Long-term stand development and demographic sustainability of tree populations in northern hardwood forests

By

Corey R. Halpin

A dissertation submitted in partial fulfillment of

the requirements for the degree of

Doctor of Philosophy

(Forestry)

at the

UNIVERSITY OF WISCONSIN-MADISON

2014

Date of final oral examination: 05/20/2014

This dissertation is approved by the following members of the Final Oral Committee:

Craig G. Lorimer, Professor, Forest and Wildlife Ecology
Ronald Gangnon, Associate Professor, Biostatistics and Medical Informatics
Stith T. Gower, Professor, Forest and Wildlife Ecology
David J. Mladenoff, Professor, Forest and Wildlife Ecology
Monica G. Turner, Professor, Zoology

General introduction

Overview

An understanding of trajectories in long-term forest development is essential for examining several fundamental issues in forest ecology and management. Long-term forest development has important implications for issues as diverse as forest productivity, carbon storage, biological diversity, and ecological forestry methods. For example, nearly all forests appear to show a decline in productivity with age (Gower et al., 1996), but it is not clear if production in old-growth forests declines to zero and becomes carbon neutral (as suggested by Odum 1969) or if old-growth stands continue to accumulate carbon (as suggested by Luyssaert et al. 2008). The sustainability of tree species populations and stand structures can likewise only be evaluated through analysis of long-term trends. Demographic sustainability of tree populations is likely to become a pressing issue in this century due to the impacts of climate change and invasive species. And while relatively few plant or animal species are true obligates for any particular stage of forest development, very substantial differences in species abundance have been associated with different developmental stages (e.g., Howe and Mossman, 1996; Werner and Raffa, 2000; Linder et al., 2006). 'Ecological forestry' practices, which seek to ensure ecosystem health, biological diversity, and long-term sustainability by emulating natural species composition, stand structure and disturbance processes, require long-term baseline data to provide sound management guidelines (Seymour and Hunter, 1999; Franklin et al., 2007).

While much has been learned from ecological studies of old-growth forests, a limitation is

that these have necessarily been brief snapshots of species composition and forest structure at one point in time from which accurate time trends can be difficult to infer. Permanent plot records are rarely available for more than a couple of decades and are usually only available in relatively young second-growth forests. More rigorous examination of these issues requires access to both long-term permanent plot data as well as forest models capable of providing accurate demographic predictions with a fairly high degree of resolution. Models are an especially promising approach because they provide the only practical means of studying changes in forests over many decades or centuries.

Forest models are available for a variety of objectives and levels of resolution. For studies of forest population dynamics, 'individual tree' models are among the most useful and versatile (Busing and Mailly, 2004; Berger et al., 2008). Individual tree models simulate the recruitment, growth, and mortality of individual trees based on local neighborhood (small plot) characteristics. Aggregate stand properties are predicted from a 'bottom up' approach in which stand growth and structure is determined form the sum of the growth and characteristics of the individual trees. As long as the calibration data set contains a wide variety of small plot conditions, the model can in principle predict the outcome of a wider range of stand conditions and treatments than may be present in the stand-level data.

There are currently three widely used families of individual-tree models: the JABOWA/ FORET family (Shugart, 1984), the SORTIE family (Pacala et al., 1996), and the FVS/ STEMS family (Belcher et al., 1982). All of these models have been useful in examining a wide range of forest ecology-forest management issues from climate change effects to forest successional patterns, but the current versions are not well-suited for the main task in this dissertation of understanding forest development in response to disturbances over multiple tree generations. None of them were calibrated with data from old-growth forests, and longer-term projections have not been rigorously tested against permanent plot data. In a medium-term comparison against permanent plot data in second-growth forest, Yaussy

(2000) found that NE-TWIGS (in the STEMS family) provided more accurate predictions of size distributions and stand volume than ZELIG (in the JABOWA/FORET family), which underestimated stand volume by 30-55% over a span of only 30 years. Yet the STEMS model, in spite of its very large calibration data set, has also been found to be inaccurate in its diameter growth and volume projections functions (Crow, 1986; Pokharel and Froese, 2008); the latter authors suggested that STEMS growth functions need to be replaced. The FVS/STEMS family, designed primarily for short-term forest inventory updates, also lacks a formal regeneration module, making long-term projections essentially impossible.

The JABOWA/FORET and SORTIE families take a more ecological approach to forest development and incorporate fairly detailed recruitment functions, but neither family contains calibration data on actual growth and mortality of mature and old trees. Models in both families also assume very simple crown and gap geometry. In JABOWA/FORET, crowns are modeled as flat circular disks, with the leaf area of each tree uniformly distributed across the plot, so that the leaf area of any tree lies completely above the leaf area of all shorter trees, even when height differences are minor. In SORTIE, crowns are modeled as cylinders that expand uniformly in all directions. These simplifications would likely impair the accuracy of simulated gap dynamics, gap closure, and gap capture, which are fundamental processes in the development of uneven-aged stands. The impact on long-term stand projections, however, is uncertain.

The overarching themes of the dissertation are the long-term trajectory of forest development and the impact of the historic natural disturbance regime on that trajectory. Embedded within these themes are three important and related concepts in forest ecology: the dynamics of changes among stand developmental stages, resilience of stand structure to the impact of disturbances, and long-term demographic sustainability of tree species. These concepts will be examined specifically using the metrics of above-ground biomass, diameter distributions, and structurally defined stand developmental stages.

The overall objectives of this dissertation are: (1) to quantify long-term trends in above-ground tree biomass and stand structure in northern hardwood forests, and to examine how these are affected by the historic natural disturbance regime, (2) to estimate the frequency of structural stand stages, along with residence times and transition rates among stages, under several disturbance regimes, and (3) to develop a quantitative approach to evaluating demographic sustainability of tree populations under several conditions, including those of restricted recruitment. Computer simulation with the CANOPY model was used to clarify long-term trends. However, to the extent possible, these model predictions were verified against available field data, including an extensive field survey and a 30-year record of change on permanent plots in the Porcupine Mountains Wilderness State Park.

Analyses in all chapters make frequent use of a structural stand stage classification system that is fully described in Appendix A. This system uses a dichotomous key based on basal area distribution among broad size classes (e.g., sapling, pole) to recognize forest developmental stages. Stand stages in this system are sorted in order of increasing modal diameter and degree of understory development. It extends the system of Lorimer and Frelich 1991a with the ability to differentiate different stages of old growth, and it is based on commonly collected field measurements. Appendix A contains background material on this classification system and may aid in interpreting these results in all chapters, especially chapters 1 and 2.

Field data sets

Two field data sets were employed in various phases of the project, one that captures a snapshot of the landscape over a broad spatial scale and another that documents small-scale forest change over a multi-decade temporal scale. The spatially broad data is a survey of 70 half-hectare plots in northern hardwood stands that were randomly located in three wilderness landscapes in Upper Michigan in 1981-84, and include stands in all stages of development. In the earlier study, stand histories for each plot were reconstructed from increment cores

(Frelich and Lorimer, 1991). A subset of eight of these plots from representative stages were stem-mapped and measured four times from 1981-2011, providing a second data set spanning three decades. On all plots in both datasets, diameter at breast height, species, and crown class were recorded for all individuals larger than 2 cm dbh.

Forest model

The modeling portion of this study employs CANOPY, an empirical, spatially explicit, individual-tree model calibrated for northern hardwood forests. The model is designed to provide detailed simulations of gap dynamics and responses to canopy disturbances over a period of several tree generations. Output is provided in the form of individual-tree measurements (dbh, height, crown radii) at 5-year intervals, which can be analyzed to produce plot-level metrics such as basal area, volume, above-ground tree biomass, and size distributions for individual species and all species pooled.

The model has several characteristics that make it especially well suited for the questions posed in this study. To our knowledge, it is the only forest model that has been calibrated with field data derived from stands in all stages of development, including highly uneven-aged, old-growth stands with trees near the end of their lifespan (350+ years). The entire model has been designed and calibrated specifically to capture the essential features of the dynamics of late-successional forests in general and northern hardwoods in particular. Predictions from CANOPY have been extensively evaluated in the past 15 years in both managed and unmanaged stands, using not only standard statistical techniques with subsets of the data withdrawn for validation purposes (e.g., Mayer and Butler, 1993; Vanclay and Skovsgaard, 1997) but also with direct comparisons of independent data from long-term experimental forest sites and other archival data (Choi et al., 2001, 2007; Halpin, 2009; Hanson et al., 2011, 2012).

Calibration data used in CANOPY reflect growth, recruitment, and mortality rates of

individual trees over a time span from about 1953 to 2011. Estimates of frequency and severity of natural disturbances were derived from tree-ring reconstructions of primary forests in upper Michigan for the time period 1830-1960 (Frelich and Lorimer, 1991). Like many empirical or mechanistic models (e.g., SORTIE and STEMS/FVS) the CANOPY model is designed only to provide predictions under the environmental conditions of the calibration data set, which are assumed to remain constant during the simulations. In its current form, CANOPY is not designed to predict specific responses to climate change, invasive species, and other major environmental changes. In this study, simulations of 500-1000 years were therefore conducted only to understand the long-term response of forests to disturbance under the baseline environmental conditions, and not to provide forecasts at any particular future time. The primary value of this approach is in enhancing the historical understanding of forest dynamics as a baseline for assessing alterations caused by global environmental change. However, in Chapter 3, CANOPY is used to explore some of the long-term implications of sapling and canopy recruitment limitations that might be induced by any number of potential environmental stressors such as climate change, invasive weedy plants, exotic insects and diseases, exotic earthworms, and excessive deer browsing.

Details of the CANOPY design have been previously published in scientific journals (Choi et al., 2001, 2007; Hanson et al., 2011, 2012) , and a brief summary of key design elements is included in the following individual chapters. But because of the limited space that can be allocated to model design in these chapters, a more comprehensive summary is included below for reader convenience, along with changes made to the model for the present study. All of the principal equations with supporting statistics are also included in Appendix B. The code of the model (in C++ programming language) is included as Appendix C.

Site quality. CANOPY uses floristic habitat types from Kotar et al. (2002) to represent site quality. These habitat types have been shown to reflect different levels of aboveground net primary production (Fassnacht and Gower, 1997) as well as the 'site index' commonly

used in forestry studies (Coffman, 1984). Habitat types were used instead of site index or direct measures of production because estimates of primary production were not available, and site index cannot be reliably determined in uneven-aged stands, which make up most of the calibration data set. The model includes separate sets of equations for the *Acer-Ozmorhza-Caulophyllum*, *Acer-Tsuga-Dryopteris*, and *Acer-Tsuga-Maianthemum* habitats, the most common and widely distributed northern hardwood habitat types in the calibration region. Habitat type can be specified independently for each 10 x 10 m cell, allowing the model to simulate stands that include a mixture of these three habitats.

Subplot-level competition. Competition in CANOPY is evaluated using regional northern hardwood stocking charts (Tubbs, 1977). These charts provide empirical measures of crowding by comparing current stand basal area with the regional average basal area for stands of the same mean diameter of dominant and codminant trees. Stocking charts are conceptually related to self-thinning diagrams (e.g., Osawa and Sugita, 1989; Westoby, 1984) in which current tree density is compared with an observed biological maximum. Previous tests indicated that the stocking levels provide a slightly better correlation with observed diameter growth and mortality than standard competition indices (Choi et al., 2001), and unlike competition indices, the numerical value of stocking level is not highly influenced by the plot size (cf. Lorimer, 1983). For growth and mortality predictions in CANOPY, competition level is evaluated for each 10 x 10 m cell by computing the stocking of the surrounding 30 x 30 m subplot. This evaluation proceeds in a 'moving-window' fashion throughout the entire stand. By using a fixed 30 x 30 m subplot size, crowding is evaluated on the scale of small neighborhoods, allowing trees to respond to local gap formation.

Diameter growth. Diameter growth of overstory trees is predicted as a function of current diameter, species, habitat type, and local subplot competition. Stochastic variation in diameter growth is incorporated based on the mean squared error of the regressions. As each tree recruits into the stand, a 'growth modifier' variable is set following a normal distribution.

This sets some trees on faster or slower growth trajectories (e.g., 10% higher than the mean), but note that the mean values themselves can change in response to changing forest density.

The current revision of CANOPY (version 3) includes an option to use an alternative set of growth and mortality equations that take into account small but statistically significant differences in the growth rates of trees in even-aged vs. uneven-aged stands. The user may select the default option that distinguishes the two stand structures only by local differences in stocking at the subplot level, or else use the alternative equations that include categorical variables and interaction terms to further distinguish even- and uneven-aged stands. If the equations with age-structure variables are selected for use, even-aged growth and mortality rates will be applied to any tree that recruits into a sapling patch until the tree reaches age 200, which is the mean longevity of canopy trees (Lorimer et al., 2001). After that point, the patch is assumed to develop the gap structure of an uneven-aged stand, and the uneven-aged equations are applied. When new recruits are added, the developmental stage of the stand (i.e., sapling or otherwise) is evaluated on a 50x50 m patch size, in order to have a reasonably sized sub-population.

Mortality. Background mortality (i.e., resulting from senescence or competition rather than disturbance) of all trees is modeled stochastically. Probabilities of mortality for each species are predicted by logistic regressions in terms of initial diameter and stocking. Mortality functions for each species follow a U-shaped trend, with very high rates of mortality for small trees, decreasing to a minimum level for most species at around 30 cm dbh, then increasing again after around 66 cm dbh as trees become larger and older (Clark, 1992; Coomes and Allen, 2007).

CANOPY includes an option to simulate the historic natural disturbance regime based on the recurrence intervals in Frelich and Lorimer (1991). In each year, an expected proportion of canopy removal is stochastically determined. For years where this removal is less than 10% of the crown area, no disturbance is simulated and instead the background mortality

subroutine is used. Otherwise, individual tree probability of windthrow equations are used stochastically to allocate disturbance mortality to particular trees. These equations predict probability of mortality for each species in terms of storm severity index and initial size (Hanson, 2009). The storm severity index follows the definition of Canham et al. (2001), and is a 0-1 parameter estimated for each calibration plot by maximum likelihood that uses plot-level degree of damage to assess the severity of disturbance without the need for explicit wind speed measurements. A regression was used to relate the percent crown area removal used in studies of recurrence intervals with the storm severity index used to develop probability of windthrow equations. This regression is used to compute an initial storm severity index, but then CANOPY checks that this storm severity index is sufficient to remove the target proportion of crown area determined from the recurrence intervals. If not, the severity is adjusted accordingly. This procedure is iterated until the target amount of crown area is removed. The recurrence intervals used in CANOPY incorporate all disturbances removing overstory trees, including not only windstorms but also other disturbances like drought and disease. Because explicit size-dependent probability of mortality equations for these other disturbances are not available, CANOPY makes a simplifying assumption that these disturbances also preferentially remove large trees and uses the probability of windthrow equations for all disturbances.

Gap dynamics and sapling height growth. In CANOPY v1, tree growth and mortality was directly influenced by crown projection area and exposed crown area. However, when the calibration data set was greatly expanded to include other sites where crown data was not available, the model had to be simplified to exclude crown variables as independent predictors of overstory growth and mortality. In CANOPY v2 and v3, crown variables are used to delineate the boundaries of canopy gaps; crown competition variables and gap area are then used as independent predictors of sapling height growth.

Asymmetric crown growth of overstory trees is simulated in response to available growing

space. Each crown is divided into quarter ellipses, one for each cardinal direction, and growth of each crown radius is independently simulated. When a crown quarter abuts the crown of a taller tree, no further lateral crown growth is predicted for that crown quarter on the shorter tree. Growth resumes as normal after the crowns no longer touch (e.g., when the taller tree dies). Growth also continues as normal for the other crown quarters. Overtopped trees use crown growth rates based differentiating an allometric relationship between diameter and mean crown radius. Overstory trees use rates from Webster (2002), which depend on their species, diameter, and degree of crown exposure.

For small trees, height growth is simulated directly, rather than being estimated from diameter growth via an allometric relationship as in most other models. Height growth for each species is predicted from initial height, gap area, and crown competition variables. For each tree, the eight closest overstory trees in cardinal and intercardinal directions are identified. The closest points to the subject tree along the driplines of these trees form the vertices of an irregular octagon used to estimate the 'gap area' for each tree (if there is no gap, then gap area = 0). The crown area of other saplings on a competition subplot centered on the sapling is summed for the crown term. Similar to the diameter growth predictions, stochastic variation is also incorporated into predictions of height growth using normally distributed error based on the MSE of the regression fits. As with the diameter growth, a stochastic modifier is assigned to each tree when it recruits into the stand.

Sapling recruitment. The number and species of 2-6 cm trees entering each 10x10 m cell determined by the species composition and stocking level of the surrounding forest. Stocking values are computed for the central cell and for the 8 cells that share a common border, allowing CANOPY to recognize gaps. Based on these stocking values, the number of 2-6 cm trees expected given the measured stocking is predicted from a field-calibrated regression. If there is a deficit of saplings compared to the density predicted for that level of subplot competition, new saplings are added to make up the deficit. Initial diameters of

these new recruits follow a lognormal distribution of diameters between 2 and 6 cm based on the calibration dataset. When there is a fractional deficit, then this is used as a stochastic probability of adding a single tree. For example, if the deficit is 0.3 trees, then there will be a 30% probability of adding a single tree. Initial species for the newly added individual are determined stochastically based on overstory density and species composition, following a method similar to Vanclay (1994).

A refinement was also made to the recruitment mechanism in CANOPY v.3 to allow it to mimic the sparse recruitment commonly observed in stands with a high proportion of pole and mature trees. Even on plots relatively unaffected by deer browsing, high concentrations of pole and mature trees were frequently associated with low density of large saplings and small poles ($p<0.001$). Based on this observation, a rule was added to CANOPY that delays further recruitment of saplings whenever pole trees comprise $\geq$20% of the basal area or mature trees comprise $\geq$ 35% of the basal area. This rule is enabled or disabled for each simulation run by a user-selected switch. When the switch is enabled, this rule is evaluated for each 10 x 10 m cell, based on the basal area percentages on the 50 x 50 m (0.25 ha) patch centered on that cell. Uneven-aged stands experiencing small gap formation typically have basal area of pole and mature trees below these thresholds, and therefore are not affected by this rule even when it is enabled.

Overview of chapters 1-3

Chapter 1 evaluates long-term stand development, specifically in relation to biomass trends and the underlying demographic changes that drive them. This chapter employs both long-term simulation and analysis of field evidence to provide a systematic test of the Bormann and Likens (1979) hypothesis that biomass developing after stand-replacing disturbance reaches a peak and then declines to a lower steady-state level having approximately zero net growth. An alternative hypothesis, that biomass shows an asymptotic trend (e.g., Lich-

stein et al., 2009; Keeton et al., 2011) is also tested. Simulations track biomass development after clearcutting for 1,000 years under a disturbance regime comprised only of individual treefall gaps caused by background mortality and under the historic natural disturbance regime. Two independent tests of these hypotheses are also conducted using field data. Biomass from the full 70-plot upper Michigan data set was arranged by stand developmental stage to determine if levels of biomass are lower late in stand development. Changes in biomass over the 30-year measurement period were also analyzed on the permanent plots to evaluate trends in net growth across a gradient of developmental stages.

Whether or not there is a peak in the biomass trend could partly depend on the ability of secondary cohorts to compensate for overstory attrition during the transition from even-aged to all-aged structure. In this chapter, this mechanism is tested both under a regime of small treefall gaps and under the historic natural disturbance regime. Biomass net growth from both simulations and field data is computed for individual trees and summarized by cohorts. The frequency with which biomass net growth of the understory cohorts can compensate for overstory attrition is evaluated.

The Bormann-Likens hypothesis predicts that the above-ground component of old-growth forests will be approximately carbon neutral, whereas the alternative hypothesis predicts that carbon accumulation may continue long into stand development (>800 years). The Bormann-Likens peak/decline trend is supported by several simulation models (e.g., Shugart, 1984; Bragg et al., 2004; Vanderwel et al., 2013), while the asymptotic trend is supported by multiple chronosequence analyses of field data (e.g., Lichstein et al., 2009; Keeton et al., 2011). The possibility that 'noise' in estimates of stand age may prevent chronosequence techniques from detecting a biomass peak is also explored.

Chapter 2 evaluates developmental dynamics at both the stand-level and larger scales from the perspective of forest structural change. Disturbance history (e.g., the timing and severity of disturbances) has frequently been used to evaluate stand development (e.g.,

Franklin et al., 2002; Zenner, 2005; D'Amato et al., 2009), but structural criteria for stand development may be both easier to evaluate in the field and more closely related to a variety of ecologically important processes. When disturbance history becomes sufficiently complex, with a number of layered past disturbances of various severities and ages, each leaving a ragged structure of surviving legacy trees, it becomes essentially impossible to define a meaningful 'stand age'. While much of a landscape may indeed have been recently disturbed, it may also have had sufficient resilience to disturbance that much of it remains essentially structurally unchanged. Therefore, a large proportion of the landscape may be occupied by stands that are structurally not distinct from those undisturbed for a long time.

In this chapter, 1,000 year simulations were conducted under two contrasting disturbance regimes: a 'dichotomous' regime of frequent small gap formation with infrequent severe disturbance, and the historic natural disturbance regime comprising a range of disturbance frequency and severity. From these simulations, a second-order Markov chain was constructed and used to compute the balance of stand structural stages for an infinitely sized population of plots at equilibrium. Markov-chain results were then used to compute state transition diagrams with residence times, illustrating the way that individual stands flow through structural stages as well as the proportion of a landscape-scale population of stands in each stage. Simulation experiments were also performed that test the impact of either single or repeated light and moderate disturbances on old-growth stands.

Chapter 3 quantitatively evaluates the ability of individual species to sustain their populations under specified conditions. In contrast to the 'minimum viable population' of animal ecology (cf. Boyce, 1992), there are no widely accepted quantitative criteria to assess population sustainability in forests. A typical *ad hoc* criterion for sustainability of forest tree populations is the presence of descending monotonic size distributions so that higher numbers of small trees can compensate for attrition due to mortality (Smith et al., 1997; Nyland, 2007). This criterion also attempts to loosely mimic the descending monotonic form observed

in many old-growth stands of late successional species.

Meyer and Stevenson (1943) observed a descending monotonic demographic distribution in unmanaged hemlock/hardwood forests, and proposed the negative exponential distribution as a mathematical model to describe this pattern. Under this model, the ratios of the numbers of trees in successive size classes (called the 'q-ratio' in forestry literature) is constant. Goff and West (1975) suggested that the constant rates of mortality across size classes implied by constant q-ratios may be untenable. They proposed a 'rotated sigmoid' distribution form, which has a variable q-ratio across size classes. There is ongoing debate in the forestry literature regarding which of these forms may be correct, with some investigators suggesting that a rotated sigmoid form may reflect either recent disturbance or represent a sampling artifact (Schmelz and Lindsey, 1965; Leak, 2002; Rubin et al., 2006).

But regardless of the specific mathematical form taken by a descending monotonic curve, the mere presence of a descending monotonic shape does not definitively indicate a sustainable distribution because the populations of small trees may still be insufficient to balance mortality. Forests around the world are increasingly faced with a growing complex of stressors that may threaten their long-term sustainability, including climate change as well as invasive and exotic pests. Under these circumstances, quantitative tools to assess the long-term sustainability of tree populations are urgently needed. In this chapter, a quantitative metric of tree population sustainability was developed that evaluates the persistence of tree populations based on their initial demographic structure. This demographic sustainability index was then used in a series of simulation experiments designed to determine the minimum number of saplings and the proportions of trees in different size classes (q-ratios) needed to sustain the current overstory basal area.

Literature Cited

Belcher, D. M., M. R. Holdaway, and G. J. Brand (1982). A description of STEMS – the stand and tree evaluation and modeling system. General Technical Report NC-79, USDA Forest Service.

Berger, U., C. Piou, K. Schiffers, and V. Grimm (2008). Competition among plants: Concepts, individual-based modeling approaches, and a proposal for a future research strategy. *Perspectives in Plant Ecology, Evolution, and Systematics 9*, 121–135.

Bormann, F. H. and G. E. Likens (1979). *Pattern and process in a forested ecosystem.* New York: Springer-Verlag.

Boyce, M. S. (1992). Population viability analysis. *Annual Review of Ecology and Systematics 23*, 481–506.

Bragg, D. C., D. W. Roberts, and T. R. Crow (2004). A hierarchical approach for simulating northern forest dynamics. *Ecological Modelling 173*, 31–94.

Busing, R. T. and D. Mailly (2004). Advances in spatial, individual-based modeling of forest dynamics. *Journal of Vegetation Science 15*, 831–842.

Canham, C. D., M. J. Papaik, and E. F. Latty (2001). Interspecific variation in susceptibility to windthrow as a function of tree size and storm severity for northern temperate tree species. *Canadian Journal of Forest Research 31*, 1–10.

Choi, J., C. G. Lorimer, J. Vanderwerker, W. G. Cole, and G. L. Martin (2001). A crown model for simulating long-term stand and gap dynamics in northern hardwood forests. *Forest Ecology and Management 152*(1), 235–258.

Choi, J., C. G. Lorimer, and J. M. Vanderwerker (2007). A simulation of the development and restoration of old-growth structural features in northern hardwoods. *Forest Ecology and Management 249*(3), 204–220.

Clark, J. S. (1992). Density-independent mortality, density compensation, gap formation, and self-thinning in plant populations. *Theoretical Population Biology 42*(2), 172–198.

Coffman, M. S. (1984). *Field guide: Habitat classification system for upper peninsula of Michigan and northeast Wisconsin.* Houghton, MI, USA: CROFS, School of Forestry and Wood Products, Michigan Technological University.

Coomes, D. A. and R. B. Allen (2007). Mortality and tree-size distributions in natural mixed-age forests. *Journal of Ecology 95*(1), 27–40.

Crow, T. R. (1986). Comparing the performance of two northern hardwood growth projection systems. *Northern Journal of Applied Forestry 3*(1), 28–32.

D'Amato, A. W., D. A. Orwig, and D. R. Foster (2009). The influence of successional processes and disturbance on the structure of *Tsuga canadensis* forests. *Ecological Applications 18*, 1182–1199.

Fassnacht, K. S. and S. T. Gower (1997). Interrelationships among the edaphic and stand characteristics, leaf area index, and aboveground net primary production of upland forest ecosystems in north central Wisconsin. *Canadian Journal of Forest Research 27*, 1058–1607.

Franklin, J. F., R. J. Mitchell, and B. Palik (2007). Natural disturbance and stand development principles for ecological forestry. General Technical Report NRS-19, USDA Forest Service.

Franklin, J. F., T. A. Spies, R. van Pelt, A. B. Carey, D. A. Thornburgh, D. R. Berg, D. B. Lindenmayer, M. E. Harmon, W. S. Keeton, D. C. Shaw, K. Bible, and J. Chen (2002). Disturbance and structural development of natural forest ecosystems with silvicultural implications, using Douglas-fir forests as an example. *Forest Ecology and Management 155*, 399–423.

Frelich, L. E. and C. G. Lorimer (1991). Natural disturbance regimes in hemlock-hardwood forests of the upper Great Lakes region. *Ecological Monographs 61*(2), 145–164.

Goff, F. G. and D. West (1975). Canopy-understory interaction effects on forest population structure. *Forest Science 21*, 98–108.

Gower, S. T., R. E. McMurtie, and D. Murty (1996). Aboveground net primary production decline with stand age: Potential causes. *Trends in Ecology and Evolution 11*, 378–382.

Halpin, C. R. (2009). Simulated long-term effects of group selection on production, efficiency, composition, and structure in northern hardwood and hemlock-hardwood forests. Master's thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J. (2009). *Emulating natural disturbance dynamics in northern hardwood forests: Long-term effects on species composition, forest structure, and yield.* Ph. D. thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J., C. G. Lorimer, and C. R. Halpin (2011). Predicting long-term sapling dynamics and canopy recruitment in northern hardwood forests. *Canadian Journal of Forest Research 41*, 903–919.

Hanson, J. J., C. G. Lorimer, C. R. Halpin, and B. J. Palik (2012). Ecological forestry in an uneven-aged, late-successional forest: Simulated effects of contrasting treatments on structure and yield. *Forest Ecology and Management 270*, 94–107.

Howe, R. W. and M. Mossman (1996). The significance of hemlock for breeding birds in the western Great Lakes region. In G. Mroz and A. J. Martin (Eds.), *Proceedings, Hemlock Ecology and Management, September 27-28, 1995*, pp. 125–139. University of Wisconsin – Madison.

Keeton, W. S., A. A. Whiteman, G. C. McGee, and C. L. Goodale (2011). Late-successional biomass development in northern hardwood-conifer forests of the northeastern United States. *Forest Science 57*(6), 489–505.

Kotar, J., T. L. Burger, and J. A. Kovach (2002). *A guide to forest communities and habitat types of northern Wisconsin.* Madison, WI: Department of Forest Ecology and Management, University of Wisconsin – Madison.

Leak, W. B. (2002). Origin of sigmoid diameter distributions. Research Paper NE-718, USDA Forest Service.

Lichstein, J. W., C. Wirth, H. S. Horn, and S. W. Pacala (2009). Biomass chronosequences of United States forests: Implications for carbon storage and forest management. In C. Wirth, G. Gleixner, and M. Heimann (Eds.), *Old growth forests: Function, fate, and value*, pp. 301–341. Berlin: Springer-Verlag.

Linder, D. L., H. H. Burdsall, and G. R. Stanosz (2006). Species diversity of polyporoid and corticioid fungi in northern hardwood forests with differing management histories. *Mycologia 98*, 195–217.

Lorimer, C. G. (1983). Tests of age-independent competition indices for individual trees in natural hardwood stands. *Forest Ecology and Management 6*(4), 343–360.

Lorimer, C. G., S. E. Dahir, and E. V. Nordheim (2001). Tree mortality rates and longevity in mature and old-growth hemlock-hardwood forests. *Journal of Ecology 89*(6), 960–971.

Luyssaert, S., E. Schulze, A. Börner, A. Knohl, D. Hessenmöller, B. E. Law, P. Ciais, and J. Grace (2008). Old-growth forests as global carbon sinks. *Nature 455*, 213–215.

Mayer, D. G. and D. G. Butler (1993). Statistical validation. *Ecological Modeling 68*(1), 21–32.

Meyer, H. A. and D. D. Stevenson (1943). The structure and growth of virgin beech-birch-maple-hemlock forests in northern Pennsylvania. *Journal of Agricultural Research 67*(2), 465–484.

Nyland, R. D. (2007). *Silviculture: Concepts and applications.* Long Grove, IL, USA: Waveland Press.

Odum, E. P. (1969). The strategy of ecosystem development: An understanding of ecological succession provides a basis for resolving man's conflict with nature. *Science 164*, 262–270.

Osawa, A. and S. Sugita (1989). The self-thinning rule: Another interpretation of Weller's results. *Ecology 70*, 279–283.

Pacala, S. W., C. D. Canham, J. Saponara, J. A. S. Jr, R. K. Kobe, and E. Ribbens (1996). Forest models defined by field measurements: Estimation, error analysis and dynamics. *Ecological Monographs 66*(1), 1–43.

Pokharel, B. and R. E. Froese (2008). Evaluating alternative implementations of the Lakes States FVS diameter increment model. *Forest Ecology and Management 255*, 1759–1771.

Rubin, B. D., P. D. Manion, and D. Faber-Langendoen (2006). Diameter distributions and structural sustainability in forests. *Forest Ecology and Management 222*, 427–438.

Schmelz, D. V. and A. A. Lindsey (1965). Size-class structure of old-growth forests in Indiana. *Forest Science 11*, 258–264.

Seymour, R. S. and M. L. Hunter (1999). Principles of ecological forestry. In M. L. Hunter (Ed.), *Maintaining Biodiversity in Forest Ecosystems*, pp. 22–62. Cambridge University Press.

Shugart, H. H. (1984). *A theory of forest dynamics: The ecological implications of forest succession models*. New York: Springer-Verlag.

Smith, D. M., B. C. Larson, M. J. Kelty, and P. M. Ashton (1997). *The practice of silviculture: Applied forest ecology*. New York: John Wiley and Sons.

Tubbs, C. H. (1977). Manager's handbook for northern hardwoods in the North Central states. General Technical Report NC-39, USDA Forest Service.

Vanclay, J. K. (1994). *Modeling forest growth and yield: Applications to mixed tropical forests*. Wallingford, U.K.: C.A.B. International.

Vanclay, J. K. and J. P. Skovsgaard (1997). Evaluating forest growth models. *Ecological Modelling 98*(1), 1–12.

Vanderwel, M. C., D. A. Coomes, and D. W. Purves (2013). Quantifying variation in forest disturbance, and its effects on aboveground biomass dynamics, across the eastern United States. *Global Change Biology 19*, 1504–1517.

Webster, C. R. (2002). *Comparative effects of single-tree and group selection on the diversity and productivity of hemlock-hardwood forests*. Ph. D. thesis, University of Wisconsin – Madison, Madison, WI, USA.

Werner, S. M. and K. F. Raffa (2000). Effects of forest management practices on the diversity of ground-occurring beetles in mixed northern hardwood forests of the Great Lakes region. *Forest Ecology and Management 139*, 135–155.

Westoby, M. (1984). The self-thinning rule. *Advances in Ecological Research 15*, 167–225.

Yaussy, D. A. (2000). Comparison of an empirical forest growth and yield simulator and a forest gap simulator using actual 30-year growth from two even-aged forests in Kentucky. *Forest Ecology and Management 126*(3), 385–398.

Zenner, E. K. (2005). Development of tree size distributions in Douglas-fir forests under differing disturbance regimes. *Ecological Applications 15*(2), 701–714.

Dedication

To Ronda and Kyra – I finished page three.

Acknowledgments

I thank my adviser, Dr. Craig Lorimer, for his guidance, support, and mentoring over the course of this project and also my entire stay in graduate school. The dedication, detail, and care he brings to his work serve as a model as I continue to improve my own. I also thank the other members of my committee, Drs. Gangnon, Gower, Mladenoff, and Turner, for the discussions and critical reviews that were instrumental in honing this research. I further thank the institutions that provided financial support for this research, including the McIntire-Stennis Cooperative Forestry Research Program, project WIS01514; the School of Natural Resources, College of Agricultural and Life Sciences, University of Wisconsin – Madison, and the Internet Scout Research Group, Department of Computer Science, University of Wisconsin – Madison.

I am grateful for permission from the administration of the Porcupine Mountains State Park to re-measure permanent plots located there, and to Kelsey Egelhoff for her help in doing so. I am indebted to the people who established and previously measured the Porcupine Mountains plots, Drs. Lee Frelich, Sally Dahir, and Jake Hanson. Without the foundation provided by their work, my own could not have been so strong. I am also indebted to Jayne Vanderwerker, Chris Webster, Jungkee Choi, and Jake Hanson for their work on earlier versions of the CANOPY model, without which this project would not have been possible.

Lastly, I thank my family for their patience and support, without which this work and so many other things could never have been.

Contents

CHAPTER 1

Long-term trends in biomass and tree demography in northern hardwood forests: An integrated field and simulation study

Abstract

Long-term trends in biomass development in forests have important implications for carbon management, but chronosequence field studies and simulation projections have produced differing conclusions about patterns of biomass accumulation and the magnitude of a biomass peak. In this paper, the Bormann-Likens hypothesis of a peak and decline in biomass followed by zero net growth after several centuries was tested against an alternative hypothesis of asymptotic biomass development having positive net growth even at advanced ages. Trends in aboveground tree biomass and large tree density (>50 cm dbh) were examined from 30-year permanent plot data and replicated, multi-century simulations using CANOPY, an empirical, individual-tree, spatially explicit model. Simulations were performed on a variety of habitat types and species mixtures. Both field data and simulations indicated a decline in aboveground tree biomass in later stages of old growth, with a corresponding decline in the number of large trees. This decline was robust to modeling assumptions and occurred to varying degrees on all habitat types and species mixtures tested. Peak aboveground tree biomass occurred at a time when age structure was changing from even-aged to multi-aged, and when the underlying size distribution showed a transition from

unimodal to descending monotonic form. Biomass net growth of secondary cohorts was usually found to be insufficient to compensate for attrition of the initial even-aged cohort. A simulation analysis mimicking chronosequence methods, however, produced a misleading asymptotic biomass trend because the mean age of the largest trees was not equivalent to time since clearcut after age 300, which distorted the time scale. Incorporating natural disturbances into the simulations lowered the level of the biomass peak, but a subsequent decline was still predicted. Net growth of aboveground tree biomass in old-growth stands with descending monotonic size distributions was approximately zero for both the field data and simulations, even when natural disturbances were included. Results of this study suggest that the aboveground component of old-growth forests is unlikely to continue accumulating carbon at very advanced ages, even when averaging over a large area containing some younger stands with positive net growth.

1.1 Introduction

Aboveground forest biomass is an important part of the total carbon budget for terrestrial ecosystems, comprising 33% of total carbon in temperate forests (Turner et al., 1995), and is highly responsive to anthropogenic and natural disturbances. Carbon storage capacity in forests has recently become a topic of increasing global concern as carbon management plans are devised (Heimann, 2009; Ashton et al., 2012). An understanding of long-term trends in aboveground tree biomass will be necessary to assess forest carbon storage potential and the degree to which carbon storage in forests can be maintained. Relevant data, however, are difficult to obtain because older forests are globally uncommon, landscapes incorporating a wide range of forest developmental stages are rare in many temperate ecosystems, and permanent plot records are usually of short duration (e.g., van Doorn et al., 2011).

Based on an early forest model (JABOWA), Bormann and Likens (1979) constructed a systematic hypothesis regarding the trend of tree biomass following stand-replacing dis-

turbance. They suggested a 'reorganization' stage where trees become established, followed by an 'aggradation' stage where a single dominant cohort increases rapidly in biomass and attains maturity. In the subsequent 'transition' stage, the initial cohort breaks up and gives way to a broader range of age classes. In late-successional forests with a gap-based disturbance regime, it then enters a 'steady state' stage where the forest is composed of a shifting mosaic of patches of various age classes. This has been the most common hypothesis for the past 35 years, and is the baseline from which many subsequent studies begin (Keeton et al., 2011; Lichstein et al., 2009). Under this hypothesis, peak biomass is attained at the beginning of the transition phase when the initial even-aged cohort reaches its maximum development. During the transition phase, biomass declines to a lower level before entering the steady-state, when net growth is approximately zero.

The hypothesis of a decline in biomass in the steady state is biologically plausible, but JABOWA was an early prototype that lacked a formal calibration data set, and few data in New England were available from actual old-growth forests. JABOWA also uses a simple crown geometry, modeling crowns as flat circular discs and distributing leaf area of individual trees evenly across the entire 10 x 10 m simulation plot (Botkin et al., 1972; Busing and Mailly, 2004). It is uncertain if the simplified crown geometry would accurately simulate gap capture and gap closure, both fundamental processes in the growth and development of uneven-aged stands. Additionally, the early JABOWA simulations did not incorporate periodic moderate-severity disturbances, which would likely affect the biomass trajectory in actual stands. Subsequent field studies and simulations in northern hardwood/conifer ecosystems have suggested that periodic moderate disturbances (30-60% canopy removal) have such a pervasive impact on actual landscapes that moderate rather than catastrophic disturbances overwhelmingly dominate the stand development pathways (Frelich and Lorimer, 1991a,b; Splechtna et al., 2005; Papaik and Canham, 2006; Hanson and Lorimer, 2007; D'Amato and Orwig, 2008; Fraver et al., 2009).

Several recent field studies of chronosequences have also raised some doubt regarding the hypothesized peak and decline, with their data being more consistent with an asymptotic trend. In a chronosequence analysis of US forest inventory data, Lichstein et al. (2009) documented an apparent asymptotic increase in biomass with age, and hypothesized that the expectation of a decline may have been an artifact of how mortality is handled in models. Keeton et al. (2011) found a similar asymptotic trend in biomass even in stands with trees up to 400 years old and partially attributed this continued increase to a concurrent linear increase in the number of large trees. Luyssaert et al. (2008) reported an apparent increase in biomass with age well into the old-growth stage. They suggested that a biomass decline was absent because losses due to mortality of large trees were balanced by increased productivity in the smaller trees released when the larger trees die. According to this hypothesis, old-growth forests function as a carbon sink even at very advanced ages (>500 years).

Permanent plot re-measurements at the Hubbard Brook Experimental Forest also appear to document an asymptotic trend in biomass (Siccama et al., 2007; van Doorn et al., 2011). Surprisingly, an asymptote appears to have been reached at a predominant tree age of only 80-90 years, well before the transition to old growth, and at a much lower biomass (245 Mg/ha) than originally expected (regression estimate of 350 Mg/ha; Whittaker et al. 1974). They hypothesized that local factors (e.g., site degradation by acid rain, beech bark disease) may be driving this trend, so it is not clear if this pattern can be broadly generalized. Unfortunately, there are few other permanent plot data sets measured over a similar timeframe for comparison.

In this paper, the Bormann and Likens (1979) peak/decline hypothesis in northern hardwood forests of the Great Lakes region is evaluated against the alternative hypothesis of asymptotic biomass development. Changes in biomass production are analyzed in relation to underlying changes in tree mortality and tree size distributions. The effects of periodic light and moderate disturbances on biomass trajectories are also examined. If the Bormann-

Likens peak/decline hypothesis is valid, we would expect the following individual hypotheses to be supported by field data and simulations. (1) In stands developing after stand-replacing disturbance and then subjected only to individual treefall gaps (i.e. background mortality), aboveground live-tree biomass and number of large trees reach maximum levels at a time corresponding to the average longevity of canopy trees in the post-disturbance cohort. (2) The rate of biomass increase of the younger trees in the understory and in canopy gaps is insufficient to balance the biomass loss from overstory mortality, resulting in a subsequent decline in live biomass. (3) Several hundred years after stand initiation, stands reach a steady-state with lower biomass and approximately zero net biomass increment. (4) In the first several centuries after the initial disturbance, diameter distributions progress from a unimodal to a compound (bimodal) to a steeply descending monotonic curve form in one maximum lifespan of the dominant trees, with peak biomass accumulation coinciding with the development of a compound diameter distribution (unimodal distribution of overstory trees and steeply descending 'tail' corresponding to understory trees). (5) When the historic natural disturbance regime is incorporated, moderate disturbances interrupt the trajectory of biomass accumulation, preventing a peak from developing, and resulting in an asymptotic trend when averaged over a number of sites. For individual stands, however, quiescent periods are sometimes long enough that a biomass peak followed by a decline still occurs.

Rather than using a single line of evidence, we test these hypotheses using an integrated approach that combines abundant field data on unmanaged mature and old-growth northern hardwood forests with simulations using a well-tested, individual-tree model calibrated with data from the same region. The field data include a broad, landscape-level survey in extensive tracts of old-growth forest as well as 30-year permanent plot measurements from stands ranging in age from young pole stands to broadly uneven-aged old growth. The model (CANOPY) is calibrated with data from >8,000 trees from permanent and temporary plots in second-growth and old-growth stands spanning a similarly wide range of development

(Hanson et al., 2011).

1.2 Methods

1.2.1 Study areas

Measurements were conducted in three upper Michigan natural areas dominated by northern hardwoods and having little or no history of logging: core areas in the Porcupine Mountains Wilderness state park, Sylvania Wilderness in the Ottawa National Forest, and a tract of protected private lands owned by the Huron Mountain Wildlife Foundation (Christy, 1929; Rafferty and Sprague, 2001). Northern hardwoods are widely distributed across the northeastern US and are dominated by late-successional species such as sugar maple (*Acer saccharum*), yellow birch (*Betula alleghaniensis*), and eastern hemlock (*Tsuga canadensis*). The natural disturbance regime is dominated by frequent small gap formation, occasional moderate-severity disturbance, and infrequent stand-replacing disturbance (Frelich and Lorimer, 1991a; Schulte and Mladenoff, 2005; Fraver et al., 2008; D'Amato and Orwig, 2008). Climate is humid continental with short cool summers. Average July temperatures are 20 °C and average January temps are -7.5 °C near Lake Superior, and -10.9 °C further inland. Annual precipitation averages 80-90 cm and is fairly evenly distributed throughout the year. Elevations range from 182 m to ∼600 m. Soils are primarily Fragiorthods and Haplorthods with a loam or sandy loam texture. The majority of plots are on the *Acer-Tsuga-Dryopteris* (ATD) habitat type of Kotar et al. (2002), which has sugar maple site index of ∼19-20 m at base age 50 (Coffman, 1984) and ANPP in second-growth stands of approximately 7.2-9.5 Mg/ha/yr (Fassnacht and Gower, 1997). In nearby experimental forests on similar sites, sugar maple site index is 18-21 m for base age 50 years (Erdmann and Oberg, 1973; Crow et al., 1981). The other less common habitats (Frelich, 1986) are also mesic and above-average in productivity, but with some variation in site quality and species composition.

1.2.2 Field methods

Seventy half-hectare plots were positioned by random coordinates on maps of primary forest in advance of the initial field survey. On each plot, diameter at breast height (DBH) and crown class were recorded for all trees $\geq$ 1.4 meters tall. Increment cores were taken from canopy trees nearest to 10-30 randomly located points. Initial measurement was in the summers of 1981-1984, with eight plots permanently marked for remeasurement. Permanent plots were selected to span a wide range of developmental stages, from pole through old-growth stands approaching a steady-state structure. On these plots, an average of 68 increment cores were taken per plot (range 39-99). Remeasurements were conducted in 1992, 2004, and 2011. Over the 30-year measurement period, the permanent plots have not experienced significant windstorms but were subjected to a severe drought in 1988 that caused high mortality of large yellow birch (Lorimer et al., 2001).

1.2.3 Model description

CANOPY is a spatially explicit, individual-tree model that simulates gap capture and closure via height growth of saplings and lateral crown growth by overstory trees (Hanson et al., 2011, 2012). Calibration data for the model were collected in northern Wisconsin and upper Michigan and span a wide range of stand developmental stages, including young even-aged stands through broadly uneven-aged old-growth. CANOPY version 3, which is used in this paper, expands the calibration dataset to include a substantial amount of long-term permanent plot data from managed and unmanaged second-growth hardwood stands in the Argonne Experimental Forest of northern Wisconsin (Niese and Strong, 1992).

CANOPY predicts the number of 2-6 cm saplings recruiting into the stand based on overstory density and species composition. Height growth of saplings is predicted for each species based on initial height, gap area, and crown competition. Diameter growth of canopy trees is predicted for each species as a function of initial tree size and plot level competition.

Crowns of canopy trees grow asymmetrically in four cardinal directions; growth rate along each crown radius depends on the available growing space in that direction. Background mortality of trees is modeled as a logistic function of initial tree size and plot competition level. The growth, mortality, and recruitment equations all incorporate categorical variables and interaction terms to account for habitat variation among the three floristic habitat types represented in the calibration dataset (*Acer-Ozmorhza-Caulophyllum*, *Acer-Tsuga-Dryopteris*, and *Acer-Tsuga-Maianthemum* types of Kotar et al. 2002). Predictions of height and diameter growth include stochastic variation following normal distributions based on the mean squared error of the regression fits. Once trees die, they move through decay classes following a Markov process, and estimates of dead volume or biomass can be computed.

CANOPY includes a subroutine to emulate the historic natural disturbance regime of the upper Great Lakes region. Disturbances removing variable amounts of crown cover are simulated following the recurrence intervals from Frelich and Lorimer (1991a). CANOPY then applied individual-tree probability of blowdown equations from Hanson (2009). These are a function of the size and species of individual trees, as well as the storm severity index, an empirical estimate of storm severity based on the overall amount of blowdown on a plot (Canham et al., 2001). A regression was developed to relate storm severity index to the amount of canopy removal assessed in the earlier studies of the natural disturbance regime. CANOPY checks that the estimated storm severity was sufficient to cause the required amount of canopy removal, and iteratively adjusts it if it was not. If the actual removed canopy area differs by more than 5% from the target, the storm severity is adjusted and new individual-tree windthrow probabilities are calculated and applied. This procedure is repeated until the target amount of canopy area is removed.

Previous validation tests have demonstrated good correspondence of CANOPY's predictions of forest structural change with archival data and independent experimental trials. Long-term predictions of stand basal area and size distributions under small-gap distur-

bance regimes match closely with archival data on uneven-aged, old-growth stands from the region. For example, in a multi-century simulation of a mixed hardwood site under background mortality (Halpin, 2009), CANOPY's predicted basal area stabilized at 34 m^2/ha within the first 300 years of simulation, compared to 34 m^2/ha reported by Janowiak et al. (2008) and Goodburn and Lorimer (1999) on similar sites. After basal area stabilized, only minor fluctuations were predicted, suggesting little evidence of error propagation even in multi-century predictions. Predicted numbers of trees per ha in sapling, pole, mature and large size categories fell near the middle of the range observed in old-growth stands (Hanson et al., 2012). At the end of 1,000-year simulations, size distributions of mixed hardwood stands were nearly identical to the mean distributions of the 18 steady-state stands in the field data (Fig. 2 in Chapter 3).

A modification was made to the recruitment module in CANOPY v.3 to mimic sparse understory development often observed in earlier stages of stand development. The field data contain evidence of low ingrowth and high mortality for small trees in stands with high concentrations of pole and mature trees, even with low levels of deer browsing (Chapter 2). The recruitment mechanism in CANOPY v.2 predicted fairly abundant regeneration in some of these situations. Based on the negative correlation between percent basal area of mature trees and sapling density (r=-0.48, p=0.005), a switch was added to the recruitment module that delays further addition of saplings whenever pole trees occupy $\geq$ 20% or mature trees $\geq$ 35% of the basal area in the 50 x 50 m patch centered on each 10 x 10 m cell. Uneven-aged stands with small gap formation normally have basal area levels lower than these thresholds and therefore are not affected by this rule, generally allowing development of a dense understory.

Growth and mortality equations in CANOPY v.3 were also refined to account for subtle differences between even- and uneven-aged stands. The CANOPY calibration dataset contains small but statistically significant differences between growth and mortality rates for

trees in even- vs. uneven-aged stands for some species. Given the more prominent role that even-aged development plays in the current study, growth and mortality equations were both augmented with categorical variables and interaction terms for even-aged structure. As each new recruit is added, stand developmental stage is evaluated on a 0.25 ha patch centered on the new recruit. If the developmental stage of the patch is classified as 'sapling' (less than 10 m^2/ha of basal area in trees > 10 cm dbh), even-aged growth and mortality rates were used for individual recruits for a period of 200 years after establishment (corresponding to the mean age at time of death for canopy trees; Lorimer et al. 2001). Uneven-aged growth and mortality equations were used for all trees that did not originate in a sapling patch or were more than 200 years old. Mortality equations were also augmented with categorical variables and interaction terms to allow different mortality rates among habitat types.

1.2.4 Simulation design

Development following a complete clearcut (no residual trees > 2 cm dbh) was simulated at annual time-steps for 1,000 years on half-hectare plots, with 20 replicates performed for each simulation experiment. In year 1 of the simulations, CANOPY's recruitment mechanism begins adding new 2-6 cm dbh saplings, with their density determined by field-calibrated regressions. Trials were performed on all three habitat types simulated by CANOPY, both under background mortality alone and under the historic natural disturbance regime (1850-1980). Simulation results presented here focus on the ATD habitat type because the majority of the field data are on plots of that type.

Sensitivity analyses were also performed to evaluate the effect of changes in CANOPY v.3 described above: i.e., incorporating age structure variables into the growth and mortality equations, and changes to CANOPY's recruitment mechanism.

Species composition in the first 20 years of simulation was set to match common observations of the composition of young stands after clearcutting in the region on these habitat

types (Buttrick, 1923; Eyre and Zillgitt, 1953; Kotar et al., 2002). For AOCa habitat, initial sapling composition was 60% sugar maple, 20% basswood (*Tilia americana*), 15% white ash (*Fraxinus americana*), and 5% yellow birch. Initial composition on ATD was 80% sugar maple, 5% basswood, 5% yellow birch, 5% red maple (*Acer rubrum*), and 3% white ash. For ATM, initial composition was 40% sugar maple, 30% yellow birch, 15% red maple, and 5% basswood and white ash. After the first 20 years, CANOPY's normal mechanism to determine species composition based on the composition and density of the overstory was employed.

Initial simulations demonstrated slow but steady increases in hemlock dominance over the 1,000 year simulation under background mortality on ATD and ATM habitat types (see also Woods, 2000), which obscured age-related biomass trends. However, simulations under the historic disturbance regime tended to remain maple dominated. In order to compare these two disturbance regimes and also to evaluate age-related biomass trends separately from changes in biomass related to changing species domiance, most simulations for this study included only mixed hardwood species; hemlock-dominated stands were analyzed separately. To clarify the relative influence of sugar maple and the less abundant hardwoods on biomass trends, some simulations also included only sugar maple.

1.2.5 Data analysis

Whole-tree aboveground biomass (including branches and leaves) was estimated from stem diameter using published biomass equations. Equations were selected from studies in the northern hardwood region, with preference given to locations with species composition similar to the study areas, and data sets that included large trees ($\geq$50 cm dbh). Sources of biomass equations were: sugar maple and yellow birch (Young et al., 1980), red maple (Crow and Erdmann, 1983), white ash (Ker, 1980), ironwood and hemlock (Monteith, 1979), and basswood (Perala and Alban, 1994).

Analysis of biomass increment partitioned the stand-level response into the following components, following definitions and size thresholds from Erdmann and Oberg (1973) and Crow et al. (1981):

Ingrowth Total biomass of live trees with dbh < 11.7 cm at the beginning of a decade that grew larger than 11.7 cm by the end of the decade.

Survivor growth Total biomass increment of trees ≥ 11.7 cm that were alive at the beginning and end of a decade.

Mortality Total biomass of trees with dbh ≥ 11.7 cm alive at the beginning of a decade that died by the end of the decade, measured at the beginning of the decade. For net growth calculations, no growth was assumed on trees that died during the interval. However, this simplifying assumption is restricted to the net growth analysis, and therefore does not affect estimates of total biomass which were tabulated at annual time-steps.

Net growth The overall change in biomass by decade, equivalent to ingrowth + survivor growth - mortality.

Biomass growth was further partitioned into cohorts. For simulations, the initial cohort was defined to be all trees that recruited in the first 20 years after the clearcut and were larger than 25 cm dbh when they died. Individual trees in the initial cohort were tracked throughout the simulations and mean ages at time of death were computed. For field data from the permanent plot remeasurements, the initial overstory was defined to be all trees larger than 25 cm dbh at first measurement (1981-1982).

Structure of simulated and field stands was also classified using a revision of the stand stage definitions from Frelich and Lorimer (1991a). Stages were sequentially arranged in order of increasing modal diameter and degree of understory development. Typically, this results in a progression in diameter distribution shape from skewed unimodal to unimodal to

descending monotonic (Appendix A). Actual distinctions between stand stages are based on distribution of basal area among broad size categories (Appendix B, Table 1). For example, old-growth stands must have at least 45% of the stand basal area in trees >46 cm dbh, corresponding to an average tree age of ∼150 years (Lorimer and Frelich, 1989). Simulations of post-clearcut stands for 1,000 years as well as available permanent plot data, supported the interpretation that the stand structural stages follow the expected temporal sequence (Appendix A).

1.3 Results

1.3.1 Aboveground biomass trends among stand structural stages

When mean aboveground live-tree biomass (hereafter "biomass" except as noted) was summarized by stand stage for the full 70 plot field data set, a higher level of biomass was observed in the transition stages than in steady state (Fig. 1). Mid-transition had on average 61 Mg/ha more biomass than steady state (t-test: p=0.004; t=3.192, df=22.54; 95% CI for the difference: 21.7-102 Mg/ha), implying a decline of 19% between the two stages.

When the eight permanent plots were arranged in order of increasing modal diameter and degree of understory development, an increasing trend in biomass was observed through the early stages of old growth (Fig. 2). The pattern in steady state then becomes somewhat ambiguous, with two plots showing a lower level of biomass than early stages of old growth and a third showing a higher level. For six of the eight permanent plots, estimated biomass fell within the range of variation of 20 replicated CANOPY simulations for stands of the same developmental stage.

CANOPY simulations of mean biomass for each stage correspond well with field-measured averages in most cases, with overlapping 95% confidence intervals for all structural stages except mature-sapling mosaic. Formal two-sample t-tests showed no significant difference between CANOPY predictions under the historic natural disturbance regime and field mea-

surements for the mature stage and the last three stages of old growth ($0.12 \leq p \leq 0.71$). For the pole stage, CANOPY biomass predictions were lower than the field average (t-test: p=0.017, t=2.65, df=16.79; 95% CI for the difference: -72 to -7 Mg/ha) when the historic natural disturbance regime was imposed, but simulations under background mortality alone did not differ from the field averages (t-test: p=0.65,t=-0.466, df=10.53). For the early transition stage, t-tests indicated that both sets of simulations differed from field averages (p=0.019 for the historic disturbance regime and p=0.013 for background mortality).

1.3.2 Trends in simulated biomass over time

Under background mortality, CANOPY simulations of biomass development on ATD habitat showed a peak followed by a decline. The average live AGB across the 20 replicates declined by 16%, from a peak of 350 Mg/ha at stand age 180 years to a low of 294 Mg/ha 320 years after clearcutting (Fig. 3 a). Subsequently, live AGB mostly recovered to a stable value of approximately 320 Mg/ha 450 years after clearcutting. Among individual replicates, peak biomass ranged from 350 Mg/ha to 397 Mg/ha and occurred between stand age 180 and 230. The subsequent reduction averaged 28% on the 17 of 20 replicates that showed a decline.

The predicted biomass peak/decline behavior persisted when alternate versions of CANOPY were used for sensitivity analyses. When CANOPY's default recruitment mechanism was employed, lacking the heuristic to mimic sparse understory development in mature and old even-aged stands, live AGB of the 20 replicates declined by 9%, from 300 Mg/ha 200 years after clearcut to 272 Mg/ha 370 years after clearcut. When growth and mortality equations were used that did not include categorical variables to distinguish even vs. uneven-aged structure, live AGB declined by 22%, from 320 Mg/ha 180 years after clearcut to 249 Mg/ha 310 years after clearcut. Finally, when neither the sparse regeneration heuristic nor the age-structure sensitive growth and mortality equations were used, live AGB declined by 13%, from 311 Mg/ha 170 years after clearcut to 272 Mg/ha 360 years after clearcut.

Simulations that incorporated the historic natural disturbance regime in mixed hardwood forests on ATD habitat resulted in lower peak biomass values (300 Mg/ha at age 200) compared to background mortality. But there was still a peak followed by a decline of 23% to 232 Mg/ha 320 years after clearcut (Fig. 3 c). Ten of the twenty individual replicates still displayed a decline in live AGB, with an average decline of 46% on replicates that had one. Disturbance in the first 300 years on many replicates usually removed $< 30\%$ of the aggregate crown area and often did not disrupt the overall peak/decline pattern (Fig. 4).

Simulated aboveground biomass of dead trees on ATD habitat peaked 20-40 years after the peak in live aboveground biomass under both background mortality (Fig. 3 a) and the historic natural disturbance regime (Fig. 3 c). After the peak, both dead and live biomass declined together, rather than demonstrating a compensating trend. The trend in total aboveground biomass was therefore similar to the trend in live AGB alone.

In sensitivity analyses conducted to evaluate the influence of species and habitat type under background mortality, biomass peaks were generally observed except on pure maple stands on ATD habitat (Table 1). However, stands on the more productive AOCa habitat reached a lower biomass peak at earlier ages compared to ATD. Biomass then recovered after the initial decline to approximately peak levels. On the less productive habitat types, biomass either never increased after the initial decline (ATM) or the recovery was only partial (ATD). Differences in the magnitude and timing of the biomass peaks were influenced by the fertility gradient, habitat-influenced differences in species composition, and differences in wood density and growth rates among species. For example, basswood and ash grow faster than sugar maple but have lower wood density (Miles and Smith, 2009).

1.3.3 Demographic drivers of biomass trends

Changes in size distributions and mortality patterns over time

When the 70 field plots were arranged by structural stand stage, diameter distributions progressed from a skewed unimodal form in the pole stage to more symmetric unimodal in the mature and early transition. The mid-transition stage typically had compound distributions, whereas the late transition and steady state stages typically had irregular or relatively smooth descending monotonic curves (Fig. 5 a-f). The CANOPY simulations of diameter distributions produced similar curve shapes and confirmed that these occurred in the temporal sequence expected from the field data on structural stages (Fig. 5 g-l).

Peak biomass and maximum number of large trees occurred in the mid-transition stage for both field and simulation data (Figs. 2,3), with a trough and subsequent partial recovery in the steady state. At the time of the biomass peak, the diameter distributions were compound in form in both field and simulation data (Fig. 5 c,i). Simulated mean age at time of death for canopy trees coincided with peak biomass under both background mortality and the historic natural disturbance regime, and also coincided with the peak number of large trees in background mortality simulations (t-tests; $0.20 \leq p \leq 0.91$). However, the peak number of large trees under the historic natural disturbance regime occurred about 20 years after the mean age at time of death (t-test; $p<0.01$). The addition of exogenous natural disturbance to background mortality also reduced the mean age at time of death from 209 to 180 years.

Aboveground live biomass had a nearly linear relationship with the number of large trees (Fig. 6 a) in both the field data and simulations. In the simulations, however, the relationship was different for even-aged stands younger than 200 years than for older uneven-aged stands with a complex disturbance history. When simulated biomass was plotted against the mean age of the largest 5% of trees, in order to roughly emulate a chronosequence analysis of multi-aged stands, the observed relationship was essentially asymptotic (Fig. 6 b). The mean age of the largest 5% of trees reached a maximum (Fig. 6 b) at the same time as peak biomass

(Fig. 3 c).

On a shorter time scale, some of the irregularities present in the biomass trends on the permanent plots (Fig. 2) are partly explained by disturbance-induced changes in species composition. The effect of the severe 1988 drought during the first measurement period was apparent on the net growth of yellow birch, which was negative on 6 of the 7 plots where it was present. High yellow birch mortality at that time was responsible for the temporary drop in biomass in an early transition stand (plot 6 in Table 3) and fully or partly accounts for declines of variable length in two steady-state stands (plots 2 and 3 in Table 3). A net decline in sugar maple biomass was also a contributing factor in the temporary biomass declines on two of the plots (plots 3 and 6). The unexpectedly level biomass trend for the early transition stand was due to a combination of high hemlock and maple mortality and nearly zero ingrowth for all species (plot 4 in Table 3) resulting from many decades of high deer browsing (Frelich and Lorimer, 1985). Yet, overall biomass net growth of hemlock was positive on 7 of 8 plots despite a paucity of hemlock <10 cm dbh.

Production rates of initial and secondary cohorts

On the pole permanent field plot, both initial and secondary cohorts were aggrading over the 30-year period (Table 2). On the mature and early transition plots, biomass increased overall with slight losses in the understory. Of the five other old-growth permanent plots, three had declining overstory biomass. In two of three cases, the negative net growth of the overstory cohort was compensated by the growth of the secondary cohorts, resulting in a net increase in biomass. However, none of the permanent plots occurred in the late transition stage, where an overall decline in biomass would be most likely under the Bormann-Likens hypothesis.

For the three permanent plots classified as steady-state based on structural attributes, cumulative 30-year net growth averaged -3.9 Mg/ha; i.e., essentially zero (Annualized range: -1.1 to 0.4 Mg/ha/yr). All three plots had positive net growth of the secondary cohorts, but

two plots had negative net growth in the overstory cohort. On one of these, understory net growth was sufficient to compensate for the losses, but not on the other plot in which the overstory cohort decreased in biomass by 60 Mg/ha (Table 2).

Trends in the field data were clarified with simulations that included the late transition stage. Simulated average biomass net growth of secondary cohorts under background mortality did not generally compensate for the loss of the initial cohort during the transition stages (Fig. 7). The initial cohort reached zero net growth at age ≈ 200, roughly at the mean age at time of death for trees in that cohort (209 years). From ages 200-310, while overall biomass was declining, average net growth was negative (t-test: p=0.014, t=-2.81, df=14; 95% CI: -0.60 to -0.08 Mg/ha/yr). After age 310, biomass net growth hovered about zero, and the average was not distinct from zero (t-test: p=0.40, t=0.85, df=67; 95% CI: -0.05 to 0.13 Mg/ha/yr).

To more directly compare simulation results with the 30-yr permanent plot records, biomass net growth was computed for 20 replicates under background mortality at 30 year intervals from simulation years 200-290, the span of time during which a decline might be expected (total of 60 intervals). In 78% of 60 cases, overall biomass declined over the 30-year interval. In all but one of those, the secondary cohort net growth was positive but was not sufficient to balance the loss of the initial cohort. In 22% of the 60 cases (n=13), overall biomass increased during a 30-year interval. In 7 of these, both initial and secondary cohorts were aggrading, and in the 6 others, the positive growth of the secondary cohort more than balanced losses in the initial cohort. In one remaining case of the 60 total intervals, both cohorts declined in biomass.

When the historic natural disturbance regime was incorporated into the simulations, the loss of the initial cohort was somewhat more irregular, but was still not balanced on average by the growth of the secondary cohorts (Fig. 8). From ages 200-300, while the overall biomass was declining, average net growth was negative with marginal statistical significance (t-test:

p=0.052, t=-2.16, df=12; 95% CI: -0.93 to 0.005 Mg/ha/yr). After age 300, average biomass net growth was not distinct from zero (t-test: p=0.74, t=0.33, df=67; 95% CI: -0.14 to 0.20 Mg/ha/yr).

To analyze the natural disturbance simulations from Fig. 8 in a manner comparable to the permanent plot field data, net growth was again computed for the 20 individual replicates at 30 year intervals from years 200-290. Since the 30-yr permanent plot record was free of moderate or severe disturbance, we sought to determine if secondary cohorts can balance overstory losses when the late transition stage occurs during a 'quiescent' period. Disturbances prior to year 200 were included in the analysis, but intervals from 200-290 that contained a disturbance were excluded along with all subsequent intervals for that replicate (n=39 intervals remaining out of 60 total). Biomass declines were more common than increases, with 59% of 39 replicates having a declining initial cohort that was not balanced by the aggrading secondary cohorts. Biomass increased in 41% of the 39 cases; in 7 of these cases, both initial and secondary cohorts were aggrading, and in 9 cases, the loss of the initial cohort was balanced or exceeded by the growth of the secondary cohorts.

1.4 Discussion

1.4.1 Evaluating the Bormann-Likens hypothesis

While the various lines of evidence from this study are not completely supportive of one hypothesis, the preponderance of evidence supports the basic outline of the Bormann and Likens (1979) hypothesis. In both field data and simulations, peak biomass occurred toward the end of the lifespan of the initial overstory cohort, followed by declining biomass and partial recovery in a steady state (Fig. 1,3) with zero net growth (Figs 7,8, Table 2). The inclusion of mild and moderate disturbances did not produce an average asymptotic trend but continued to show a peak-decline pattern, although natural disturbances lowered the mean biomass levels (Fig. 3 c), as suggested by Bragg et al. (2004) and Keeton et al. (2011).

Some disturbances prevented or delayed attainment of peak biomass, but most mild or moderate disturbances were not sufficiently frequent or severe to disrupt the peak-decline pattern (Fig. 4).

The magnitude of the simulated peak-decline pattern however, did vary under different conditions, sometimes only showing a 3-10% decline under certain combinations of species composition, habitat type, and model specifications. These declines might be too subtle to detect in 'noisy' chronosequence data. Even under conditions where the mean decline was 15-25%, a decline was not universal among all replicates due to stochastic variation in growth and mortality, at least at the 0.5 ha scale. For example, in simulations under background mortality, 22% of the measurement intervals after age 200 had level or increasing biomass, and in 10% of all cases, this increase occurred because overstory attrition was balanced or exceeded by growth of secondary cohorts.

In both field and simulation data, biomass trends of stands and individual cohorts also showed short-term fluctuations that could render the trend ambiguous over short time periods (e.g., Figs. 1,3, Table 2). Sometimes, limitations in the field data also augmented the level of ambiguity. For example, permanent plot field data show that secondary cohorts can in many cases compensate for overstory losses. But late transition stands, when a large net decline is most likely to be observed, were not present in the permanent plot data set. Likewise, the ambiguous 30-yr trend among transition and steady-state permanent plots in Fig. 2 appears to be largely an artifact of site variation. Specifically, the steady-state plot with high biomass had the highest site quality of the 8 permanent plots. Its biomass of 366 Mg/ha was the highest of any of the 18 steady state stands in the larger data set, well above the mean level of 264 Mg/ha, while the biomass of the other two steady-state permanent plots was close to the mean. Given these circumstances, even the trend in Fig. 2 appears to be more indicative of a peak-decline pattern.

The peak-decline pattern is consistent with the underlying demographic processes of the

tree species. The peak biomass in simulations occurred at about age 200, which corresponds to the average longevity of canopy trees in both field data (Lorimer et al., 2001) and simulations. This stage of stand development also had the greatest density of large trees (Figs. 3,5), which is highly correlated with aboveground biomass (Keeton et al., 2011). In both field data and simulations, the biomass peak also occurred at the time of transition in diameter distributions from a predominantly unimodal to a descending monotonic form (Fig. 5). Simulations indicate that after age 200, mortality exceeded growth on average for the first time, reflecting the breakup of the even-aged cohort (Fig. 7 b).

The peak-decline pattern of the Bormann-Likens hypothesis is also consistent with what is known about structural differences between even-aged and all-aged stands. Periodic mortality from senescence of large trees and resulting gap formation in all-aged forests leads to a complex structural mosaic that includes numerous sapling and pole groups exposed to direct skylight. In the present data set, this is evident in the size distributions of gap saplings and canopy trees (dark bars in Fig. 5 f) and in crown maps that clearly show clusters of sapling and pole trees in old canopy gaps in various stages of regrowth (Fig. 6b in Lorimer, 1985). Biomass per unit area is lower for sapling and pole stands relative to mature and old growth (e.g., Figs. 1,2 in this paper, cf. also Lichstein et al., 2009). Thus, the displacement of some mature and large trees by sapling and pole groups will likely lower the level of aboveground biomass. While it is conceivable that the multi-layered structure of all-aged forests could augment total biomass and prevent a decline, both field (Fig. 1) and simulation data (Fig. 3) generally indicated about 20% less biomass in these all-aged northern hardwood forests compared to earlier stages of old growth.

Simulated peak and decline was robust to a number of modeling assumptions, with similar conclusions based on both sets of growth and mortality equations, both recruitment mechanisms, and both background mortality and natural disturbance regime, and also when assessed at larger spatial scales (1, 2, 4 ha). Simulations are also able to factor out site-to site

variability and represent behavior of identical sites allowed to develop under stable dynamics. However, a limitation of the model is that CANOPY can't predict the timing of stochastic disturbance events like the 1988 drought, which accounts for some of the discrepancies between observed and predicted biomass in Fig. 2.

1.4.2 How general are peak/decline dynamics?

The question of whether a broad generalizable trend can be inferred in biomass development of northern hardwoods and other late-successional temperate forests depends partly on the relative strengths of field and simulation evidence. Several models that differ substantially in design from CANOPY have also predicted a peak in biomass followed by a decline, including JABOWA (Whittaker et al., 1974), FORET (Shugart, 1984), FORCLIM (Bugmann, 1996), SORTIE (Pacala et al., 1996), NORTHWDS (Bragg et al., 2004), and CAIN (Vanderwel et al., 2013). However, the magnitude and timing of the predicted peaks differ among models. For example, simulations with the NORTHWDS model predicted biomass peaks of ≈340 Mg/ha in the absence of windthrow and ≈ 290 Mg/ha under a natural disturbance regime (Bragg et al., 2004), with peaks in both cases occurring around stand age 70. In contrast, simulation with the CAIN model Vanderwel et al. (2013) predicted peak biomass of ≈500 Mg/ha at approximately age 200, declining to ≈ 300 Mg/ha. And despite its simplicity and minimal calibration data, JABOWA gave predictions of biomass similar to those of CANOPY under background mortality. Whittaker et al. (1974) reported simulated peak biomass of 390 Mg/ha at age 200, declining to a long-term average of ≈300 Mg/ha by age 400. In the present study, CANOPY's predictions of the magnitude of peak biomass are in agreement with observations from field surveys of old-growth stands in the region (Fig. 1; likewise Mroz et al. 1985; Gries 1995). Also, the onset of the decline coincides with both predictions of average canopy-tree longevity generated by the field-calibrated mortality equations as well as field evidence on longevity based on age determinations of recently dead

trees (Lorimer et al., 2001).

At Hubbard Brook, recent permanent plot remeasurements show biomass leveling off at ≈245 Mg/ha (Siccama et al., 2007), rather than the expected increase to ≈350 Mg/ha (Whittaker et al., 1974). The asymptotic trend was believed to be influenced by anthropogenic factors such as the exotic beech bark disease and acid rain. In the present study, there has been little consistent evidence of growth stagnation in recent years, especially for the mature and transition stages, and most stands seem to be trending toward a maximum biomass of about 300-350 Mg/ha (Fig. 2). A possible exception is the pole-mature stand at Summit Peak (Plot 5 in Table 2). In the first two measurement periods, biomass net growth in this stand averaged 2.09 Mg/ha/yr, the highest value observed on any of the permanent plots, and slightly lower than the 2.4 Mg/ha/yr reported at Hubbard Brook (van Doorn et al., 2011). In the last 7-yr measurement interval, however, net growth averaged only 0.51 Mg/ha/yr, and its biomass may be trending toward a peak value of only 250 Mg/ha. This could merely indicate a short-term lull in net growth, or alternatively it may be a valid longer-term trend reflecting the stand's location on an upper south-facing slope.

Other field studies in northern hardwoods aside from Hubbard Brook have reported asymptotic biomass trends based on chronosequence analysis in which a time trend is inferred from the mean age of larger trees in the stand. However, it is unclear if the assumption of approximate equality between the mean age of the largest trees and time since major disturbance (i.e. stand age) is consistently reliable. For stands younger than the mean lifespan of the dominant species and with no old residual trees as a result of heavy partial disturbance, this assumption works quite well (Fig. 6 c). As the initial cohort breaks up and the stand begins to have a multi-aged structure, the relationship becomes increasingly less reliable. Maple-dominated stands 600 years after stand initiation had a simulated mean age of the largest 5% of trees closer to 300 years. When a trend in biomass development is inferred using this noisy approximation of age, a largely asymptotic trend results (Fig. 6 b) even when

the actual temporal trend is a peak followed by a decline (Fig. 3 c). In the current study, repeated measurements on identical sites were used rather than a space-for-time substitution across a mixture of sites. Another concern regarding chronosequence analysis is that many forest inventory data sets, such as the US Forest Service FIA data, are comprised largely of second-growth stands with few trees older than 150 years. At least for northern hardwood systems, these ages would not be sufficient to observe the biomass peak (ca. 180-200 years) or the subsequent decline (ca 250-300 years).

Some authors have hypothesized that disturbances may interfere with biomass development, forestalling any peak that might theoretically exist. For example, Odum (1969) suggested that some ecosystems may experience a dynamic in which periodic disturbances maintain an ecosystem at an intermediate developmental stage. Based on their theoretical REGIME model, however, Weng et al. (2012) computed that disturbances removing all of the trees at $>$ 120 year intervals would result in a $<$20% reduction in ecosystem carbon content. The disturbance regime used by CANOPY has recurrence intervals longer than 120 years for all severity classes greater than 10% canopy removal (Frelich and Lorimer, 1991a; Zhang et al., 1999; Schulte and Mladenoff, 2005). Many temperate forests with humid climates in Europe and Asia have similar disturbance regimes in which severe disturbance is rare (Masaki et al., 1999; Schelhaas et al., 2003; Splechtna et al., 2005; Fraver et al., 2008). This raises the possibility that late successional forests in humid climates may have similar peak-decline dynamics provided that recovery time is shorter than the interval between severe disturbances (*sensu* Turner et al., 1993).

While the field and simulation evidence in this study support the hypothesis of positive net growth of aboveground tree biomass well into the old-growth stages, the question of whether old-growth natural areas continue to accumulate carbon also depends partly on the spatial scale. The large landscapes of northern hardwoods in the present study include a mosaic of stands in all stages of development, but with late transition and steady-state

stands comprising about 40% of the total area (Appendix A). In simulations using 20 independent replicates subjected to the historic natural disturbance regime, pooled estimates of net growth in aboveground tree biomass hovered about zero after simulation year 200 (Fig. 8a). This implies that large wilderness areas with a high proportion of old-growth northern hardwoods may have zero net growth, even though some young and mature stands with positive net growth are present. Several lines of field and simulation evidence, in fact, suggest that the aggregate study areas meet criteria of an equilibrium landscape, in which net growth of zero would also be expected (Chapter 2). It is possible that carbon may still accumulate in the soil in old-growth stands, an issue that is still poorly known (Gleixner et al., 2009; Price et al., 2012). However, Tang et al. (2008, 2009) reported field-measured soil carbon accumulation in old-growth northern hardwoods of <0.04 Mg/ha/yr, supported also by simulations using Biome-BGC showing approximately zero NEP for old-growth forests (Peckham and Gower, 2011). Regardless of the specific soil carbon accumulation rate, the aboveground biomass and thus carbon storage in old-growth forests nevertheless remains substantial, considerably higher than in younger stands (e.g., Fig. 1; see also Lichstein et al., 2009; Keeton et al., 2011; Carroll et al., 2012) and should be explicitly considered in carbon management plans.

Literature Cited

Ashton, M. S., M. L. Tyrrell, D. Spalding, and B. Gentry (2012). *Managing forest carbon in a changing climate*. New York: Springer.

Bormann, F. H. and G. E. Likens (1979). *Pattern and process in a forested ecosystem*. New York: Springer-Verlag.

Botkin, D. B., J. F. Janak, and J. R. Wallis (1972). Some ecological consequences of a computer model of forest growth. *Journal of Ecology 60*(3), 849–872.

Bragg, D. C., D. W. Roberts, and T. R. Crow (2004). A hierarchical approach for simulating northern forest dynamics. *Ecological Modelling 173*, 31–94.

Bugmann, H. K. M. (1996). A simplified forest model to study species composition along climate gradients. *Ecology 77*, 2055–2074.

Busing, R. T. and D. Mailly (2004). Advances in spatial, individual-based modeling of forest dynamics. *Journal of Vegetation Science 15*, 831–842.

Buttrick, P. L. (1923). Second growth hardwood forests in Michigan. Special Bulletin 23, Agricultural Experiment Station, Michigan Agricultural College, East Lansing, MI.

Canham, C. D., M. J. Papaik, and E. F. Latty (2001). Interspecific variation in susceptibility to windthrow as a function of tree size and storm severity for northern temperate tree species. *Canadian Journal of Forest Research 31*, 1–10.

Carroll, M., B. Milakovsky, A. Finkral, A. Evans, and M. S. Ashton (2012). Managing carbon sequestration and storage in temperate and boreal forests. In M. S. Ashton, M. L. Tyrrell, D. Spalding, and B. Gentry (Eds.), *Managing forest carbon in a changing climate*, pp. 205–226. New York: Springer.

Christy, B. H. (1929). *The book of Huron Mountain.* Chicago, IL, USA: Huron Mountain Club.

Coffman, M. S. (1984). *Field guide: Habitat classification system for upper peninsula of Michigan and northeast Wisconsin.* Houghton, MI, USA: CROFS, School of Forestry and Wood Products, Michigan Technological University.

Crow, T. R. and G. G. Erdmann (1983). Weight and volume equations and tables for red maple in the Lake States. Research Paper NC-242, USDA Forest Service.

Crow, T. R., C. H. Tubbs, R. D. Jacobs, and R. R. Oberg (1981). Stocking and structure for maximum growth in sugar maple selection stands. Research Paper NC-199, USDA Forest Service.

D'Amato, A. W. and D. A. Orwig (2008). Stand and landscape-level disturbance dynamics in old-growth forests in western Massachusetts. *Ecological Monographs 78*(4), 507–522.

Erdmann, G. G. and R. R. Oberg (1973). Fifteen-year results from six cutting methods in second-growth northern hardwoods. Research Paper NC-100, USDA Forest Service.

Eyre, F. H. and W. M. Zillgitt (1953). Partial cuttings in northern hardwoods of the Lake States: Twenty-year experimental results. Technical Bulletin 1076, USDA Forest Service.

Fassnacht, K. S. and S. T. Gower (1997). Interrelationships among the edaphic and stand characteristics, leaf area index, and aboveground net primary production of upland forest ecosystems in north central Wisconsin. *Canadian Journal of Forest Research 27*, 1058–1607.

Fraver, S., B. G. Honsson, M. Jönsson, and P.-A. Esseen (2008). Demographics and disturbance history of a boreal old-growth *Picea abies* forest. *Journal of Vegetation Science 19*, 789–798.

Fraver, S., A. S. White, and R. S. Seymour (2009). Natural disturbance in an old-growth landscape of northern Maine, USA. *Journal of Ecology 97*(2), 289–298.

Frelich, L. E. (1986). *Natural disturbance frequencies in the hemlock-hardwood forests of the upper Great Lakes region.* Ph. D. thesis, University of Wisconsin – Madison, Madison, WI, USA.

Frelich, L. E. and C. G. Lorimer (1985). Current and predicted long-term effects of deer browsing in hemlock forests in Michigan, USA. *Biological Conservation 34*(2), 99–120.

Frelich, L. E. and C. G. Lorimer (1991a). Natural disturbance regimes in hemlock-hardwood forests of the upper Great Lakes region. *Ecological Monographs 61*(2), 145–164.

Frelich, L. E. and C. G. Lorimer (1991b). A simulation of landscape-level stand dynamics in the northern hardwood region. *Journal of Ecology 79*, 145–164.

Gleixner, G., C. Tefs, A. Jordan, M. Hammer, C. Wirth, A. Nueske, A. Telz, U. E. Schmidt, and S. Glatzel (2009). Soil carbon accumulation in old-growth forests. In C. Wirth, G. Gleixner, and M. Heimann (Eds.), *Old growth forests: Function, fate, and value*, pp. 231–266. Berlin: Springer-Verlag.

Goodburn, J. M. and C. G. Lorimer (1999). Population structure in old-growth and managed northern hardwoods: An examination of the balanced diameter distribution concept. *Forest Ecology and Management 118*, 11–29.

Gries, J. F. (1995). Biomass and net primary production for a northern hardwood stand developmental sequence in the upper peninsula, Michigan. Master's thesis, University of Wisconsin – Madison, Madison, WI, USA.

Halpin, C. R. (2009). Simulated long-term effects of group selection on production, efficiency, composition, and structure in northern hardwood and hemlock-hardwood forests. Master's thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J. (2009). *Emulating natural disturbance dynamics in northern hardwood forests: Long-term effects on species composition, forest structure, and yield.* Ph. D. thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J. and C. G. Lorimer (2007). Forest structure and light regimes following moderate windstorms: Implications for multi-cohort management. *Ecological Applications 17*, 1325–1340.

Hanson, J. J., C. G. Lorimer, and C. R. Halpin (2011). Predicting long-term sapling dynamics and canopy recruitment in northern hardwood forests. *Canadian Journal of Forest Research 41*, 903–919.

Hanson, J. J., C. G. Lorimer, C. R. Halpin, and B. J. Palik (2012). Ecological forestry in an uneven-aged, late-successional forest: Simulated effects of contrasting treatments on structure and yield. *Forest Ecology and Management 270*, 94–107.

Heimann, M. (2009). *Old-growth forests: Function, fate, and value*. New York: Springer.

Janowiak, M. K., L. M. Nagel, and C. R. Webster (2008). Spatial scale and stand structure in northern hardwood forests: Implications for quantifying diameter distributions. *Forest Science 54*(5), 497–506.

Keeton, W. S., A. A. Whiteman, G. C. McGee, and C. L. Goodale (2011). Late-successional biomass development in northern hardwood-conifer forests of the northeastern United States. *Forest Science 57*(6), 489–505.

Ker, M. F. (1980). Tree biomass equations for ten major species in Cumberland County, Nova Scotia. Information Report MX-108, Maritime Forest Research Centre, Canada.

Kotar, J., T. L. Burger, and J. A. Kovach (2002). *A guide to forest communities and habitat types of northern Wisconsin*. Madison, WI: Department of Forest Ecology and Management, University of Wisconsin – Madison.

Lichstein, J. W., C. Wirth, H. S. Horn, and S. W. Pacala (2009). Biomass chronosequences of United States forests: Implications for carbon storage and forest management. In C. Wirth, G. Gleixner, and M. Heimann (Eds.), *Old growth forests: Function, fate, and value*, pp. 301–341. Berlin: Springer-Verlag.

Lorimer, C. G. (1985). Methodological considerations in the analysis of forest disturbance history. *Canadian Journal of Forest Research 15*, 200–213.

Lorimer, C. G., S. E. Dahir, and E. V. Nordheim (2001). Tree mortality rates and longevity in mature and old-growth hemlock-hardwood forests. *Journal of Ecology 89*(6), 960–971.

Lorimer, C. G. and L. E. Frelich (1989). A methodology for estimating canopy disturbance frequency and intensity in dense temperate forests. *Canadian Journal of Forest Research 19*(5), 651–663.

Luyssaert, S., E. Schulze, A. Börner, A. Knohl, D. Hessenmöller, B. E. Law, P. Ciais, and J. Grace (2008). Old-growth forests as global carbon sinks. *Nature 455*, 213–215.

Masaki, T., H. Tanaka, H. Tanouchi, T. Sakai, and T. Nakashizuka (1999). Structure, dynamics and disturbance regime of temperate broad-leaved forests in Japan. *Journal of Vegetation Science 10*, 805–814.

Miles, P. D. and W. B. Smith (2009). Specific gravity and other properties of wood and bark for 156 tree species found in North America. Research Note NRS-38, USDA Forest Service.

Monteith, D. B. (1979). Whole tree weight tables for New York. AFRI Research Report 40, Applied Forestry Research Institute, Syracuse, NY.

Mroz, G. D., M. R. Gale, M. F. Jurgensen, D. J. Frederick, and A. C. III (1985). Composition, structure, and aboveground biomass of two old-growth northern hardwood stands in upper Michigan. *Canadian Journal of Forest Research 15*(1), 78–82.

Niese, J. N. and T. F. Strong (1992). Economic and tree diversity trade-offs in managed northern hardwoods. *Canadian Journal of Forest Research 22*(11), 1807–1813.

Odum, E. P. (1969). The strategy of ecosystem development: An understanding of ecological succession provides a basis for resolving man's conflict with nature. *Science 164*, 262–270.

Pacala, S. W., C. D. Canham, J. Saponara, J. A. S. Jr, R. K. Kobe, and E. Ribbens (1996). Forest models defined by field measurements: Estimation, error analysis and dynamics. *Ecological Monographs 66*(1), 1–43.

Papaik, M. J. and C. D. Canham (2006). Species resistance and community response to wind disturbance regimes in northern temperate forests. *Journal of Ecology 94*, 1011–1026.

Peckham, S. D. and S. T. Gower (2011). Simulated long-term effects of harvest and biomass residue removal on soil carbon and nitrogen content and productivity for two Upper Great Lakes forest ecosystems. *GCB Bioenergy 3*(2), 135–147.

Perala, D. A. and D. H. Alban (1994). Allometric biomass estimators for aspen-dominated ecosystems in the upper Great Lakes. Research Paper NC-314, USDA Forest Service.

Price, S. P., M. A. Bradford, and M. S. Ashton (2012). Characterizing organic carbon stocks and flows in forest soils. In M. S. Ashton, M. L. Tyrrell, D. Spalding, and B. Gentry (Eds.), *Managing forest carbon in a changing climate*, pp. 7–30. New York: Springer.

Rafferty, M. and R. Sprague (2001). *Porcupine Mountains companion: Inside Michigan's largest state park*. White Pine, MI, USA: Neqauket Natural History Associates.

Schelhaas, M.-J., G.-J. Nabuurs, and A. Schuck (2003). Natural disturbances in the European forests in the 19th and 20th centuries. *Global Change Biology 9*(11), 1620–1633.

Schulte, L. A. and D. J. Mladenoff (2005). Severe wind and fire regimes in northern forests: Historical variability at the regional scale. *Ecology 86*(2), 431–445.

Shugart, H. H. (1984). *A theory of forest dynamics: The ecological implications of forest succession models*. New York: Springer-Verlag.

Siccama, T. G., T. J. Fahey, C. E. Johnson, T. W. Sherry, E. G. Denny, E. B. Girdler, G. E. Likens, and P. A. Schwarz (2007). Population and biomass dynamics of trees in a northern hardwood forest at Hubbard Brook. *Canadian Journal of Forest Research 37*, 737–749.

Splechtna, B. E., G. Gratzer, and B. A. Black (2005). Disturbance history of a European old-growth mixed-species forest – a spatial dendro-ecological analysis. *Journal of Vegetation Science 16*, 511–522.

Tang, J., P. V. Bolstad, and J. G. Martin (2009). Soil carbon fluxes and stocks in a Great Lakes forest chronosequence. *Global Change Biology 15*, 145–155.

Tang, J., P. V. Bolstand, A. R. Desai, J. G. Martin, B. D. Cook, K. J. Davis, and E. V. Carey (2008). Ecosystem respiration and its components in an old-growth forest in the Great Lakes region of the United States. *Agricultural and Forest Meteorology 148*, 171–185.

Turner, D. P., G. J. Koerper, M. E. Harmon, and J. J. Lee (1995). A carbon budget for forests of the conterminous United States. *Ecological Applications 5*(2), 421–436.

Turner, M. G., W. H. Romme, R. H. Gardner, R. V. O'Neill, and T. K. Kratz (1993). A revised concept of landscape equilibrium: Disturbance and stability on scaled landscapes. *Landscape Ecology 8*(3), 213–227.

van Doorn, N. S., J. J. Battles, T. J. Fahey, T. G. Siccama, and P. A. Schwarz (2011). Links between biomass and tree demography in a northern hardwood forest: A decade of stability and change in Hubbard Brook Valley, New Hampshire. *Canadian Journal of Forest Research 41*, 1369–1379.

Vanderwel, M. C., D. A. Coomes, and D. W. Purves (2013). Quantifying variation in forest disturbance, and its effects on aboveground biomass dynamics, across the eastern United States. *Global Change Biology 19*, 1504–1517.

Weng, E., Y. Lou, W. Wang, H. Wang, D. J. Hayes, A. D. McGuire, A. Hastings, and D. S. Schimel (2012). Ecosystem carbon storage capacity as affected by disturbance regimes: A general theoretical model. *Journal of Geophysical Research 117*, G03014.

Whittaker, R. H., F. H. Bormann, G. E. Likens, and T. G. Siccama (1974). The Hubbard Brook ecossytem study: Forest biomass and production. *Ecological Monographs 44*, 233–252.

Woods, K. D. (2000). Long-term change and spatial pattern in a late-successional hemlock-northern hardwood forest. *Journal of Ecology 80*, 267–282.

Young, H. E., J. H. Ribe, and K. Wainwright (1980). Weight tables for tree and shrub species in maine. Miscellaneous report 230, University of Maine, Life Sciences and Agriculture Experiment Station, Orono, ME.

Zhang, Q., K. S. Pregitzer, and D. D. Reed (1999). Catastrophic disturbance in the pre-settlement forests of the upper peninsula of Michigan. *Canadian Journal of Forest Research 29*(1), 106–114.

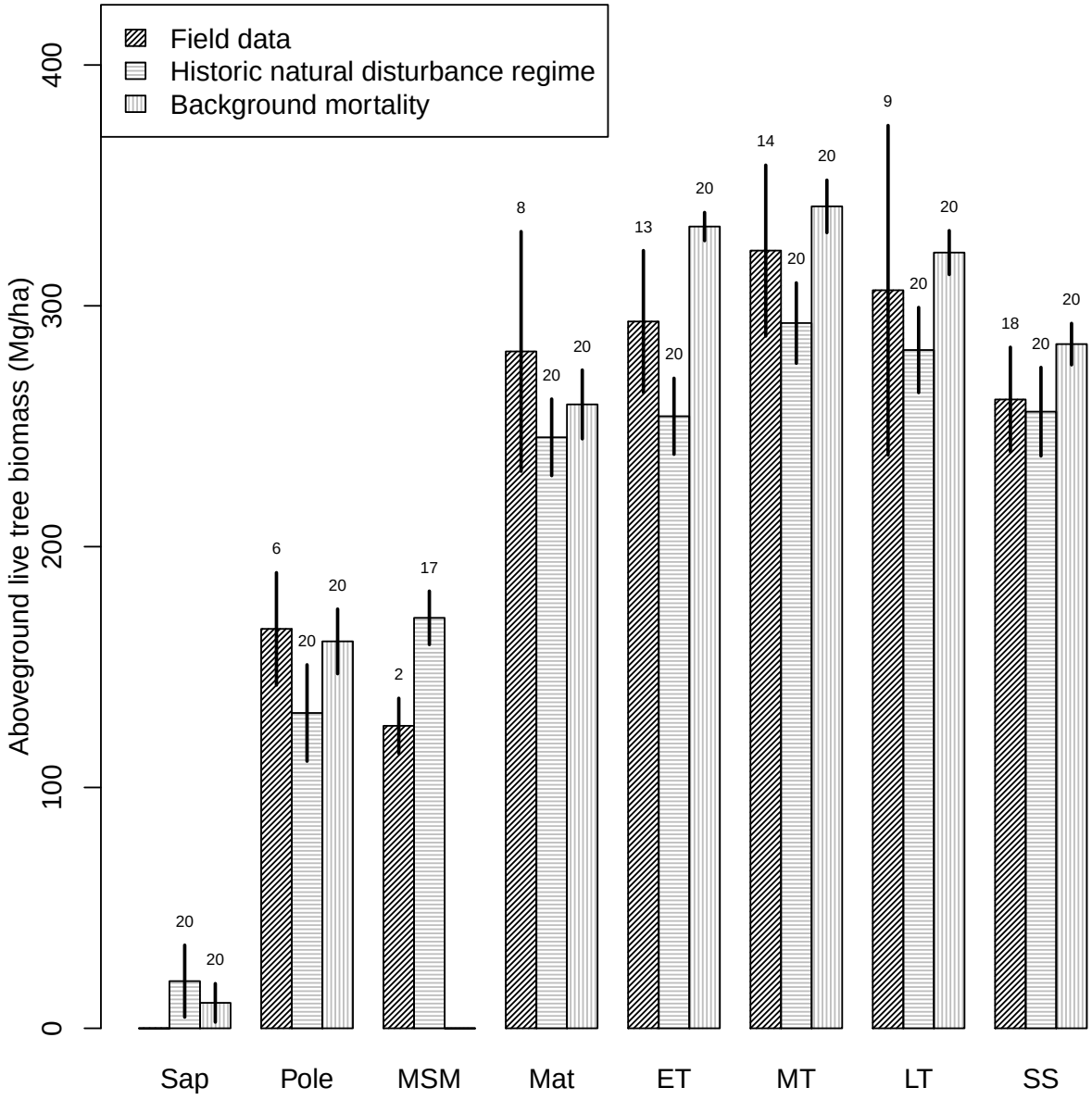

Figure 1: Biomass by stand stage of the seventy 0.5 ha field plots, compared to CANOPY simulations of 20 replicate 0.5 ha plots subjected to either background mortality or the historic natural disturbance regime. Simulations were conducted for mixed hardwood stands on *Acer-Tsuga-Dryopteris* habitat type, starting immediately after a clearcut and leaving no residual trees larger than 2 cm dbh. For simulation of biomass trends in the even-aged and transition stages under background mortality, data were selected only for the first 300 years after the clearcut. Vertical lines give 95% confidence intervals; numbers above them reflect sample size. Sap: Sapling; MSM: Mature-sapling mosaic; Mat: Mature; ET: Early transition; MT: Mid-transition; LT: Late transition; SS: Steady state.

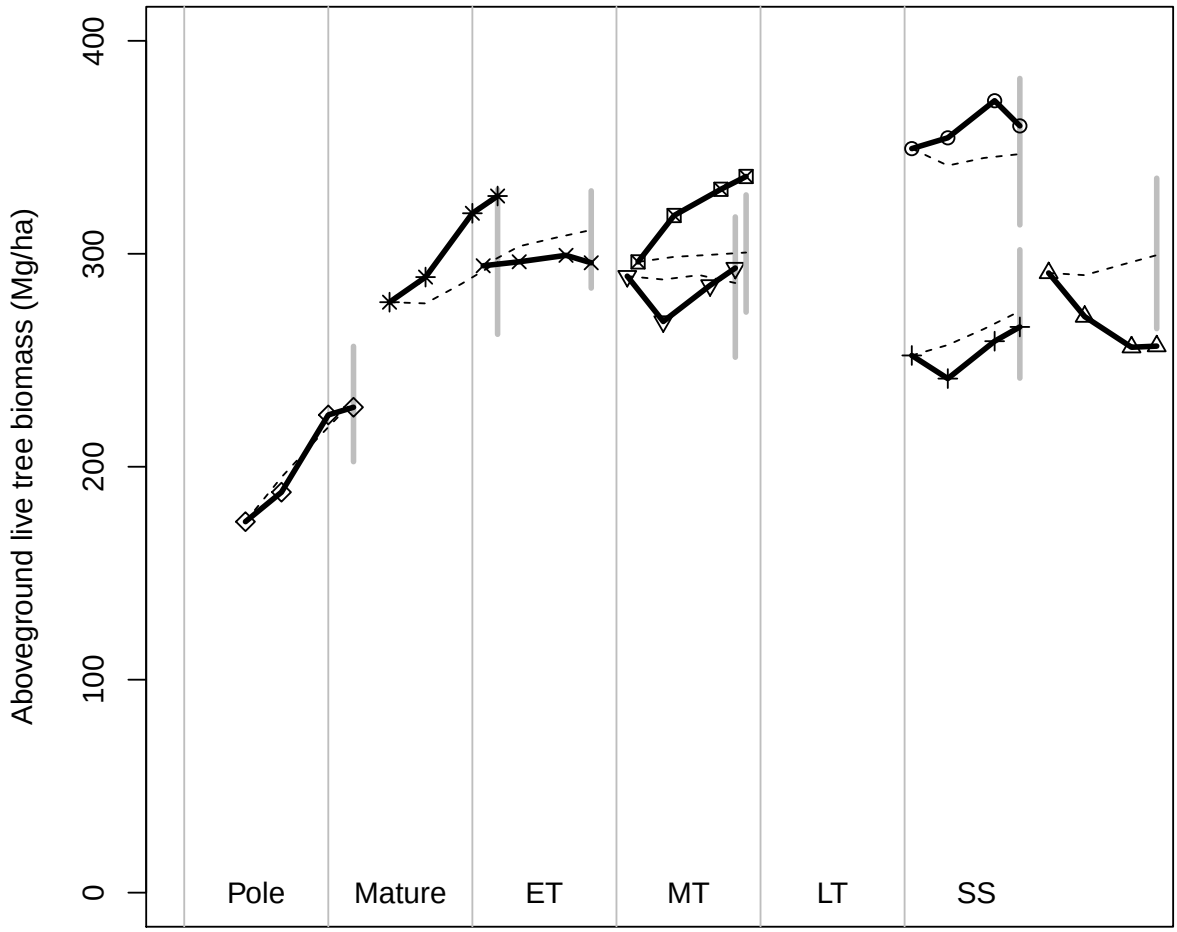

Figure 2: Biomass trajectories of the eight Porcupine Mountains permanent 0.5 ha plots over the 30 year remeasurement period (thick lines with symbols), sorted by stand stages. Grey bars give the range of variation for 20 CANOPY simulations of the same plots assuming only background mortality, with thin dotted lines showing the average trajectory over the 20 replicates. Abbreviations as in Fig. 1.

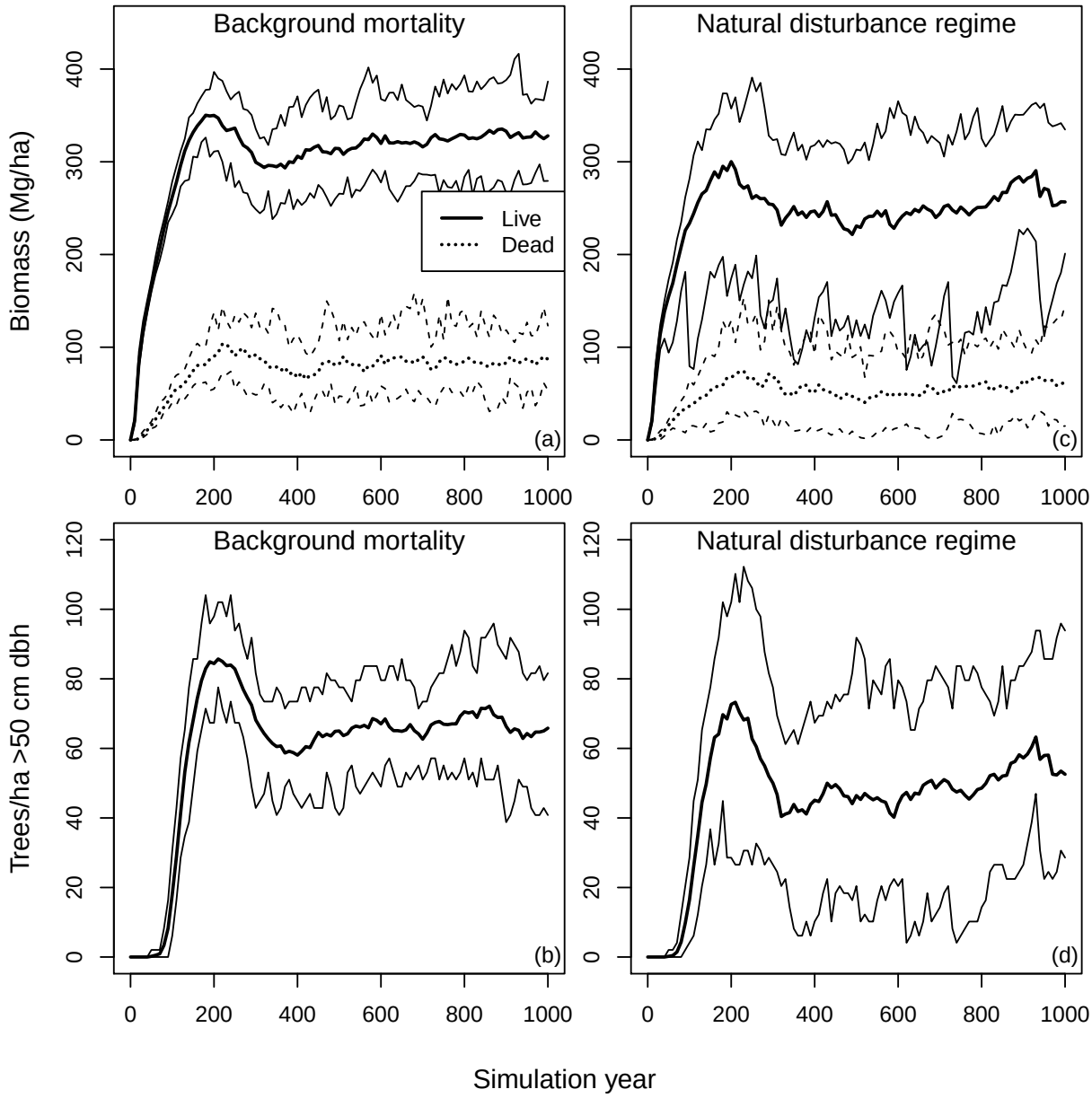

Figure 3: Simulated trends in aboveground tree biomass and number of large trees in mixed hardwoods on the *Acer-Tsuga-Dryopteris* habitat type. Thick lines give means, thin lines ranges of variation for 20 replicates using 0.5 ha plots. Note that later simulation years under the historic natural disturbance regime reflect the average of a mixture of twenty 0.5 ha stands encompassing a wide range of developmental stages rather than a uniform old-growth stand.

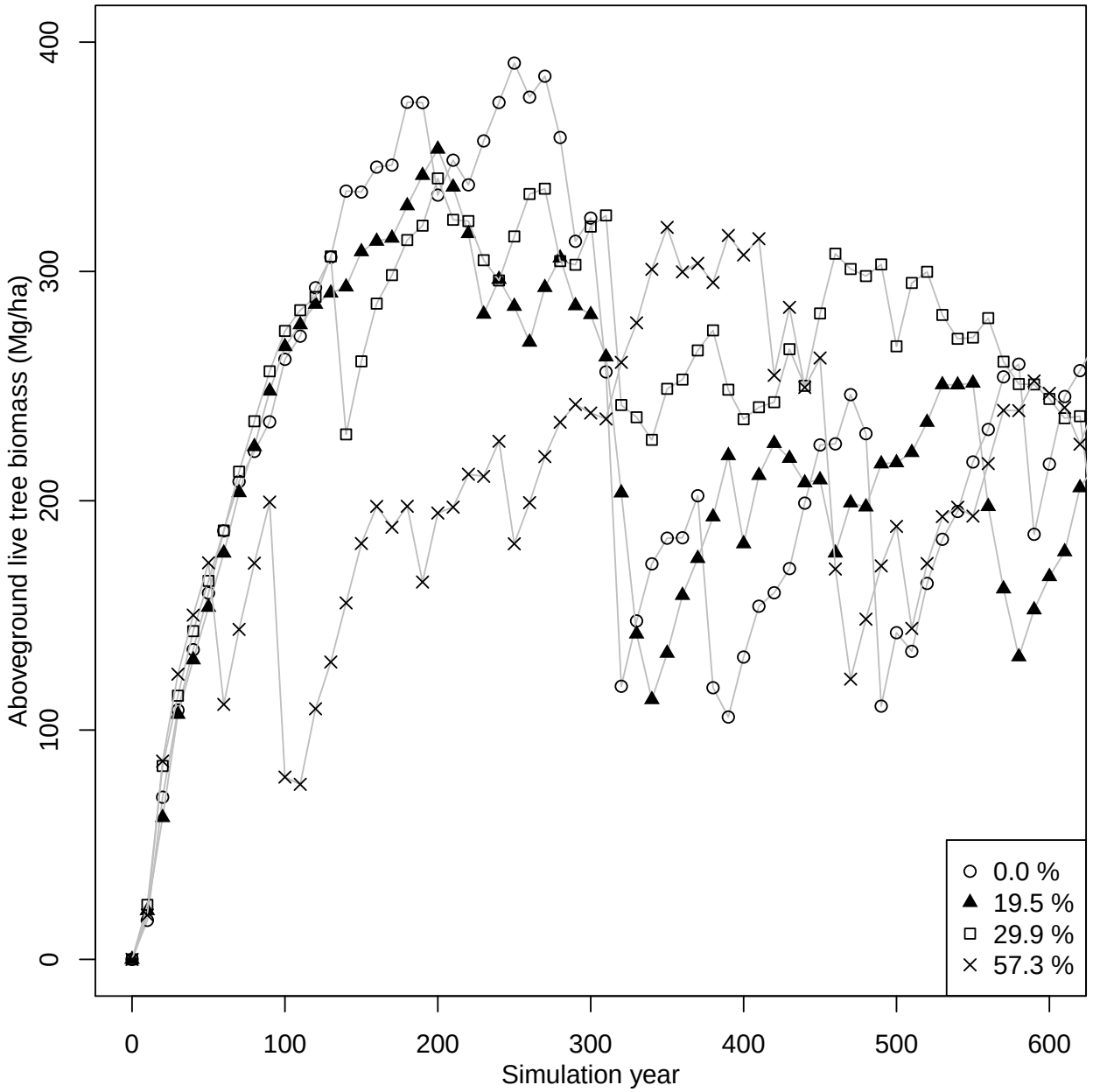

Figure 4: Simulated biomass trajectories of four replicate 0.5 ha plots simulated under the historic natural disturbance regime (a subset of replicates from Fig. 3 c). Symbols refer to the % crown area removed during the most severe disturbance on each replicate in the first 300 years. Selected cases illustrate biomass trajectories for replicates with the most severe and least severe disturbance during the first 300 years, with two replicates subjected to intermediate-severity disturbances.

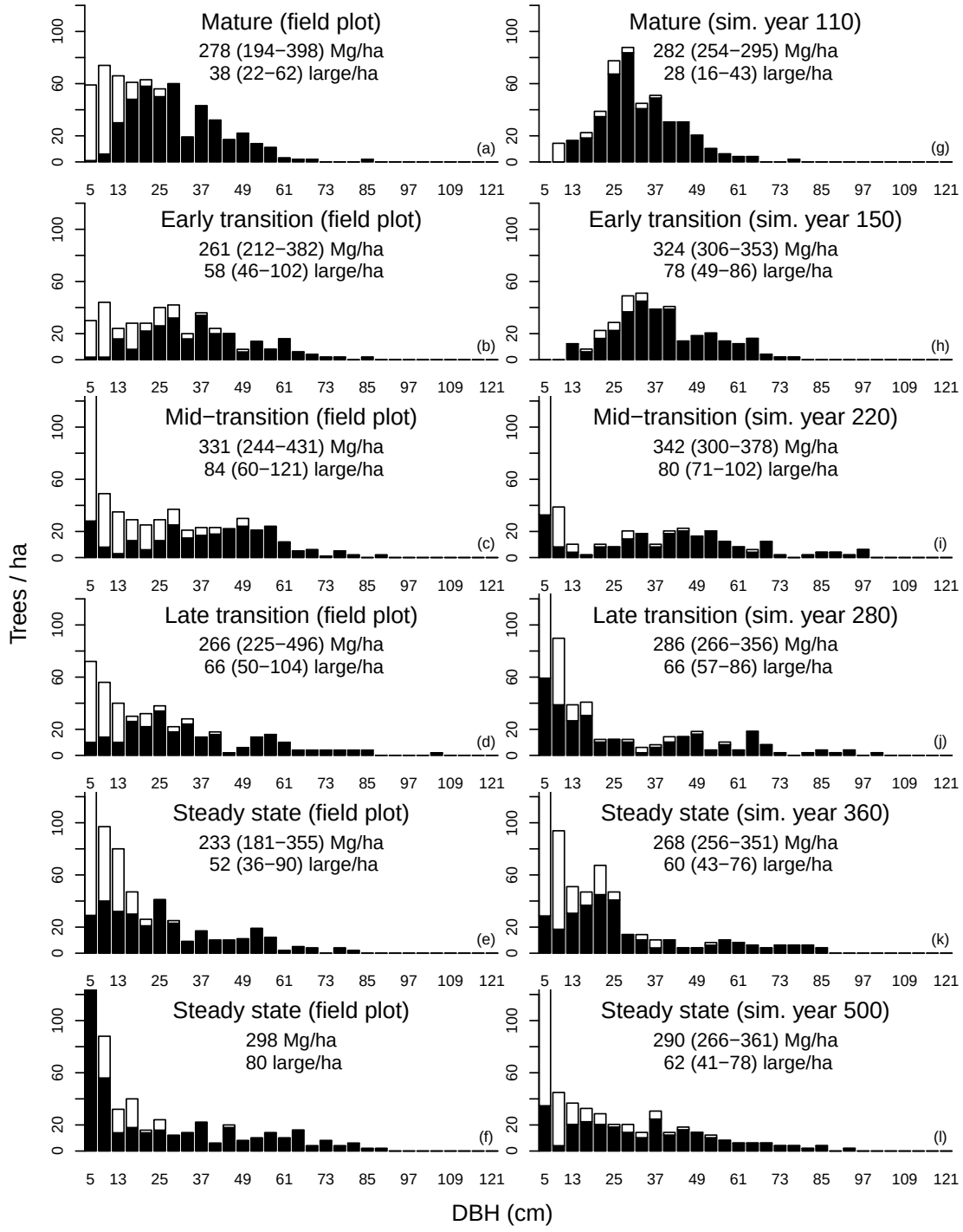

Figure 5: Diameter distributions of mixed hardwood stands from field data (a-f) and simulations (g-l) representative of the five mature and old growth stages on *Acer-Tsuga-Dryopteris* habitat. Simulations were conducted under a disturbance regime of background mortality only following an initial clearcut in year 0. Black bars denote canopy trees and gap saplings. Hollow bars indicate shaded understory trees. First row of numbers under the stand stage designation indicate the biomass of the particular stand or simulation replicate shown, as well as the range (in parentheses) among all stands or replicates in that stage. Second row of numbers shows the density per ha of trees >50 cm dbh for the stand or replicate shown.

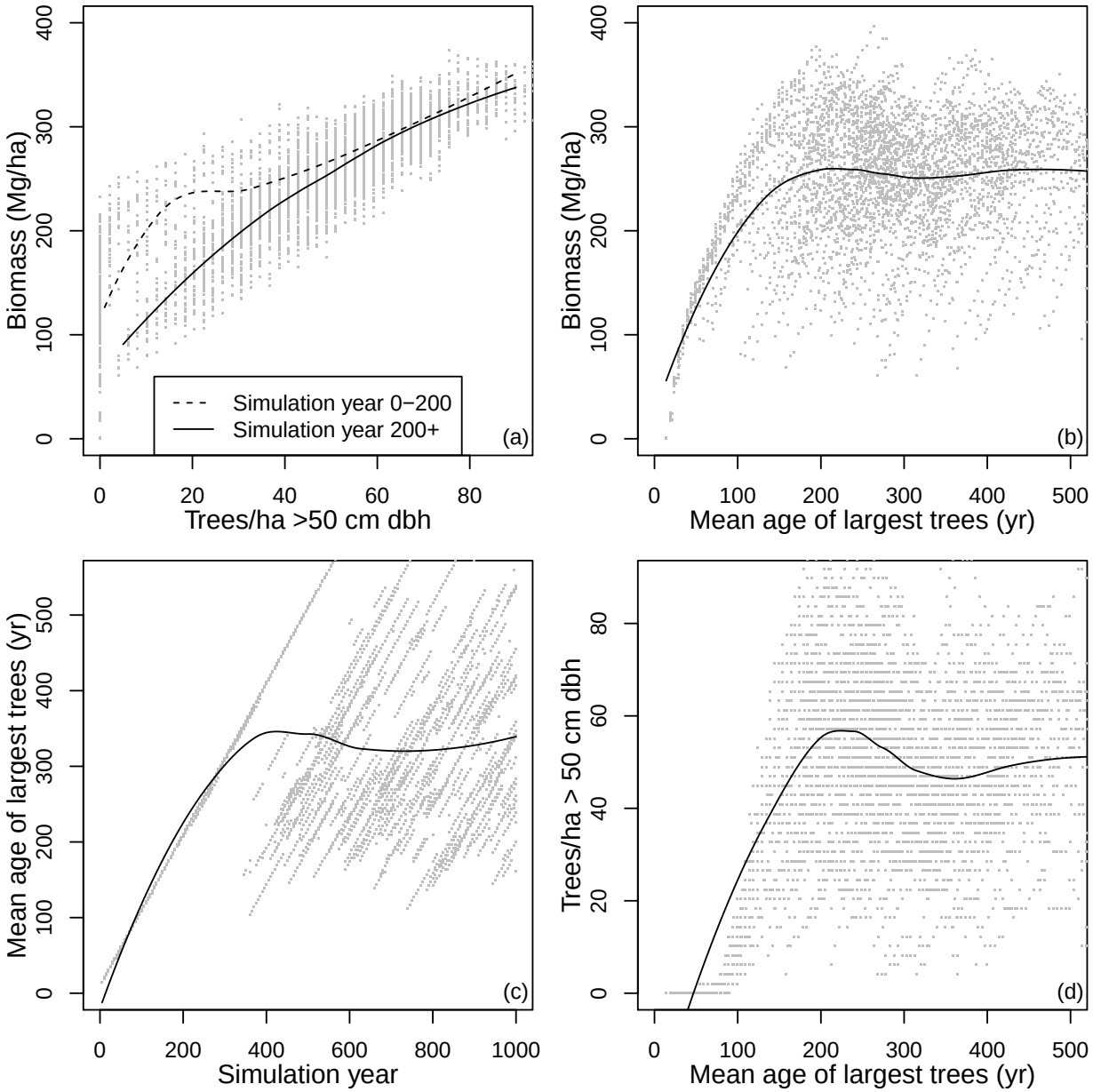

Figure 6: Simulated relationships between aboveground live tree biomass, number of large trees (>50 cm dbh), and mean age of the largest 5% of trees for the simulations in Fig. 3 c. Each dot represents a single simulation year; curves on all panels show a loess fit to the simulation data. Mean age of largest 5% of trees was computed at 5 year intervals for 20 replicate simulations developing after a clearcut on *Acer-Tsuga-Dryopteris* habitat with mixed hardwood regeneration, followed by 1,000 years of simulation under the historic natural disturbance regime. In panel (c), individual replicates follow a 45 degree trajectory in between mortality events, as trees age. Mortality events cause a vertical drop in average age as younger trees enter the largest 5% size class, resulting in an overall sawtooth trajectory for each replicate. The diagonal line from year 0 to approximately year 400 depicts the aging of the initial even-aged cohort with time, and the subsequent irregular trend shows the variable mean age of the largest 5% once the initial cohort is no longer among the largest 5% of the trees.

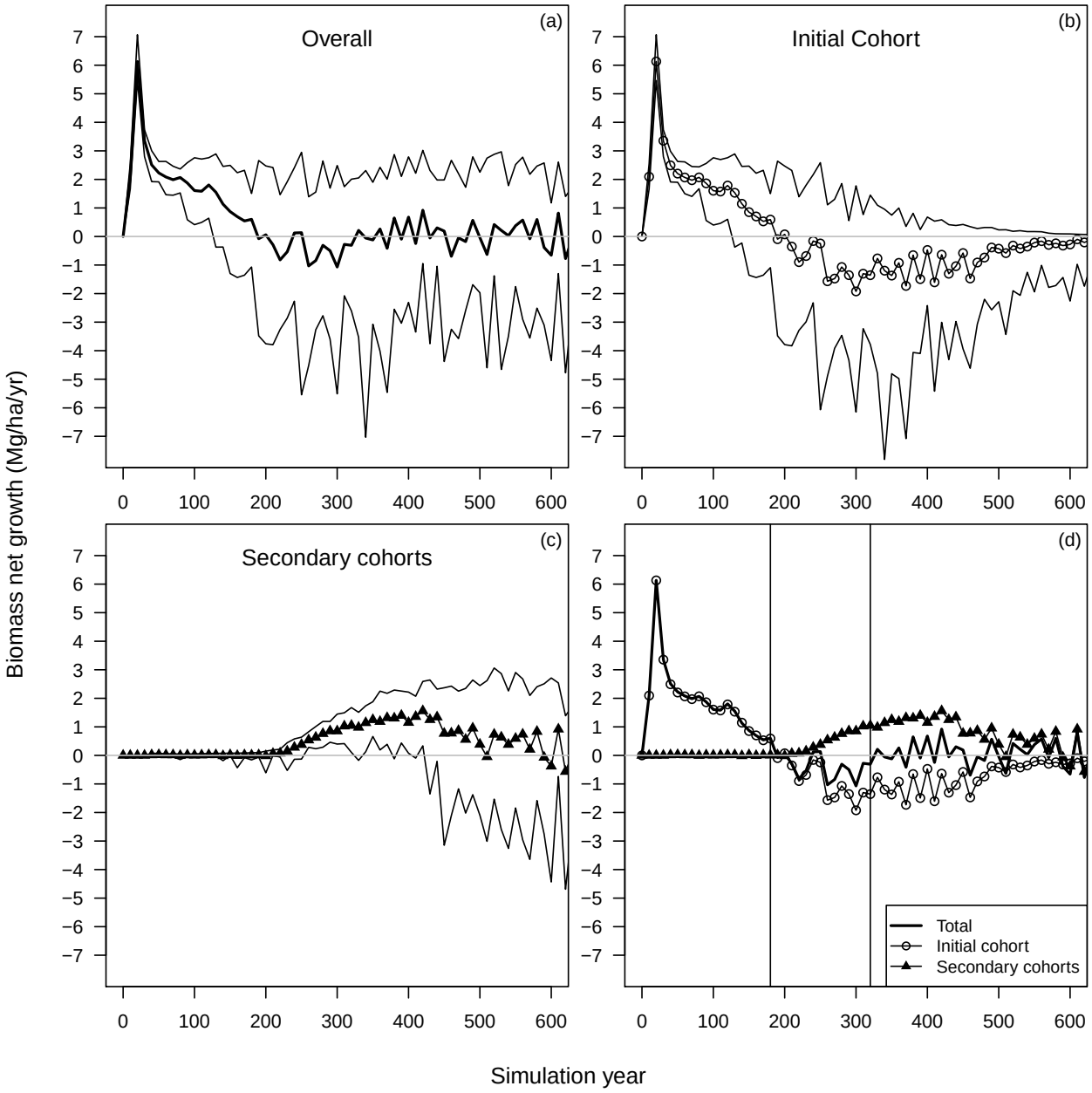

Figure 7: Simulated aboveground live tree biomass net growth (Mg/ha/yr) for initial and secondary cohorts after clearcutting in year 0, with no further disturbance except background mortality (simulations from Fig. 3 a). Vertical lines (panel d) mark the time of the initial peak biomass and subsequent trough in Fig. 3 a. Initial cohort comprises all trees that recruited into the stand from year 0-20.

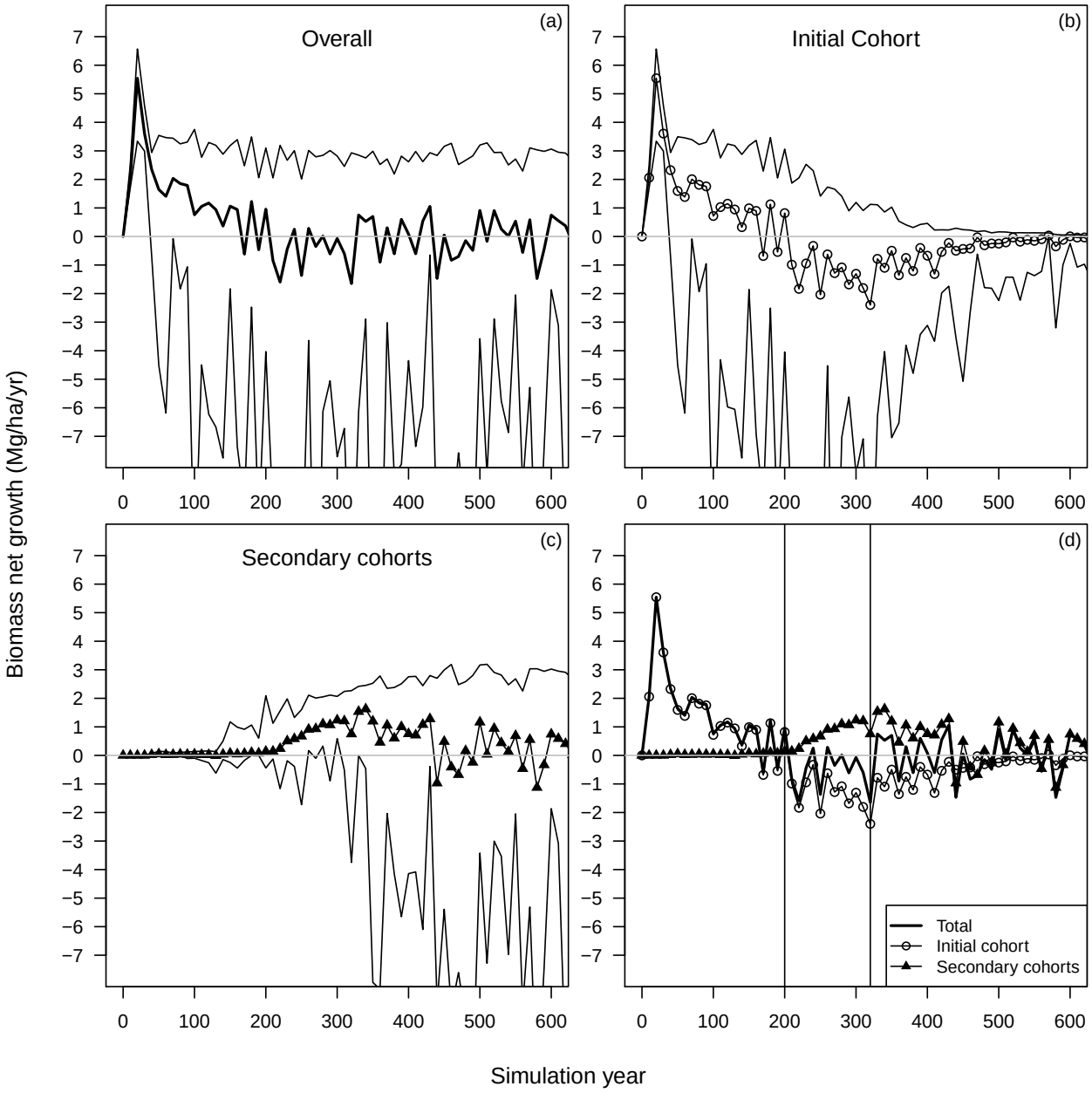

Figure 8: Simulated aboveground live tree biomass net growth (Mg/ha/yr) of initial and secondary cohorts after clearcutting in year zero; replicates were then subject to the historic natural disturbance regime (simulations from Fig. 3 c). Vertical lines (panel d) mark the time of peak biomass and subsequent trough in Fig. 3 c. Initial cohort comprises all trees that recruited into the stand from year 0-20.

Table 1: Effect of habitat type and species composition on the timing and magnitude of a biomass peak in CANOPY simulations under background mortality.

	Peak biomass		Trough biomass			Stable value	
	Mg/ha	yr	Mg/ha	yr	% Decline	Mg/ha	yr
Acer-Ozmorhza-Caulophyllum							
Mixed hardwood	314	120	286	230	9%	320	500
Pure maple	340	190	295	300	13%	330	400
Acer-Tsuga-Dryopteris							
Mixed hardwood	350	180	294	320	16%	320	450
Pure maple	330	220	320	350	3%	320	350
Pure hemlock	400	200	254	430	12%	300	600
Acer-Tsuga-Maianthemum							
Mixed hardwood	307	220	256	330	17%	256	330
Pure maple	276	200	236	320	14%	236	320

Table 2: Cumulative thirty year biomass growth (Mg/ha) of overstory and secondary cohorts from the Porcupine Mountains permanent plots. The overstory cohort is all trees $\geq$ 25 cm in 1981; secondary cohorts are all other trees, including ingrowth after 1981.

	Pole	Mature	Early trans.	Mid-trans.		Steady state		
				Plot number				
	5	8	4	6	7	1	2	3
Overstory cohort survivor growth	76.1	61.3	31.9	69.7	56.1	60.4	52.4	69.8
Overstory cohort mortality	45.0	10.5	28.6	83.1	38.7	75.2	113.0	68.7
Secondary cohort survivor growth	40.3	29.5	29.2	25.5	33.5	35.3	39.5	38.0
Secondary cohort mortality	32.7	36.0	31.9	13.2	12.3	12.4	15.3	35.2
Ingrowth after 1981	0.8	1.5	0.0	1.8	1.3	3.0	3.6	5.9
Overstory cohort net growth	31.1	50.8	3.3	-13.4	17.4	-14.8	-60.5	1.1
Secondary cohort net growth	8.5	-5.0	-2.7	14.0	22.4	26.0	27.8	8.7
Total net growth	39.6	45.8	0.6	0.6	39.8	11.2	-32.7	9.8
Mean annual net growth (Mg/ha/yr)	1.32	1.53	0.02	0.02	1.33	0.37	-1.09	0.32

Net growth = ingrowth + survivor growth - mortality. ingrowth = 0 for 1981 overstory.

Table 3: Cumulative thirty-year biomass growth (Mg/ha) by species from the Porcupine Mountains permanent plots.

		Pole	Mature	Early trans.	Mid-trans.		Steady state		
		Plot number							
		5	8	4	6	7	1	2	3
Sugar maple	Net growth	44.3	36.9	-7.0	-8.3	-1.6	7.2	-9.3	22.6
	Ingrowth	0.7	0.1	0.0	1.0	0.3	2.2	2.6	5.9
	Survivor growth	81.6	58.1	10.1	17.4	15.0	56.5	75.7	97.2
	Mortality	38.0	21.4	17.0	26.7	17.0	51.5	87.7	80.5
Hemlock	Net growth	0.4	7.2	5.9	16.4	36.2	10.8	-4.0	3.5
	Ingrowth	0.0	0.4	0.0	0.0	0.0	0.4	0.1	0.0
	Survivor growth	0.8	11.1	43.8	38.9	53.7	20.1	13.5	3.5
	Mortality	0.4	4.3	37.9	22.4	17.6	9.7	17.6	0.0
Yellow birch	Net growth	0.0	-0.8	0.0	-35.8	0.6	-14.1	-14.9	-16.4
	Ingrowth	0.0	0.5	0.0	0.3	0.7	0.1	0.1	0.0
	Survivor growth	8.7	4.3	0.0	7.0	11.4	3.5	0.3	6.4
	Mortality	8.8	5.5	0.0	43.2	11.6	17.7	15.3	22.8
Other	Net growth	-5.1	2.5	1.7	28.3	4.7	7.3	-4.6	0.2
	Ingrowth	0.1	0.4	0.0	0.4	0.3	0.3	0.8	0.0
	Survivor growth	25.3	17.3	7.2	31.9	9.4	15.6	2.5	0.8
	Mortality	30.5	15.3	5.5	4.1	5.0	8.6	7.8	0.6

Net growth = ingrowth + survivor growth - mortality.

CHAPTER 2

Trajectories of stand structural development in response to variable disturbance severities in northern hardwoods

Abstract

In late successional forests, analysis of long-term stand development using structural criteria are often more tractable and more closely related to key ecological parameters than more commonly used metrics like stand age or time since stand-replacing disturbance. In this paper, the effects of various disturbance regimes on long-term stand structural development in northern hardwoods at the stand and landscape scale were analyzed using the CANOPY individual-tree model. Percentages of a simulated equilibrium population among stages were similar to conventional chronology-based analyses for the earlier stages of development. However, steady-state stands as defined by structure were much more frequent than when the steady state is defined by the timing and severity of disturbances. Under the historic natural disturbance regime, characterized by predominately mild or moderate severity disturbances, mean residence times among all structural stages were predominately short (8 to 35 years) and followed descending monotonic distributions. Sequences of mild and moderate disturbances were sufficient to generate multi-aged unimodal size distributions that were similar in form to size distributions of even-aged stands with comparable modal diameters. However, simulation experiments often demonstrated structural resilience to repeated mild disturbance even at the 0.5 ha scale. For example, stands in the early stages of old growth did not revert

to earlier structural stages even after three disturbances removing 20% crown area at 30 year intervals. Recovery from heavy partial disturbances was markedly faster than recovery after clearcutting, with stands recovering to a steady state 175 years after disturbances removing 60% of the crown area, compared to 280 years after a clearcut. State transition diagrams based on a second-order Markov chain supported earlier evidence that the existing 23,000 ha study areas closely approach the threshold criteria for equilibrium landscapes.

2.1 Introduction

Many structural attribute of forests, like size distributions, are strongly shaped by disturbance history (Zenner, 2005; D'Amato et al., 2009). Disturbance histories, especially in late-successional forests, are often highly variable (Henry and Swan, 1974; Fraver et al., 2009; Vanderwel et al., 2013). The occurrence of infrequent stand-replacing events, upon which are superimposed many minor and moderate-severity disturbances, can result in stand developmental trajectories that resemble a complex web rather than a simple repeating cycle (Frelich and Lorimer, 1991a). These repeated partial disturbances can generate structurally complex multi-aged stands that would not be expected under a simpler disturbance model. For example, these landscapes can contain pole and mature stands that are highly uneven-aged and yet structurally almost indistinguishable from even-aged stands of similar mean diameter (Appendix A). And while severe disturbances usually cause reversion to early developmental stages (Bormann and Likens, 1979; Oliver and Larson, 1990), recent field evidence suggests that moderate disturbances in structurally uniform mature stands can at least temporarily accelerate the development of gap structure and a multi-layered canopy (Hanson and Lorimer, 2007). But it is not currently clear whether these changes might either set back or accelerate the long-term trajectories of stand structural development.

In complex multi-aged stands, 'stand age' has no clear meaning and 'time since last major disturbance' may often be impossible to determine from tree-ring evidence. Structural stages

of stand development may be more easily recognized (Appendix A). For some purposes, structural metrics may be more ecologically informative than disturbance chronologies. For example, wildlife habitat, forest production rates, and resilience to disturbance (i.e., the ability to 'absorb' disturbances without reversion to an earlier structural stage or to quickly regain pre-disturbance structural stage after disturbances) are probably more closely related to structural variables than to the timing of disturbances (cf. Franklin et al., 2002; Canham et al., 2001; Caspersen et al., 2011). Structure is also more easily monitored, and can be more quickly and precisely assessed, than disturbance history. Furthermore, land managers of late-successional temperate forests often have little or no information about stand age or disturbance history.

A structural classification of forest patches within a landscape could produce results quite different than one based on stand age or disturbance history. For example, based solely on the timing and severity of natural disturbances, only 4% of the area of three large landscapes of primary northern hardwood forest in Michigan could satisfy fairly lenient steady-state criteria (Frelich and Lorimer, 1991a). Yet old-growth stands with size distributions approaching steady-state conditions are frequently observed in field studies of these and other late-successional forests (Hett and Loucks, 1971; Goodburn and Lorimer, 1999; Emborg et al., 2000; Antos and Parish, 2002; Piovesan et al., 2005; Motta et al., 2011). Forest dynamics are most often interpreted using a non-equilibrium framework (Mori, 2011), but the possibility that disturbance history data and stand structural dynamics could lead to differing conclusions about equilibrium or non-equilibrium dynamics has not often been considered.

The overall goal of this paper is to quantitatively analyze stand developmental pathways from a structural perspective. Specifically, we ask the following questions:

- How does the complex natural disturbance regime, with its mixture of predominantly mild and moderate disturbances, influence the frequency of stand developmental stages

on a landscape compared to a 'dichotomous' disturbance regime of frequent small treefall gaps with infrequent severe events?

- How does the frequency of stand developmental stages on a landscape under the natural disturbance regime differ when stages are defined by structural criteria vs. disturbance history criteria? What are the corresponding transition rates between stages, and how variable are the resulting residence times?

- How resilient are old-growth stands to the impacts of mild and moderate disturbance? Specifically, what cumulative level of mild and moderate disturbances can be 'absorbed' by old-growth forests without retrogressing to earlier developmental stages? And when retrogression to earlier stages occurs, how does the recovery time to regain old-growth structure compare to recovery time after stand-replacing disturbance?

- For stands in the earliest stages of old growth, do periodic light to moderate disturbances (10-30% canopy removal) always cause retrogression to earlier stages, or can these disturbances accelerate the development of later developmental stages?

To achieve these objectives, simulation experiments were performed using CANOPY, an individual-tree model of forest gap dynamics. CANOPY was designed for long-term projections of demographic change such as size-class distributions, and it can simulate effects of the historic natural disturbance regime. A recently developed structural stand stage classification (Appendix A) was then used to analyze these simulation results. CANOPY has been calibrated using data from a >8,000 trees from stands representing a broad range of developmental stages, including pole, mature, and various stages of old growth. The model includes explicit simulation of gap dynamics by incorporating gap closure via asymmetric lateral growth of overstory crowns and height growth of saplings in response to opening size. Extensive validation of CANOPY has been performed for gap capture, recruitment,

size distributions in old-growth stands, and the response of stands to both natural and anthropogenic disturbance (Choi et al., 2007; Halpin, 2009; Hanson et al., 2011, 2012).

2.2 Methods

2.2.1 Study areas

Data were collected from three large natural areas with 23,000 ha of primary forests in upper Michigan: the Porcupine Mountains Wilderness State Park, the Sylvania Wilderness Area in the Ottawa National Forest, and a tract of primary forest west of Marquette, MI owned by the Huron Mountain Club (see Frelich and Lorimer, 1991b, for detailed site locations). These areas largely escaped 19th and early 20th century logging (Christy, 1929; Rafferty and Sprague, 2001). Climate in this region is humid continental with short and cool summers (mean summer temperature 20 °C; mean winter temperature -7.5°C near Lake Superior and -11°C further inland). Precipitation is fairly evenly distributed throughout the year and averages 80-90 cm annually. Elevations range from 182 m to ≈600 m. Soils belong primarily to the Fragiorthod and Haplorthod groups. Soils in Sylvania are generally of sandy loam texture, whereas soils in the other two study areas are generally of loam, sandy loam, or silt loam texture. The large majority of plots were classified as the *Acer-Tsuga-Dryopteris* (ATD) floristic habitat type of Kotar et al. (2002); other habitat types were present but all were mesic or dry-mesic sites. Site index of sugar maple (*Acer saccharum*) on ATD habitat is approximately 19-20 m at base age 50 (Coffman, 1984). In nearby experimental forests on similar sites, sugar maple site index is 18-21 m for base age 50 (Erdmann and Oberg, 1973; Crow et al., 1981). Aboveground net primary productivity young second-growth stands on ATD habitat is approximately 7.2-9.5 Mg/ha/yr (Fassnacht and Gower, 1997).

2.2.2 Field methods

Seventy half-hectare plots were originally located in 1981-1984 using random coordinates on maps of the primary forest zones. A half hectare plot size was selected to provide a

sample large enough to reasonably be considered a population (generally >165 trees more than 10 cm dbh) with a relatively homogeneous disturbance history and site quality. Species composition and structure of 0.5 ha individual plots were typically representative of the larger stands in which they occurred. Plots were divided into seven contiguous 70.7 x 10.1 m strips, although in some younger stands with higher stem densities, only a subset of strips were censused. Eight of the 70 plots, spanning a wide range of developmental stages, were permanently marked, mapped, and were remeasured in 1992, 2004, and 2011.

On each plot, diameter at breast height (DBH) and crown class was recorded for all trees $\geq$ 1.4 m tall. Increment cores were taken from 10-30 randomly located canopy trees for stand history reconstruction. On the permanent plots, an average of 68 increment cores were obtained per plot (range 39-99).

2.2.3 Model description

Simulations were performed using CANOPY, an empirical, individual tree, spatially explicit model (Hanson et al., 2011, 2012). Calibration data for the model were collected in northern Wisconsin and upper Michigan and span a wide range of stand developmental stages, from young even-aged through uneven-aged old-growth. For the version of CANOPY used in this paper (v.3), the calibration data set was expanded to include a substantial amount of long-term permanent plot data from managed and unmanaged second-growth hardwood stands in the Argonne Experimental Forest of northeastern Wisconsin (Niese and Strong, 1992). CANOPY has submodels for the *Acer-Ozmorhza-Caulophyllum*, *Acer-Tsuga-Dryopteris*, and *Acer-Tsuga-Maianthemum* floristic habitat types of Kotar et al. (2002).

CANOPY predicts species composition and density of 2-6 cm saplings entering a stand as a function of overstory species composition and density. Height growth of small trees is modeled for each species as a function of initial height, gap area, and competitor crown variables. Diameter growth of overstory trees is simulated as a function of initial size, species,

and local crowding. Height and diameter growth equations both include categorical variables and interaction terms for habitat type. Asymmetric crown expansion of overstory trees is simulated as a function of crown position and predicted diameter growth. Background mortality is modeled as a logistic process based on species, initial tree size, and subplot competition level. Growth predictions incorporate normally distributed stochastic variation based on the mean squared errors estimated by the regression procedure. Each tree is assigned a stochastic growth modifier when it recruits into the stand that is used throughout its life. These modifiers raise or lower the growth of individual trees relative to the predicted mean, which itself changes in response to local crowding.

In addition to simulations of background mortality, CANOPY is also able to simulate disturbance-related mortality. Each year, a random variable was used to determine if a disturbance should occur, with recurrence intervals for a given level of crown removal from Frelich and Lorimer (1991b). The maximum canopy area removed in a disturbance was 70% because of limited evidence on the frequency of more severe events. CANOPY then applied individual-tree probability of blowdown equations from Hanson (2009), which are a function of the size and species of individual trees, as well as the storm severity index, an empirical estimate of storm severity based on the overall amount of blowdown on a plot (Canham et al., 2001). A regression was developed to relate storm severity index to the amount of canopy removal assessed in the earlier studies of the natural disturbance regime. If crown removal at the regression-estimated storm severity index differed by more than 5% from the target given by the recurrence intervals, CANOPY adjusted the severity upward or downward, then repeated the removal procedure until the target amount of crown area was removed.

The current version of CANOPY (v.3) adds the ability to simulate disturbances that are correlated across plots. Previous work showed that disturbances that removed $\geq 40\%$ of the crown area on a plot tended to occur in clusters affecting approximately four plots (equivalent to $\sim$1,300 ha of forest), while smaller disturbances did not have statistically

significant spatial correlation (Frelich and Lorimer, 1991b). In each year, disturbances were simulated independently for each plot. When a disturbance removed $\geq 40\%$ of the aggregate exposed crown area on one plot, three other plots were also selected at random and subjected to a disturbance of equal severity. To maintain the same recurrence intervals between the spatially correlated and uncorrelated simulations, the next three subsequent disturbances of that severity were then skipped.

To mimic the sparse recruitment commonly observed in the earlier stages of stand development, a refinement was made to the recruitment module in CANOPY v3. Even on plots relatively unaffected by deer browsing, the field data often contain evidence of low ingrowth and high mortality for saplings in stands with high concentrations of pole and mature trees (Appendix A). Based on the observed negative correlation between percent basal are of mature trees and sapling density ($p=0.005$), a switch was added to CANOPY delaying further recruitment whenever pole trees comprise $\geq 20\%$ of the basal area or mature trees comprise $\geq 35\%$ of the basal area. This rule is evaluated for each 10 x 10 m cell by assessing basal area allocation on the 50 x 50 m (0.25 ha) patch centered on that cell. Uneven-aged stands undergoing small gap formation as a result of background mortality typically have basal area levels below these thresholds, and are therefore not generally affected by this rule.

2.2.4 Simulation design

Field measurements from the seventy 0.5 ha plots from the early 1980s were used as initial conditions, and stand development was simulated for 1,000 years with 10 replications per plot. For the temporary plots that were not stem-mapped, tree locations were randomly generated given previous work on stem spatial patterns (e.g., Frelich and Graumlich, 1994; Chokkalingam and White, 2001), with a different random stem map used in each replication. Plots were subjected to a number of stochastically-timed disturbances, with the frequency and severity based on the data in Frelich and Lorimer (1991b). Half-hectare plot sizes

were used for simulations involving the natural disturbance regime because the recurrence intervals in CANOPY were based on plots of that size. To evaluate the immediate response of stands with varied histories to a new disturbance, structural stage following each stochastic disturbance was tabulated.

A sensitivity analysis was also conducted under a 'dichotomous disturbance regime' comprised of only small gap formation due to background mortality and infrequent disturbances removing 70% of the aggregate exposed crown area (the latter with a 3734 year mean recurrence interval (Frelich and Lorimer, 1991b)). For the 0.5 ha plot size, all 70 plots were simulated starting from the 1981-84 plot data. To evaluate the effect of plot size on the resulting developmental trajectories, 4 ha plots were constructed by tiling the 0.5 ha plot data in a 4x2 grid (i.e., using the measured plot in a repeating pattern). Because of the long time required for 4 ha simulations, a subset of 30 plots were randomly selected from the larger set of 70 plots for the 4 ha sensitivity analysis.

Simulation experiments were also performed that subjected 0.5 ha old-growth plots to both a single moderate severity disturbance and repeated low severity disturbances at 30 year intervals. Starting conditions for these simulations were plots in the early transition stage (representing the earliest stage of old-growth development with a non-equilibrium size distribution) and the steady state (representing the latest stage of old-growth with a potentially stable size distribution). To control for the influence of past disturbance, initial conditions for these experiments were drawn from 20 replicated 1,000-year simulations of stand development starting from a clearcut and with no intermediate disturbances aside from small gap formation due to background mortality. This approach has the additional advantage of accounting for the structural diversity of stands in old growth by giving each replication a different set of initial old-growth conditions.

2.2.5 Analytical techniques

Structural stages of simulated stands were classified using a refinement of the stand stage definitions from Frelich and Lorimer (1991a). The structural stages reflect a generally progressive increase in modal overstory diameter and increasing understory development as stands progress from young to old-growth stands (Appendix A). Revised criteria are based entirely on aggregate basal area of size classes rather than crown data or disturbance history, and the old-growth stage has been subdivided into four stages. Individual stages from sapling stage to old-growth are determined primarily by whether saplings (0-10.9 cm), poles (11-25.9 cm), mature (26-45.9 cm) or large trees ($>$ 46 cm) comprise the largest proportion of stand basal area. For example, in old-growth stands, large trees (with ages usually $>$150 years) make up $>$ 45% of the stand basal area (Fig. 1 in Appendix A).

The first three stages of old growth (early, mid-, and late transition) represent the transition from a unimodal size distribution and generally even-aged structure to the descending monotonic size distribution of an all-aged forest. Early transition has limited understory development and the lowest modal diameter of all the old growth stages (Fig. 1 b). Mid-transition has variable understory development, but among the mature and old-growth stages, it has the lowest mean proportion of basal area in saplings and poles and the highest proportion of large trees. Late transition has a well-developed understory with a reduced component of large trees.

The structural steady state stage includes stands with descending monotonic diameter distributions that have the potential to be self-sustaining (Fig. 1 d). The sustainability is based on the concept of a balanced sized distribution where each size class occupies roughly equal amounts of growing space (Smith et al., 1997; Goodburn and Lorimer, 1999; Nyland, 2007). Note that the steady-state stage in this classification does not imply either a lack of exogenous disturbance or long-term demographic stability. In fact, residence times in the structural steady state can be quite short and, at the 0.5 ha scale, retrogression from steady

state can occur during decades of above-average background mortality. This stage only signifies the potential for a self-sustaining size distribution that, while not entirely static, would not experience large changes in form.

In order to be classified as steady state, an old-growth stand had to have a well-developed understory with a significant proportion of saplings growing in gaps (gap saplings >0.6% of stand basal area). In the original field data, gap saplings were identified by the degree of crown exposure in gaps. In simulations, gap saplings were identified using similar criteria, but gap sizes were based on predicted rather than observed crown radii of gap border trees.

The stand stage classification system also includes a 'mature-sapling mosaic' stage. These stands have some mature or large trees (10-20 m^2/ha of basal area), but not enough to meet the criteria for mature or old-growth forests. These appear to be formerly mature or old-growth stands that were subjected to moderate severity disturbance that left a number of legacy trees.

A second-order Markov chain was constructed from CANOPY simulations, as it enables the equilibrium distribution of stand stages to be directly computed, rather than only approximated with large numbers of simulations. States were defined that included the current and previous stages of each plot (e.g., pole from sapling). A second-order formulation allows transition probabilities that are conditional on the previous stand stage. Including this historical data allows the model to take into account structural features, like large legacy trees, which may affect the residence time in a given stage. For each of the 10 replicates of the 70 plots in the data set, transition rates between stages were computed from the last 800 years of a 1000 year simulation. These transition rates were then turn used to approximate transition probabilities. A Markov state transition matrix M was constructed, and the equilibrium was computed by finding i such that $M^i = M^{(i+1)}$, then taking the diagonal of M^i. Residence times in each state at equilibrium were computed using the expectation of a negative binomial variable with $p = M[j, j]$ and $r = 1$. To evaluate the impact of species

dominance, an additional Markov analysis was performed that augmented state definitions with species dominance (e.g., maple sapling from hemlock (*Tsuga canadensis*) mature). For this analysis, a plot was classified as dominated by hemlock when hemlock comprised $\geq 35\%$ of the stand basal area.

2.2.6 Evaluation of model predictions

CANOPY has been extensively validated in previous studies both for natural stand development and responses to silvicultural treatments. Hanson et al. (2012) found that simulated volumes harvested under single-tree selection over a 300-yr span differed by <6% from harvest volumes in comparable experimental field studies. For untreated stands in northern Wisconsin, Hanson et al. (2011) demonstrated close correspondence of stand basal area and tree density by size class between 1,000 year CANOPY simulations and old-growth forests on similar sites. CANOPY's recruitment mechanism also replicated observed trends in the number, species composition, and spatial pattern of saplings for both treated and untreated stands.

The ability of CANOPY to predict changes in size distributions and stand developmental stages was evaluated in the present paper by comparing 30 year changes in observed vs predicted diameter distributions and stand stage on the permanent plots (Fig. 1). Overall, the observed 2011 field measurements fell within the range of variation for CANOPY simulations for 82% of the 4 cm diameter classes. On individual plots, observations fell within the predicted range of variation for 79%-95% of diameter classes except for the pole plot, where observation fell within the predicted range of variation for 67% of diameter classes. In most cases, the observed diameter distributions after 30 years were near the middle of the range of values predicted by CANOPY (Fig. 1).

CANOPY also predicted stand stage in 2011 correctly for the large majority of replicates. On 6 of 8 plots, the majority of individual replicates of each plot correctly matched field

observations of stand stage (70% correct predictions overall on these plots), including the two plots that changed stage in the field observation. For the pole stand, all 20 replicates correctly predicted the development from pole to mature stage, and 12 of 20 replicates for the mature hardwood stand correctly predicted the development from mature to the early transition stage during the 30-year period. Four plots were correctly predicted to remain in their initial stages, including an early transition plot, a mid-transition plot, and two steady state plots. The two plots where CANOPY predictions did not agree with field observations included a mid-transition plot that was predicted to advance to late transition, but did not mainly because of heavy deer browsing that slowed the pace of understory development. One of the three steady plots was also predicted to retrogress to late transition from stochastic background mortality but actually remained in steady state.

In long-term (1000 year) simulations of the upper Michigan field data, predicted mean numbers of trees per size class under background mortality were very close to the mean numbers in each size class of the 18 steady-state forests measured in the field (Fig. 2 in Chapter 3).

2.3 Results

2.3.1 Initial distribution of structural stages and predicted changes

The initial distribution of stand stages on the study area landscapes in 1981-1984 was 9% pole, 11% mature, 3% mature-sapling mosaic, 19% early transition, 20% mid-transition, 13% late transition, and 26% steady state. Initial stand structural classification in the simulations differed for a few plots from the field classification because of the need in simulations to generate stem and crown maps on unmapped plots, which altered the gap status of some saplings compared to direct field observations.

The final predicted balance of stand stages on the landscape was very close to initial conditions. As simulations proceeded, CANOPY predicted a decrease in the proportion of

pole stands over time from 9% to 2%. Corresponding fluctuation in the mature, transition, and steady-state stages occurred as the existing cohort of pole stands advanced through the older structural stages (Fig. 2). Most stages then returned to essentially their initial levels except for the mature-sapling mosaic and pole stages. Mature-sapling mosaic was predicted to increase to 8% of the population in simulations, becoming more common than either sapling or pole stands. However, the sum of sapling, pole, and mature-sapling mosaic stages – all of which reflect fairly heavy recent disturbance – was similar between simulations and field measurements (11% for simulations vs 12% for field data).

2.3.2 Residence times among stages from CANOPY simulations

Mean residence times in each stage ranged from about 8 years for sapling stands to 35 years for mid-transition stands (Fig. 3). The distribution of residence times in each stage was descending monotonic in form, except for the pole and mid-transition stages, which had an irregular curve shape. A negative exponential distribution had no significant lack of fit for residence times in the sapling stage (chi-squared test; $p=0.14$), but had significant lack of fit for all other stages ($p<0.001$). Residence times were highly variable in most stages. For example, while the pole stage had a mean residence time of only 27 years, some plots remained in the pole stages for up to 75 years. The steady-state stage, which also had mean residence time of 27 years, had some stands remaining in that stage for over 100 years. Mean residence time in old growth was 87 years, but ranged as high as 500 years for some plots. Mean residence time in the last two old-growth stages combined (late transition and steady state) was 35 years, but ranged up to 170 years for some plots.

Residence times were markedly shorter under the complex natural disturbance regime than when simply developing along an even-aged pathway for several reasons. For example, average residence in the mature stage under the historic natural disturbance regime was only about 20 years, but averaged about 70 years in even-aged stands (Appendix A). This

was due partly to the large residual trees remaining after moderate disturbances, permitting shorter recovery times to old-growth structure. Secondly, mild or moderate disturbances often caused retrogression to mature-sapling mosaic and other earlier stages before the stand had a chance to progress to the next stage. Thirdly, many of the short residence times in uneven-aged stands in this analysis occurred in stands near the boundary of two structural stages. In these stands, higher than average background mortality was often sufficient to cause a stand to retrogress to an earlier stage. Residence times from this paper, therefore, cannot be summed among consecutive stand stages to estimate the time required to reach a particular stage along an even-aged pathway.

2.3.3 Predicted equilibrium structural composition from the Markov analysis

In the equilibrium distribution of stand stages predicted by the second order Markov analysis of CANOPY simulations under the historic natural disturbance regime, $\sim$ 80% of the plots were in the old-growth stage (Fig. 4). Most of the stage transitions were among the early transition, late transition, and steady-state stages, among which individual plots had a random-walk behavior. The highest proportion of plots (26-27%) occurred in the early transition and steady state stages. The proportion of plots in the pole stage was lower than in the field data (2% vs 9%) and the proportion in mature-sapling mosaic was higher than in the field data (8% vs 3%).

When simulations were conducted under a dichotomous disturbance regime (frequent small-gap formation and infrequent severe disturbance), there was a dramatic increase in the proportion of the Markov equilibrium distribution in steady state, as expected (Fig. 5). However, the developmental pathways in this situation were more complex than expected, with 11% of the plots in early transition, 15% in mid-transition, and a significant amount of retrogression and recovery between steady state and these earlier stages. Over the span of a typical century, nearly all the stands in steady state and late transition shifted back and

forth between those two stages. About 20-30% of the steady state stands per century shifted back and forth between that stage and early transition. This diversity of structural stages appeared to be related to the relatively small scale of the 0.5 ha plots. When larger 4 ha plots were simulated (Fig. 6), the proportion of plots in early transition and mid-transition decreased dramatically to 2.4% and 3.4%, respectively, and retrogressions from steady state to earlier stages of stand development decreased to $< 5\%$ of the plots per century. At the 4 ha scale, late transition occupied 44% of the distribution and steady state occupied 49%, with the overwhelming majority of the transfers between stages occurring between these two as a result of stochastic variation in background mortality (Fig. 6).

A Markov chain equilibrium analysis that took into account the species composition of individual stands revealed little difference in the residence times between sugar maple and hemlock dominated stands (Fig. 7). Residence times for hemlock stands in the transition stages of old growth were slightly longer than maple stands, though both species had residence times in these stages that were quite short. Shifts in dominance between maple and hemlock (e.g., reciprocal replacement *sensu* Woods, 1979) were rare in the simulations (Table 1).

2.3.4 Effect of disturbance on structural development

Based on the field estimates of recurrence intervals for natural disturbances incorporated into CANOPY, most disturbances were of low severity. For example, of the more than 30,000 five-year intervals simulated in Table 2, 93% had mortality levels not distinguishable from background mortality. Seventy-five percent of the more than 2100 disturbances beyond background level had severities of 20% canopy removal or less (98% of all intervals).

In simulations of the historic natural disturbance regime, disturbances of 10% canopy removal had little immediate effect on the simulated stand stage of old-growth plots. Disturbances of 20% removal caused elevated frequency of retrogression relative to background

mortality, but more than 50% of the stands remained in, or advanced beyond, their pre-disturbance structural stage (Table 2). As disturbance severity increased to 30% removal or more, the large majority of old-growth stands underwent immediate major retrogression to earlier stages. For 40-50% removal, retrogression to mature-sapling mosaic was the most common outcome. For 60-70% removal, retrogression to the sapling stage or mature-sapling mosaic was the most common result.

Simulation experiments that subjected plots to a controlled sequence of disturbance timing and severity, and which tracked the response over several hundred years, provided insights into long-term recovery patterns. Background mortality alone on the 0.5 ha scale was sufficient to cause some retrogression of steady-state stands to earlier stages, such that the average stand stage often 'drifted' into the late transition stage over the course of the simulations (Fig. 8 a). For both steady state and early transition stands, single disturbances removing 10%, 20%, or 30% of the crown area resulted in stand stages that were not statistically distinct from simulations with only background mortality.

Simulation experiments that subjected plots to sequences of light and moderate disturbances demonstrated the cumulative effect of multiple disturbances (Fig. 8 b,d). Repeated disturbances removing 10% of the crown area at 30-year intervals continued to show essentially no effect on stand stage. Repeated 20% removals caused steady state to revert to earlier stages after the second removal and caused early transition to revert to earlier stages after the fourth removal. Repeated 30% removals caused both steady state and early transition to revert to mature or mature-sapling mosaic stages after the second removal.

There were surprising differences in the impact of a given disturbance severity on different stages of old growth (results for early transition and steady state shown in Table 2 and Fig. 8). After 10-20% canopy removal, the percentage of stands remaining in their pre-disturbance stage or advancing to a later stage was similar for early transition and steady state. But immediately after 30-40% canopy removal, early transition stands were 2-4X more likely

than steady state to remain in the pre-disturbance stage or advance to a later stage (Table 2). Recovery times were also quite different in these two old-growth stages, as can be seen in the different slopes of the curves in Fig. 8 a,c. For example, after 60% canopy removal that set both of these old-growth stages back to mature-sapling mosaic, it took an average of ∼90 years for a steady-state stand to recover to the early transition stage compared to ∼40 years for a stand initially in the early transition stage.

There was some, but limited, evidence that light or moderate disturbances in mature or early transition stands could accelerate development to a later stage by increasing gap formation and lowering the high concentration of mature trees. After 2-3 disturbances of 20% canopy removal in early transition stands, stand stage was significantly more advanced 25-100 years later compared to 'control' stands under background mortality (Fig. 8 d). In the simulation experiments with a single 10-20% disturbance, stand stage after 25-100 years was nominally more advanced on average than the controls, but differences were not significant (chi-squared test, n=40 plots, $p>0.05$).

2.4 Discussion

2.4.1 The effect of complex disturbance regimes on stand structural development

Simulations based on the historic natural disturbance regime produced more complex pathways of structural development than a 'dichotomous' regime of small gap formation with severe disturbance at long intervals. Under the historic natural disturbance regime, there was a greater diversity and more equitable distribution of stand stages. The natural disturbance regime on 0.5 ha plots had only 64% as much forest satisfying the structural criteria for a steady state, but 6X as much forest in the mature stage and twice as much forest in the early and late transition stages.

The distribution of residence times in Fig. 3 likely reflects a number of different interacting

processes, both progressive and retrogressive. Stands can progress to higher developmental stages either as a result of growth increasing the modal diameter, or as a result of mortality increasing the structural complexity (e.g., increasing the proportion of gap saplings). For upward transitions driven by growth, a normal distribution of residence times might be expected, with mean determined by the average of the underlying growth equations and variance determined by the stochastic variation in growth. In 4 ha simulations under background mortality, residence times were normally distributed for the sapling and pole stages (Shapiro-Wilk test; sapling: p=0.32, pole: p=0.95).

Stands can also retrogress to earlier developmental stages as a result of disturbance mortality or sometimes even higher-than-average background mortality. By definition, residence times within a stage are analogous to age distributions (i.e., a patch having a 10 year residence in the mature stage is a 10-year old mature patch). Age distributions of forest patches, assuming independent probability of disturbance, tend to follow a negative exponential distribution (Van Wagner, 1978). If the probability of a disturbance sufficient to cause retrogression is constant within a stage, then disturbance-truncated residence times should also represent a superposition of negative exponential residence times. Additional work is necessary to develop a detailed mathematical model of residence times among structural stages, and in particular to examine the role played by resilience to disturbance.

Contrary to expectation, simulations using temporally correlated disturbances produced essentially identical estimates of transition rates and equilibrium structural composition from those using uncorrelated disturbances. The apparent reason is that the Markov chain analysis is based only on transition probabilities between stages, which are determined by the recurrence intervals of disturbance without regard to their spatial configuration. We had also expected that temporally correlated disturbances (as in Fig. 1) would produce a 'jumpier' developmental trajectory for an individual 70 plot replication than independent disturbances (data not shown). However, because of the disturbance patch sizes involved for

northern hardwoods, multiple plot disturbances still only affected four plots at a time and were therefore easily lost in the average when the other 66 plots were included.

2.4.2 Implications of a structural approach to large scale stand dynamics

The structural approach used in classifying stands in this paper leads to different insights compared to a disturbance chronology approach (Frelich and Lorimer, 1991a), despite the use of the same disturbance recurrence intervals. The largest differences in the two analyses relate to how old-growth stands were classified. In the disturbance chronology approach, 50% of the landscape was occupied by non-equilibrium, old-growth, uneven-aged stands with widely varying disturbance histories. Steady-state stands occupied only 4% of the landscape. In the present structural approach, non-equilibrium old-growth stands likewise occupy the largest portion of the landscape (51%), but stands satisfying structural criteria for the steady state occupied 26% (field data) or 27% (simulation with Markov analysis) of the study areas.

Both approaches together enhance our understanding of the underlying dynamics of the system. For example, the disturbance chronology approach isolates the even-aged pathway following catastrophic disturbance (e.g., 18.7% of the study areas were subjected to severe disturbance of >66% canopy removal within the previous 250 years, in agreement with Canham and Loucks 1984). In the structural approach, as in a classical first-order Markov analysis, the pathway of how a stand arrived in a particular structural stage is not directly considered. On the other hand, an advantage of the structural approach is much greater structural uniformity within a developmental stage. For example, the 'old multi-aged' category in the disturbance chronology approach is structurally (and probably functionally) quite heterogeneous, containing stands that would in this study be classified in the early-, mid-, late-transition, and steady-state stages. Both field and simulation evidence indicate that these different structural stages have substantially different patterns of biomass accu-

mulation and net growth, and that these differences are more closely tied to structure than disturbance history (Chapter 1).

Although 74% of the individual stands were in non-equilibrium structural stages, the aggregate study areas (23,000 ha) at the time of the field survey in 1981-84 were close to the equilibrium distribution predicted from the Markov analysis of CANOPY simulations. Earlier studies (Frelich and Lorimer, 1991a,b) concluded that these study areas met criteria for equilibrium landscapes based on a uniform frequency of gap formation by decade from tree-ring analysis, as well as comparisons of field data with long-term simulations using a simpler size-class model (STORM). The current results support these earlier conclusions with an additional line of evidence based on forest structure and using a much more detailed individual-tree model in which competition level dynamically affects recruitment, growth rate, and mortality. Sensitivity analyses suggest that the close structural similarity of the simulated and observed landscape would not likely have been possible if there were inaccuracies in the disturbance recurrence intervals estimated from tree-ring methodology or in CANOPY's predictions of long-term stand development. In addition, the close similarity between the predicted equilibrium landscape distribution and the 1981 landscape provides evidence that the high dominance of old-growth in the field survey was not an artifact of preservation bias in the acquisition of these natural reserves.

The issue of whether individual stands, or even forest landscapes, can reach a steady state under prevailing disturbance regimes has been controversial, and the steady state has also been defined in a variety of ways. The key concept underlying many definitions has been a shifting mosaic of age classes and a stable age distribution (Bormann and Likens, 1979; Shugart, 1984; Johnson and Van Wagner, 1985; Turner et al., 1993). Some investigators have added other criteria such as species composition and ecosystem processes (Odum, 1969; Mori, 2011). Clearly, if criteria for equilibrium status are numerous and stringent, few or no stands would ever qualify and the debate becomes moot (Sprugel, 1991). The dominance of

some ecosystems by recurrent stand-replacing disturbances such as crown fires, landslides, hurricanes, and insects is well documented and is beyond dispute (e.g., Batzer and Popp, 1985; Guariguata, 1990; Heinselman, 1973; Foster and Boose, 1992).

But in some humid temperate regions with late successional hardwood and conifer species, the relatively mild disturbance regimes appear to allow some forests to approach steady-state age or size structure either at the stand or landscape level during multi-century periods of a fairly stable climate (Emborg et al., 2000; Antos and Parish, 2002; Motta et al., 2011). The state diagram of Turner et al. (1993) accommodates these disparate views and illustrates how equilibrium and non-equilibrium states are determined by specific properties of the disturbance regime. In northern hardwoods, the evidence suggests that true equilibrium age distributions seldom occur at the 0.5 ha scale even when criteria allow for several periodic light disturbances (Frelich and Lorimer, 1991a). However, the present study illustrates that stands at that scale can often satisfy structural criteria of a steady state. More importantly, these stands exhibit the expected behavior of a steady state, such as stable size distributions and zero net growth, over a 30-yr monitoring period (Fig. 1 d and Appendix A). This illustrates the point that forest stands may not satisfy strict disturbance history criteria for a steady state and yet may be largely indistinguishable structurally – and perhaps functionally – from stands with a truly balanced age distribution. It may also be the case that in some systems where disturbance intervals are long, stands may seldom reach a steady state but still have a strong tendency to converge toward it (i.e., a 'domain of attraction' *sensu* Holling 1973) At the same time, however, it must be emphasized that this overall system is highly dynamic; steady-state stands are subject to frequent retrogression caused by moderate disturbances and have a mean residence time of less than 30 years.

2.4.3 The influence of light and moderate disturbance on stand structure

The simulations in this study help clarify the relative role of treefall gaps, moderate disturbance, and severe disturbance in shaping the structure and composition of these forests. Our evidence suggests that combinations of light and moderate disturbance, as well as severe disturbance, are both capable of causing structural retrogression to earlier stand stages (Fig. 8) and generating the commonly observed non-equilibrium diameter distributions (Appendix A). Stand-replacing disturbances are too infrequent to have a major impact on the landscape (Canham and Loucks, 1984; Frelich and Lorimer, 1991b; Zhang et al., 1999; Schulte and Mladenoff, 2005). The simulations in the present study explain how a stand with repeated light or moderate disturbances can have a nearly identical size distribution to an even-aged mature or early transition stand and yet have many age classes and a complex disturbance history. There was also little evidence that descending monotonic distributions could spontaneously revert to unimodal or compound distributions under background mortality at the 4 ha scale. Sometimes modest peaks (but not a unimodal distribution) could develop in all-aged stands at the 0.5 ha scale because of stochastic mortality.

Once a stand has developed a unimodal size distribution from the impact of repeated light or moderate disturbances, both field and simulation data suggest that small gap formation has a limited effect on size distributions except over a long period of time. For example, the 30-yr permanent plot data show that in the mature and early transition stands, the bulge in the middle size classes actually became more pronounced over time, and there were net losses of small trees despite periodic gap formation by dying canopy trees (Fig 1a, b). Simulations suggest that it will take about 150 years for a 100-yr old mature stand on above-average site to develop a descending monotonic distribution (late transition stage) under small gap dynamics. And while single- and small multiple- tree gaps can help avoid the local extirpation of the less shade-tolerant species, the simulations suggest that moderate disturbances are necessary to maintain the current abundance of yellow birch and other midtolerant species

(Chapter 3). Cumulatively, this evidence suggest that, despite many studies devoted to the extremes of either small-gap dynamics or stand-replacing disturbance in northern hardwoods (see review in Seymour et al., 2002), moderate disturbances removing 20-50% of the canopy have a more dominant role in shaping the key structural and compositional attributes in protected natural areas in this ecosystem.

Numerous studies have documented that both mild and severe disturbances removing primarily mature canopy trees (e.g., wind, ice storms, disease, drought) often accelerate succession, rather than reversing the trend, due to release of advance regeneration of shade tolerant species (Abrams and Scott, 1989; Webb and Scanga, 2001; Arévalo et al., 2009). Simulation results in this paper provide some support for the hypothesis that mild disturbances might also accelerate structural development in mature and early old-growth stages by reducing the high density of mature trees and increasing structural complexity through gap formation. The clearest evidence was found over a span of 50-100 years during and after several mild disturbances in early-transition stands, which accelerated the trend to late transition compared to undisturbed stands. There was only weak evidence that a single disturbance could have this effect. Too many mild disturbances, on the other hand, caused retrogression to earlier stages.

2.4.4 Resilience of old-growth forest under the natural disturbance regime

Holling (1973) defined 'resilience' as "a measure of the persistence of systems and of their ability to absorb change and disturbance and still maintain the same relationships between populations or state variables". Many authors have focused on the "persistence of systems" aspect of this definition, and examined (for example) the ability of forests to remain forests rather than converting to some other vegetation type (e.g., Gunderson, 2000; Scheffer et al., 2001). However, in management and conservation contexts, the second portion of this definition is of equal or even greater importance and has been the subject of recent attention (e.g.,

Drever et al., 2006; O'Hara and Ramage, 2013). The ability of old-growth forests to absorb disturbance without reverting to an earlier structural stage is of particular interest given the current rarity of old-growth forests in many ecosystems and the vulnerability of existing remnants to destruction by natural disturbances (e.g., Henry and Swan, 1974; Peterson and Pickett, 1995). In the present study, resilience of old growth might be indicated by its mean residence time, its ability to tolerate disturbance without retrogressing to mature or younger stages, and the recovery time to old growth in cases of retrogression.

In our simulations, old-growth stands were able to absorb multiple disturbances of 10-20% canopy removal at 30-yr intervals and still maintain old-growth structure. However, even two 30% removals or a single removal of $\geq$40% was sufficient to cause reversion to earlier stages. Under the historic natural disturbance regime, the resulting average residence time in old-growth forest was only 87 years, which is shorter than the time required (150 years) to reach old growth after stand-replacing disturbance. Residence times, however, were highly variable and ranged up to 500 years in some replicates. Also, residence times in some old-growth stands might in principle be higher in topographically protected locations (e.g., Foster and Boose, 1992). Earlier results in this region indicated no significant difference in the frequency of light, moderate, and severe disturbance on plots in differing topographic position (Frelich and Lorimer, 1991b). However, topographic position might possibly exert a more subtle influence on structural response to disturbance. Hanson and Lorimer (2007) found that even within a single moderate disturbance event, larger individual trees on ridgetops had higher probabilities of windthrow than those in lower topographic positions. Possible reasons could include greater crown exposure of large trees and thinner soils on ridge tops. Since stand structural stage is highly sensitive to the proportions of mature and large trees, disturbances of the same severity in different topographic positions could therefore result in different effects on stand stage. Because of its design, the disturbance module in CANOPY v. 3 doesn't incorporate these more subtle influences. However, the close correspondence of the

Markov predictions of stand stage allocations with the field data suggest that the overall bias resulting from the simplifying assumption about topographic effects may not be very strong.

Previous studies have indicated that resilience of forest stands to windthrow is influenced by the size distribution, with larger trees being more susceptible than medium or small trees (Foster, 1988; Canham et al., 2001). At the same time, there is some evidence that multi-aged stands may be more resilient to windthrow than even-aged stands due to the diversity of size classes and greater windfirmness of trees growing within more open canopies (Mason, 2002; O'Hara and Ramage, 2013). In the present study, simulated patterns of recovery after single episodes of disturbance appeared to differ among different stages of old growth. For steady state stands, disturbances of any severity resulted in some retrogression, whereas early transition stands appeared to be more resilient (Table 2). Early transition stands typically had fewer large trees than steady state and hence may be inherently less susceptible to windthrow. But even after disturbances that removed identical amounts of crown area, steady-state stands took nearly twice as long to recover as the early transition stands (Fig. 8 a,c). Early transition stands also had about twice as much basal area in mature trees compared to steady state (simulated mature tree basal area of 16.9 m^2/ha in early transition compared to 6.7 m^2/ha for steady state). This would likely provide early transition stands with a much larger population of survivors that are not only more wind-resistant but also more resilient and capable of rapidly closing gaps formed during a moderate disturbance.

Compared to development along an even-aged pathway, simulated residence times in the younger stages were much shorter under the historic natural disturbance regime. In a management context, this may provide an incentive to utilize ecological forestry methods (e.g. Franklin et al., 2007), which attempt to silviculturally emulate patterns of natural disturbance to help maintain biodiversity and ecosystem function. When residual trees are

retained, many of which need not be large, stands can recover to steady state within 175 years after a 60% canopy removal, compared to 280 years after a clearcut that leaves no residual trees. These shorter recovery times, in addition to the mild disturbance regimes, are major reasons for the high dominance of these landscapes by old-growth forest.

Literature Cited

Abrams, M. D. and M. L. Scott (1989). Disturbance-mediated accelerated succession in two Michigan forest types. *Forest Science 35*(1), 42–49.

Antos, J. A. and R. Parish (2002). Structure and dynamics of a nearly steady-state subalpine forest in south-central British Columbia, Canada. *Oecologia 130*, 126–135.

Arévalo, J. R., J. K. DeCoster, S. D. McAlister, and M. W. Palmer (2009). Changes in two Minnesota forests during 14 years following catastrophic windthrow. *Journal of Vegetation Science 11*(6), 833–840.

Batzer, H. O. and M. P. Popp (1985). Forest succession following a spruce budworm outbreak in Minnesota. *The Forestry Chronicle 61*(2), 75–80.

Bormann, F. H. and G. E. Likens (1979). *Pattern and process in a forested ecosystem.* New York: Springer-Verlag.

Canham, C. D. and O. L. Loucks (1984). Catastrophic windthrow in the presettlement forests of Wisconsin. *Ecology 65*(3), 803–809.

Canham, C. D., M. J. Papaik, and E. F. Latty (2001). Interspecific variation in susceptibility to windthrow as a function of tree size and storm severity for northern temperate tree species. *Canadian Journal of Forest Research 31*, 1–10.

Caspersen, J. P., M. C. Vanderwel, W. G. Cole, and D. W. Purves (2011). How stand productivity results from size-and competition-dependent growth and mortality. *PloS one 6*(12), e28660.

Choi, J., C. G. Lorimer, and J. M. Vanderwerker (2007). A simulation of the development and restoration of old-growth structural features in northern hardwoods. *Forest Ecology and Management 249*(3), 204–220.

Chokkalingam, U. and A. White (2001). Structure and spatial patterns of trees in old-growth northern hardwood and mixed forests of northern Maine. *Plant Ecology 156*(2), 139–160.

Christy, B. H. (1929). *The book of Huron Mountain.* Chicago, IL, USA: Huron Mountain Club.

Coffman, M. S. (1984). *Field guide: Habitat classification system for upper peninsula of Michigan and northeast Wisconsin.* Houghton, MI, USA: CROFS, School of Forestry and Wood Products, Michigan Technological University.

Crow, T. R., C. H. Tubbs, R. D. Jacobs, and R. R. Oberg (1981). Stocking and structure for maximum growth in sugar maple selection stands. Research Paper NC-199, USDA Forest Service.

D'Amato, A. W., D. A. Orwig, and D. R. Foster (2009). The influence of successional processes and disturbance on the structure of *Tsuga canadensis* forests. *Ecological Applications 18*, 1182–1199.

Drever, C. R., G. Peterson, C. Messier, Y. Bergeron, and M. Flannigan (2006). Can forest management based on natural disturbance maintain ecological resilience? *Canadian Journal of Forest Research 36*, 2285–2299.

Emborg, J., M. Christensen, and J. Heilmann-Clausen (2000). The structural dynamics of Suserup Skov, a near-natural temperate deciduous forest in Denmark. *Forest Ecology and Management 126*, 173–189.

Erdmann, G. G. and R. R. Oberg (1973). Fifteen-year results from six cutting methods in second-growth northern hardwoods. Research Paper NC-100, USDA Forest Service.

Fassnacht, K. S. and S. T. Gower (1997). Interrelationships among the edaphic and stand characteristics, leaf area index, and aboveground net primary production of upland forest ecosystems in north central Wisconsin. *Canadian Journal of Forest Research 27*, 1058–1607.

Foster, D. R. (1988). Species and stand response to catastrophic wind in central New England, USA. *The Journal of Ecology 76*, 135–151.

Foster, D. R. and E. R. Boose (1992). Patterns of forest damage resulting from catastrophic wind in central New England, USA. *Journal of Ecology 80*, 79–98.

Franklin, J. F., R. J. Mitchell, and B. Palik (2007). Natural disturbance and stand development principles for ecological forestry. General Technical Report NRS-19, USDA Forest Service.

Franklin, J. F., T. A. Spies, R. van Pelt, A. B. Carey, D. A. Thornburgh, D. R. Berg, D. B. Lindenmayer, M. E. Harmon, W. S. Keeton, D. C. Shaw, K. Bible, and J. Chen (2002). Disturbance and structural development of natural forest ecosystems with silvicultural implications, using Douglas-fir forests as an example. *Forest Ecology and Management 155*, 399–423.

Fraver, S., A. S. White, and R. S. Seymour (2009). Natural disturbance in an old-growth landscape of northern Maine, USA. *Journal of Ecology 97*(2), 289–298.

Frelich, L. E. and L. J. Graumlich (1994). Age-class distribution and spatial patterns in an old-growth hemlock-hardwood forest. *Canadian Journal of Forest Research 24*(9), 1939–1947.

Frelich, L. E. and C. G. Lorimer (1991a). A simulation of landscape-level stand dynamics in the northern hardwood region. *Journal of Ecology 79*, 145–164.

Frelich, L. E. and C. G. Lorimer (1991b). Natural disturbance regimes in hemlock-hardwood forests of the upper Great Lakes region. *Ecological Monographs 61*(2), 145–164.

Goodburn, J. M. and C. G. Lorimer (1999). Population structure in old-growth and managed northern hardwoods: An examination of the balanced diameter distribution concept. *Forest Ecology and Management 118*, 11–29.

Guariguata, M. R. (1990). Landslide disturbance and forest regeneration in the upper Luquillo Mountains of Puerto Rico. *Journal of Ecology 78*, 814–832.

Gunderson, L. H. (2000). Ecological resilience – in theory and application. *Annual Review of Ecology and Systematics 31*, 425–439.

Halpin, C. R. (2009). Simulated long-term effects of group selection on production, efficiency, composition, and structure in northern hardwood and hemlock-hardwood forests. Master's thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J. (2009). *Emulating natural disturbance dynamics in northern hardwood forests: Long-term effects on species composition, forest structure, and yield.* Ph. D. thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J. and C. G. Lorimer (2007). Forest structure and light regimes following moderate windstorms: Implications for multi-cohort management. *Ecological Applications 17*, 1325–1340.

Hanson, J. J., C. G. Lorimer, and C. R. Halpin (2011). Predicting long-term sapling dynamics and canopy recruitment in northern hardwood forests. *Canadian Journal of Forest Research 41*, 903–919.

Hanson, J. J., C. G. Lorimer, C. R. Halpin, and B. J. Palik (2012). Ecological forestry in an uneven-aged, late-successional forest: Simulated effects of contrasting treatments on structure and yield. *Forest Ecology and Management 270*, 94–107.

Heinselman, M. L. (1973). Fire in the virgin forests of the Boundary Waters Canoe Area, Minnesota. *Quaternary Research 3*(3), 329–382.

Henry, J. D. and J. M. A. Swan (1974). Reconstructing forest history from live and dead plant material – an approach to the study of forest succession in southwest New Hampshire. *Ecology 55*(4), 772–783.

Hett, J. M. and O. L. Loucks (1971). Sugar maple (*Acer saccharum* marsh.) seedling mortality. *The Journal of Ecology*, 507–520.

Holling, C. S. (1973). Resilience and stability of ecological systems. *Annual Review of Ecology and Systematics 4*, 1–23.

Johnson, E. A. and C. E. Van Wagner (1985). The theory and use of two fire history models. *Canadian Journal of Forest Research 15*(1), 214–220.

Kotar, J., T. L. Burger, and J. A. Kovach (2002). *A guide to forest communities and habitat types of northern Wisconsin.* Madison, WI: Department of Forest Ecology and Management, University of Wisconsin – Madison.

Mason, W. L. (2002). Are irregular stands more windfirm? *Forestry 75*, 347–355.

Mori, A. S. (2011). Ecosystem management based on natural disturbances: Hierarchical context and non-equilibrium paradigm. *Journal of Applied Ecology 48*, 280–292.

Motta, R., R. Berretti, D. Castagneri, V. Dukić, M. Garbarino, Z. Govedar, E. Lingua, Z. Maunaga, and F. Meloni (2011). Toward a definition of the range of variability of central european mixed *FagusâĂŞAbiesâĂŞPicea* forests: the nearly steady-state forest of Lom (Bosnia and Herzegovina). *Canadian Journal of Forest Research 41*, 1871–1884.

Niese, J. N. and T. F. Strong (1992). Economic and tree diversity trade-offs in managed northern hardwoods. *Canadian Journal of Forest Research 22*(11), 1807–1813.

Nyland, R. D. (2007). *Silviculture: Concepts and applications.* Long Grove, IL, USA: Waveland Press.

Odum, E. P. (1969). The strategy of ecosystem development: An understanding of ecological succession provides a basis for resolving man's conflict with nature. *Science 164*, 262–270.

O'Hara, K. L. and B. S. Ramage (2013). Silviculture in an uncertain world: Utilizing multi-aged management systems to integrate disturbance. *Forestry 86*, 401–410.

Oliver, C. D. and B. C. Larson (1990). *Forest stand dynamics.* New York: McGraw-Hill.

Peterson, C. J. and S. T. A. Pickett (1995). Forest reorganization: a case study in an old-growth forest catastrophic blowdown. *Ecology*, 763–774.

Piovesan, G., A. DiFilippo, A. Alessandrini, F. Biondi, and B. Schirone (2005). Structure, dynamics and dendroecology of an old-growth *Fagus* forest in the Apennies. *Journal of Vegetation Science 16*, 13–28.

Rafferty, M. and R. Sprague (2001). *Porcupine Mountains companion: Inside Michigan's largest state park.* White Pine, MI, USA: Neqauket Natural History Associates.

Scheffer, M., S. Carpenter, J. A. Foley, C. Folke, and B. Walker (2001). Catastrophic shifts in ecosystems. *Nature 413*, 591–596.

Schulte, L. A. and D. J. Mladenoff (2005). Severe wind and fire regimes in northern forests: Historical variability at the regional scale. *Ecology 86*(2), 431–445.

Seymour, R. S., A. S. White, and P. G. deMaynadier (2002). Natural disturbance regimes in northeastern North America – evaluating silvicultural systems using natural scales and frequencies. *Forest Ecology and Management 155*, 357–367.

Shugart, H. H. (1984). *A theory of forest dynamics: The ecological implications of forest succession models*. New York: Springer-Verlag.

Smith, D. M., B. C. Larson, M. J. Kelty, and P. M. Ashton (1997). *The practice of silviculture: Applied forest ecology*. New York: John Wiley and Sons.

Sprugel, D. G. (1991). Disturbance, equilibrium, and environmental variability: What is 'natural' vegetation in a changing environment? *Biological Conservation 58*, 1–18.

Turner, M. G., W. H. Romme, R. H. Gardner, R. V. O'Neill, and T. K. Kratz (1993). A revised concept of landscape equilibrium: Disturbance and stability on scaled landscapes. *Landscape Ecology 8*(3), 213–227.

Van Wagner, C. E. (1978). Age-class distribution and the forest fire cycle. *Canadian Journal of Forest Research 8*(2), 220–227.

Vanderwel, M. C., D. A. Coomes, and D. W. Purves (2013). Quantifying variation in forest disturbance, and its effects on aboveground biomass dynamics, across the eastern United States. *Global Change Biology 19*, 1504–1517.

Webb, S. L. and S. E. Scanga (2001). Windstorm disturbance without patch dynamics: Twelve years of change in a Minnesota forest. *Ecology 82*, 893–897.

Woods, K. D. (1979). Reciprocal replacement and the maintenance of codominance in a beech-maple forest. *Oikos*, 31–39.

Zenner, E. K. (2005). Development of tree size distributions in Douglas-fir forests under differing disturbance regimes. *Ecological Applications 15*(2), 701–714.

Zhang, Q., K. S. Pregitzer, and D. D. Reed (1999). Catastrophic disturbance in the pre-settlement forests of the upper peninsula of Michigan. *Canadian Journal of Forest Research 29*(1), 106–114.

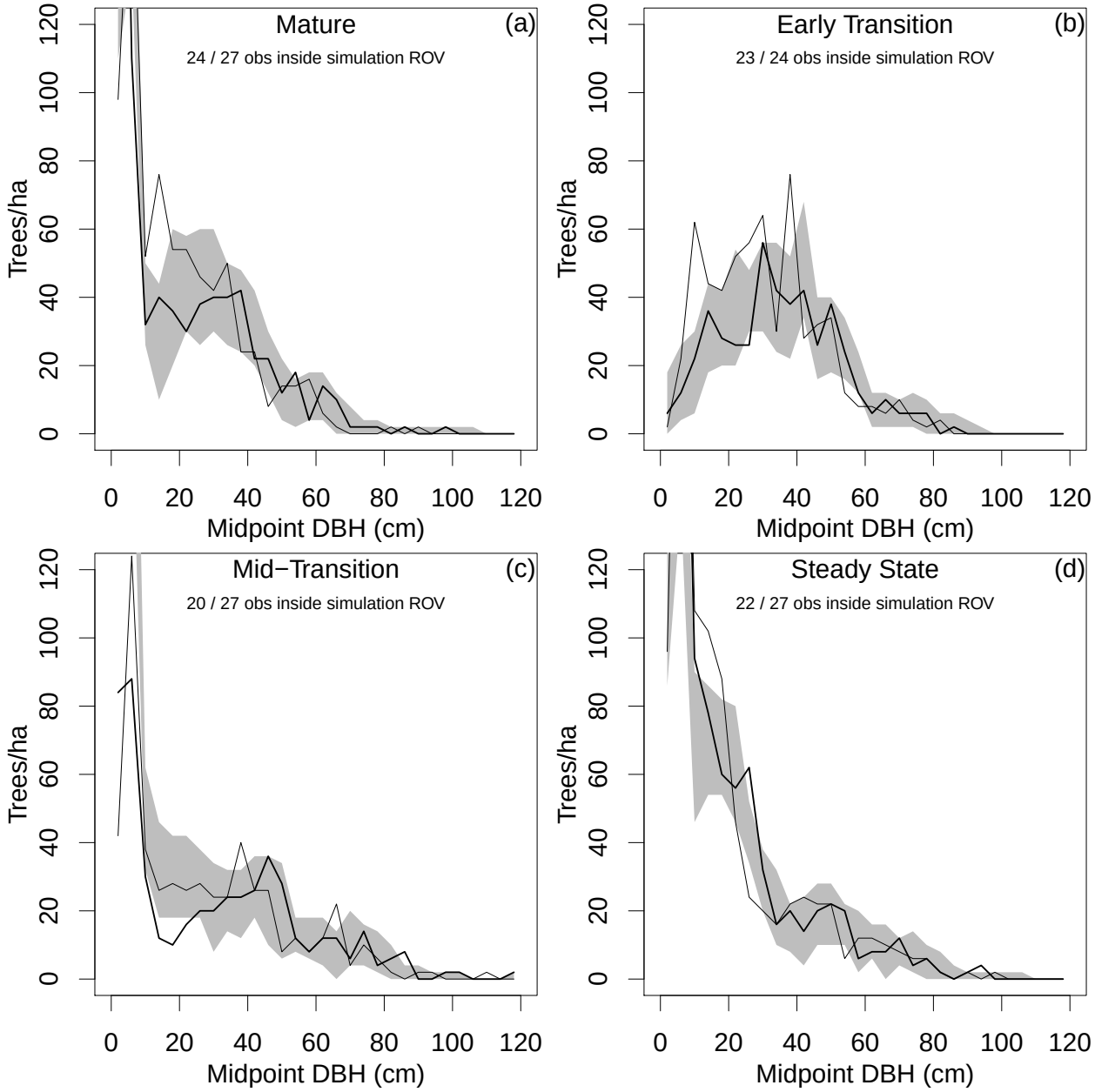

Figure 1: Thirty-year change in diameter distribution from 0.5 ha permanent plots representing four mature and old growth stages, compared to CANOPY predictions of these changes. Thick lines give 2011 measurement and thin lines 1981 measurements. grey bands give ranges of variation (ROV) over 20 replicated CANOPY simulations.

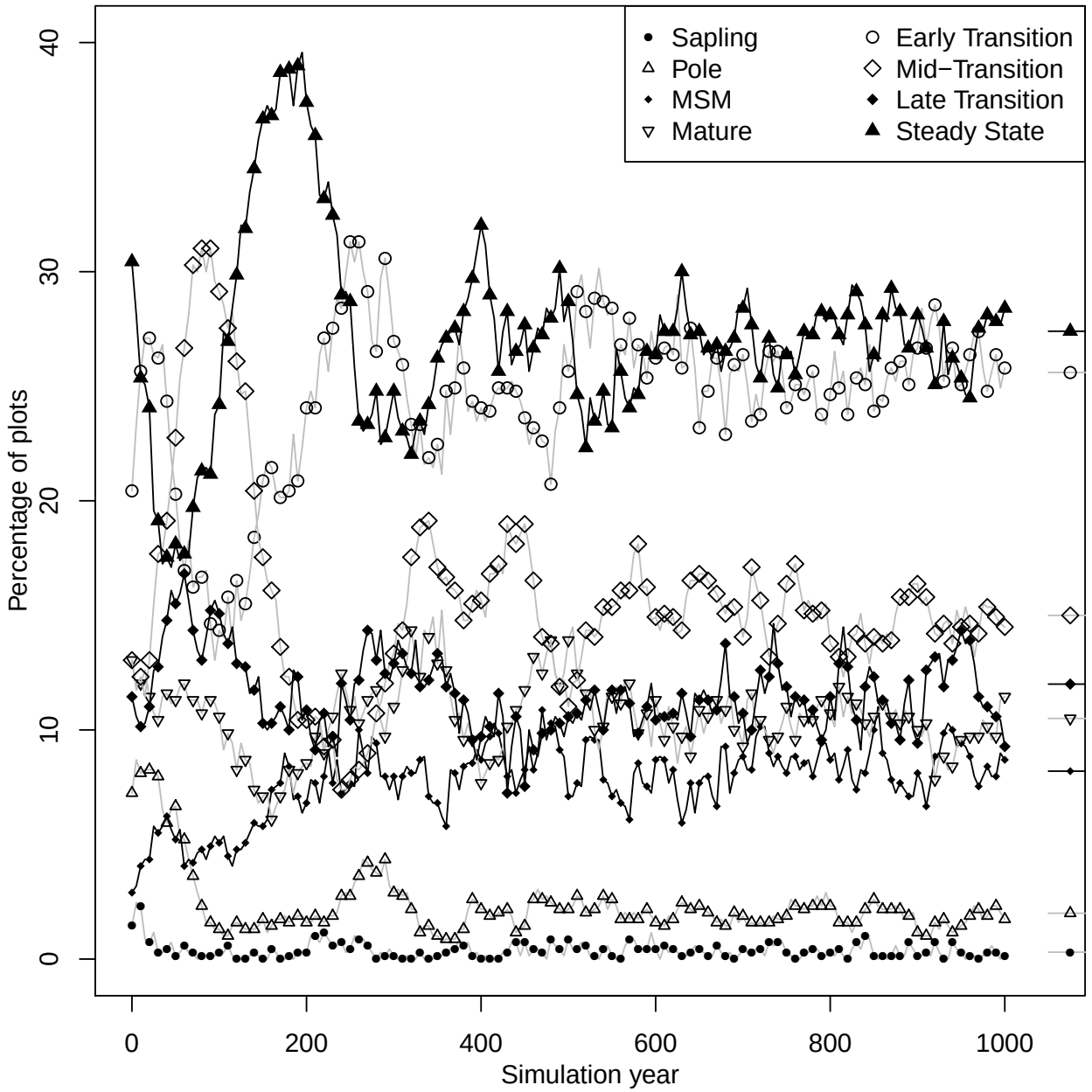

Figure 2: Simulated changes over time in the percentage of the 70 plot population (0.5 ha plots, 10 reps per plot, 1000 years per rep) in each structural stage under the historic natural disturbance regime. Lines to the right of the trajectories show the equilibrium state predicted by the second order Markov chain analysis. MSM: Mature-sapling mosaic.

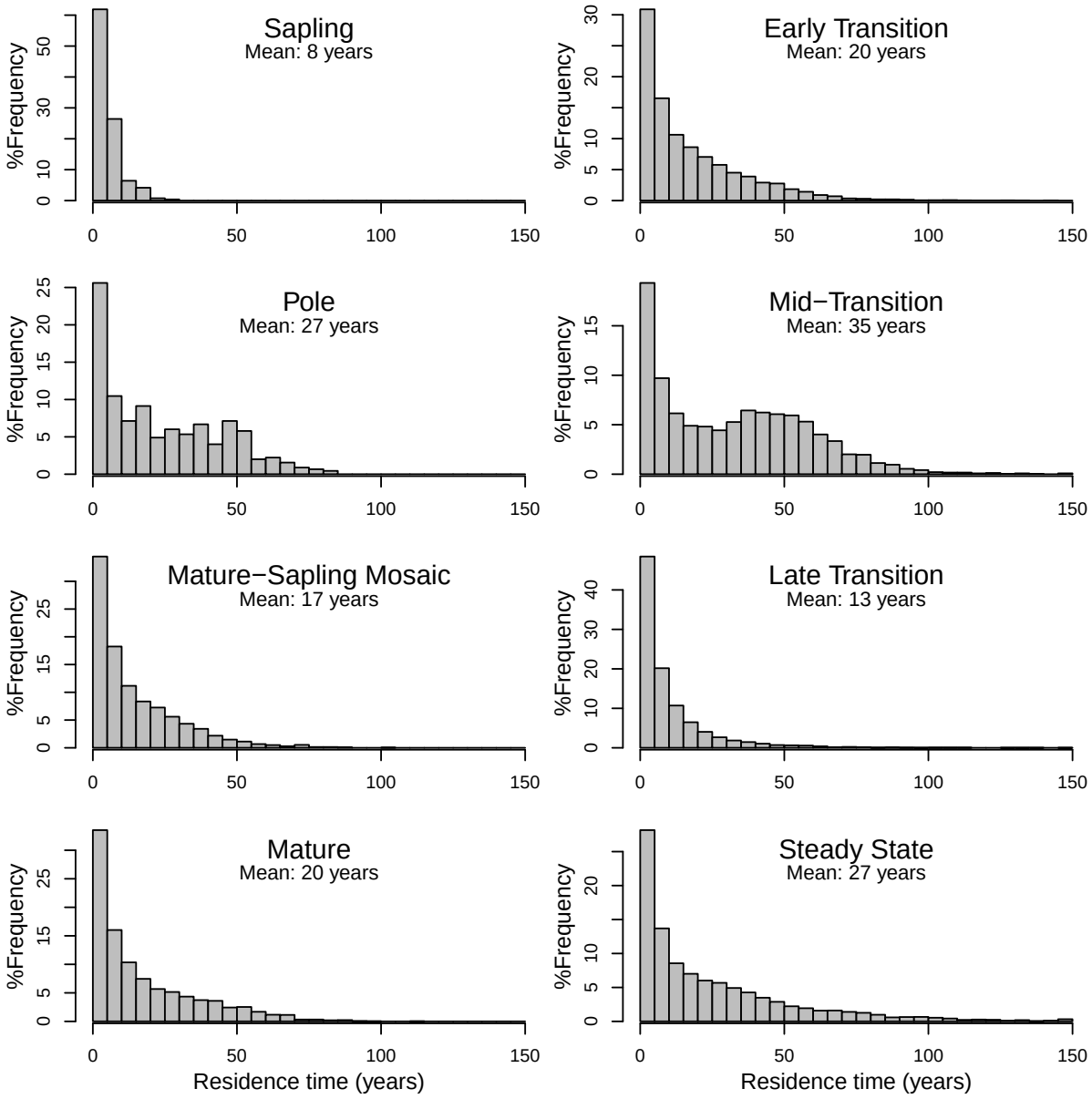

Figure 3: Frequency distributions of residence times in each structural stage of the 70 plot population (CANOPY simulations, 0.5 ha plots, 10 reps per plot, 1000 years per rep) under the historic natural disturbance regime. The first 100 years of each simulation were excluded to reduce correlation between replicates of the same plot.

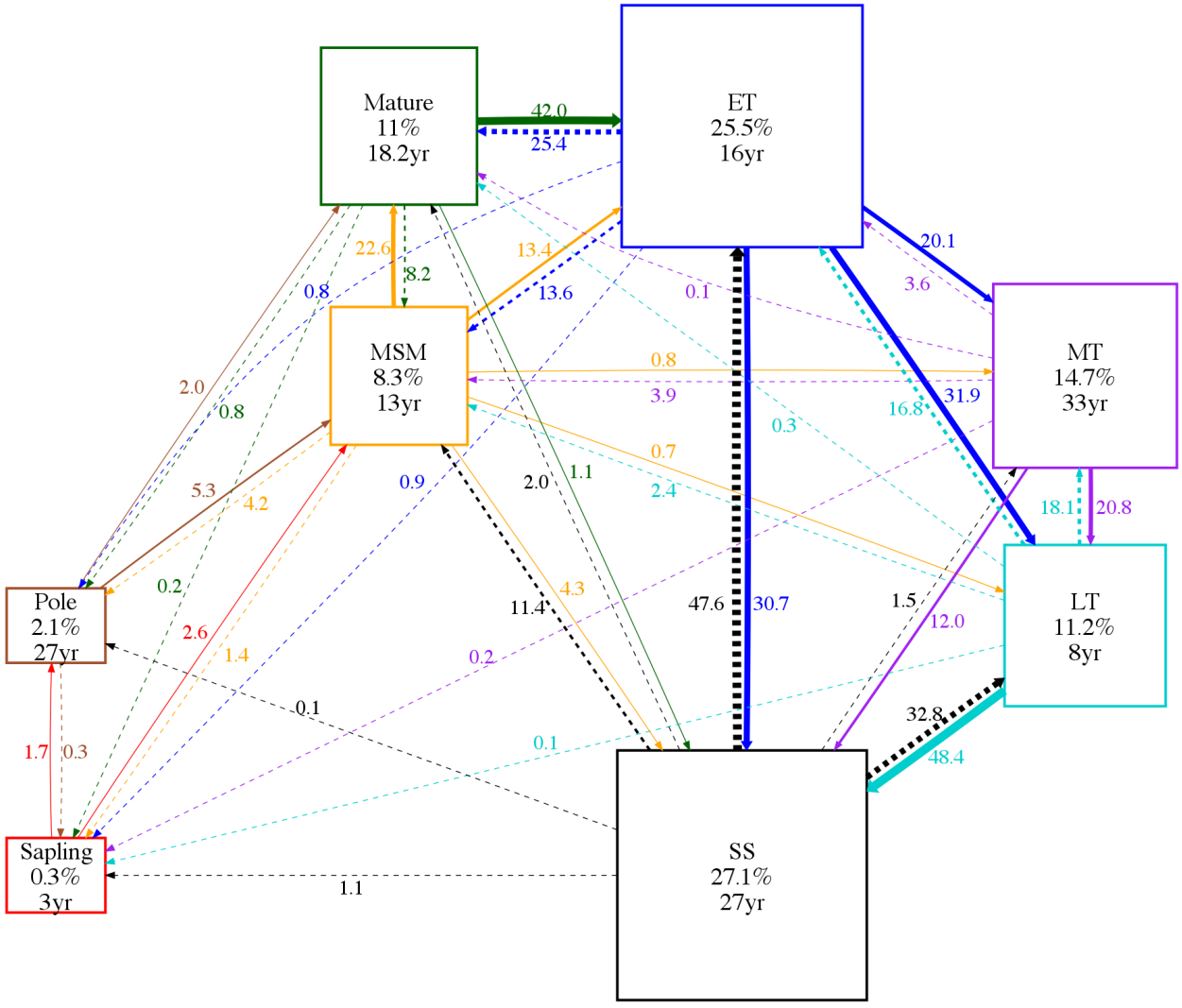

Figure 4: State transition diagram for the historic natural disturbance regime on 0.5 ha plots based on a second order Markov chain analysis that computed the equilibrium structural composition. Numbers in the boxes represent the percentage of the Markov population in a given stage at equilibrium and the average residence time for a plot in that stage. Labeling on the arrows give the percentage of the Markov population transferred per century between stages, and arrow widths are scaled according to these percentages. Transition probabilities between stages were taken from CANOPY simulations in Fig. 3. MSM: Mature-sapling mosaic, ET: Early transition, MT: Mid-transition, LT: Late transition, SS: Steady state.

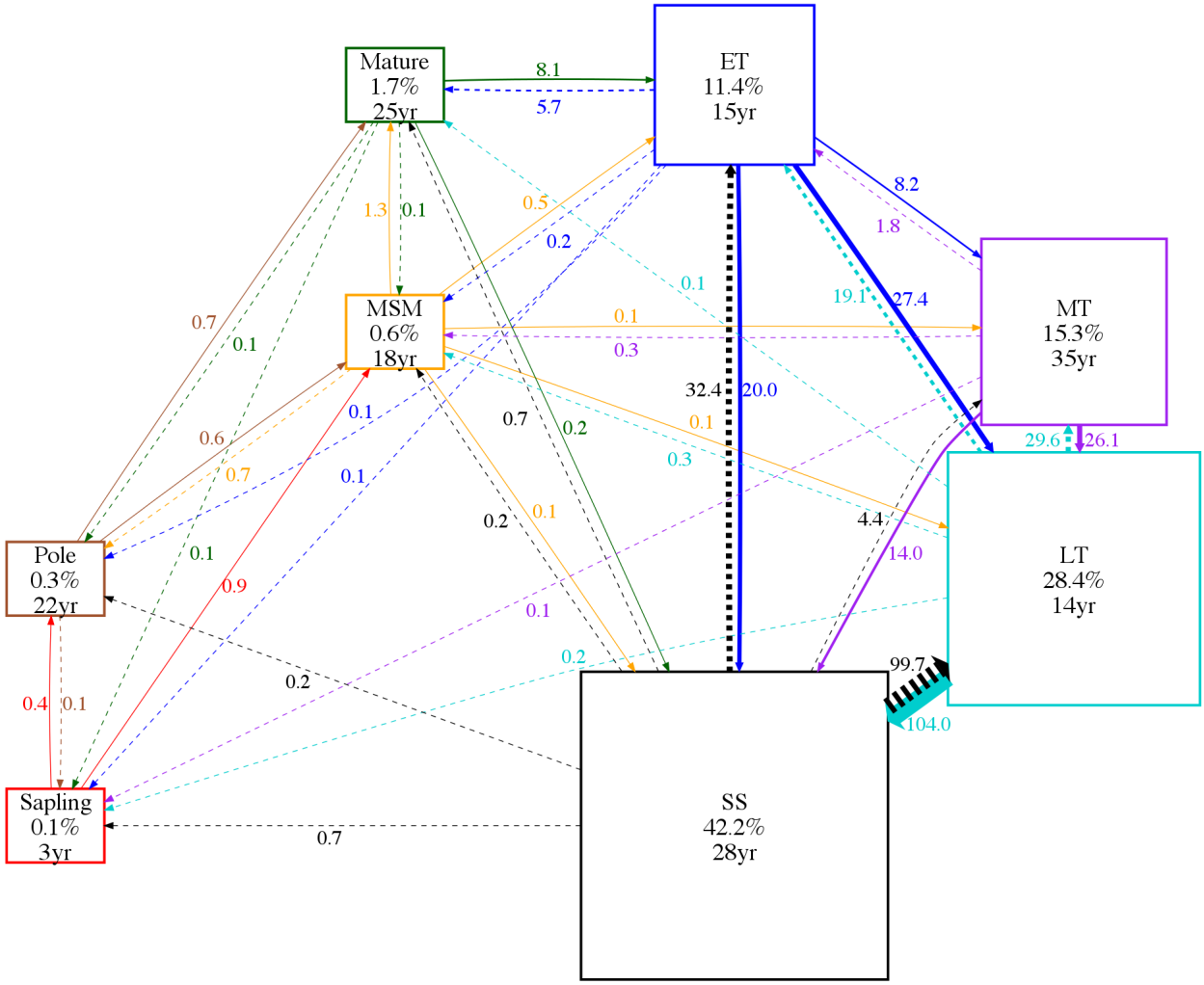

Figure 5: State transition diagram for a dichotomous disturbance regime on 0.5 ha plots that included only background mortality in most decades and removal of 70% canopy area at mean intervals of 3734 years (from Frelich and Lorimer, 1991b) . Numbers in the boxes represent the percentage of the Markov population in a given stage at equilibrium and the average residence time for a plot in that stage. Labeling on the arrows give the percentage of the population transferred per century between stages, and arrow widths are scaled according to these percentages. Abbreviations as in Fig. 4.

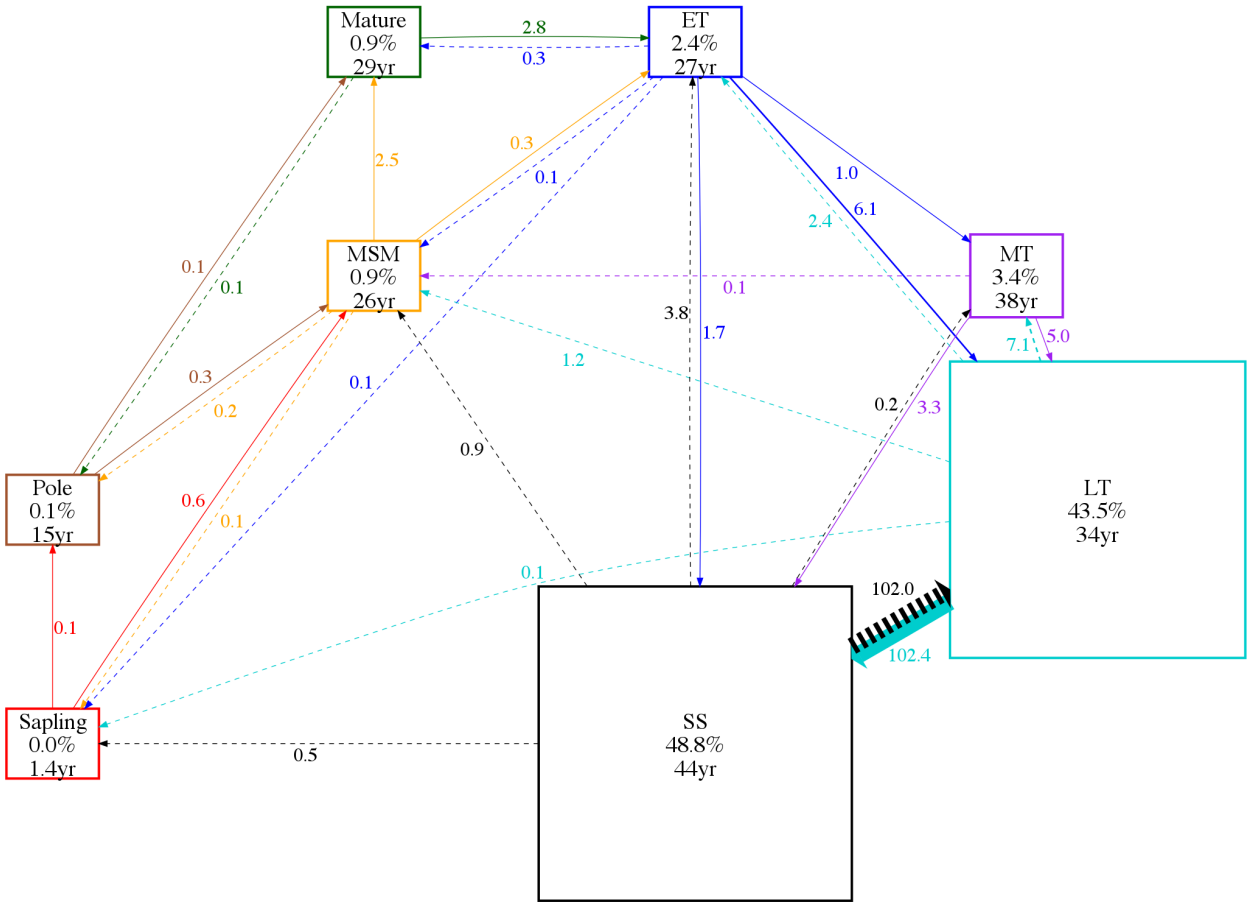

Figure 6: State transition diagram for 4 hectare simulations of a dichotomous disturbance regime that included only background mortality in most decades and severe events removing 70% canopy area at mean recurrence interval of 3734 years. Numbers in the boxes represent the percentage of the Markov population in a given stage at equilibrium and the average residence time for a plot in that stage. Labeling on the arrows give the percentage of the population transferred per century between stages, and arrow widths are scaled according to these percentages. Abbreviations as in Fig. 4.

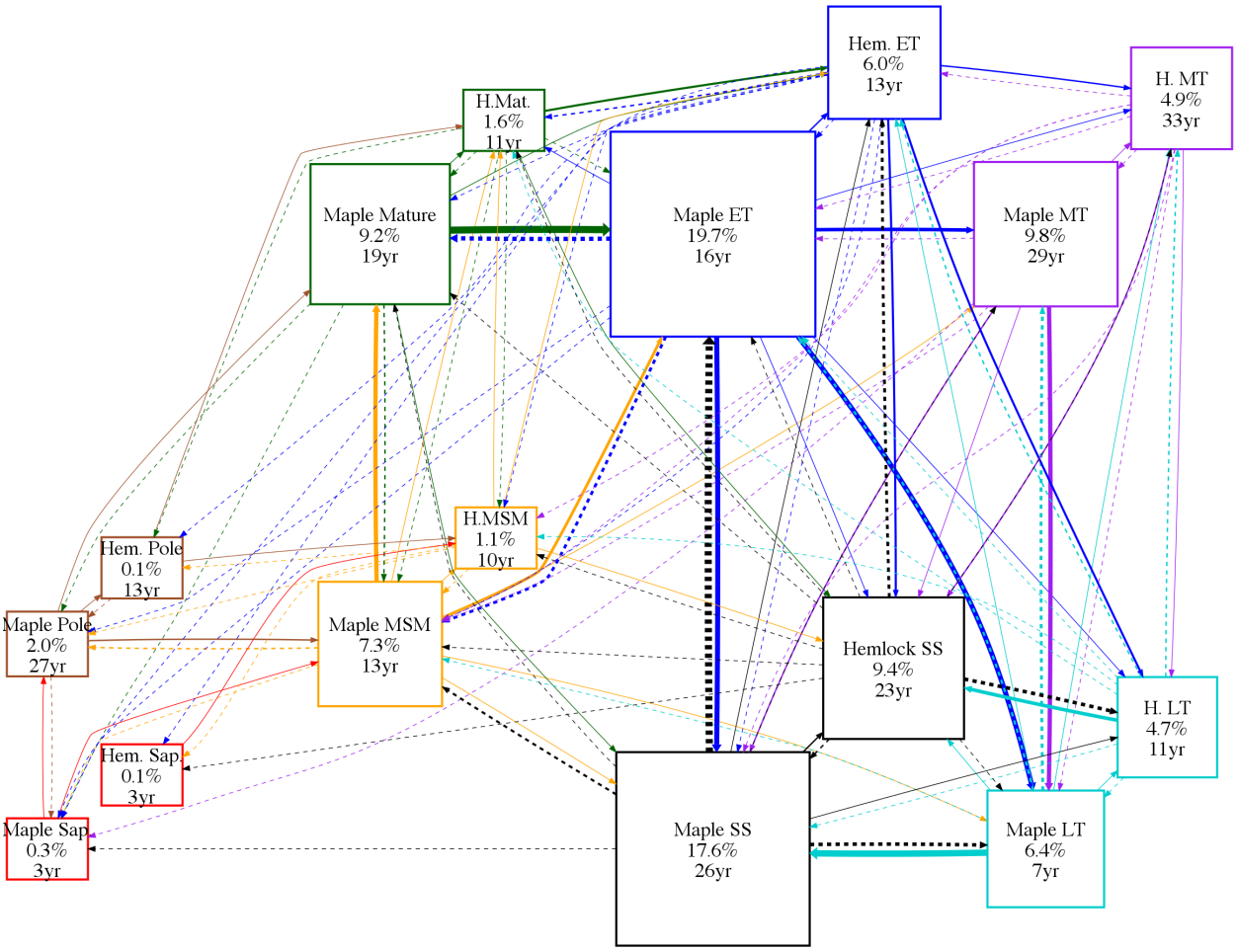

Figure 7: State transition diagram for the historic natural disturbance regime, analyzed separately for maple- and hemlock- dominated stands (defined as $\geq 35\%$ hemlock basal area). Numbers in the boxes represent the percentage of the Markov population in a given stage at equilibrium and the average residence time for a plot in that stage. Arrow widths are scaled according to the percent transferred per century between stages. Transition probabilities taken from simulations in Fig. 3. Transition rates among stages are shown in Table 1. Abbreviations as in Fig. 4.

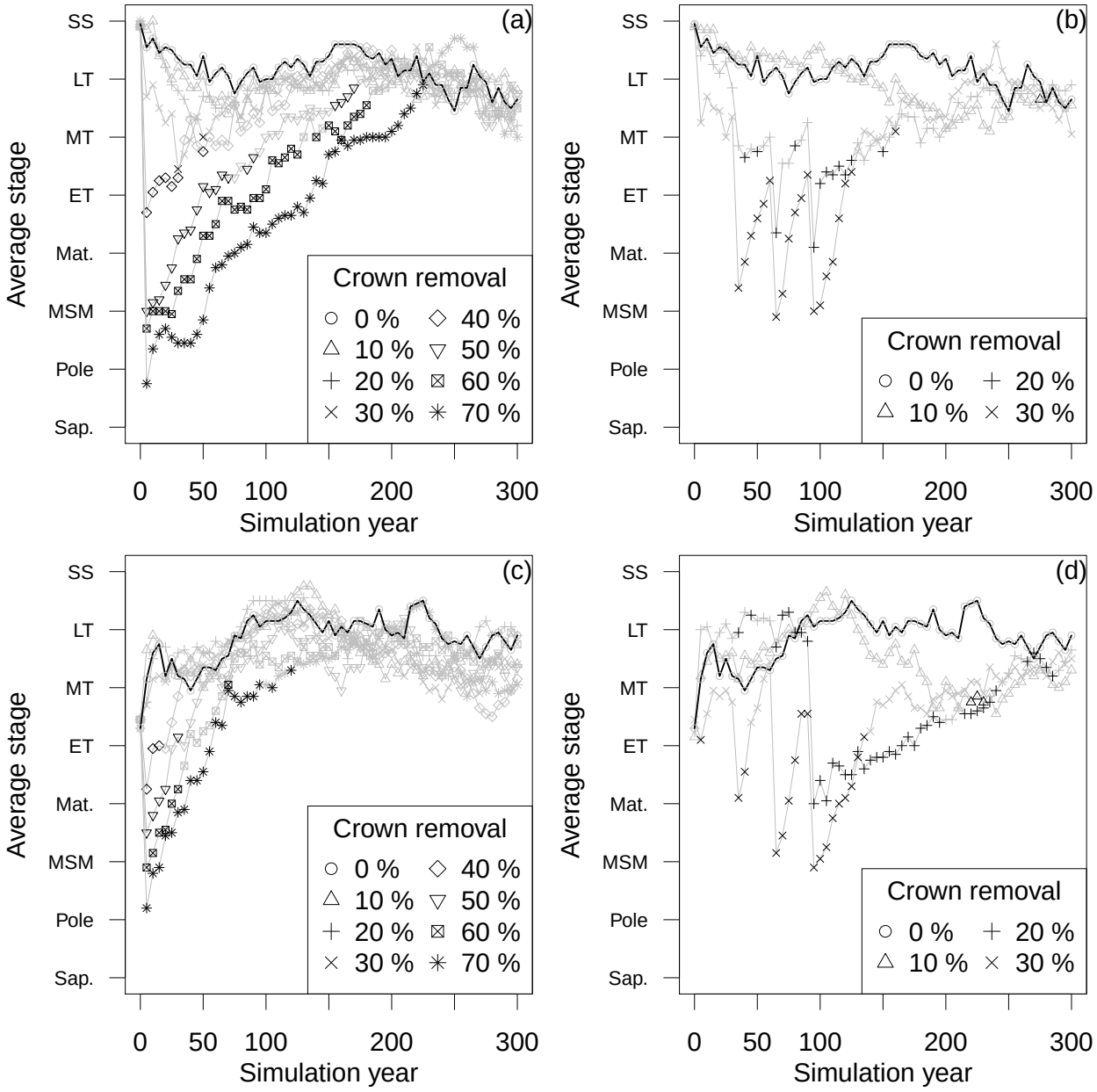

Figure 8: Simulated structural recovery of old-growth stands from either single or repeated disturbances compared to development under background mortality. Thick line shows trajectory under background mortality. Black symbols denote $p<0.05$ in a chi-squared test between the distribution of stages with disturbance vs background mortality. (a) single disturbance in steady state stands. (b) Repeated disturbances (year 1, 31, 61, 91) in steady state stands. (c) Single disturbance in early transition stands. (d) repeated disturbance in early transition stands. Abbreviations as in Fig. 4.

Table 1: Percentage of the landscape transferred between stages by century from the Markov model in Fig. 6. Rows give the initial stages, columns the destination. Empty cells represent transitions never observed.

		Maple								Hemlock							
		Sap	Pole	MSM	Mat	ET	MT	LT	SS	Sap	Pole	MSM	Mat	ET	MT	LT	SS
Maple	Sap		1.6	2.1							<0.1	<0.1					
	Pole	0.3		5.0	1.8						0.2	<0.1					
	MSM	1.2	3.8		19.4	11.3	0.7	0.7	3.4	<0.1		0.6	0.2	<0.1			<0.1
	Mat	0.2	0.6	7.0		33.3			0.7			<0.1	1.6	0.5			<0.1
	LA	0.7	0.6	11.2	19.9		15.0	21.7	21.5			<0.1	0.3	4.5	0.3	0.2	0.3
	ET	0.1		2.8		2.1		18.2	4.6			<0.1		<0.1	2.0	0.1	0.2
	LT	0.1		1.7	<0.1	10.4	12.0		30.4			<0.1		0.1	0.2	1.0	0.4
	SS	0.8	<0.1	8.6	1.2	35.0	0.3	14.8		<0.1		0.1	<0.1	0.5	<0.1	0.3	3.9
Hemlock	Sap	<0.1	<0.1								0.1	0.5					
	Pole	<0.1	0.2							<0.1		0.4	0.2				
	MSM	<0.1	0.1	0.9	<0.1	<0.1				0.1	0.4		3.2	2.2	0.1	<0.1	0.9
	Mat		<0.1	0.3	0.9	0.1					0.1	0.9		8.4			0.3
	LA	0.1	<0.1	0.8	0.7	2.9	<0.1	<0.1	0.3	0.1	0.1	1.7	4.8		4.8	9.8	8.8
	ET			0.3	<0.1	0.2	1.9	0.2	0.3	<0.1		0.8		1.2		2.6	6.8
	LT			0.1	<0.1	0.3	<0.1	0.5	0.5	<0.1		0.6	0.2	5.9	5.8		16.8
	SS	0.1	<0.1	0.7	0.2	0.8	<0.1	0.2	3.9	0.2	<0.1	2.2	0.6	11.6	1.1	16.7	

Table 2: Short-term response of steady state and early transition stands five years after disturbances of various severity during simulations of the historic natural disturbance regime. Results were constructed by tabulating structural stage following stochastic disturbances from the simulations in Fig. 2. Abbreviations as in Fig. 4. n: number of 5-yr simulation intervals.

		Percentage of stands 5 years after disturbance								n
		Sap	Pole	MSM	Mat	ET	MT	LT	SS	
Steady state	Background Mortality	<0.1	0.0	0.5	0.2	8.1	0.3	6.4	84.3	32522
	10% disturbance	0.0	0.0	3.4	1.1	16.0	0.0	0.8	78.8	1139
	20% disturbance	0.0	0.0	19.2	4.0	23.6	0.2	0.2	52.8	598
	30% disturbance	0.0	0.0	66.5	3.4	13.2	0.0	0.0	16.9	266
	40% disturbance	1.9	0.0	91.1	0.6	1.9	0.0	0.0	4.5	157
	50% disturbance	4.9	0.0	95.1	0.0	0.0	0.0	0.0	0.0	61
	60% disturbance	52.5	5.0	42.5	0.0	0.0	0.0	0.0	0.0	40
	70% disturbance	67.7	9.7	22.6	0.0	0.0	0.0	0.0	0.0	31
Early transition	Background Mortality	<0.1	<0.1	0.9	4.7	77.8	4.1	6.5	6.0	28965
	10% disturbance	0.0	0.1	7.1	8.5	68.5	3.6	4.9	7.4	1100
	20% disturbance	0.0	0.4	29.8	11.4	43.3	1.1	4.1	9.9	543
	30% disturbance	0.4	1.2	56.3	10.7	24.5	1.5	0.8	4.6	261
	40% disturbance	2.1	5.5	68.5	6.2	13.0	0.7	0.0	4.1	146
	50% disturbance	8.1	14.5	75.8	0.0	1.6	0.0	0.0	0.0	62
	60% disturbance	19.4	22.6	58.1	0.0	0.0	0.0	0.0	0.0	31
	70% disturbance	64.3	9.5	26.2	0.0	0.0	0.0	0.0	0.0	42

CHAPTER 3

A demographic approach to evaluating tree population sustainability

Abstract

Currently, there are no widely accepted quantitative criteria to assess the demographic sustainability of tree populations. Such criteria would be useful in forest management as climate change and a growing complex of invasive pests are likely to drive forests outside their historic range of variability. In this paper, a quantitative index of tree population sustainability was defined based on the ratio of future simulated basal area to current basal area. This index assesses the proportion of existing species or stand basal area that is sustained over a specified time period and under specified environmental conditions. Simulations were conducted to compute the index under several disturbance regimes for seventy northern hardwood stands having a range of initial size distributions. These simulations were conducted using CANOPY, an empirical, individual-tree model that provides projections of density-dependent recruitment, growth, and mortality. Only steeply descending monotonic size distributions were indicated to be moderately or highly sustainable (final BA / initial BA $\geq$ 0.7) and stable in the sense that their size distributions maintained the same form. Regression analyses indicated that demographic variables (e.g., initial basal area, quadratic mean diameter, etc) were often predictive of demographic sustainability, both overall and for individual species. When the population density and species composition of trees 2-6 cm

dbh were maintained at current levels, all species examined had demographic sustainability index values $\geq$0.95 except for red maple (*Acer rubrum*). A series of simulation experiments were conducted which manipulated the number of 2-6 cm dbh saplings in uneven-aged, old growth stands and forecast the long-term demographic sustainability. A minimum sapling density of 300 trees per hectare (2-6 cm dbh) was required to sustain the initial basal area, but increasing the density of 2-6 cm trees above 300 did not result in increased basal area because of coincident increases in mortality. Applying fixed shallow slopes to the size distribution of the 2-22 cm size classes (q ratio < 2.3) reduced the population of small trees below that required to sustain the overstory. A variable slope was found to necessary in the understory to maintain the existing overstory of mature and old-growth stands. These results suggest that current uneven-aged management guidelines specifying residual curves with a constant q-ratio would not be sustainable if small trees were harvested to the specified residual density.

3.1 Introduction

In plant ecology, there is no close analog to the concept of 'minimum viable populations' in animals. Extending this concept to plants has proven difficult because of the increased role that spatial processes play in non-motile populations (Reed et al., 2002). Nevertheless, several investigators have applied the population viability analysis framework from animal ecology to understory plants (cf. McGraw and Furedi, 2005; Obioh and Isichei, 2007). In this framework, Monte Carlo simulation is used, generally with a probabilistic life-table model, to assess a population's probability of persistence (Reed et al. 2002), oftentimes including the influence of stochastic disturbance. Various persistence criteria have been used. But because population viability analysis is most often applied to threatened or endangered species, persistence criteria are often satisfied if the target population does not go extinct, and little emphasis given to maintaining present levels of abundance. To my knowledge,

there have not been any formal population viability analyses conducted to date for tree species.

The common *ad hoc* criterion used to judge the demographic sustainability of tree populations is that size distributions have a descending monotonic form, so that higher numbers of small trees can compensate for mortality (Smith et al., 1997; Nyland, 2007). In the forestry literature, the steepness of slope has been quantified by the 'q-ratio', defined as the ratio of the number of trees in successive diameter classes (Guldin, 1991). But a descending monotonic form is not a definitive criterion of demographic sustainability because many size distributions of this form could be unsustainable if mortality rates are higher than recruitment rates, and neither of these rates can be determined without long-term data. And under certain conditions, a steeply descending monotonic form might be unnecessary for sustainability. For example, gap-phase species might not need to maintain large sapling populations to compensate for mortality if sapling growth rates within gaps are high and mortality rates are low, suggesting that a relatively 'flat' size distribution may be sustainable. Likewise, a unimodal size distribution could conceivably have an underlying stable age distribution in some cases if fast growth in some smaller trees permits them to escape the high mortality rates in the sapling classes, but low mortality and declining growth with increasing diameter causes trees to 'pile up' in the larger size classes.

There are a number of important North American species for which existing means of assessing sustainability do not provide a definitive assessment. Coast redwood (*Sequoia sempervirens*), a canopy dominant species over a 500,000 acres of coastal forests in California and Oregon, often has relatively flat size and age distributions in which recruitment rates seem to be low (Lorimer et al., 2009). Yellow birch (*Betula alleghaniensis*), a moderately shade tolerant gap-phase species in northern hardwood forests, oftentimes also has flat, shallow distributions with few young trees (Tyrrell and Crow, 1994). It is not entirely clear if either of these species is maintaining sustainable size distributions in such stands. In other

cases, a clear change in the disturbance regime can affect population sustainability. Eastern hemlock (*Tsuga canadensis*), which was the most abundant tree in northern Wisconsin at the time of the 19$^{\text{th}}$ century land survey (Fahey et al., 2012), is now in region-wide decline, which is at least partially attributable to recruitment limitation caused by chronic deer over-browsing (Rooney et al., 2000; Witt and Webster, 2010).

The topic of population sustainability is likely to become increasingly important in this century because of large numbers of anthropogenically induced stressors such as climate change, exotic plants, invasive earthworms, and exotic insects and diseases. For example, there is evidence that invasive earthworms may be hindering the establishment of tree seedlings of some of the most common and economically valuable tree species, such as sugar maple (*Acer saccharum*; Holdsworth et al., 2007). The future population viability of even major tree species is uncertain because of introduced insects and diseases such as hemlock woolly adelgid (*Adelges tsugae*) and Asian longhorned beetle (*Anoplophora glabripennis*) (Orwig et al., 2002; Dodds and Orwig, 2011). Under these conditions, we need much more rigorous criteria for judging the long-term population sustainability of various size distributions.

The main objective of this paper is to develop quantitative guidelines for interpreting the long-term demographic sustainability of tree species at scales from small stands to landscapes. The specific hypotheses of interest were:

- Slopes of minimally sustainable size distributions will vary by species and size class, reflecting systematic differences in mortality rates. Distributions with shallow slopes (e.g., q-ratios of less than 1.5 in the smaller size classes) will not be sustainable at the stand level for most tree species.

- Current overstory importance of late successional species will be sustainable at both the stand and aggregated population level in the absence of novel stressors such as high

deer populations or invasive pests. Gap-phase species will have sustainable populations at the larger landscape level, but not within individual stands.

- For a given productivity level and species mixture, and under constant environmental conditions and a disturbance regime limited to individual treefall gaps, stands dominated by shade-tolerant species will tend to converge toward a common and sustainable equilibrium size distribution.

In this paper, we use a well-tested, individual-tree model to examine the degree to which various initial size distributions in mature and old-growth forests are sustainable over time under different assumptions regarding recruitment limitations and disturbance regimes. We then develop a quantitative index of tree population sustainability that measures the degree to which the initial basal area is maintained. Simulation experiments were also conducted to determine the minimum number of saplings needed to sustain the overstory basal area of a species and the overall stand.

3.2 Methods

3.2.1 Study areas

Field sites were located in portions of three large upper Michigan natural areas that largely escaped logging in the 19th and 20th centuries: the Porcupine Mountains Wilderness State Park, Sylvania Wilderness in the Ottawa National Forest, and a tract of protected private lands owned by the Huron Mountain Wildlife Foundation (see Frelich and Lorimer, 1991a, for full details). Climate is humid continental with mean summer temperatures of 20 C and mean winter temperatures between -7.5 C and -11 C depending on distance from Lake Superior. Annual precipitation is approximately 80-90 cm and is well distributed throughout the year. Soils are typically of loam, sandy loam, or silt loam texture and are primarily Fragiorthods and Haplorthods. Elevations range from ˜ 182 m near the shore of Lake Superior to 600 m

further inland. All plots were on mesic or dry-mesic habitats, with the majority of plots on the *Acer-Tsuga-Dryopteris* (ATD) floristic habitat type of Kotar et al. (2002). The ATD habitat is above average in productivity, with a sugar maple site index of approximately 19-20 m for base age 50 years (Coffman, 1984).

3.2.2 Field methods

Plot locations were limited to zones of primary forest and were determined by random coordinates on maps in advance of the field survey. Seventy 0.5 ha plots were surveyed in the summers of 1981-1984. Each plot was divided into seven contiguous 10.1 x 70.7 m strips. Species, diameter at breast height (dbh) and crown class were recorded for all trees taller than 1.4 m. On some younger dense stands comprised primarily of small trees, a subset of strips were tallied. Each plot generally included a population of > 165 trees larger than 10 cm dbh and was representative of the species composition and size structure of the surrounding stand.

3.2.3 Model description

CANOPY is a spatially explicit, individual-tree model that incorporates direct simulation of gap capture and simulation of natural disturbances (Hanson et al., 2011, 2012). It has been calibrated from a data set containing more than 8,000 trees, many on permanent plots. Stands in the calibration data set span a wide range of forest developmental stages, from young even-aged stands to broadly uneven-aged old growth. Measurements were conducted on temporary and permanent plots between 1953 and 2007. For each species and floristic habitat type, height growth is simulated for small trees in response to canopy gap area and competitor crown area. For larger trees, diameter growth is simulated as a function of competition level (crowding or stocking level) in a 900 m^2 competition neighborhood. Background mortality is determined by initial tree size and plot competition level. Mortality from natural disturbances is simulated following the recurrence intervals in Frelich and Lorimer

(1991a), with size-dependent individual tree probability of mortality based on equations in Hanson (2009). Stochastic variation is incorporated into predictions of species composition of new recruits, height and diameter growth, background mortality, and disturbance events.

CANOPY's recruitment module assesses the density of saplings (2-6 cm dbh) on each 10x10 m cell, adding new individuals when the measured density is lower than the expected density predicted by the regeneration equation, given the overstory density on the surrounding 900 m^2 patch (Hanson et al., 2011). Species of new individuals are determined stochastically based on the local overstory competition and density following a method similar to Vanclay (1994). CANOPY is currently calibrated for late-successional northern hardwood forests on mesic upland sites in which recruitment is primarily through advance regeneration of mostly shade-tolerant species and short-distance colonization by midtolerant gap-phase species. Wind disturbance usually does not facilitate the establishment of pioneer species and largely leaves dominance by late-successional species unchanged (Dahir and Lorimer, 1996; Peterson, 2000). Long-distance colonization by species of relatively low shade tolerance (e.g., *Betula papyrifera*, *Populus tremuloides*, *Pinus strobus*) occurs primarily after fire. But because fires severe enough to cause overstory mortality occur infrequently in northern hardwoods (Frelich and Lorimer, 1991a; Schulte and Mladenoff, 2005), and early successional trees of all species combined comprise only 3.3% of the basal area in the study area forests, post-fire and long-distance recruitment is not considered in this paper. The recruitment equations for hemlock in this paper are based on calibration data only from stands having low levels of deer herbivory.

3.2.4 The demographic sustainability index

An index of demographic sustainability was defined as the ratio of final simulated basal area to initial observed basal area. This index evaluates the degree to which the existing stand basal area can be sustained over time in a particular stand or landscape given a specified

disturbance regime, recruitment mechanism, and environmental conditions. Spatial extent and time frame will affect the numerical value of the index, but both can be adjusted based on the questions of interest. For example, a species that is not sustainable in any individual stand may nevertheless be sustainable at a larger spatial scale if there is a shifting spatial pattern of establishment and subsequent decline. Likewise, if an even-aged stand of pioneer species has deficient regeneration, the demographic sustainability index may be quite low over a time span of about a century. But if subsequent disturbance or harvest/planting operations stimulate adequate regeneration, then the index may be close to 1.0. In the northern hardwood ecosystem considered here, the natural disturbance regime produces mostly uneven-aged stands (Frelich and Lorimer, 1991b), and so our simulations largely evaluate the degree to which the species or stand basal area may be sustainable in a context where the canopy is usually not completely removed in a disturbance.

In this paper, we evaluated the index at multi-century time scales and a dual stand/landscape-level spatial scale. We examined both 500 and 1,000 year time frames, a period about two to four times the average age at time of death for canopy trees in this ecosystem (Lorimer et al., 2001). These simulations were designed to examine the long-term consequences of a particular demographic structure under current environmental conditions, not to predict future consequences of global environmental change. Note that the index can exceed 1.0 if the population is increasing, and that it can't be computed on plots where a species was initially absent.

We computed the index for all species pooled as well as for each of the major canopy species (sugar maple, hemlock, yellow birch, green ash (*Fraxinus pennsylvanica*), basswood (*Tilia americana*), and red maple (*Acer rubrum*)). While the reasons for investigating sustainability of individual species are evident, pooled species sustainability is also relevant in evaluating maintenance of current forest density in cases of severe recruitment limitations. For example, exotic earthworms or exotic pests like the Asian longhorned beetle can limit

recruitment of most species in this ecosystem (Holdsworth et al., 2007; Dodds and Orwig, 2011), potentially leading to chronically understocked stands.

3.2.5 Analytical techniques

Regressions were performed to assess the degree to which the sustainability index can be predicted based on initial stand structure. Several structural variables were computed as a function of the basal area distributions of the underlying stands, which sum the basal area in each 4 cm diameter class. Predictors of the sustainability index included initial stand basal area, initial number of 2-6 cm dbh trees present, the percentage of stand basal area occupied by the five classes centered on the mode of the basal area distribution, and a number of other variables. Sustainability index was regressed individually against each variable and a multiple regression including all the variables was also constructed. Stepwise selection based on AIC was used to remove non-significant terms. Residual plots were manually inspected for evidence of pattern, and normality of residuals was verified with Shapiro-Wilk tests. All statistical analysis was performed in GNU R version 3.0.1 (R Core Team, 2013).

3.2.6 Simulation design

Eight separate simulation designs, within four general groups, were performed to evaluate patterns in population sustainability. Group 1 includes both individual stand and larger-scale simulations, while groups 2-4 include only stand-scale simulations.

Projections of the upper Michigan field data (Group 1). Several sets of simulations were performed that projected the development of the 70 upper Michigan field plots for 500 years, with 10 replications per plot. To assess the stability of the current population structure, one set of simulations (Group 1a) was conducted for individual stands in which the total number of trees in the 2-6 cm size class (the smallest class tracked by CANOPY) was maintained constant at the initial levels with background mortality only. Group 1a allowed CANOPY to determine the species composition of new recruits based on local overstory density and

species composition. However, this procedure does not maintain constant numbers of 2-6 cm saplings for any individual species, only for all species pooled.

To evaluate the sustainability of the overstory for individual stands in the absence of recruitment limitation, Group 1b simulations were also performed in which CANOPY's normal recruitment mechanism sets species composition and permits variable understory density, again with background mortality only.

In Group 1c simulations, CANOPY set both species composition and understory density on each of the 70 plots, this time in concert with simulation of the historic natural disturbance regime.

In all Group 1 a-c simulations, results of individually simulated plots were also subsequently pooled to estimate sustainability at a larger scale (results shown in the last column of Table 1).

In Group 1d, the 70 plots were pooled together initially and simulated as a single unit to represent the landscape scale, so that the total numbers of trees of each species at the larger scale could be experimentally manipulated. Also, in contrast to Group 1a, which controlled only the total number of saplings of all species, the simulations in Group 1d held the initial numbers of 2-6 cm trees of each species constant over the 500-year period. During these simulations, only background mortality was employed, but note that the initial number of 2-6 cm trees may also reflect the influence of recent episodic disturbance in the field data.

Simulation experiments with pre-determined numbers of 2-6 cm saplings (Group 2). To more directly assess the minimum population necessary to sustain a stand, a group of designed simulation experiments was performed that systematically varied the number of 2-6 cm trees. Initial conditions for this group consisted of simulated size distributions after 1000 years of forest development with only small gap formation from background mortality. These initial conditions have the advantages of removing the complicating influence of past partial disturbances and having an initial size distribution shown in previous simulation experiments

to sustain the initial basal area (Appendix A). In Group 2a, recruitment was held fixed at pre-determined constant levels and, for simplicity, only sugar maple saplings were added. Levels of recruitment selected ranged from 15% to 150% of the average number of 2-6 cm saplings in the initial conditions.

In Group 2b simulations, CANOPY was allowed to regulate recruitment levels within limits, but recruitment was restricted so as not to exceed the quotas used in Group 2a. This was done in order to avoid forcing biologically unrealistic numbers of saplings into a stand. For example, in one trial, 600 trees/ha was the maximum allowed number of 2-6 cm trees, but natural mortality simulated by CANOPY could cause numbers to fall below this level. To assess the influence of species, this experiment was repeated using pure sugar maple stands, mixed hardwood stands without hemlock, and lastly with mixed hemlock/hardwood stands.

Simulations imposing a constant q-ratio on 2-22 cm trees (Group 3). These 1000-year simulation experiments imposed pre-specified and constant maximum q-ratios on the 2-22 cm dbh classes (using the conventional 5 cm class width). In each year, CANOPY counted the number of trees in the 17-22 cm class, and determined the number that should then be expected in the 12-17 cm, 7-12 cm, and 2-7 cm classes based on the specified q-ratio. If at any time, the model detected surplus trees in a size class beyond the number specified by the q-ratio, these were removed to maintain the specified q-ratio. If mortality in the smaller classes was high enough that the 17-22 cm class was not replenishing itself, then the target number of trees in the other classes was adjusted downward to account for the change in 17-22 cm density in order to maintain the specified q-ratio.

Simulations assessing minimum sustainable q-ratios (Group 4). Whereas the experiments in Group 1-3 generally examined constant numbers of 2-6 cm trees or constant q-ratios, Group 4 allowed for the possibility that variable q-ratios within a stand may be needed for sustainability. This group assessed the minimum number of understory trees needed in three consecutive 5 cm understory classes. Based on the simulated mean number of 17-22 cm

trees in a steady state stand (49 trees/ha), we determined the minimum number of trees in the next smaller class needed to sustain the 49 trees/ha, given the growth and mortality functions in the model. This procedure was iteratively repeated in the 3 smallest size classes for 500 year time spans.

3.3 Results

3.3.1 Range of demographic sustainability index values for individual stands

When individual plots were simulated for 500 years with the total number of 2-6 cm saplings held constant at their initial levels (Group 1a simulations), a variety of demographic sustainability index values were predicted, with values ranging from 0.2 to 1.6. Final sustainability values depended on initial structure, recruitment dynamics, and disturbance regime; a subset of 5 plots that ranged widely in initial structure is shown in Fig. 1. Within any particular plot, a range of values was predicted because of the stochastic growth and mortality functions in the CANOPY model. For any fixed population of 2-6 cm trees, size distributions under a disturbance regime of single treefall gaps tended toward a descending monotonic form, but with differing final basal areas and sustainability.

A number of general trends emerged from simulations of individual plots based on an assumption of constant recruitment (Group 1a). All size distributions with high sustainability (index values $\geq$0.9) were initially descending monotonic in form (data not shown). Shallow descending monotonic curves were only moderately sustainable, typically having sustainability index values of about 0.7. No flat or unimodal distributions were identified that were both sustainable and stable in form. For example, the rather flat, unimodal distribution in Fig. 1 j likewise had an index of about 0.7. Young pole stands with large numbers of small trees were often able to sustain the initial basal area (e.g., Fig. 1 m). But these pole stands did not have stable size distributions and were able to maintain basal area sustainability only by shifting the form of their size distribution over several centuries from unimodal to

descending monotonic.

When CANOPY's normal recruitment mechanism was allowed to operate for 500 years (i.e. no constraints were placed on the number of 2-6 cm trees; Group 1b simulations), the five stands in Fig. 1 tended to converge toward similar descending monotonic curves. These were also close to the average observed size distributions from field measurements of steady-state, old-growth stands on ATD habitat (Fig. 2 a). While there were differences in the final number of trees in the 10-40 cm classes among some replicates, the simulation envelopes for all five curves overlapped (Fig. 2 b), suggesting little inherent difference among replicates. For all 70 plots, mean diameter distributions of simulated steady-state plots fell near the middle of the range of field observations (Fig. 2 c).

Calculations of the sustainability index for individual species, based on 500 year CANOPY simulations of the 70 individual plots, also indicated a wide range of index values depending on recruitment limitations and disturbance regime (Table 1, Group 1a simulations). Sugar maple, green ash, and all species pooled had index values > 0.8 for all conditions tested. Yellow birch and red maple had low sustainability under most conditions, with $\geq 70\%$ of plots having index values < 0.4 for both species and average plot-level index values < 0.5. Hemlock and basswood had moderate levels of sustainability, except that hemlock showed a very gradual and progressive increase in relative basal area on ATD habitat under a regime of only small treefall gaps and low levels of deer browsing.

3.3.2 Assessing sustainability at the landscape scale

The initial landscape-level diameter distributions for most species from the 70-plot field survey were descending monotonic in form, with the exceptions of hemlock, white cedar, and northern red oak (Fig. 3). The descending distributions might be considered sustainable under traditional, *ad hoc* criteria. However, it is difficult to make reliable assessments of sustainability without consideration of the underlying recruitment, growth, and mortality

relationships. Therefore, the sustainability of initially observed landscape-level size distributions in Fig. 3 was evaluated by projecting them for 500 years using the demographic equations in CANOPY.

Two sets of simulations evaluated sustainability at the landscape scale with constant numbers of 2-6 cm trees (Group 1a and 1d simulations). These two analyses gave substantially different results. In simulations where CANOPY was allowed to determine species composition of the 2-6 cm trees (Group 1a), sugar maple and ash had sustainability > 1.0 (rightmost column of Table 1). Other species had only moderate or low sustainability (<0.8), including major gap-phase species such as yellow birch. In Group 1d, where numbers of trees in the 2-6 cm class were maintained at initial levels for each species, all species had sustainability >1.0 except red maple (Fig. 4).

3.3.3 Predictors of demographic sustainability

Multiple regressions using demographic variables based on initial plot conditions were able to explain 47-77% of the variation in sustainability index for all species combined, and 27-68% of the variation for individual species, depending recruitment mechanism and disturbance regimes (Table 2). The two best single predictors in bivariate regressions were the initial basal area and quadratic mean diameter (R^2 of 0.47 and 0.24, respectively, under the historic natural disturbance regime without recruitment limitation). Demographic sustainability index was most predictable when the initial number of saplings was maintained under a disturbance regime comprised only of background mortality. Sustainability of individual species was least predictable ($0.27 \leq R^2 \leq 0.45$) under background mortality alone but without any recruitment limitations. Incorporating the historic disturbance regime resulted in the lowest predictability for overall sustainability (R^2=0.47). However, predictability for sugar maple and yellow birch was higher under the historic natural disturbance regime than under background mortality alone and roughly equal to simulations that maintained the

initial number of saplings (Table 1).

In the original field data, increasing initial basal area was negatively correlated with the number of 2-6 cm saplings in a log-linear fashion (Fig. 5 a). Simulations suggested that the sustainability index is positively correlated with the number of 2-6 cm trees and negatively correlated with initial basal area (Fig. 5 b). Sustainability index of 1.0 was predicted at 37 m^2/ha of basal area and 320 trees/ha in the 2-6 cm class (Fig. 5 b,c). These two levels of initial stand basal area and number of 2-6 cm saplings are very similar to the mean observed values of stands in the field data (Fig. 5 d,e)

3.3.4 Minimum number of understory trees for sustainability

Controlled experiments that started with a steady-state stand (Group 2a simulations) and maintained the initial number of 2-6 cm dbh trees at constant levels from 50-600 trees/ha demonstrated an asymptotic trend in sustainability with increasing sapling density. Index values of about 1.0 were reached when stands contained approximately three hundred 2-6 cm trees per hectare (Fig. 6 a). Additional experiments that placed a limit on the number of 2-6 cm trees but did not add more trees than predicted by CANOPY's recruitment mechanism (to avoid biologically unreasonable sapling densities) produced similar trends (Group 2b; Fig. 6 b). Results were similar under all species mixtures tested.

Equilibrium diameter distributions for pure maple stands with fixed numbers of 2-6 cm trees had shallower slopes as the number of saplings was decreased, but were all descending monotonic in form (Fig. 7). For all stands with sapling densities > 150/ha, size distributions for trees above 30 cm dbh converged to essentially identical form as long as the stands were fully stocked. Higher densities of saplings resulted in higher densities of trees less than 30 cm dbh. However, mortality rates were correspondingly higher in high-density stands, especially for the smaller trees.

Controlled experiments that started from steady-state stands and applied constant q-

ratios to the 2-22 cm trees (Group 3 simulations) demonstrated that q-ratios of less than about 2.3 resulted in stands with low index values averaging only 0.3-0.4(Fig. 8). Results were similar for mixed hardwood stands without hemlock recruitment and for stands that included mixed hemlock/hardwood recruitment.

Iterative experiments that controlled q-ratios in the 2-7, 7-12, and 12-17 cm diameter classes (Group 4 simulations) showed that a variable q-ratio was necessary to maintain target tree densities in the smaller diameter classes (Table 3). The predicted minimum number of saplings was close to the average sapling populations in CANOPY simulations of steady-state stands under a disturbance regime exclusively of individual treefall gaps. Predicted minimum sapling numbers were also generally close to observed values from stands on ATD habitat classified as steady state based on structural criteria (Appendix A). The simulated steady-state stands do contain some inherent variation as a result of stochastic growth and mortality in the CANOPY model, but stands on average appear to maintain populations close to the minimum number of saplings required for sustainability and without carrying large excesses.

3.4 Discussion

3.4.1 Characteristics and dynamics of sustainable size distributions

Results demonstrated that the following demographic size distributions were not generally sustainable in this forest type and region: (1) flat distributions, (2) shallow descending monotonic curves, and (3) unimodal curves. While it is conceivable that a flat or shallow descending distribution could be sustainable if mortality rates are sufficiently low, the mortality calibration data in CANOPY indicate that mortality rates are too high to sustain these distribution types under any disturbance regime and competition level tested in these experiments. These results suggest that descending monotonic curves should not always be assumed to indicate sustainable populations in the absence of specific knowledge of mortal-

ity and recruitment rates. Likewise, we could find no evidence that unimodal distributions could represent alternative stable states under the conditions examined. All sustainable and stable size distributions in this study were steeply descending in form.

The relationships in Figures 2, 5, and 6 suggest underlying feedback mechanisms in these forests that cause stand structure to trend toward a stable size distribution in the absence of moderate or severe disturbance. When basal area is relatively high, recruitment decreases (Fig. 5 a) and mortality increases (Fig. 6), driving overall stand density downward over time. But when forests are relatively sparse, the density of saplings tends to increase in the absence of recruitment limitations (Fig. 5 a). In this study, these countervailing trends balanced, with a demographic sustainability index of 1.0, at 37 m^2 of basal area and 320 saplings per hectare (Fig. 5 b,c). These values correspond with the mean of field observations (Fig. 5 d,e), suggesting that the existing landscape-level size distribution is, on average, close to CANOPY's predicted stable structure. Simulations of different stands with varied initial structures all converged toward a common size distribution on the same habitat and site productivity level (Fig. 2 a). The apparent differences among curves in the 10-40 cm dbh classes were not significant (Fig. 2 b), supporting the hypothesis of convergence toward a common stable structure for a given habitat under constant environmental conditions. The implication that size distributions at the landscape scale may be stable is consistent with earlier evidence suggesting that these study areas generally meet criteria for equilibrium landscapes (Frelich and Lorimer, 1991a,b) based on disturbance frequency and severity (*sensu* Turner et al., 1993).

Some authors have suggested that relatively low and constant q-ratios are sustainable and typical of natural stands, and they have defended the negative exponential function as the best model for uneven-aged northern hardwoods (e.g., Rubin et al., 2006). For Ponderosa pine, O'Hara (1996) provided simulation evidence that constant q-ratios of less than 1.5, or even linearly decreasing numbers of trees, may be sustainable in managed selection stands.

Q-ratios in natural northern hardwood stands, however, can be much higher. In old-growth northern hardwood stands in upper Michigan, q-ratios of the 5-10 cm class averaged 6.5 (Goodburn and Lorimer, 1999). In the present study, any q-ratio greater than 2.3 in the smaller size classes was found to be sustainable, suggesting that ratios as steep as six may reflect an excess population of small trees.

However, a variable and relatively steep q-ratio was necessary in the smaller size classes (<20 cm) to maintain a stable, uneven-aged structure. Q-ratios of 1.3 are used commonly in uneven-aged silvicultural guidelines for northern hardwoods (Leak, 1965; Marquis, 1978; Leak, 2002). But, when these were applied to the 2-22 cm dbh classes, stands had sustainability index values of only 0.3 and standing basal area of only ≈ 12 m^2/ha, about half the level in fully-stocked managed stands. Shallow q-ratios of about 1.3 do provide a reasonable approximation to the form of a natural size distribution over the ranges of diameters typically maintained in managed stands (e.g., 15-60 cm dbh). But, results of the present study suggest that a constant q-ratio of this magnitude would not be sustainable if applied across the entire range of sizes (see also Goodburn and Lorimer, 1999). Our results suggest that a rotated sigmoid size distribution (*sensu* Goff and West, 1975) with variable q-ratios may be required when management is applied across the full range of size classes, as in a forest managed for old-growth characteristics (Keeton, 2006).

The negative exponential model with constant q ratios may especially be inappropriate if some small trees are harvested from the site. Harvesting of trees < 15 cm dbh has rarely been conducted with traditional selection silviculture. But there may be a trend toward harvesting smaller trees in some stands managed for bioenergy and other alternative wood products (Janowiak and Webster, 2010; Aguilar et al., 2013). The results of the present study suggest that viable alternatives for thinning the small-tree population may be limited in uneven-aged stands where demographic sustainability is a management objective.

3.4.2 Sustainability of late-successional and gap-phase species

Overstory populations of late-successional species tended to be sustainable both at the level of individual plots and in the aggregate population. Sugar maple had high values of the sustainability index (average plot index >1.2 and aggregate population index >0.95) under all conditions tested. Results for hemlock were somewhat more varied, reflecting several likely factors. CANOPY predicts that under prevailing (late 20^{th}/early 21^{st} century) environmental conditions, hemlock on *Acer-Tsuga-Dryopteris* habitats would gradually increase in importance over a span of > 500 years under a regime of single-tree gap dynamics and low deer populations. However, CANOPY predicted that hemlock would decline in abundance in multiple-tree gaps and larger disturbances (see also Webster and Lorimer, 2002; Halpin, 2009; Hanson et al., 2012). When simulations were conducted under background mortality only and low browse levels, hemlock had an average individual-plot sustainability index of 2.3 and a landscape-scale sustainability of 1.6. CANOPY simulations of the historic natural disturbance regime suggest a corresponding reduction of hemlock following major disturbance; the average-plot sustainability of hemlock decreased to 1.2 and the aggregate-population sustainability decreased to 0.8. While these predictions of hemlock response need further field testing, the model does suggest that hemlock increasingly dominates stands under a regime of small treefall gaps, but this trend is counteracted by the occurrence of periodic canopy disturbance. Hemlock is known to become dominant under disturbance regimes in which large openings and stand-replacing disturbances are infrequent. For example, in the presettlement land surveys, hemlock was the single most abundant tree species in northern Wisconsin forests (Fahey et al., 2012). But mature hemlock is also known to suffer high mortality after exposure (Hough, 1960), and hemlock seedling establishment is often poor in large openings (Goerlich and Nyland, 2000).

As hypothesized, gap-phase species tended to be sustainable when the aggregate population was simulated at the larger scale, but not on the level of individual plots. For example,

yellow birch had sustainability less than 0.6 on 80% of plots under all conditions tested. This appears to reflect the shifting-mosaic nature of recruitment of many gap-phase species. Typical size distributions of yellow birches for the upper Michigan dataset contain occasional spikes of recruitment, but many stands lack adequate recruitment. In 57% of stands where yellow birch occurred, the average density of birches per hectare by 4 cm size class < 25 cm dbh was less than the average density by size class of birches ≥ 25 cm dbh, with 83% of these having density of small birches 20% lower than density of large birches. On these plots, the overstory would be unable to sustain the understory even in the absence of any mortality. The deficit of small trees exists despite frequent gap formation, numerous fallen rotten logs, and tip-up mounds. This evidence suggests that more intense disturbance than small treefall gaps (large openings or partial shade in low-density stands) is necessary to sustain the gap-phase species.

The two alternative evaluations of landscape-scale sustainability for gap-phase species (Group 1c and 1d simulations) gave dramatically different results. For example, in Group 1c simulations, where CANOPY selects the species composition of new recruits, yellow birch had a sustainability index averaging only $\approx$0.2. These low index values occurred even under the historic natural disturbance regime and even though moderate and severe disturbance in the simulations were often followed by a pulse of yellow birch recruitment. These pulses, however, were not sufficient to maintain the initial number of yellow birch saplings observed in the field data or the overall level of yellow birch basal area. In contrast, Group 1d simulations, which ensured the observed initial numbers of 2-6 cm trees for each species, had average sustainability values of yellow birch of $\approx$1.3. This difference arose because in Group 1c, the magnitude and duration of the yellow birch pulses as determined by CANOPY's recruitment equations likely reflects the limited recruitment of yellow birch in the calibration data set. Second-growth stands in the calibration data seldom had adequate seedbed conditions (tip-up mounds, rotten logs; e.g., Bolton and D'Amato, 2011). In addition, while the timespan of the

old-growth field survey (1981-2011) included periods during which some plots had suitable seedbed conditions, episodes of moderate or severe disturbances did not occur during the calibration period, which otherwise could have provided light regimes favorable for yellow birch establishment and survival. For these reasons, we consider the Group 1d simulations, which hold constant the initial observed sapling densities of each species, to provide the most plausible assessments of landscape-level sustainability of gap-phase species under the stated environmental conditions.

3.4.3 Potential applications

The demographic sustainability index could be used in conservation planning to identify stands and species with potentially unsustainable populations. Estimates of the index could then be used to prioritize management activity, focusing interventions (e.g., planting operations, population control of herbivores, insect pests, invasive plants, etc) on those sites where they are most likely to be effective. The index could easily be integrated into an Adaptive Management approach, as these already make prominent use of models (Schreiber et al., 2004; Williams, 2011).

In this paper, we used CANOPY, a complex, individual-tree model that has many components such as a natural disturbance simulator, to provide the raw data for calculating the demographic sustainability index. Other individual-tree models or even whole-stand models could be employed, and they need not necessarily be complex to compute the index. For example, some models that utilize non-spatial forest inventory data (e.g., Kaya and Buongiorno, 1987; Caspersen et al., 2011) might also be suitable. Essential requirements for computing the index would be a model that simulates interacting recruitment, growth, and mortality rates under varying competition levels.

In this study, long-term sustainability was evaluated for existing species under dynamics of the historic natural disturbance regime. The CANOPY calibration data set includes

recruitment, growth, and mortality data from the mid 20^{th} to early 21^{st} centuries and disturbance regime data from the early 19^{th} to late 20^{th} centuries (Frelich and Lorimer, 1991a; Hanson et al., 2012). However, climate change and the oncoming complex of invasive and exotic pests will likely have a profound impact on these forest dynamics. Regardless of the specific model used, the recruitment, growth, and mortality functions in empirical and mechanistic models would need to be recalibrated under changing environmental conditions.

As an example, consider the potential impact of the exotic hemlock woolly adelgid. When twenty replications of an old-growth stand with ≈70% hemlock basal area and no adelgid present was simulated for 100 years under background mortality, hemlock had an average sustainability index of 0.9. In a simulation including a single adelgid infestation that killed 25% of the live hemlocks (as in Eschtruth et al., 2006), the predicted mean sustainability index was reduced to 0.74. When a persistent infestation was assumed, such that hemlock mortality was 25% in each decade, mean index value was further reduced to 0.07. The index could be especially useful in comparing the potential impact of pests and pathogens that kill only mature trees (e.g., Dutch Elm disease) with those that kill trees in all life stages (e.g., Hemlock woolly adelgid). As data on these and other stressors becomes available, future work could integrate their effects into models such as CANOPY, and revised estimates of demographic sustainability index could help guide conservation and restoration efforts.

Literature Cited

Aguilar, F. X., M. J. Daniel, and L. L. Narine (2013). Opportunities and challenges to the supply of woody biomass for energy from Missouri nonindustrial privately owned forestlands. *Journal of Forestry 111*, 249–260.

Bolton, N. W. and A. W. D'Amato (2011). Regeneration responses to gap size and coarse woody debris within natural disturbance-based silvicultural systems in northeastern Minnesota, USA. *Forest Ecology and Management 262*, 1215–1222.

Caspersen, J. P., M. C. Vanderwel, W. G. Cole, and D. W. Purves (2011). How stand productivity results from size-and competition-dependent growth and mortality. *PloS one 6*(12), e28660.

Coffman, M. S. (1984). *Field guide: Habitat classification system for upper peninsula of Michigan and northeast Wisconsin.* Houghton, MI, USA: CROFS, School of Forestry and Wood Products, Michigan Technological University.

Dahir, S. E. and C. G. Lorimer (1996). Variation in canopy gap formation among developmental stages of northern hardwood stands. *Canadian Journal of Forest Research 26*(10), 1875–1892.

Dodds, K. J. and D. A. Orwig (2011). An invasive urban forest pest invades natural environments – Asian longhorned beetle in northeastern US hardwood forests. *Canadian Journal of Forest Research 41*, 1729–1742.

Eschtruth, A. K., N. L. Cleavitt, J. J. Battles, R. A. Evans, and T. J. Fahey (2006). Vegetation dynamics in declining eastern hemlock stands: 9 years of forest response to hemlock woolly adelgid infestation. *Canadian Journal of Forest Research 36*, 1435–1450.

Fahey, R. T., C. G. Lorimer, and D. J. Mladenoff (2012). Habitat heterogeneity and life-history traits influence presettlement distribution of early-successional tree species in a late-successional, hemlock-hardwood landscape. *Landscape Ecology 27*(8), 999–1013.

Frelich, L. E. and C. G. Lorimer (1991a). Natural disturbance regimes in hemlock-hardwood forests of the upper Great Lakes region. *Ecological Monographs 61*(2), 145–164.

Frelich, L. E. and C. G. Lorimer (1991b). A simulation of landscape-level stand dynamics in the northern hardwood region. *Journal of Ecology 79*, 145–164.

Goerlich, D. L. and R. D. Nyland (2000). Natural regeneration of Eastern Hemlock: A review. General Technical Report GTR-NE-267, USDA Forest Service.

Goff, F. G. and D. West (1975). Canopy-understory interaction effects on forest population structure. *Forest Science 21*, 98–108.

Goodburn, J. M. and C. G. Lorimer (1999). Population structure in old-growth and managed northern hardwoods: An examination of the balanced diameter distribution concept. *Forest Ecology and Management 118*, 11–29.

Guldin, J. M. (1991). Uneven-aged BDq regulation of Sierra Nevada mixed conifers. *Western Journal of Applied Forestry 6*(2), 27–32.

Halpin, C. R. (2009). Simulated long-term effects of group selection on production, efficiency, composition, and structure in northern hardwood and hemlock-hardwood forests. Master's thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J. (2009). *Emulating natural disturbance dynamics in northern hardwood forests: Long-term effects on species composition, forest structure, and yield.* Ph. D. thesis, University of Wisconsin – Madison, Madison, WI, USA.

Hanson, J. J., C. G. Lorimer, and C. R. Halpin (2011). Predicting long-term sapling dynamics and canopy recruitment in northern hardwood forests. *Canadian Journal of Forest Research 41*, 903–919.

Hanson, J. J., C. G. Lorimer, C. R. Halpin, and B. J. Palik (2012). Ecological forestry in an uneven-aged, late-successional forest: Simulated effects of contrasting treatments on structure and yield. *Forest Ecology and Management 270*, 94–107.

Holdsworth, A. R., L. E. Frelich, and P. B. Reich (2007). Effects of earthworm invasion on plant species richness in northern hardwood forests. *Conservation Biology 21*(4), 997–1008.

Hough, A. F. (1960). Silvical characteristics of Eastern Hemlock (tsuga canadensis). Station paper SP-NE-132, USDA Forest Service.

Janowiak, M. K. and C. R. Webster (2010). Promoting ecological sustainability in woody biomass harvesting. *Journal of Forestry 108*(1), 16–23.

Kaya, I. and J. Buongiorno (1987). Economic harvesting of uneven-aged northern hardwood stands under risk: A markovian decision model. *Forest Science 33*(4), 889–907.

Keeton, W. S. (2006). Managing for late-successional/old-growth characteristics in northern hardwood-conifer forests. *Forest Ecology and Management 235*(1), 129–142.

Kotar, J., T. L. Burger, and J. A. Kovach (2002). *A guide to forest communities and habitat types of northern Wisconsin.* Madison, WI: Department of Forest Ecology and Management, University of Wisconsin – Madison.

Leak, W. B. (1965). The j-shaped probability distribution. *Forest Science 11*(4), 405–409.

Leak, W. B. (2002). Origin of sigmoid diameter distributions. Research Paper NE-718, USDA Forest Service.

Lorimer, C. G., S. E. Dahir, and E. V. Nordheim (2001). Tree mortality rates and longevity in mature and old-growth hemlock-hardwood forests. *Journal of Ecology 89*(6), 960–971.

Lorimer, C. G., D. J. Porter, M. A. Madej, J. D. Stuart, S. D. V. jr, S. P. Norman, K. L. O'Hara, and J. J. Libby (2009). Presettlement and modern disturbance regimes in coast redwood forests: Implications for the conservation of old-growth stands. *Forest Ecology and Management 258*(7), 1038–1054.

Marquis, D. A. (1978). Application of uneven-aged silviculture and management on public and private lands. In *Uneven-aged silviculture and management in the United States*, Timber Management Research, Washington, DC, pp. 25–61. USDA Forest Service.

McGraw, J. B. and M. A. Furedi (2005). Deer browsing and population viability of a forest understory plant. *Science 307*(5711), 920–922.

Nyland, R. D. (2007). *Silviculture: Concepts and applications.* Long Grove, IL, USA: Waveland Press.

Obioh, G. I. B. and A. O. Isichei (2007). A population viability analysis of serendipity berry (*Dioscoreophyllum cumminsii*) in a semi-deciduous forest in Nigeria. *Ecological Modeling 201*(3-4), 558–562.

O'Hara, K. L. (1996). Dynamics and stocking-level relationships of multi-aged Ponderosa pine stands. *Forest Science 42*(Supplement 33).

Orwig, D. A., D. R. Foster, and D. L. Mausel (2002). Landscape patterns of hemlock decline in New England due to the introduced hemlock woolly adelgid. *Journal of Biogeography 29*(10-11), 1475–1487.

Peterson, C. J. (2000). Damage and recovery of tree species after two different tornadoes in the same old growth forest: a comparison of infrequent wind disturbanes. *Forest Ecology and Management 135*, 237–252.

R Core Team (2013). *R: A language and environment for statistical computing.* Vienna, Austria: R Foundation for Statistical Computing.

Reed, J. M., L. S. Mills, J. B. Dunning, E. S. Menges, K. S. McKelvey, R. Frye, S. R. Beissinger, M.-C. Anstett, and P. Miller (2002). Emerging issues in population viability analysis. *Conservation Biology 16*(1), 7–19.

Rooney, T. P., R. J. McCormick, S. L. Solheim, and D. M. Waller (2000). Regional variation in recruitment of hemlock seedlings and saplings in the upper Great Lakes, USA. *Ecological Applications 10*(4), 1119–1132.

Rubin, B. D., P. D. Manion, and D. Faber-Langendoen (2006). Diameter distributions and structural sustainability in forests. *Forest Ecology and Management 222*, 427–438.

Schreiber, S. G., A. R. Bearlin, S. J. Nicol, and C. R. Todd (2004). Adaptive management: A synthesis of current understanding and effective application. *Ecological management and restoration 5*(3), 177–182.

Schulte, L. A. and D. J. Mladenoff (2005). Severe wind and fire regimes in northern forests: Historical variability at the regional scale. *Ecology 86*(2), 431–445.

Smith, D. M., B. C. Larson, M. J. Kelty, and P. M. Ashton (1997). *The practice of silviculture: Applied forest ecology.* New York: John Wiley and Sons.

Turner, M. G., W. H. Romme, R. H. Gardner, R. V. O'Neill, and T. K. Kratz (1993). A revised concept of landscape equilibrium: Disturbance and stability on scaled landscapes. *Landscape Ecology 8*(3), 213–227.

Tyrrell, L. E. and T. R. Crow (1994). Structural characteristics of old-growth hemlock-hardwood forests in relation to age. *Ecology 75*(2), 370–386.

Vanclay, J. K. (1994). *Modeling forest growth and yield: Applications to mixed tropical forests*. Wallingford, U.K.: C.A.B. International.

Webster, C. R. and C. G. Lorimer (2002). Single-tree versus group selection in hemlock-hardwood forests: Are smaller openings less productive? *Canadian Journal of Forest Research 32*(4), 591–604.

Williams, B. K. (2011). Passive and active adaptive management: Approaches and an example. *Forest Ecology and Management 92*, 1371–1378.

Witt, J. C. and C. R. Webster (2010). Regeneration dynamics in remnant *Tsuga canadensis* stands in the northern Lake States: Potential direct and indirect effects of herbivory. *Forest Ecology and Management 260*(4), 519–525.

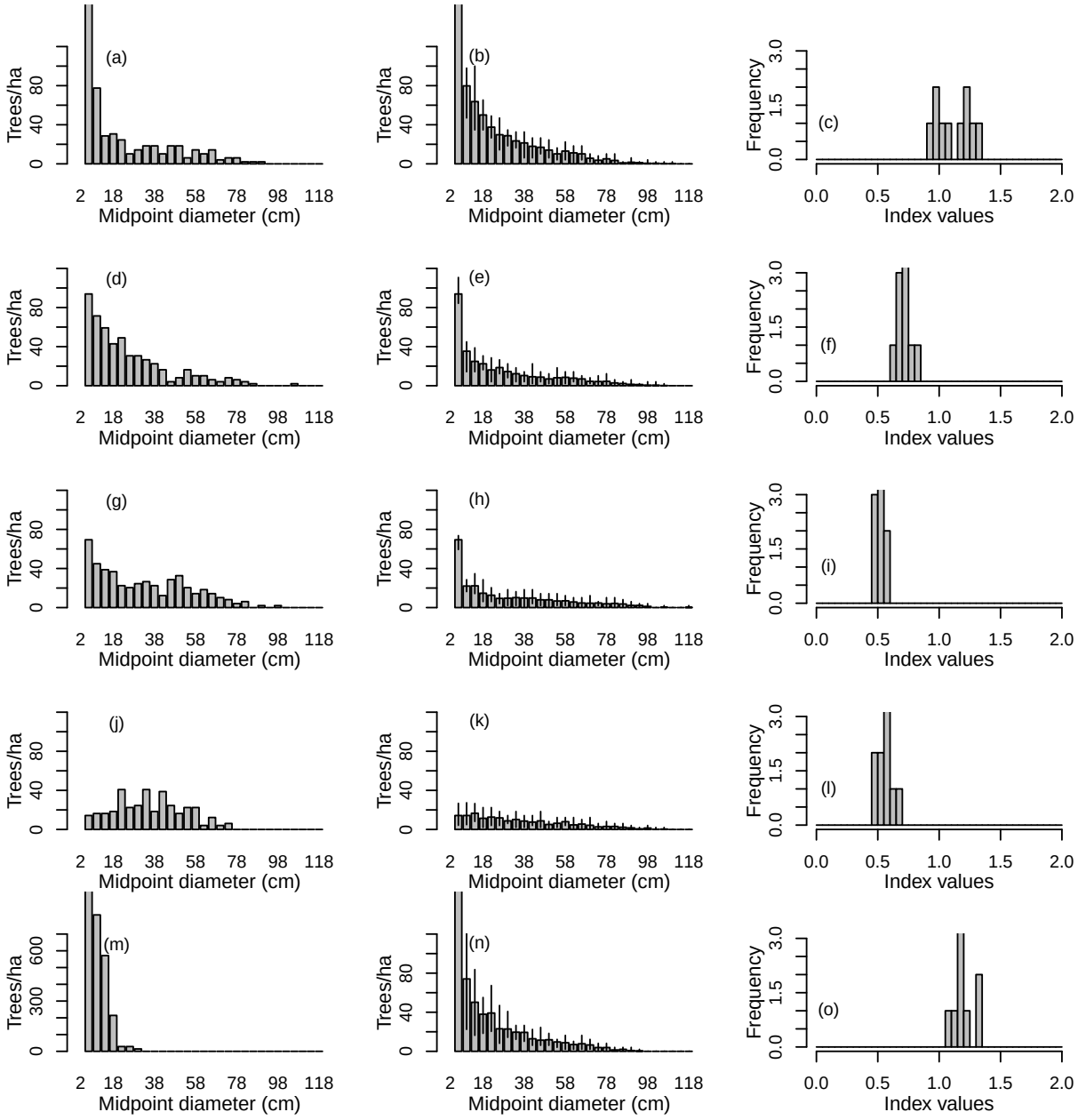

Figure 1: Predicted demographic sustainability index for five 0.5 ha plots with a variety of size initial distributions from the 1981-1984 upper Michigan field survey. Each plot was simulated 10 times for 500 years with the number of 2-6 cm trees maintained constant at initial levels (Group 1a simulations). First class shown is the 2-6 cm class. Left column: initial conditions. Middle column: Projected diameter distributions in year 500 (bars: means; whiskers: range of variation for the 10 replicates). Right column: Distribution of sustainability index values for all species pooled.

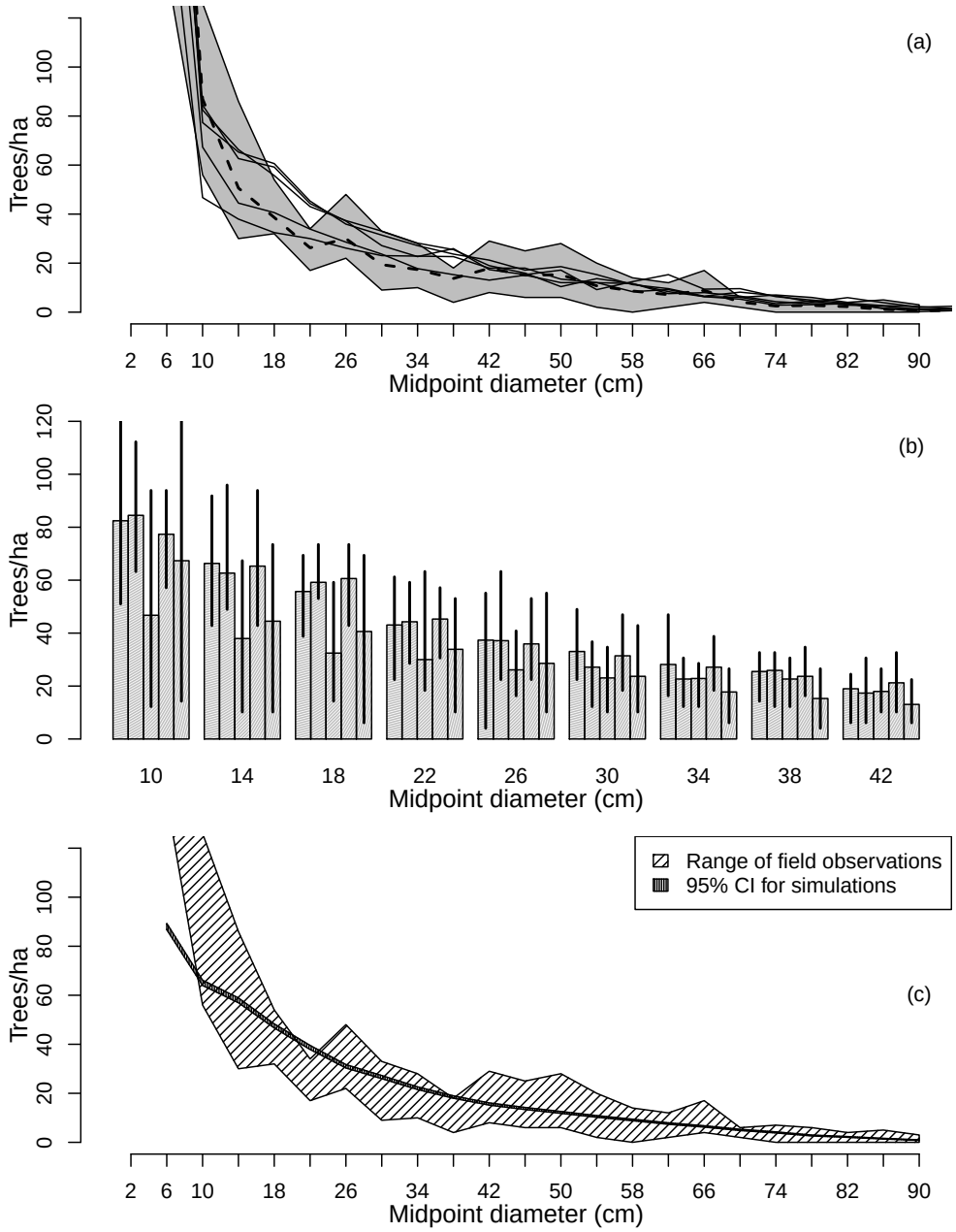

Figure 2: (a)Predicted diameter distribution in year 500 for the five plots shown in Fig. 1 when simulated without any recruitment limitation under background mortality only (Group 1b simulations). Solid lines show the mean of the 10 replicates for each plot. Dashed line gives the mean of steady-state field plots on *Acer-Tsuga-Dryopteris* habitat, with grey band showing the range of variation. (b) Mean simulated densities for trees in the 10-40 cm size classes and the range of variation among replicates of the five stands. (c) 95% confidence band (thick line) for 1,000-yr simulations under background mortality of all 70 plots compared to the range of variation among steady-state field plots (diagonal hatching). The narrow confidence band in the simulations is due to the large sample size (n = 700 independent observations after 1000 years of simulation), rather than lack of stochastic variation in the model (see panel b). For each replication, the diameter distribution selected to compute the width of the confidence band occurred in the last year in which that replication was classified as steady-state.

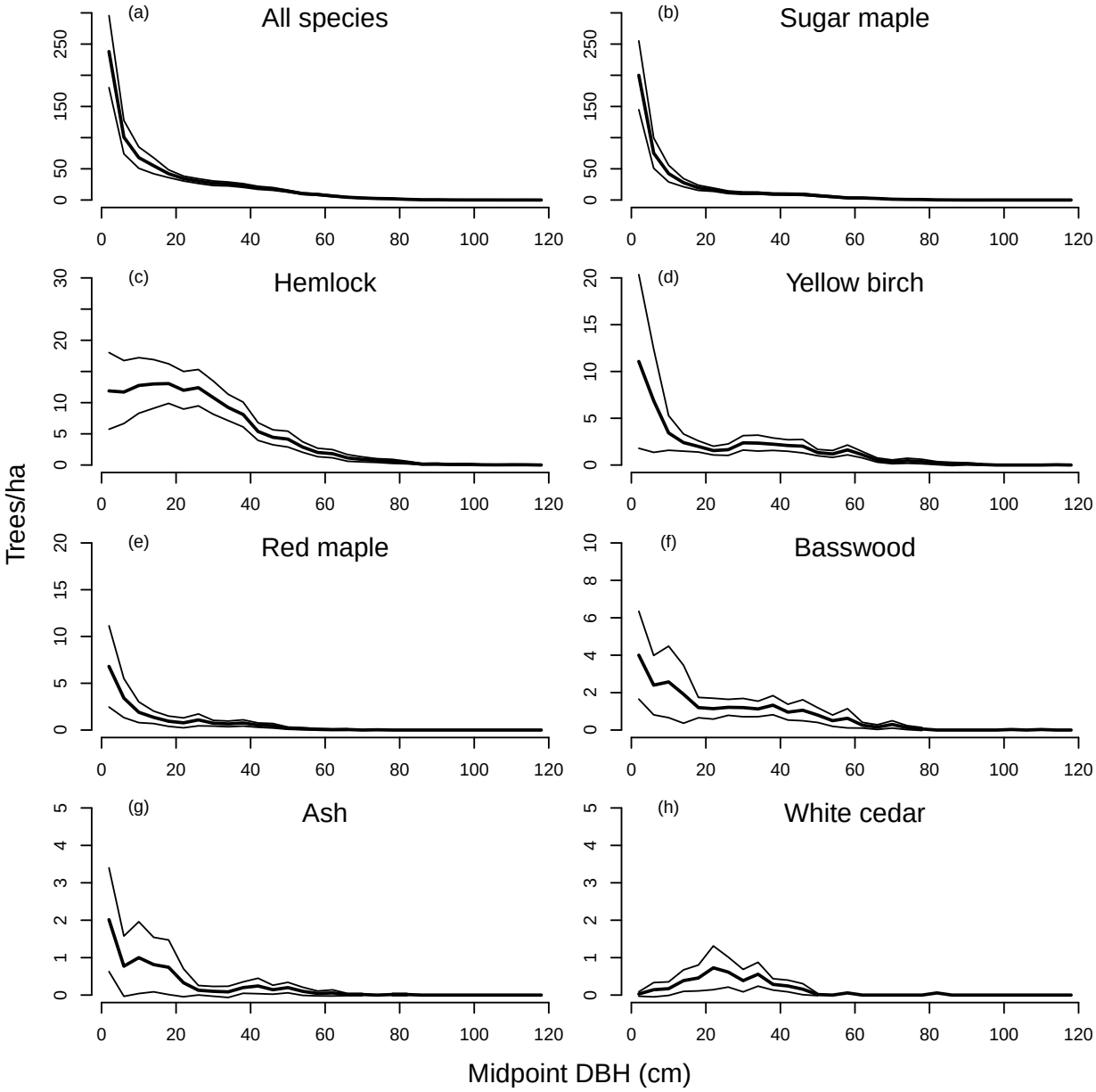

Figure 3: Observed landscape-level diameter distributions for each species in the 1981-1984 upper Michigan field survey (70 half hectare plots in natural areas containing 23,000 ha of primary forest). Thick lines give the means among plots, thin lines show a 95% confidence band without a multiple-test correction.

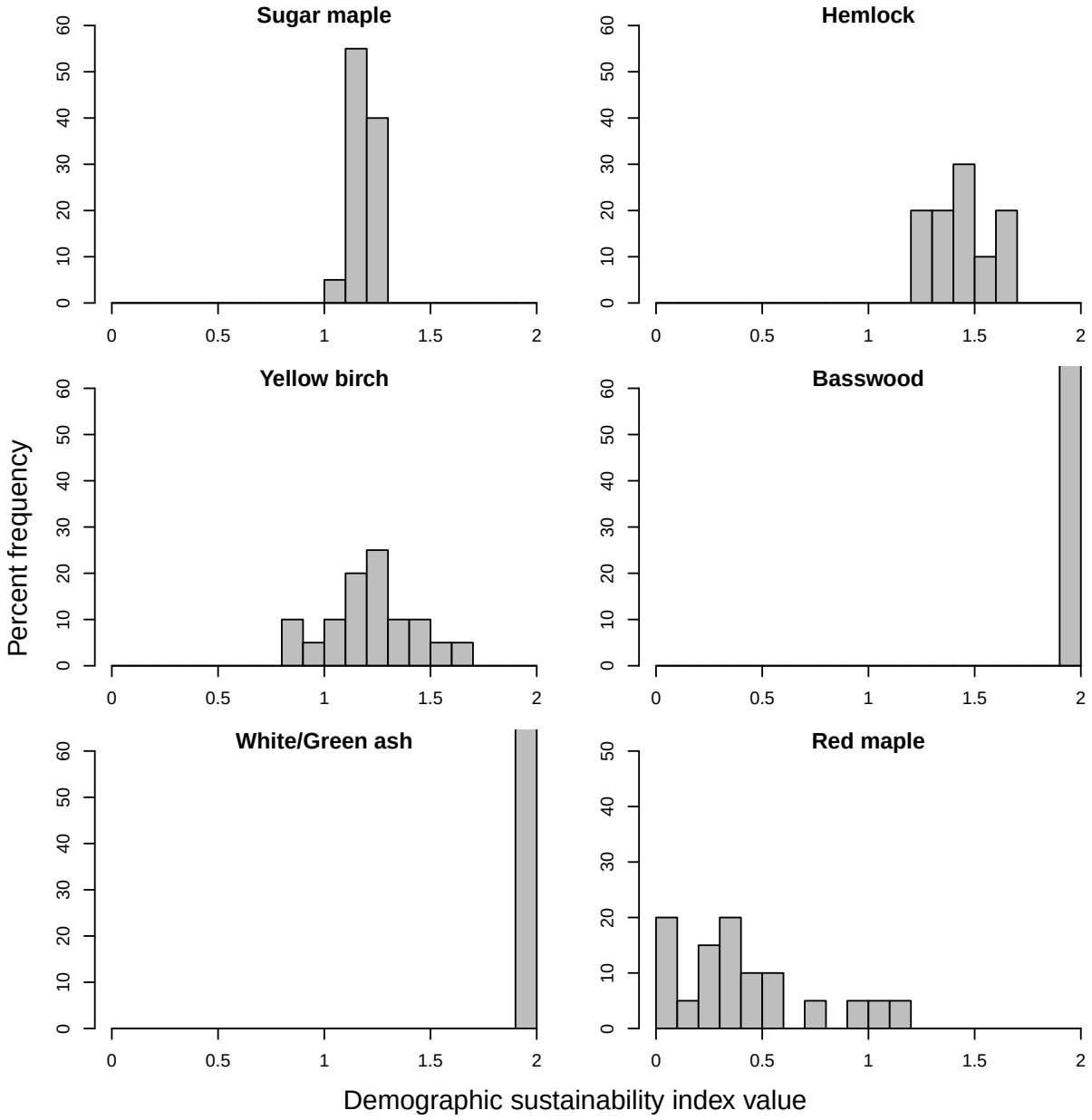

Figure 4: Demographic sustainability index of principal species based on 20 replicated simulations of the pooled 1981 field measurements. Simulations were constrained to maintain the initial observed numbers of trees of each species in the field data, assuming a disturbance regime of background mortality only (Group 1d simulations).

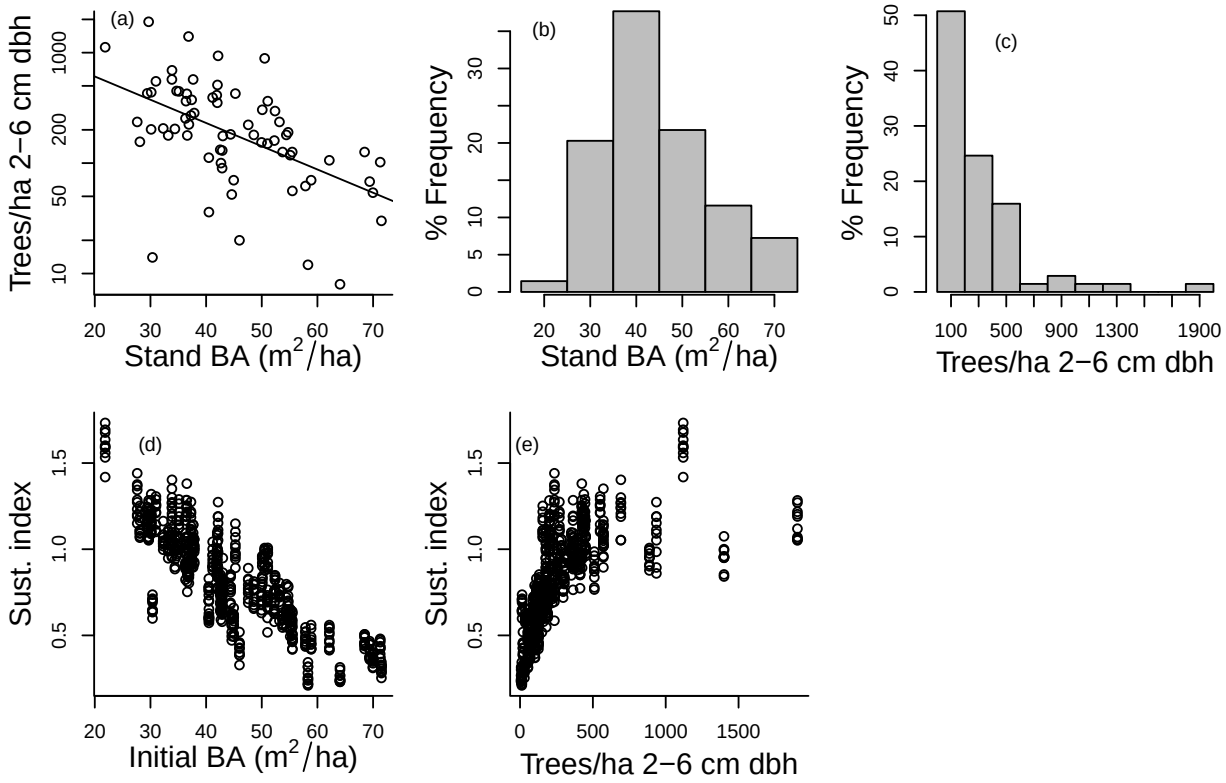

Figure 5: Relationship of initial stand basal area and sapling density with sustainability index. (a) Stand basal area and initial numbers of 2-6 cm dbh trees from the upper Michigan field data. (b,c) Initial distribution of stand basal area and density of 2-6 cm dbh trees from the 1981-1984 field survey. (d,e) Predictions of pooled species demographic sustainability index for the 70 field plots. Simulations were run for 500 years, with 10 replications per plot, and the initial number of 2-6 cm dbh trees held constant. Species composition of new recruits was determined by CANOPY's recruitment equations. (Group 1a simulations).

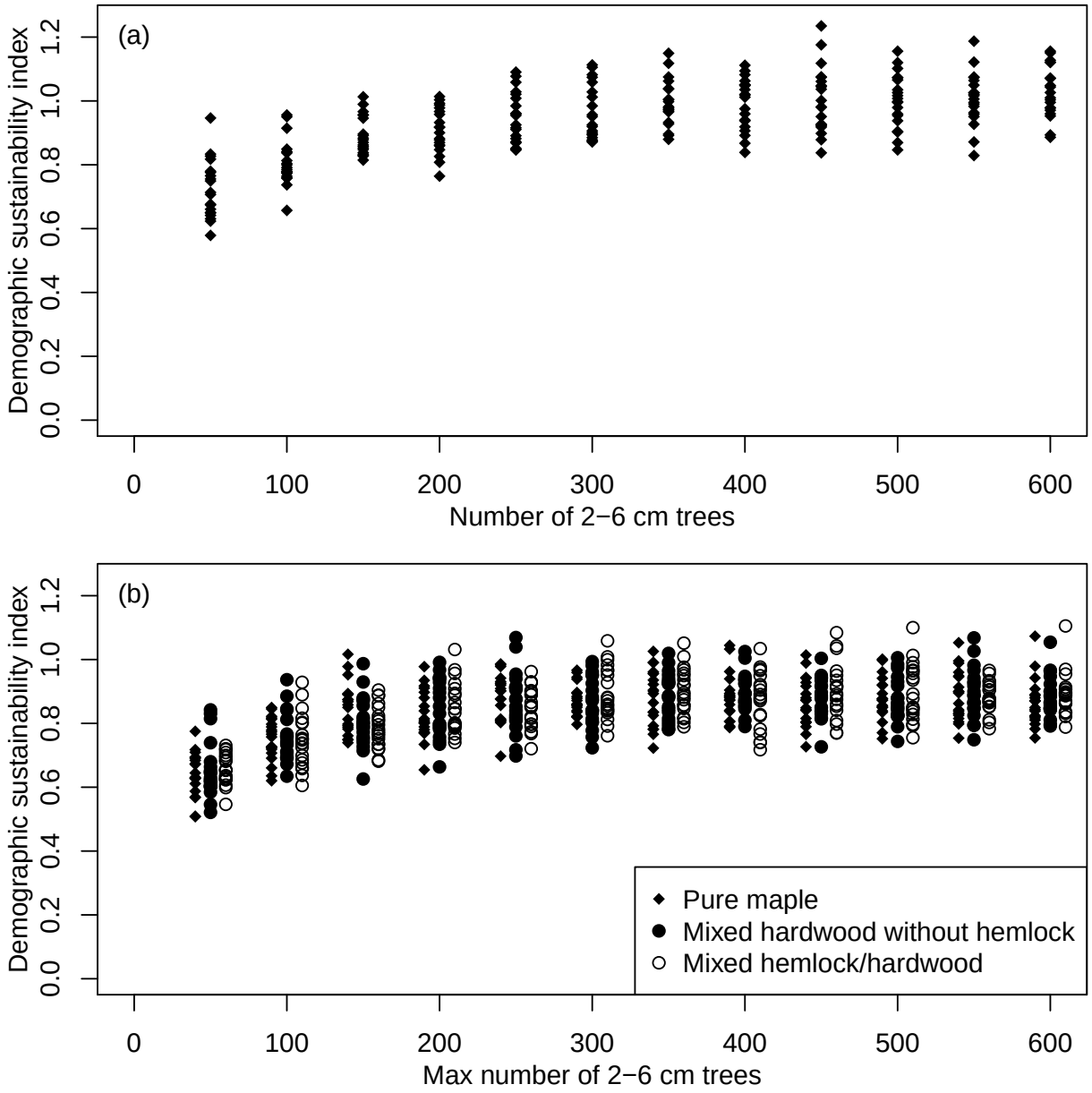

Figure 6: Pooled-species demographic sustainability index values from 1000 year simulation experiments with varying densities of small saplings (Group 2 simulations). (a) Pure maple stand where trees were added if necessary to maintain the specified sapling density (Group 2a simulations). (b) Stands where the specified density was used as a limit (i.e. additional trees were not added if mortality reduced the population below the specified density; Group 2b simulations).

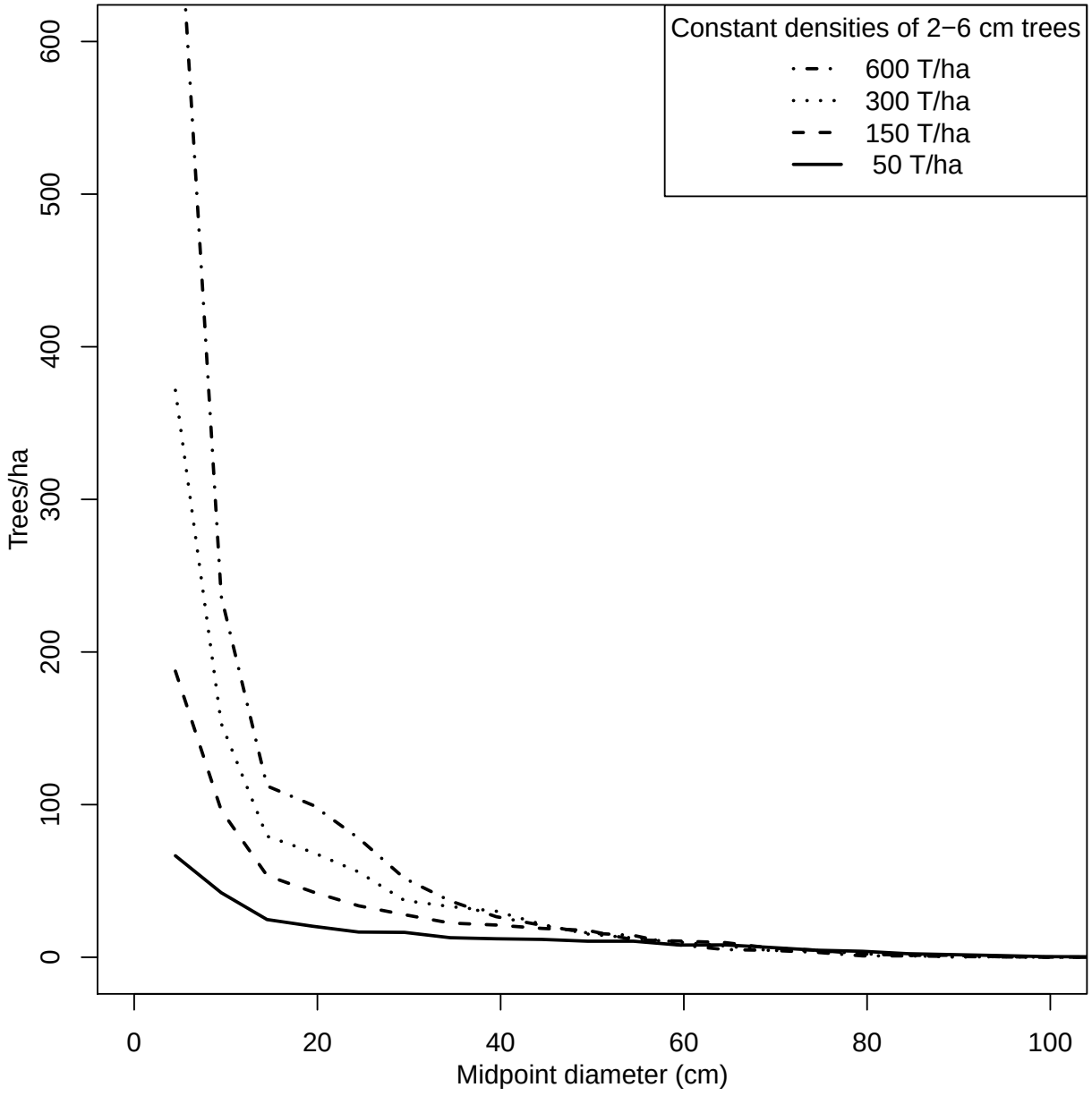

Figure 7: Simulated diameter distributions in year 1000 of pure maple stands where the specified sapling densities were maintained at constant levels (Group 2a simulations). Data shown are the mean of 20 replications.

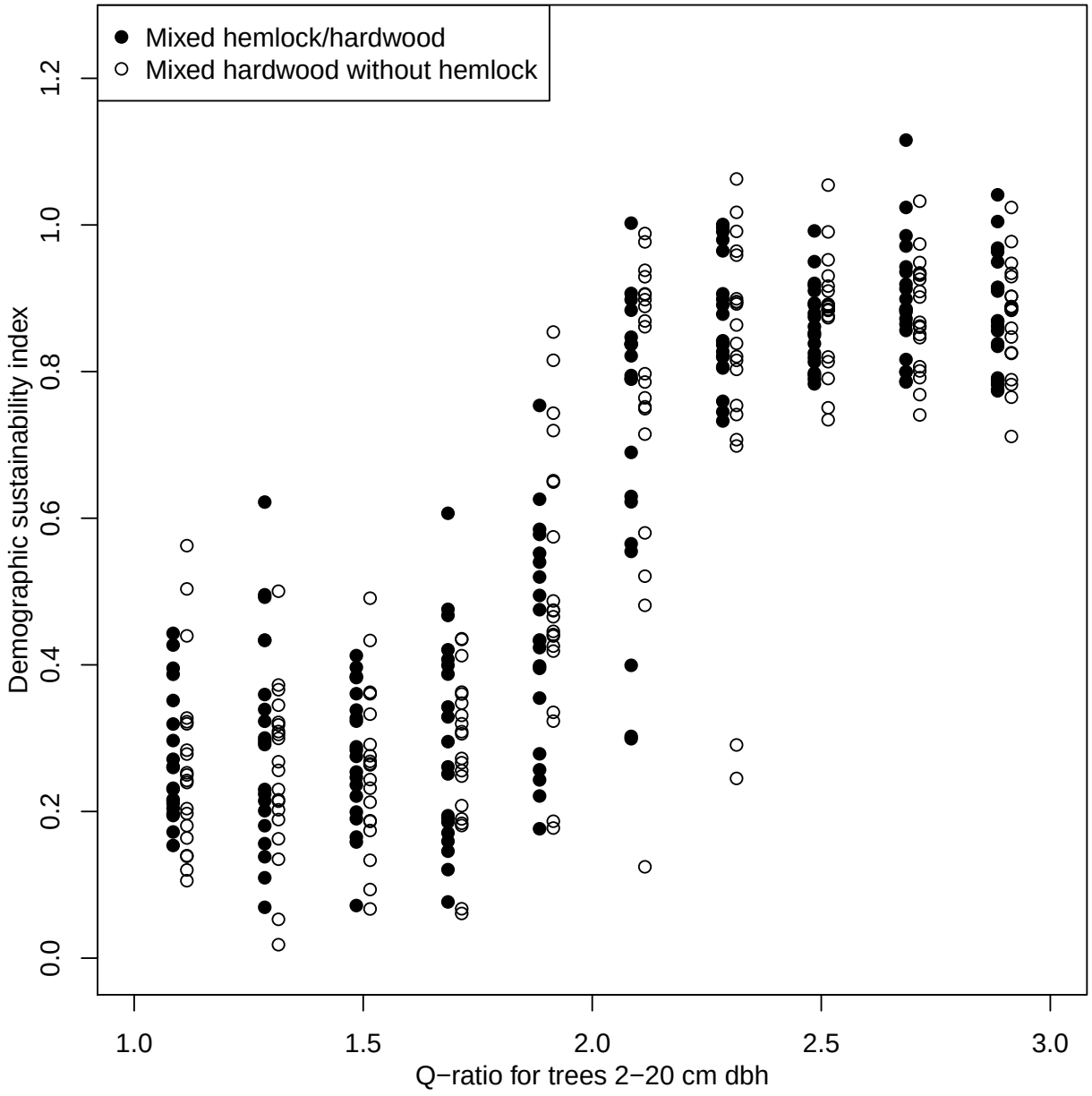

Figure 8: Pooled-species demographic sustainability index as a function of maximum q-ratio maintained in the 2-22 cm diameter classes. Simulations were conducted for 1000 years and replicated 20 times starting from uneven-aged old-growth stands (Group 3 simulations).

Table 1: Summary of demographic sustainability index from simulations of the 70 upper Michigan field plots analyzing the effect of recruitment limitation and disturbance regime (Group 1 simulations). Ten replications of each plot were simulated for 500 years.

	Percentage of plots by sustainability class							
	<0.2	<0.4	<0.6	<0.8	<0.9	0.9+	avg. plot[1]	pop. overall[2]
				All spp. pooled				
Const. 2-6 cm T/ha[3]	0	8	23	48	58	42	0.82	0.83
Background mort. only[4]	0	0	0	11	24	76	1.02	1.08
Historic disturbance[5]	0	2	16	49	66	34	0.82	0.85
				Sugar maple				
Const. 2-6 cm T/ha	1	4	11	24	32	68	1.35	1.06
Background mort. only	4	15	27	39	44	56	1.23	0.95
Historic disturbance	1	6	15	27	35	65	1.38	1.06
				Hemlock				
Const. 2-6 cm T/ha	15	27	40	50	55	45	1.31	0.72
Background mort. only	0	1	4	8	13	87	2.28	1.65
Historic disturbance	10	23	38	51	56	44	1.16	0.82
				Yellow birch				
Const. 2-6 cm T/ha	57	78	84	88	89	11	0.44	0.20
Background mort. only	48	70	81	86	87	13	0.53	0.22
Historic disturbance	59	79	86	89	90	10	0.44	0.18
				Ash				
Const. 2-6 cm T/ha	31	41	49	55	59	41	1.38	2.27
Background mort. only	16	31	44	50	53	47	1.73	2.57
Historic disturbance	24	37	46	52	57	43	1.70	2.49
				Basswood				
Const. 2-6 cm T/ha	26	37	46	53	55	45	1.86	0.78
Background mort. only	14	27	38	47	51	49	2.11	0.90
Historic disturbance	28	45	54	62	64	36	1.61	0.57
				Red maple				
Const. 2-6 cm T/ha	66	78	84	86	88	12	0.36	0.28
Background mort. only	61	73	79	84	85	15	0.50	0.29
Historic disturbance	61	74	82	86	87	13	0.49	0.28

[1] Average demographic sustainability index on plots where the subject species was initially present.

[2] Mean sustainabiilty index of all 70 plots pooled, including plots that initially lacked the subject species.

[3] Constant number of trees maintained in the 2-6 cm class, but CANOPY allowed to determining sapling species composition, under a regime of background mortality only (Group 1a simulations).

[4] CANOPY determined both density and composition of 2-6 cm dbh trees, under a regime of background mortality only (Group 1b simulations).

[5] CANOPY determined both density and composition of new 2-6 cm dbh trees, under the historic natural disturbance regime (Group 1c simulations).

Table 2: Multiple regressions to predict sustainability index based on simulations of the upper Michigan field data.

	Const. 2-6 cm[1]		Background mort.[2]		Historic Dist. Regime[3]	
Variable	Estimate	p	Estimate	p	Estimate	p
All species pooled						
(Intercept)	1.0150	<0.001	0.7190	<0.001	0.5753	<0.001
2-6 cm T/ha	0.0008	<0.001	-0.0002	<0.001	-0.0002	0.039
Initial BA	-0.0266	<0.001	-0.0128	<0.001	-0.0165	<0.001
BA-weighted avg. q			0.0320	<0.001	0.0249	0.072
Distance from SS	-0.0112	<0.001	-0.0022	0.012		
Modal DBH	-0.0020	0.452	-0.0030	<0.001	-0.0020	0.023
R^2	0.77		0.63		0.47	
Sugar maple						
(Intercept)	1.0941	<0.001	-1.5188	<0.001	1.2119	<0.001
2-6 cm T/ha	0.0012	<0.001	0.0010	<0.001	0.0009	0.003
Initial BA	-0.0254	<0.001	-0.0057	0.053	-0.0204	<0.001
Avg. q below mode	0.1478	<0.001	1.4382	<0.001	0.2942	0.008
QMD	0.0391	<0.001	0.0274	<0.001	0.0226	<0.001
2-6 cm SM/ha	-0.0010	<0.001			-0.0013	<0.001
Initial SM BA	-0.0397	<0.001	-0.0470	<0.001	-0.0379	<0.001
Avg. q below mode for SM	0.0031	0.104				
QMD for SM	-0.0170	<0.001	-0.0104	0.002	-0.0205	<0.001
R^2	0.68		0.45		0.60	
Hemlock						
(Intercept)	2.9959	<0.001	1.6742	<0.001	-0.2405	0.404
2-6 cm T/ha	-0.0012	0.002	0.0004	0.071	0.0007	0.049
Initial BA	0.0613	<0.001	-0.0041	0.159	0.0148	0.012
Avg. q below mode	-0.2965	0.071	-0.7328	<0.001		
QMD	-0.1779	<0.001	0.0128	0.065	0.0284	0.012
2-6 cm HM/ha			0.0056	0.003		
Initial HM BA	-0.0649	<0.001	-0.0307	<0.001	-0.0262	<0.001
Avg. q below mode for HM			0.0115	0.049	-0.0400	<0.001
QMD for HM					-0.0359	<0.001
R^2	0.57		0.29		0.29	
Yellow birch						
(Intercept)	-0.0820	0.740	1.0638	0.010	2.0188	0.004
2-6 cm T/ha			-0.0011	0.004		
Initial BA	0.0501	<0.001	-0.0199	<0.001	0.0409	<0.001
Avg. q below mode			-0.8325	<0.001	-1.9549	<0.001
QMD	-0.0682	<0.001			-0.0485	<0.001
2-6 cm YB/ha	-0.0033	0.079	0.0042	0.026		
Initial YB BA	-0.0458	0.002	-0.1621	<0.001	-0.0614	<0.001
Avg. q below mode for YB	-0.0473	<0.001			-0.0344	<0.001
QMD for YB	-0.0492	<0.001	0.0130	<0.001	-0.0371	<0.001
R^2	0.34		0.27		0.34	

[1] Constant number of trees maintained in the 2-6 cm dbh class, but CANOPY allowed to determine their species composition, under a regime of background mortality only (Group 1a simulations).

[2] CANOPY allowed to determine both density and composition of 2-6 cm dbh trees, under a regime of background mortality only (Group 1b simulations).

[3] CANOPY allowed to determine both density and composition of new 2-6 cm dbh trees, under the historic natural disturbance regime (Group 1c simulations).

Table 3: Determination of minimum tree densities in the three smallest size classes needed to perpetuate the steady-state structure shown in panel 1. In Panels 2-4, the number of trees in the three smallest 5 cm diameter classes were iteratively adjusted to determine the number of trees necessary to sustain the next larger size class (final result in bold). Simulations were conducted for 500 years, with 20 replications each, starting from steady state stands on ATD habitat that developed after clearcut under background mortality only (Group 4 simulations). Numbers in parentheses give 95% confidence intervals.

1.	Simulated SS stand		Steady-state field data	
	T/ha	q	T/ha	q
2-7 cm	337	–	686 (290-1080)	–
7-12 cm	88	3.83	153 (119-189)	4.52 (1.65-7.39)
12-17 cm	51	1.74	71 (55-86)	2.20 (1.84-2.56)
17-22 cm	49	1.05	45 (37-52)	1.60 (1.29-1.91)

2. Number of 12-17 cm trees needed to maintain 49 17-22 cm trees

q	12-17 cm T/ha	Predicted 17-22 cm T/ha
1.05	51	30.1 (24.9-35.3)
1.10	53	35.9 (30.7-41.2)
1.15	**56**	**48.5 (41.9-55.0)**

3. Number of 7-12 cm trees needed to maintain 56 17-22 cm trees

q	7-12 cm T/ha	Predicted 17-22 cm T/ha
1.60	90	34.5 (30.3-38.7)
1.80	101	45.0 (39.6-50.4)
1.90	**106**	**56.3 (49.3-62.9)**

4. Number of 2-7 cm trees needed to maintain 106 12-17 cm trees

q	2-7 cm T/ha	Predicted 12-17 cm T/ha
3.50	370	72.3 (65.5-79.2)
3.60	382	88.8 (79.0-100.6)
3.70	392	98.5 (88.5-108.6)
3.75	**398**	**106.7 (93.7-119.6)**

Conclusions

This dissertation has helped to clarify the relative role of treefall gaps, moderate disturbance, and severe disturbance in long-term forest development of northern hardwoods. While evidence on stand development in young even-aged stands and uneven-aged old growth stands has been abundant (e.g., Erdmann and Oberg, 1973; Marquis, 1991; Tyrrell and Crow, 1994; D'Amato and Orwig, 2008), data have been largely lacking on stands between these extremes (ages 100-300). This period is especially important because of the major changes in size distributions and biomass during the critical transition between even- and uneven-aged structure.

When a stand developing after a clearcut is simulated for several centuries with only background mortality, CANOPY predicted that the unimodal diameter distribution of a typical even-aged stand begins to develop a 'compound' form shortly before age 200. This coincided with the maximum number of large trees and peak aboveground live-tree biomass, which are highly correlated (cf. Keeton et al., 2011). On the predominant habitat type, a decline in biomass of about 16% started to occur at about age 210, which corresponded to the mean longevity of the initial even-aged cohort. Biomass net growth became negative from year 210 to 310. The apparent mechanism behind this decline was that positive net growth of secondary cohorts was insufficient to compensate for mortality in the overstory cohort. This mechanism differs from the 'compensation hypothesis' proposed by Luyssaert et al. (2008) and Lichstein et al. (2009) to explain the apparent asymptotic biomass trends in chronosequence data from second-growth forests.

The threshold for steady state structure was crossed in simulations at age 280, but biomass continued to decrease until age 320, and the number of large trees continued to decline until age 400. While this suggests that minimum criteria for steady-state structure can be achieved within the lifespan of the initial even-aged cohort, size distributions did not fully stabilize until 400-500 years after the clearcut. After the initial period of negative net growth, biomass net growth became zero after year 310 and remained non-significantly different from zero for the remainder of the simulation. Average aboveground live-tree biomass net growth of the steady-state permanent plots was likewise close to zero, supporting the early ideas of Odum (1969) and Bormann and Likens (1979) for the existence of a steady state having zero net growth. This disturbance regime of a clearcut followed by several centuries of gap formation also indicated major changes in species composition over time. Whereas midtolerant species averaged 26% of stand basal area 80 years after the clearcut, they declined to only 7% after 1000 years of background mortality.

To further clarify the effects of disturbance regime on stand stage trajectories beyond a simple clearcut followed by background mortality, simulations were conducted under two additional disturbance regimes. In the dichotomous disturbance regime, the existing stand structure of the 70 plots was simulated assuming background mortality in most decades with infrequent severe disturbance at random times uncorrelated among plots. Landscapes developing under the dichotomous regime were heavily dominated by old-growth stands (>97%). This was true for both the 0.5 ha and 4 ha scales, although the 0.5 ha plots had a more diverse distribution of old-growth stages because the smaller plot size is more subject to erratic fluctuations in background mortality.

For simulations under the historic natural disturbance regime, the mild- and moderate-severity disturbances added significant complexity to this trajectory. Landscapes developing under the historic natural disturbance regime were still dominated by old-growth, but with a much higher proportion of other stages (21% vs 3%) and not as heavily dominated by late

transition and steady state (38% vs 71%). Three-fourths of the plots were categorized as non-equilibrium stages. However, despite the frequent occurrence of moderate disturbances, a quarter of the plots were still able to attain steady state structure. The system, however, was very dynamic, with residence times in steady state often on the order of a few decades (Fig. 2-3). Structural retrogression as a result of mild disturbances resulted in shorter residence times in all structural stages and a much more diverse mixture of stages. Simulations demonstrated that sequences of mild and moderate disturbance could generate multi-aged transition stands having a unimodal diameter distributions similar to those of even-aged stands (e.g., Fig. 2-7). In spite of this added complexity, a peak and decline pattern in biomass development remained after major disturbances (Fig. 1-3 c,d). Biomass in all stages was generally lower under the historic natural disturbance regime than under a dichotomous regime (Fig. 1-2). The number of large trees decreased in the old-growth stages, but was greater in the early stages of development as a result of legacy trees remaining after disturbances. It was clear from both field data and simulations that moderate disturbances exert a defining influence on this ecosystem (compare Fig. 2-4 vs Fig. 2-5).

The resilience of old-growth forests is of current interest because of the rarity of old-growth forests worldwide and the vulnerability of old-growth remnants to destruction by natural disturbances. In this study, old-growth forests were resilient, in the sense of retaining old-growth structure after disturbance, to the impact of a single event of up to 30% canopy removal, or even multiple events of <30% removal. However, the simulations suggest that multiple disturbances of >30% removal would cause retrogression to mature or mature-sapling mosaic stages. Under the historic natural disturbance regime, mean residence time in the old-growth stages was about 90 years, but with much variability (up to 500 years).

The multiple lines of evidence employed in this dissertation (broad-scale field survey, permanent plot re-measurements, simulations) were extremely helpful in providing a clear picture of long-term forest dynamics. For example, all three of these lines of evidence were

necessary to evaluating the pattern of long-term biomass accumulation. While they are all consistent with the peak/decline trend, the combination of all three lines of evidence is much more convincing than any single line of evidence. The differing strengths of these lines of evidence allowed them to act in a complementary way. For example, the biomass trend among permanent plots was somewhat ambiguous and the lack of late transition field plots precluded a clear evaluation of the ability of secondary cohorts to compensate for overstory attrition based on field data alone. Oftentimes, biologists are more trusting of field data than simulations. However, especially in older forests, field data are often difficult to clearly interpret because of complex stand history and site variation. For example, even the pole and mature stands shown in Chapter 1, Figure 1 were usually multi-aged and hence do not technically represent even-aged development. In cases where field data are ambiguous or difficult to interpret, models can be a helpful and complementary tool.

The use of chronosequence techniques are often necessary in reconstructing long-term trends in forest development. However, the assumption of approximate equality between the mean age of the largest trees and the time since major disturbance becomes increasingly unreliable as forests become older. Results in this study imply that this limitation of chronosequences can lead to misleading inference about biomass trends. In many cases, chronosequences are the only available tool, but such results should be interpreted with caution. Structural sequences based on multiple structural attributes of the stand as a whole (e.g., Appendix A) rather than estimated maximum tree ages may provide a more accurate interpretation of the stage of stand development in multi-aged stands.

Results of this dissertation have a number of implications for several different aspects of forest management.

Carbon management. The evidence in this study is consistent with recent hypotheses that all old-growth stands do not necessarily have negative net growth (Luyssaert et al., 2008; Lichstein et al., 2009; Keeton et al., 2011). However, in the present simulations, aboveground

live-tree biomass net growth became negative at an earlier stage, at about 210 years after clearcutting, than has been proposed in these hypotheses. Biomass net growth under the historic natural disturbance regime averaged approximately zero, even when individual stands were aggregated at a larger landscape scale and included a mixture of stages (Fig. 1-8 a). The above-ground tree component of the forest is, therefore, unlikely to remain a carbon sink late into forest development or in large old-growth reserves. Presumably, the point at which biomass ceases to accumulate will be a function of the maximum age of the trees. In northern hardwoods, mean canopy tree longevity is about 200-250 years (Lorimer et al., 2001), an age which corresponded in this study with the onset of the decline in biomass. However, carbon storage in old-growth forests is substantially higher than in pole or mature stands (Ch. 1,Fig. 2) and so old-growth forests could serve as a valuable means of carbon storage, even if their net carbon accumulation rate is approximately zero.

Conventional uneven-aged management. Single-tree selection is often implemented following the DBq method, where a smooth residual curve is computed based on a fixed q-ratio (e.g., Guldin, 1991). In northern hardwoods, this distribution is typically applied to trees from 11-60 cm, with trees larger than 60 cm always harvested and trees less than 11 cm left unmanaged (Arbogast, 1957; Crow et al., 1981). However, there is recent increased interest in harvesting smaller trees for woody biomass (Janowiak and Webster, 2010; Aguilar et al., 2013). In this study, variable q-ratios were required in the understory to maintain overstory density (Fig. 3-8), implying that harvesting of small diameter timber following a strict DBq design may render a stand unsustainable.

Ecological Forestry methods. Often, ecological forestry is performed by applying DBq management with a larger maximum diameter (Birch and Johnson, 1992; Hansen et al., 1995). However, if emulation of natural conditions is a goal, then broad application of a DBq technique would produce a higher proportion of descending monotonic curves than would probably exist in nature. In this study, only about 40% of stands in an equilibrium

population of stands followed a descending monotonic diameter distribution (Steady state + late transition; Ch. 2, Fig. 4). The application of multi-cohort management, which does not prescribe any particular residual distribution but focuses instead on gap structure and residual density, would be one approach to more closely mimic natural conditions. Retention of legacy trees may also aid resilience and speed recovery, even when few of those trees are large (Chapter 2 O'Hara and Ramage, 2013). Managers could also consider the use of other descending monotonic residuals which allow a variable q-ratio (e.g., Keeton, 2006).

Demographic sustainability. In presettlement forests, essentially any size distribution of northern hardwoods would have been sustainable given a long-enough time horizon (Ch. 3, Fig. 4). However, the rising tide of global environmental change will bring recruitment limitations with it, and therefore sustainability can no longer be assumed even with very shade-tolerant species like sugar maple, which historically had prolific regeneration. Under these conditions, the ability of a lower level of recruitment to sustain an existing overstory becomes a concern. The demographic sustainability index presented in Chapter 3 provides one tool to address that concern.

Global environmental change will be a difficult challenge for all types of models, but each type of model may have a role in clarifying different aspects of this problem. Process models are designed to provide predictions of forest growth under a changing climate, provided that the underlying forest dynamics (e.g., drivers of recruitment and mortality) do not also change. However, some novel environmental stressors will change these dynamics (e.g., invasive earthworms, devastating insects pests, and pathogens), and their abundance, population dynamics, and the resulting effects will be difficult to predict using any model. At the present time, a strength of CANOPY its ability to predict long-term demographic change, including mechanistic detail about gap capture and natural disturbances. In order to continue to provide these predictions in a scenario of environmental change that includes multiple stressors, the equations in CANOPY would need to be periodically recalibrated

with terms to represent the changing effects of climate and stress on growth, mortality, and recruitment. An example of such terms for incorporating climate effects might be the growing season average Palmer drought severity index and the number of growing degree days per season. A suitable metric for the reflecting the impact of multiple stressors might be the vigor of tree crowns. These variables could enable predictions involving various climate scenarios and interacting multiple stressors without the need for detailed submodels to predict the population and environmental dynamics of any particular stressor.

Literature Cited

Aguilar, F. X., M. J. Daniel, and L. L. Narine (2013). Opportunities and challenges to the supply of woody biomass for energy from Missouri nonindustrial privately owned forestlands. *Journal of Forestry 111*, 249–260.

Arbogast, C. (1957). Marking guides for northern hardwoods under the selection system. Research Paper LS-56, USDA Forest Service.

Birch, K. R. and K. N. Johnson (1992). Stand-level wood production costs of leaving live, mature trees at regeneration harvest in coastal Douglas-fir stands. *Western Journal of Applied Forestry 7*, 65–68.

Bormann, F. H. and G. E. Likens (1979). *Pattern and process in a forested ecosystem.* New York: Springer-Verlag.

Crow, T. R., C. H. Tubbs, R. D. Jacobs, and R. R. Oberg (1981). Stocking and structure for maximum growth in sugar maple selection stands. Research Paper NC-199, USDA Forest Service.

D'Amato, A. W. and D. A. Orwig (2008). Stand and landscape-level disturbance dynamics in old-growth forests in western Massachusetts. *Ecological Monographs 78*(4), 507–522.

Erdmann, G. G. and R. R. Oberg (1973). Fifteen-year results from six cutting methods in second-growth northern hardwoods. Research Paper NC-100, USDA Forest Service.

Guldin, J. M. (1991). Uneven-aged BDq regulation of Sierra Nevada mixed conifers. *Western Journal of Applied Forestry 6*(2), 27–32.

Hansen, A. J., S. L. Garman, J. F. Weigand, D. L. Urban, W. C. McComb, and M. G. Raphael (1995). Alternative silvicultural regimes in Pacific Northwest: Simulations of ecological and economic effects. *Ecological Applications 5*, 535–554.

Janowiak, M. K. and C. R. Webster (2010). Promoting ecological sustainability in woody biomass harvesting. *Journal of Forestry 108*(1), 16–23.

Keeton, W. S. (2006). Managing for late-successional/old-growth characteristics in northern hardwood-conifer forests. *Forest Ecology and Management 235*(1), 129–142.

Keeton, W. S., A. A. Whiteman, G. C. McGee, and C. L. Goodale (2011). Late-successional biomass development in northern hardwood-conifer forests of the northeastern United States. *Forest Science 57*(6), 489–505.

Lichstein, J. W., C. Wirth, H. S. Horn, and S. W. Pacala (2009). Biomass chronosequences of United States forests: Implications for carbon storage and forest management. In C. Wirth, G. Gleixner, and M. Heimann (Eds.), *Old growth forests: Function, fate, and value*, pp. 301–341. Berlin: Springer-Verlag.

Lorimer, C. G., S. E. Dahir, and E. V. Nordheim (2001). Tree mortality rates and longevity in mature and old-growth hemlock-hardwood forests. *Journal of Ecology 89*(6), 960–971.

Luyssaert, S., E. Schulze, A. Börner, A. Knohl, D. Hessenmöller, B. E. Law, P. Ciais, and J. Grace (2008). Old-growth forests as global carbon sinks. *Nature 455*, 213–215.

Marquis, D. A. (1991). Independent effects and interactions of stand diameter, tree diameter, crown class, and age on tree growth in mixed-species, even-aged hardwood stands. In L. H. McCormick and K. W. Gottschalk (Eds.), *Proceedings, 8th Central Hardwood Forest Conference*, General Technical Report NE-148, pp. 442–458. USDA Forest Service.

Odum, E. P. (1969). The strategy of ecosystem development: An understanding of ecological succession provides a basis for resolving man's conflict with nature. *Science 164*, 262–270.

O'Hara, K. L. and B. S. Ramage (2013). Silviculture in an uncertain world: Utilizing multi-aged management systems to integrate disturbance. *Forestry 86*, 401–410.

Tyrrell, L. E. and T. R. Crow (1994). Structural characteristics of old-growth hemlock-hardwood forests in relation to age. *Ecology 75*(2), 370–386.

APPENDIX A

Classification and dynamics of developmental stages in late-successional forests[1]

A.1 Introduction

Analysis of changes in plant species composition, forest structure, and animal populations over long periods of time in forests usually necessitates development of a chronosequence or "space-for-time substitution" because of the slow rate at which forests develop. Permanent plots provide an alternative approach with fewer assumptions (Aldrich et al., 2005). But permanent plots spanning even a modest time period of 30-60 years are uncommon in most areas of the world and usually available only in certain forest types, habitats, and for a restricted range of age classes. If a sufficient number of sample plots are taken and carefully stratified by habitat and age class, chronosequences can often yield valuable insights about certain long-term ecological trends (Walker et al. 2010). In cases where stand age can be determined accurately, trends in structural features or biotic populations can be plotted as a function of stand age or time since the last stand-replacing disturbance (Crowell and Freedman 1994; DeWalt et al. 2003).

Late-successional forests of shade-tolerant species, however, present a number of special problems that often make it difficult to develop conventional chronosequences. Even when the oldest cohort can be aged accurately, it is often difficult or impossible to determine the

[1]Corey Halpin is second author on this paper and was integrally involved in the data analysis.

time since last major disturbance using conventional tree-ring analysis. The oldest cohort could represent either a remnant of a once-dominant even-aged overstory (Antos and Parish 2002) or a more limited cohort that developed after minor or moderate disturbance in an uneven-aged stand (Parish and Antos 2006).

Furthermore, the validity of chronosequences becomes increasingly doubtful as time since last major disturbance increases and the age structure diversifies. While forests of shade-tolerant species often differ markedly in structure and disturbance history, they often have many major and minor age classes, reflecting a complex pattern of past disturbance (e.g., Splechtna et al., 2005; D'Amato and Orwig, 2008; Fraver et al., 2009). A chronosequence approach has sometimes been applied to such stands by using the age of the oldest cored tree as a proxy for stand age. But investigators have recognized that the oldest tree in a stand with many age classes and a complex disturbance history may not always bear a clear relationship with time since last major disturbance (Tyrrell and Crow 1994; Lichstein et al. 2009; Keeton et al, 2011). Even relatively young stands a few decades after heavy disturbance often contain 'legacy trees' 150-200 years old or more (e.g., Henry and Swan, 1974).

Understanding dynamic changes in old-growth forests can be especially challenging. While it is possible to recognize old growth as a single stage (e.g., Oliver and Larson 1996, DeWalt et al. 2003), there has been an increasing awareness that forests continue to change in structure and production rate after the old-growth threshold is reached. Based on simulations with the JABOWA model, Bormann and Likens (1979) predicted that after about age 125, northern hardwood forests would pass through three further stages of biomass accumulation: a late aggradation phase, a transition phase with peak biomass, and a steady state with reduced biomass and zero net growth. Spies and Franklin (1988) recognized various "degrees of oldgrowthness" in Pacific Northwest forests, and a number of investigators have provided evidence of continuous change in attributes as old-growth forests develop, such as

coarse woody debris volumes and numbers of large trees (Tyrrell and Crow 1994, Keeton et al., 2011).

An alternative to a strict chronosequence in late successional forests is to arrange stands by developmental stages based on stand structure. Despite the often complex developmental pathways in late successional forests, structural variation among stands (evidenced, for example, by diameter distributions) appears to follow a more predictable trajectory over time and is more easily quantified (Lorimer and Frelich 1998; Antos and Parrish, 2002; Podlaski, 2006; D'Amato et al., 2008, Janowiak et al., 2008). A systematic approach based on quantitative structural criteria would be useful in facilitating objective and repeatable methods of classifying stand stages. Developmental stages have significant ecological ramifications for matters as diverse as growth dynamics, wildlife habitat, aesthetics, and carbon storage (Bormann and Likens, 1979; Keeton et al, 2011). An understanding of long-term stand development under natural, 'baseline' conditions is an especially urgent task, as many rare old-growth remnants are potentially at risk of functional extirpation of dominant species by exotic insects and diseases such as the hemlock woolly adelgid (*Adelges tsugae*), emerald ash borer (*Agrilis planipennis*), and the Asian longhorned beetle (*Anoplophora glabripennis*)(e.g., Poland and McCullough, 2006; Orwig et al., 2008; Dodds and Orwig, 2011).

A rare opportunity to study long-term forest development is provided by the existence of three sizable landscapes of late successional northern hardwood forest with little past human disturbance in upper Michigan, USA. Previous work has suggested that these areas contain a mosaic of stands spanning all stages of development, from young pole stands originating after stand-replacing windstorms to old-growth, all-aged forests approaching a steady state (Lorimer and Frelich 1998). Even old-growth stands differed widely in structure, with many having unimodal size distributions with variable mean diameters and a history of one or more moderate disturbances. Especially useful metrics for interpreting stand development in that study included modal overstory diameter, the percentage of aggregate crown area in

large trees (>46 cm dbh), and the ratio of crown area in large to mature trees. Nearly linear diameter-age relationships confirmed that stands with progressively larger mean overstory diameters contained correspondingly older trees. Thus, rather than using maximum tree age in uneven-aged stands for arranging stands in a developmental sequence, this approach utilizes several overall structural metrics to categorize the stand developmental stage. Despite complex disturbance histories, stands on a given site productivity class with relatively few large trees generally had a recent history of either frequent small disturbances or else one or more moderate-to-severe disturbances. Both types of stands generally represented earlier stages of structural development as indicated by the size distributions compared to those with a higher proportion of large trees.

In this study we extend the results of Lorimer and Frelich (1998) to develop a systematic and quantitative classification of forest developmental stages for forests of shade-tolerant species based on overstory and understory structure. These stages reflect progressive changes in the form of the diameter distribution from skewed unimodal to descending monotonic. While our proposed system tracks these progressive changes in the form of the diameter distribution, the diagnostic criteria utilize more easily quantified and readily interpretable basal area metrics such as percent stand basal area in pole or large trees and the large:mature ratio rather than estimated equation parameters from fitted curves. The revised classification system recognizes eight stages of development, including 4 stages of old growth. We also evaluate the degree to which the stages reflect actual underlying temporal trends by using long-term permanent plot records on sites with known stand histories, ecological inventories of stands spanning a wide range of stages on mesic sites, and computer simulation. Simulations were performed using a model (CANOPY) that has had extensive validation tests in the same ecosystem (Choi et al., 2007; Hanson et al., 2011; Hanson et al., 2012). While the classification system has been developed for northern hardwood forests in eastern North America, and specifically in the Great Lakes region, the general approach is probably

applicable with modification to a number of other late successional temperate forests around the world.

A key point is that this method only identifies the probable position of each stand in a developmental series based on its structure, and does not identify either a specific stand age or time since last stand-replacing disturbance. The reason is that the latter metrics often have no intrinsic biological meaning in multi-aged stands or cannot be determined in stands with a complex disturbance history, even when numerous trees were cored to determine ages. As with traditional chronosequences, an underlying assumption of the method is that the stands being compared are of similar productivity, unless adjustments have been made for differences in growth rates and tree size among habitats. In mixed-species forests, it is also critical that the method be restricted to forests in which the component species have similar potential maximum diameters and similar growth rates. Potential candidates are forests dominated by shade-tolerant and relatively slow growing genera such as *Fagus*, *Acer*, *Tsuga*, *Picea*, *Abies*, and *Thuja*. The method is not generally applicable as presented here in forests in which fast-growing species of low or moderate shade tolerance are mixed with slow-growing species of high shade tolerance. For example, forests of shade tolerant species mixed with a substantial component of fast-growing species like *Pinus strobus* or *Liriodendron tulipifera* may not be suitable because large trees of the latter species may often be the same age or younger than smaller shade-tolerant canopy trees (Fajvan and Seymour, 1993; Oliver and Larson, 1996). The presence of large trees of fast-growing species would not necessarily signify that such stands are in a later stage of development than those lacking large trees.

A.1.1 Study areas

The three study landscapes in Michigan include the Porcupine Mountains Wilderness State Park, the Sylvania Wilderness on the Ottawa National Forest, and private lands protected by the Huron Mountain Wildlife Foundation (14,500 ha, 6,000 ha, and 2,500 ha of primary

forest, respectively). These lands had not been logged for various reasons, including private ownership as hunting/fishing preserves or earlier ownership by mining companies lacking a strong interest in timber production. Their status as reserves dates from the late 19th to mid-20th centuries (Christy, 1929; Rafferty and Sprague, 2001).

Climate in all areas is humid continental, with mean July and January temperatures of 19-20 C and -7 to -11 C, respectively. Mean annual precipitation is 80-90 cm and is fairly well distributed throughout the year. Sylvania is situated on a ground moraine with low topographic relief, while the Porcupine and Huron Mountains have more rugged topography with elevations up to 600 m. Only mesic and dry-mesic northern hardwood and hemlock-hardwood stands, dominated by *Acer saccharum*, *Betula alleghaniensis*, and *Tsuga candensis*, were sampled in this study. Soils in the stands are primarily spodosols (Haplorthods and Fragiorthods) of sandy loam to silt loam surface texture.

The natural disturbance regime in these areas is complex, with the primary disturbance agents being windstorms of various intensities, drought, ice storms, disease, and fire. Disturbance chronologies reveal evidence of repeated disturbances removing 10 to >70% of the forest cover on 0.5 ha tracts, with corresponding mean recurrence intervals ranging from 70 to 3700 years (Frelich and Lorimer 1991a). While all stages of stand development appear to be present, about 75% of the landscapes are covered by old-growth forest, most of which is uneven-aged with numerous age classes of varying importance.

A.1.2 Field Methods

The initial survey in 1981-1984 included seventy 0.5 ha plots. Plot size was selected to provide a sample size large enough (typically more than 165 trees > 10 cm dbh) so that the size distribution and other population parameters could be examined separately from other plots, but small enough to reflect uniform site conditions and disturbance history. Locations of plots were determined by random coordinates in advance of field inspection on

maps of primary forest in the study areas. Measurements included dbh, crown class, and gap/non-gap status of saplings and poles. Site quality was judged both by floristic habitat type (Coffman et al., 1980) and use of curves (analogous to site index) showing mean height plotted against mean dbh for each stand (Curtis, 1967; Stout and Shumway, 1982; Frelich, 1986). On the 62 temporary plots, a random sample of 10-30 increment cores were taken in each stand. A subset of 8 plots, spanning a wide range of development, was selected for permanent plot status. On the permanent plots, individual trees were mapped, and an average of 68 increment cores were taken per stand (range 39-99). Increment cores were sanded to reveal ring structure and ring widths measured to the nearest 0.001 mm under a binocular microscope with stage micrometer. The permanent plots were remeasured in 1992, 2004, and 2011, providing a 30-year record of change.

A.1.3 CANOPY model and simulation design

CANOPY is a spatially explicit, individual tree model that simulates long-term gap dynamics in large plots or stands. The model is calibrated with data from >8,000 trees on permanent and temporary plots in northeastern Wisconsin and western upper Michigan. Data were obtained from stands spanning a wide range of developmental stages, including pole, mature, and old-growth stands. Recruitment, growth, and mortality of trees is predicted on 10 x 10 m cells within larger mapped stands from initial tree size and competition level of the surrounding 900 m^2 patch. The species of recruited saplings is influenced by the species composition of the surrounding overstory. Height growth of saplings is predicted in part by presence and size of canopy gaps and sapling competition. Lateral closure of canopy gaps occurs by radial crown growth of gap border trees in four cardinal directions. Stochastic variation is incorporated into a number of model elements, including species composition of sapling recruits, diameter growth, and mortality. (For details on calibration data and model design, see Hanson et al. 2011 and Hanson et al. 2012).

CANOPY has been tested extensively and has shown good correspondence with stand growth and tree mortality reported in independent field experiments. Tests also have indicated good correspondence with details of long-term stand dynamics, such as sapling recruitment, sapling gap capture, and size distributions in old-growth stands (Hanson et al., 2011, 2012).

Version 3 of CANOPY, utilized in this paper, incorporates three new refinements. An optional subroutine (enabled by a user-controlled switch) has been added to incorporate the impact of the historic natural disturbance regime, using recurrence intervals for a range of disturbance severities as reported in Frelich and Lorimer (1991a). Initial disturbance severities are determined by a regression relating the removal of canopy cover on the 0.5 ha tracts with storm severity index (Canham et al., 2001). CANOPY adjusts storm severity index up or down as necessary to remove the designated level of crown area. Removal of individual tree sizes is based equations in Hanson (2009) relating probability of windthrow in individual trees as a function of species, size, and storm severity.

A second feature is an option to include even- or uneven-aged structure as a categorical independent variable in the tree growth and mortality equations. Tests of the calibration data indicated small but statistically significant differences in individual-tree growth and mortality rates at a given tree size and plot competition level between even-aged and uneven-aged northern hardwood stands. Two versions of the growth and mortality equations were therefore used in this study, one based on a unified set of equations for all stand stages, and one based on equations including the age-structure variable and interaction terms with competition level. Equations with the age-structure variables were used in all results reported here except for a sensitivity analysis using the unified equations without these categorical variables. Even-aged growth and mortality rates were used for any trees that originated as part of a post-disturbance sapling cohort in an opening > 0.25 ha and continued for the subsequent 200 years. This temporal marker corresponds to the mean age at time of death for

canopy trees in maple-dominated forests (Lorimer et al., 2001) and the point of transition to an uneven-aged structure, when canopy gaps are becoming common in unmanaged even-aged stands. After year 200, and for trees occurring in all other stand structures and disturbance regimes, the rates for uneven-aged stands were used.

Version 3 also includes a refinement to the sapling recruitment function. Tests of CANOPY v.2 indicated that under high plot competition levels, low ingrowth of large saplings and poles occurs from recruitment limitations and high mortality until about age 190, while abundant recruitment of large saplings is predicted in uneven-aged stands with a diverse gap structure. Both predictions match well with field data. However, stands in middle stages of old growth (190-240 years old) also frequently have evidence of restricted recruitment of understory trees, even in areas of low deer browsing, and this was not being predicted by the plot competition metric in CANOPY v.2. Based on a significant negative correlation between the percent stand basal area in mature trees and sapling density (P=0.005), CANOPY v. 3 uses a tentative simplifying assumption of delayed recruitment into the 2-6 cm dbh class whenever a 0.25 ha patch has >20% of the basal area in pole trees or >35% of the basal area in mature trees. Stands with pole or mature trees above this threshold are associated with low levels of understory development, and permanent plots exceeding these thresholds have had large reductions in understory trees in the past 30 years (cf. Fig. 6). Uneven-aged forests with a developed gap structure, on the other hand, typically have pole and mature basal area levels below these thresholds, and sapling recruitment is typically prevalent (e.g., after age 180 in Fig. 3). A sensitivity analysis was conducted by also running simulations having recruitment functions from CANOPY v. 2 lacking this feature.

In this paper, simulations were conducted on 0.5 ha tracts to match the size of plots and scale of disturbances measured in the 1980s field survey. While CANOPY can simulate stand dynamics on all three of the most common northern hardwood habitat types, the *Acer-Tsuga-Dryopteris* habitat has the widest range of age-class data in the calibration data

set and was used for all simulations reported in this paper. One set of simulations with 20 replicates was started in year 0 with a clearcut (all trees removed down to 2 cm dbh) and continued for 1,000 years under a regime of background mortality only (individual treefall gaps). Another set used the same starting conditions but simulated development on the 20 replicates for 1,000 years under the historic natural disturbance regime. Initial species composition of saplings after the disturbance was set at 80% sugar maple, 5% basswood, 5% yellow birch, 5% basswood, and 3% white ash, based on a range of reported field observations.

A.1.4 Analytical Methods

For analytical purposes, trees were grouped into four broader size classes: saplings (0-10.9 cm dbh), poles (11.0-25.9 cm), mature (26.0-45.9 cm), and large (>46.0 cm). On sites of below-average productivity (usually dry-mesic sites), the minimum threshold for large trees was reduced to 44 cm dbh based on inspection of dbh-age relationships. The lower threshold size for mature and large trees corresponds to a mean age of ∼100 and 150 years, respectively, for the three dominant species (sugar maple, hemlock, and yellow birch). The lower size threshold for large trees is below the mean size of canopy trees at time of death (51 cm; Lorimer et al., 2001).

For these species there was also a highly significant relationship between dbh and age ($P<0.0001$) (Lorimer and Frelich 1998). Mean or modal overstory diameter is therefore used as an indicator of the average canopy tree age in each stand. A 'large to mature ratio' (hereafter L/M ratio) was also computed for each stand as the ratio of aggregate basal area in large trees divided by the aggregate basal area in mature trees and used as an index of the relative developmental stage among stands. Diameter distributions were examined for each stand by individual species as well as pooled species, although only the pooled species graphs are shown in this paper because of space limitations. Disturbance chronologies for stands referred to in this paper were reported in Frelich (1986), Frelich and Lorimer (1991a),

and Dahir and Lorimer (1996).

In an earlier classification (Frelich and Lorimer 1991a), 5 stand stages were recognized (sapling, pole, mature, mature-sapling mosaic, and old growth). Each stage was defined by a dominant size cohort and a secondary cohort, together representing $\geq 67\%$ of the aggregate exposed crown area of the stand. For example, a mature stand was defined as a stand with at least 67% of the crown area in pole and mature trees, with more crown area in mature than poles, or $\geq$67% of the crown area in mature and large, with mature > large.

The decision tree in Fig. 1 was designed to retain the same basic rationale for distinguishing these 5 stages. But in the classification rules shown in Fig. 1, basal area equivalents are used. Developmental stages are distinguished in part by the amount of absolute basal area in mature and large trees (e.g., a threshold ≥ 20 m^2/ha in the case of mature and old-growth stands) and also by the percentage basal area made up by certain size classes. For example, >30% basal area in pole trees helps distinguish pole stands from sapling stands, and >45% basal area in large trees helps distinguish old-growth stands from mature stands. These thresholds were chosen to make the results of the new basal area classification as close as possible to the crown-based system. Among the four stages (pole, mature, mature-sapling mosaic, and old growth) present in the 1980s field data and in common between this system and that of Frelich and Lorimer (1991a), actual percent similarity in the classification of the 70 stands is 89%. However, changes were made in two of the stages. The 'mature-sapling mosaic' category, which formerly could include some old-growth stands, has been redefined to include only mature stands and to exclude old growth. Also, the old-growth stage has been subdivided into four separate stages, reflecting the transition from a predominantly even-aged to uneven-aged structure and development of a potentially stable size distribution in the steady state.

CANOPY simulations in conjunction with field data were used to provide further input in defining threshold criteria for stand stages in Fig. 1, especially for old-growth stands.

Fig. 3 shows how the percentage of stand basal area in saplings, pole trees, mature, and large trees are predicted to change over time following a clearcut. Poles reach a peak of more than 70% of stand basal area from ages 20-40. They then decline precipitously, reaching a minimum of <5% at age 200, and eventually stabilize to levels of 9-10% after age 400. Mature trees peak at levels exceeding 60% of stand basal area between ages 80-110, reach a minimum of ∼15% at age 270, and then increase to stable values of ∼24%. Large trees reach a peak of >70% from ages 200-300, then slowly decline to a stable level of ∼64%.

A.2 Results

A.2.1 Range of structural variation in the field data

Individual northern hardwood stands in the three upper Michigan landscapes vary widely in structural characteristics on similar mesic habitats of above-average productivity. The percentage of stand basal area in large trees (≥46 cm dbh) varied from 0-75% in different stands. The ratio of basal area in large to mature trees (25-45.9 cm dbh) ranged from 0 to 4.6, and modal diameter of the overstory cohort (dominant, codominant and intermediate crown classes) ranged from 10 to >50 cm dbh.

The sample of stands shown in Fig. 2a-f illustrates the typical variation in size distributions among the 70 stands. Stands with unimodal, bimodal, and descending size distributions are all common on these landscapes. As modal overstory diameter increased from 10 to ∼35 cm dbh, the size distribution of all species and crown classes shifted from a steeply descending curve with a small mean diameter (Fig 2a) to a distinctly unimodal curve with minimal understory development (Fig 2b, c).

Stands with a higher overstory modal dbh of 40-45 cm had broader and flatter size distributions, with evidence of a re-developing understory (Fig. 2d). These distributions have a characteristic overstory 'bulge' with a steeply descending 'tail' in the smaller size classes (hereafter termed 'compound' size distributions). When the overstory attained a modal dbh

of >45-50 cm, the identity of the main overstory cohort was less distinct and secondary cohorts of smaller trees are present, causing the curve to be skewed toward smaller size classes (Fig. 2e). About 26% of the stands conformed to the classic descending monotonic curves of relatively balanced, uneven-aged forest, in which evidence of a single dominant overstory cohort is muted or lacking (Fig. 2f).

A.2.2 Stand stage criteria and descriptions

Sapling and pole stages

The residence time of sapling northern hardwood stands is very brief (<20 years in simulations after a clearcut on ATD habitat), and none of the 70 randomly-selected field stands fell into this category. Simulations after a clearcut on ATD habitat suggest that at age 10, saplings average 57% and poles 42% of the stand basal area.

Pole stands are characterized by less than 20 m^2/ha of basal area in mature and large trees, with pole trees making up >30% of the stand basal area (Fig. 1). Pole stands made up 9% of the initial (1981-84) field sample. Four of the six pole stands occurred in the west-central uplands of the Porcupine Mountains landscape; three of these developed after a historically documented severe windstorm in June 1953 (Rafferty and Sprague 2001) and the fourth after a severe disturbance about 1960 as reconstructed from increment cores. For all 6 pole stands, large trees (≥46 cm dbh) in the first measurement in 1981-84 averaged only 10% of the stand basal area, with only 6 trees/ha > 50 cm dbh (Table 1). Although young pole stands usually have a steeply descending size distribution when all species and crown classes are pooled, the size distribution is typically unimodal when only trees with crowns exposed to direct skylight are included (Fig. 2a). The 30-year permanent plot record in a Porcupine Mountains pole stand originating after the 1953 storm demonstrates a shift toward a more unimodal curve in recent measurement periods (Fig. 6).

Simulations after a clearcut on ATD habitat suggest that even-aged stands remain in the

pole stage from about age 20 to age 70. During this time, the simulated proportion of stand basal area in poles declined from 74 to 42%, mature trees increased from 4 to 54%, and large trees from 0 to 2%.

Mature stage

Mature stands are defined as having > 20 m^2 of basal area in mature and large trees, with large trees making up <45% of total stand basal area (Fig. 1). Modal overstory diameter in mature stands in the field data usually ranged from 14-42 cm. The mean proportion of stand basal area in large trees in the mature field stands averaged 36%, with a mean of 43 trees/ha >50 cm dbh (Table 1). Despite their prevalence in the modern human-modified landscape, mature stands made up only 11% of the reserved study landscapes.

Based on post-clearcut simulations, even-aged stands remained in the mature stage, as defined structurally in Fig. 1, from ages 80-140. During this time, the proportion of stand basal area in poles decreased from 32% to 6%, mature trees decreased from 62% to 46%, and large trees increased from 5 to 48%.

The mature stands in the field sample were typically uneven-aged, with some legacy trees 150-250 years old. Disturbance chronologies usually show evidence of more than one episode of moderate disturbance rather than a single stand-replacing event. Field evidence and simulations both suggest that most mature stands in the study areas have more commonly retrogressed from old-growth stands following moderate disturbance, rather than developed 'upward' from sapling-pole stands after stand-replacing disturbance (Frelich and Lorimer 1991b).

Mature-sapling mosaic

The mature-sapling mosaic stage includes stands that have 10-20 m^2/ha of basal area in mature and large trees – less than in a mature stand, but without the high concentration of poles typical of a pole stand. Only two of the 70 field stands fell into this category. Several

lines of evidence indicate that these stands were formerly older stands that experienced recent moderate disturbance. While the basal area in mature and large trees is only about half that of the average mature stand (Table 1), diameter distributions in both stands were descending monotonic with scattered large trees up to 80 cm dbh. Trees cored between 32 and 48 cm dbh ranged in age from 165 to 187 years. Also, most of the simulated mature-sapling mosaic stands had retrogressed from older stands, especially from early transition or steady state, after moderate disturbance. Simulated post-clearcut stands rarely passed through the mature-sapling mosaic stage but moved directly from pole to mature and then to early transition (Fig. 4). Based on simulations, residence time in this stage under the historic natural disturbance regime averaged only 13 years.

Early transition stage

In post-clearcut stands, the transition stages correspond to the point in time when the original even-aged overstory is starting to break up, or in the words of Bormann and Likens (1979), "starting to lose its grip on the site." It is also comparable to the 'breakdown' or 'degradation' stage recognized by some European ecologists (Emborg et al., 2000; Podlaski, 2008; Kral et al, 2010). We recognize three stages that correspond to the Bormann-Likens transition stage. The earlier phase corresponds to when the initial overstory cohort is still mostly intact and size distributions generally are still strongly unimodal (Fig. 2c). The middle stage occurs when mortality of the initial cohort is underway and an understory of trees > 2 cm dbh is beginning to develop, but the canopy is still largely closed. The late transition stage occurs when breakup of the cohort is fairly advanced and canopy gaps and gap sapling/ pole groups are common.

The early transition stage represents the earliest stage of old-growth development, with >45% of stand basal area in large trees. Understory trees > 2 cm dbh are usually sparse, although stands sometimes have a compound size distribution with a developing 'tail'. However, these larger saplings, when present, are usually suppressed beneath a closed canopy and

do not occur in gaps (Fig. 2c). L/M ratios in the early transition field stands were usually relatively low compared to other old growth stages, with 69% of the stands having L/M ratios between 1.1-1.4. Modal overstory dbh in the field data commonly ranged between 35 and 42 cm. Based on simulation trials, even-aged stands in the early transition stage on ATD habitat range in age from 140-160 years.

Early transition stands can also result from structural retrogression of old-growth stands from later developmental stages. In the field data, all of early transition stands had an uneven-aged structure, often with some trees >200 years old, even if the size distribution was strongly unimodal. Some (15-23%) of the early transition field stands also had > 14% saplings and poles or a higher L/M ratio of 1.6-1.7. Early transition stands with a unimodal distribution were not distinguished from those with a skewed descending curves in the classification because there appeared to be no consistent differences in stand history either in the field data or simulations. Simulations suggest that both types of size distributions could sometimes result from either 'upward' development from an earlier stand stage or 'retrogression' from a more advanced stage following disturbance.

Mid-transition stage

Mid-transition stands are generally characterized by compound size distributions, but they almost always retain a single and strongly identifiable bulge in the overstory size classes, with a peak typically at 45-50 cm dbh (Fig. 2d). Both the field data and the simulations show that stands with these features are characterized by maximum abundance of large trees and minimal understory development (Fig. 2d and 2i, Fig. 5, Table 1). In the decision tree, stands progress from early transition to mid-transition as soon as the L/M ratio exceeds 1.75 and the percentage of stand basal area and saplings and poles drops below 8% (maple stands) or 10% (hemlock stands). The maximum sapling-pole development in this stage is below the mean level normally found in steady state stands in the field data and simulations (Fig. 3). In simulations of post-clearcut stands on ATD habitat with no subsequent exogenous

disturbance, mid-transition is a stage of long duration (80 years), from ages 170 to 250. Together, the early and mid-transition stages occupy 39% of the study area landscapes.

All of the indices of large tree development (% basal area in large trees, L/M ratio, density of trees >50 cm dbh) reach a maximum in this stage in the field data (Table 1) and simulations (Fig. 5). For example, density of trees >50 cm dbh averaged 84 trees/ha in the mid-transition field data, compared to 63 in early transition and 66 in late transition. The simulations likewise predict a peak value of 85 trees/ha >50 cm dbh at age 220, after the midpoint of the mid-transition stage. Understory development in mid-transition was less than in any other stand stage, perhaps because of continued attrition of overtopped pole trees and despite the development of smaller saplings. Saplings and poles averaged 7.0% of stand basal area in mid-transition field stands compared to 10.3% for early transition, 12.7% in late transition, and 14.8% in steady state.

Late transition stage

Late transition stands differ from mid-transition in having much more evidence of understory development (sapling + pole basal area 10-19% of total) and a less pronounced cohort of large trees. The overall distribution is typically skewed toward the smaller size classes (Fig. 2e). However, a modest peak is usually present in the overstory size distribution at ∼50-55 cm dbh – larger than the modal diameter of mid-transition stands. This peak is much more evident if size class basal area is plotted on the y axis rather than number of trees. There is often a secondary peak in a smaller size class, usually < 35 cm dbh.

Although the understory is usually well developed, larger gap saplings 2-10 cm dbh are still relatively uncommon (<0.6% of stand basal area). For late transition stands originating after stand-replacing disturbance, this pattern would be expected from a breakup of the even-aged cohort and the incipient development of one or more secondary cohorts in the gaps. L/M ratio is set at a minimum of 1.7 but often ranges from 2.0 to 2.7. However, late transition has a lower density of large trees than in mid-transition as noted above.

In CANOPY post-clearcut simulations on ATD habitat, late transition is of short duration, lasting from ages 260-270. Functionally, this stage may last up to 80 years longer because simulated stand volume continues to decline until age 350. But from a structural perspective, the diameter distributions and volumes for 280-350 year old stands are not distinguishable from temporal variations among steady-state stands more than 400 years after the clearcut (Fig. 4). In spite of the short simulated residence time, late transition field stands as classified by the criteria in Fig. 1 are relatively common on the landscape. Simulations suggest that the reason is a high frequency of retrogressions into this stage from the steady state because of minor disturbances, as well as from random fluctuations in age-related background mortality on small tracts such as the 0.5 ha plots (Fig. 4).

Steady state stands

The steady-state stands in the field data had diameter distributions that conform reasonably closely to the descending monotonic curves (e.g., rotated sigmoid or negative exponential on semi-logarithmic axes) that have long been described for all-aged stands. Disturbance chronologies verified that these stands typically have many age classes and lack of severe disturbance for the past 150-200 years. Minimum criteria for steady state include a L/M ratio >1.4, sapling + pole component of 10-19%, and large tree component of >45% (Fig. 1). Saplings and poles in canopy gaps become abundant for the first time since stand initiation (gap saplings $\geq$ 0.6% of stand basal area). Simulated stands on ATD habitat after clearcut but with no further disturbance reached the steady state after about age 280.

Steady-state stands as defined in Fig. 1 occupied 26% of the study area landscapes, more than any other single stage (Table 1). Structurally, steady-state stands showed only minor differences from late transition, mainly in the prominence of gap saplings and (usually) a less irregular size distribution. But otherwise the structural attributes were almost identical (Table 1). Together, late transition and steady state occupied 39% of the study area landscapes, the same as with early transition + mid-transition. All four old-growth stages thus

occupied 78% of the study area landscapes.

Simulated stands subject only to small treefall gaps for many centuries had a basal area distribution of ∼2.3% saplings, 10% poles, 24% mature trees, and 64% large trees (Fig. 3). Minor disturbances might slightly decrease the proportion of large trees in actual landscapes – the field data in stands of this stage (n=18) had a mean of 3.6% saplings, 11% poles, 27% mature, and 59% large (Table 1).

Some peaks and irregularities in the diameter distributions, resulting from relatively minor disturbances, are accepted in this stage as long as the quantitative criteria in Fig. 1 are met. The rationale for a minimum gap sapling component of 0.6% of total basal area (mean for steady-state stage in the field data = 1.4%, maximum = 4.7%) is that a highly balanced, all-aged stand is presumed to have a fairly constant background mortality rate of canopy trees and recruitment of gap saplings in each decade. If there is a quiescent period of 20 or 30 years with little gap formation, a stand can relapse into the late transition stage even without exogenous disturbance. Simulations suggest that this could often happen on relatively small tracts $\leq$ 0.5 ha, but is much less common on larger tracts such as 4 ha (Fig. 4). Alternatively, if a disturbance increases the component of saplings and poles above 19% of total basal area and lowers the proportion of large trees, the stand may relapse into the early transition stage (Fig. 1).

A.2.3 Evaluation of the classification system

Several lines of evidence were used to evaluate the degree to which the recognized stages mimic underlying temporal trends in stand dynamics. The first line of evidence is independent verification of trends in the size distributions using the CANOPY model. The simulated distributions for a wide range of stand age and time since clearcutting (Fig. 2 g-l) were compared with the developmental sequence of size distributions reconstructed from structural variation in the field data (Fig. 2a-f). The model mimics the inferred sequence in the field

data as size distributions progress from steeply descending to unimodal and then to broad, flat distributions with a developing understory. By age 280-300, old-growth stands develop a descending monotonic distribution that remains largely stable for hundreds of years in the absence of further disturbance (Fig. 2f, 4). Aside from the general similarity between observed and simulated size distributions, comparisons also show a close correspondence between specific metrics such as basal area or number of trees in pole, mature, and large size categories within a given stage (Fig. 5).

A second line of evidence is the trajectories of stand development observed on the eight permanent 0.5 ha mapped tracts over a span of 30 years (Fig. 6). Two of the stands advanced in stage during the 30 years, and in the expected manner. The pole hardwood stand met the threshold criteria of a mature forest by 2004, and the mature hardwood stand had also crossed the threshold for late aggradation by 2004. The three hemlock-dominated stands in early and mid- transition remained in those stages, as did the three steady-state hardwood stands.

Diameter distributions on these permanent plots also changed in ways that would be expected, based on the developmental sequence postulated in Fig.1. During the shift from pole to mature and from mature to early transition, the stands had major losses in the number of small trees, making the diameter distributions more unimodal in form. The mid-transition hemlock stands also had net losses in small-medium trees, although this was likely aggravated by the high levels of deer browsing in hemlock stands bordering Lake Superior (Frelich and Lorimer 1985). The three steady-state hardwood stands, in contrast, had little change in the descending monotonic (rotated sigmoid) size distributions, aside from the expected increase in the mean dbh and corresponding reduction in stem density of trees in a few peak size classes (Fig. 6).

A third line of evidence is the order of stand stage development in the simulations. In post-clearcut simulations, stands progressed in order through each one of the stages sequen-

tially, from sapling stand to steady state (stages 1 to 8, but excluding mature-sapling mosaic; Fig. 4). Once stands reached the steady state, they remained in that stage through the end of the simulation (year 1000) in the absence of further disturbance.

Simulations also supported the interpretation that structural retrogression of stands to earlier stages of development after a series of light or moderate disturbances could result in stands with diameter distributions similar to those of even-aged stands. For example, Fig. 7 shows the retrogression of a steady state stand to a mid-transition stand with a prominent bulge in the size distribution as predicted by CANOPY.

A.3 Discussion

A.3.1 Long-term trends in stand development

The stand stages as described here confirm previously hypothesized temporal trends in structural development as mean tree diameter increases and age structure diversifies (e.g., Kral et al. 2010), but supporting systematic and quantitative data across a complete range of stand stages have previously been difficult to obtain. Both the field data and the CANOPY simulations support the hypothesis of Bormann and Likens (1979) that the number of large trees in a northern hardwood stand reaches a maximum toward the end of the lifespan of an even-aged cohort and declines thereafter. In the field data, the maximum abundance of trees $>$ 50 cm dbh (84 trees/ha) is reached in the mid-transition stage when there is still a distinct bulge in the overstory size distribution but minimal understory development. The CANOPY model predicts a peak of 86 trees/ha of this size range at age 220 on the ATD habitat type, which corresponds approximately to the mean age of death (216 years) of canopy sugar maples observed in the field data (Lorimer et al. 2001). In both field data and simulations, there is an apparent decline to a level of $\sim$60 large trees/ha in the steady state, as well as an increase in the prominence of sapling and pole trees. Similar results were obtained with simulations on the more productive *Acer-Osmorhiza-Caulophyllum* habitat

type and the less productive *Acer-Tsuga-Maianthemum* type.

Both field data and simulations also support the traditional concept that an old-growth stand with a descending monotonic size distribution has a relatively stable and sustainable population over long periods of time in the absence of major disturbance (e.g., Motta et al., 2011; Kucbel et al., 2012). One might also be able to make an a priori case for stability of stands with unimodal or compound distributions in certain cases or on certain sites. For example, slow growth of large trees and limited recruitment under heavy crown cover (e.g., Goff and West, 1975) could possibly result in a stable bulge in the overstory and a persistent deficit in small trees. However, no evidence was found that unimodal or compound distributions were stable for more than a decade on any of the habitat types investigated here. All such stands showed progressive changes over time in field data and simulations. Similar progressive changes in the form of 'compound' size distributions based on permanent plot records were reported over a span of 28 years for a sugar maple stand in Wisconsin (Lorimer and Krug, 1983) and over 35-40 years in European beech stands (Fagus sylvatica) (von Oheimb et al., 2005; Kucbel et al., 2012).

The rather high proportion of steady state stands (26%) defined by structural criteria in this analysis contrasts with an earlier analysis indicating only 4% steady state when the steady state is defined by disturbance history criteria (Frelich and Lorimer, 1991b). The differences are explained by evidence in both field data and simulations that moderately light disturbances may not always have enough impact on stand structure to cause retrogression to an earlier structural stage. For example, the three steady-state permanent plots in this study all have evidence of several decades with 10-20% canopy removal in the past 150 years (Frelich and Lorimer, 1991a; Dahir and Lorimer, 1996), but these caused only minor irregularities in the size distributions (Fig. 6).

The close correspondence of the multi-aged stands in the field data with simulated development after clearcutting supports the hypothesis that structural stages in the even-aged and

multi-aged pathways are very similar. In most cases, even-aged and multi-aged stands with similar modal overstory diameters cannot be readily distinguished by the size distribution (e.g., Fig. 2). The main exception may be that old-growth stands in later developmental stages may retain more understory trees following moderate disturbance than even-aged stands with a comparable mean overstory diameter. Also, a mature or old growth stand with 2 or 3 widely separated age classes would likely be distinguishable structurally from an even-aged stand of similar mean age.

A.3.2 The role of simulation in inferring developmental trajectories

The CANOPY model does not include features intrinsically related to stand stages or diameter distributions, and so the field data and model predictions provide largely independent evidence of long-term stand development. The predictions in Fig. 2 g-l are only determined by the underlying recruitment, growth, and mortality equations of individual trees in response to tree size and local competitive environment in a 900 m^2 neighborhood, as calibrated from a large regional data set. And while both field and simulated stage boundaries are defined by constraining certain parameters within a specified range (e.g., L/M ratio in some cases), there are no imposed limits within a stage on the number per ha or basal area per ha for trees of any size class. The close correspondence of predicted vs. observed metrics (Fig. 5) suggests that the equations in CANOPY are providing reasonable long-term projections of stand development patterns.

The sensitivity analysis in which simulations used the earlier recruitment function from CANOPY v. 2 indicated that the modified function in v. 3 provides better predictions of size distributions in the mid-transition stage. The function in CANOPY v.2, for example, doesn't predict the low level of pole basal area (mean of ∼5%) evident in the mid-transition field data. Consequently, with the earlier recruitment function, even-aged stands often progressed directly from stage 5 (early transition) to stage 8 (steady state), skipping over the structural

features of mid- and late transition stages seen in the field data (e.g., Fig. 2d).

Sensitivity analysis with the unified growth and mortality equations, on the other hand, suggested relatively little difference in magnitude or timing of model predictions. For example, when using separate equations for even-aged and uneven-aged stands, most simulation replicates had reached the early transition stage by year 150; with the unified equations, this occurred about 10 years later. But there was no consistent trend among all stand stages. With the unified equations, many replicates had reached the mature stage and steady state about 10 years sooner than with the separate equations.

A.3.3 Applications of the method

The method for identifying stand stages in this paper is quantitative, easy to apply, and easily replicated since only basal area calculations of specific size classes are needed. Other approaches and criteria for distinguishing stand stages are also possible. For example, various mathematical distributions such as the single or double Weibull distribution can be fit to observed size distributions and inferences made about stand stages based on the equation parameters or the theoretical distributions having the best fit (e.g., Podlaski, 2006). An advantage of the classification system described here is that each stage is based on multiple criteria that can be precisely measured (basal area of saplings, poles, L/M ratio, etc.) and applied uniformly to any stand of comparable species composition and site quality. By contrast, curve-fitting methods in our experience are often subject to more error in obtaining a good fit to the raw data because of limited flexibility in the underlying equations and the complexity of non-linear curve fitting, especially when applied to irregular or multimodal size distributions. It may be possible to classify stages by identifying threshold values in the equation parameters, but probably these threshold values would have to be determined by examining the same kinds of variables discussed in this paper.

While several lines of evidence suggest that these developmental stages tend to follow a

logical chronological development following major disturbance, the classification is nevertheless not a strict chronology and is based only on measurable differences in stand structure. CANOPY simulations suggest that stands may sometimes skip one or more structural stages or retrogress one or more stages, especially in old-growth stands after minor or moderate disturbance. The reason is that after moderate disturbance, old-growth stands often retain many large and small trees that place them near the threshold of two or three different structural stages. For example, a steady-state stand may retrogress after moderate disturbance to early transition because of a drop in the number of large trees. But it may retain sufficient small, medium and large trees – as well as a descending monotonic size distribution – that allow it to quickly recover to a steady state structure without going through the mid- or late transition stages (cf. Fig. 2). This is especially likely if the steady-state stand initially had a low L/M ratio (e.g., 1.4-1.6) and only modest densities of mature trees capable of growing into larger size classes. Also, it is conceivable that two stands with overstory trees of the same age sometimes could be placed in different stages if they differ markedly in other attributes such as understory density. From a structural perspective, different stages may be warranted since they differ in attributes of ecological importance (e.g., songbird habitat), and the time required for the two stands to reach a more advanced stage of development would likely be different in the two cases.

The basic method described here could likely be adapted for other suitable forest types. This could be done by recognizing site-specific or species-specific size thresholds for sapling, pole, mature, and large trees based on local dbh-age relationships. One could likewise modify the stand stage criteria (Fig. 1) to fit observed structural variations among the diameter distributions in other forest types, analogous to those shown in Fig. 2. Size distributions with these curve forms can be observed across a fairly wide range of stand size, but investigators should keep in mind that size distributions (and hence thresholds for defining stand stages) can be affected by the spatial extent of sampling. In order to minimize potential confounding

effects of variation in habitat features and disturbance history on the size distribution, we recommend that sample plots within a single stand not be distributed over more than 8-10 hectares unless site features and stand history are known to be uniform over a larger area.

Literature Cited

Aldrich, P.R., Parker, G.R., Romero-Severson, J., Michler, C.H. 2005. Confirmation of oak recruitment failure in Indiana old-growth forest: 75 years of data. For. Sci. 51, 406-416.

Antos, J.A., Parish, R., 2002. Dynamics of an old-growth, fire-initiated, subalpine forest in southern interior British Columbia: tree size, age, and spatial structure. Can. J. For. Res. 32, 1935-1946.

Bormann, F.H., Likens, G.E. 1979. Pattern and process in a forested ecosystem. Springer-Verlag, New York.

Canham, C.D., Papaik, M.J., Latty, E.J., 2001. Interspecific variation in susceptibility to windthrow as a function of tree size and storm severity for northern hardwood tree species. Can. J. For. Res. 31, 1-10.

Christy, B.H. (ed)., 1929. The Book of Huron Mountain. Huron Mountain Club, Chicago, IL.

Coffman, M.S., Alyanak, E., Resovsky, R. 1980. Field guide, habitat classification system for upper peninsula of Michigan and northeast Wisconsin. Michigan Technological University, Houghton, MI.

Crowell, M., Freedman, B. 1994. Vegetation development in a hardwood-forest chronosequence in Nova Scotia. Can. J. For. Res. 24, 260-271.

Curtis, R.O. 1967. Height-diameter and height-diameter-age equations for second-growth Douglas-fir. For. Sci., 13, 365-375.

D'Amato, A.W., Orwig, D.A., 2008a. Stand and landscape-level disturbance dynamics in old-growth forests in western Massachusetts. Ecol. Monogr. 78, 507-522.

D'Amato, A.W., Orwig, D.A., Foster, D.R., 2008b. The influence of successional processes and disturbance on the structure of Tsuga canadensis forests. Ecol. Appl. 18, 1182-1199.

Dahir, S.E., Lorimer, C.G., 1996. Variation in canopy gap formation among developmental stages of northern hardwood stands. Can. J. For. Res. 26, 1875-1892.

DeWalt, S.J., Maliakal, S.K., Denslow, J.S. 2003. Changes in vegetation structure and

composition along a tropical forest chronosequence: implications for wildlife. For. Ecol. Manage. 182, 139-151.

Dodds, K.J., D.A. Orwig. 2011. An invasive urban pest invades natural environmentsâĂŤAsian longhorned beetle in northeastern US hardwood forests. Can. J. For. Res. 41, 1729-1742.

Emborg, J., Christensen, M., Heilmann-Clausen, J. 2000. The structural dynamics of Suserup Skov, a near-natural temperate deciduous forest in Denmark. For. Ecol. Manage. 126, 173-189.

Fajvan, M.A., Seymour, R.S., 1993. Canopy stratification, age structure, and development of multi-cohort stands of eastern white pine, eastern hemlock, and red spruce. Can. J. For. Res., 23, 1799-1809.

Fraver, S., White, A.S., Seymour, R.S., 2009. Natural disturbance in an old-growth landscape of northern Maine, USA. J. Ecol. 97, 289-298.

Frelich, L.E., Lorimer, C.G., 1985. Current and predicted long-term effects of deer browsing in hemlock forests in Michigan, USA. Biol. Conserv. 34, 99-120.

Frelich, L.E., Lorimer, C.G. 1991a. Natural disturbance regimes in hemlock-hardwood forests of the upper Great Lakes region. Ecol. Monogr. 61, 145-164.

Frelich, L.E., Lorimer, C.G. 1991b. A simulation of landscape-level stand dynamics in the northern hardwood region. J. Ecol. 79, 223-233.

Goff, F.G., West, D. 1975. Canopy-understory interaction effects on forest population structure. For. Sci. 19, 97-104.

Hanson, J.J.. 2009. Emulating natural disturbance dynamics in northern hardwood forests: long-term effects on species composition, structure, and yield. Ph.D. Dissertation, Univ. of Wisconsin-Madison, Madison, WI, USA.

Hanson, J.J., Lorimer, C.G., Halpin, C.H. 2011. Predicting long-term sapling dynamics and canopy recruitment in northern hardwood forests. Can. J. For. Res. 41, 903-919.

Hanson, J.J., Lorimer, C.G., Halpin, C.R., Palik, B.J., 2012. Ecological forestry in an uneven-aged, late-successional forest: Simulated effects of contrasting treatments on structure and yield. For. Ecol. Manage. 270, 94-107.

Janowiak, M.K., Nagel, L.M., Webster, C.R., 2008. Spatial scale and stand structure in northern hardwood forests: implications for quantifying diameter distributions. For. Sci. 54, 497-506.

Kral, K, Vrska, T., Hort, L., Adam, D., Samonil, P., 2010. Developmental phases in a

temperate natural spruce-fir-beech forest: determination by a supervised classification method. Eur. J. For. Res. 129, 339-351.

Kucbel, S., Saniga, M., Jaloviar, P., Vencurik, J., 2012. Stand structure and temporal variability in old-growth beech-dominated forests of the northwestern Carpathians: a 40 years perspective. For. Ecol. Mange. 264, 125-133.

Lorimer, C.G., Krug, A.G., 1983. Diameter distributions in even-aged stands of shade-tolerant and midtolerant tree species. Am. Midl. Nat. 109, 331-345.

Lorimer, C.G., Frelich, L.E., 1984. A simulation of equilibrium diameter distributions of sugar maple (*Acer saccharum*). Bull. Torrey Bot. Club 111, 193-199.

Lorimer, C.G., Frelich, L.E., 1998. A structural alternative to chronosequence analysis for uneven-aged northern hardwood forests. J. Sustain. For. 6, 347-366.

Lorimer, C.G., Dahir, S.E., Nordheim, E.V., 2001. Tree mortality rates and longevity in mature and old-growth hemlock-hardwood forests. J.Ecol. 89, 960-971.

Motta, R., Berretti, R., Castagneri, D., Dukic, V., Garbarino, M., Govedar, Z., Lingua, E., Maunaga, Z., Meloni, F., 2011. Toward a definition of the range of variability of central European mixed Fagus-Abies-Picea forests: the nearly steady-state forest of Lom (Bosnia and Herzegovina). Can. J. For. Res. 41, 1871-1884.

Oliver, C.D., Larson, B.C. 1996. Forest Stand Dynamics. Update ed., John Wiley & Sons, New York.

Orwig, D.A., Cobb, R.C., D'Amato, A.W., Zizlinski, M.L., Foster, D.R., 2008. Multi-year ecosystem response to hemlock woolly adelgid infestation in southern New England forests. Can. J. For. Res. 38, 834-843.

Parish, R., Antos, J.A., 2006. Slow growth, long-lived trees, and minimal disturbance characterize the dynamics of an ancient, montane forest in coastal British Columbia. Can. J. For. Res. 36: 2826-2838.

Podlaski, R., 2006. Suitability of the selected statistical distributions for fitting diameter data in distinguished development stages and phases of near-natural mixed forests in the Swietokrzyski National Park (Poland). For. Ecol. Manage. 236, 393-402.

Podlaski, R., 2008. Dynamics in central European near-natural Abies-Fagus forests: Does the mosaic cycle approach provide an appropriate model? J. Veg. Sci. 19, 173-182.

Poland, T.M., McCullough, D.G., 2006. Emerald ash borer: invasion of the urban forest and the threat to North America's ash resource. J. For. 104, 118-124.

Rafferty, M., Sprague, R., 2001. Porcupine Mountains Companion: Inside Michigan's Largest State Park , 4th ed., Neqauket Natural History Associates, White Pine, MI.

Spies, T.A., Franklin, J.F. 1988. Old growth and forest dynamics in the Douglas-fir region of western Oregon and Washington. Nat. Areas J. 8, 190-201.

Splechtna, B.E.,Gratzer, G., Black, B.A., 2005. Disturbance history of a European old-growth mixed-species forest – a spatial dendro-ecological analysis. J. Veg. Sci. 16, 511-522.

Stout, B.B., Shumway, D.L., 1982. Site quality estimation using height and diameter. For. Sci., 28, 639-645.

Tyrrell, L.E., Crow, T.R., 1994. Structural characteristics of old-growth hemlock-hardwood forests in relation to age. Ecology 75, 370-386.

Von Oheimb, G., Westphal, C., Tempel, H., Hardtle, W., 2005. Structural pattern of a near-natural beech forest (Fagus sylvatica) (Serrahn, North-east Germany). For. Ecol. Manage. 212, 253-263.

Walker, L.R., Wardle, D.A., Bardgett, R.D., Clarkson, B.D., 2010. The use of chronosequences in studies of ecological succession and soil development. J. Ecol., 98, 725-736.

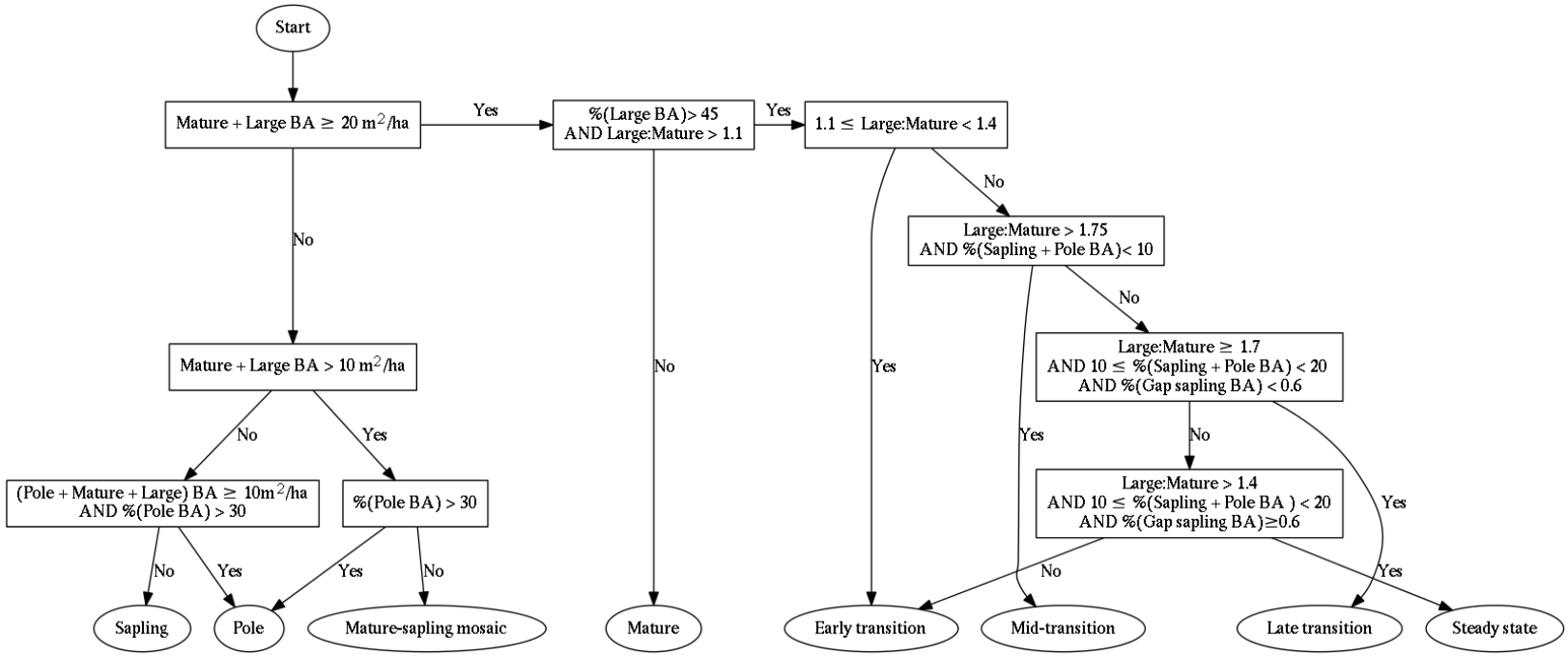

Figure 1: Dichotomous key for classifying structural stand stages. All metrics (including large:mature ratio) are expressed in terms of absolute or relative basal area (BA) of broad size classes. Size classes are: sapling (<11 cm dbh), pole (11 - 25.9 cm), mature (26-45.9), and large (≥46 cm).

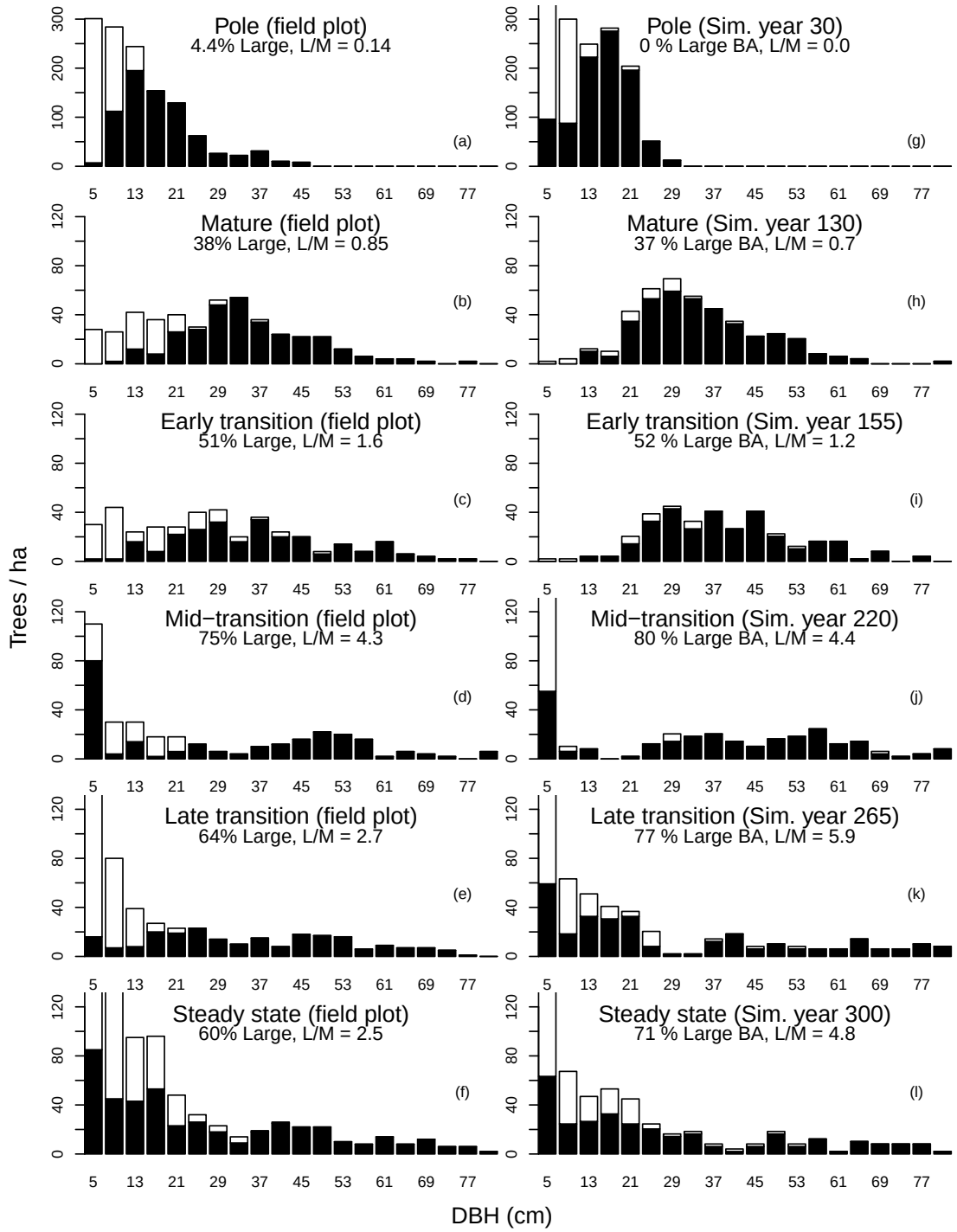

Figure 2: Representative size distributions for each structural stage on *Acer-Tsuga-Dryopteris* habitat. (a-f): Distributions from half-hectare field plots. (g-l): distributions from simulations developing after clearcut that removed all trees larger than 2 cm dbh and subsequently with only background mortality. Note that the earlier stages of field plots may include some older residual trees.

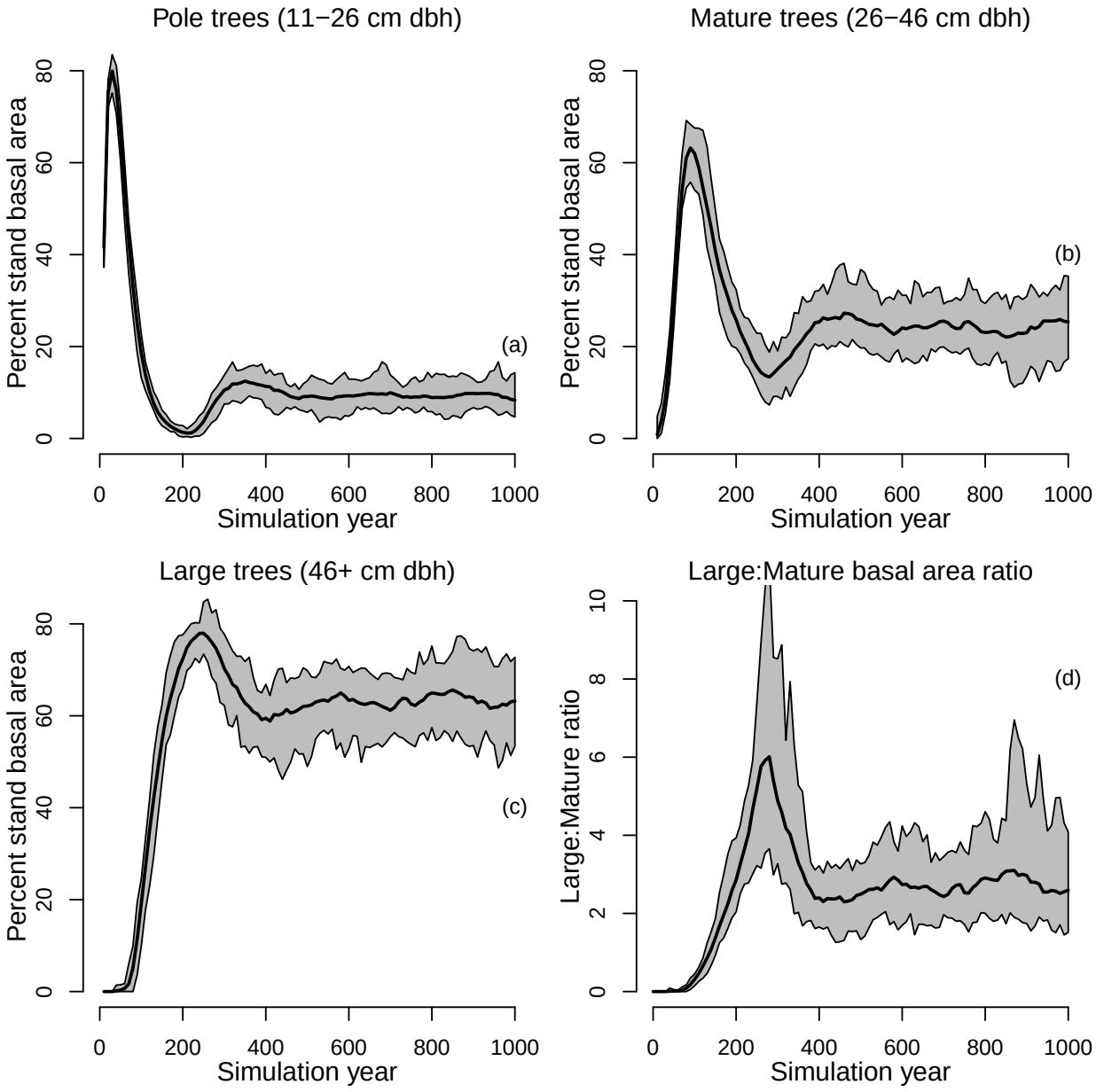

Figure 3: CANOPY simulations of the percent basal area occupied by various size classes as a function of time since clearcutting on half hectare plots. Thick line denotes the mean and the grey band denote the range of variation over 20 replications.

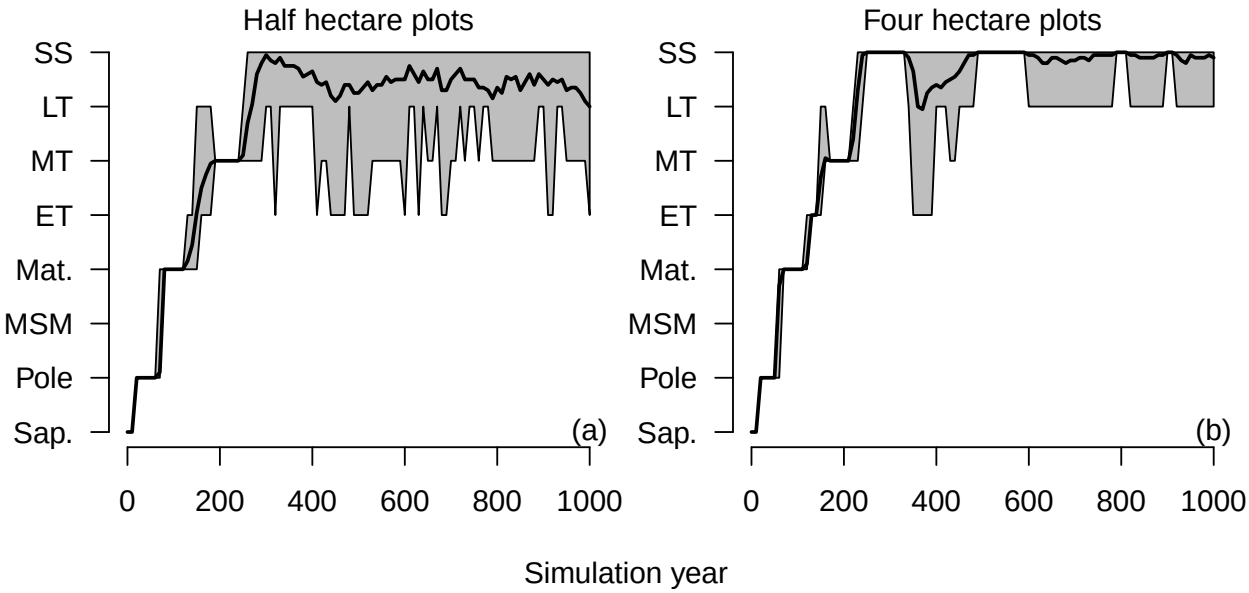

Figure 4: Mean simulated stand stage and range of variation on 20 replications after clearcutting and subsequent background mortality on (a) half-hectare plots and (b) four-hectare plots. Abbreviations were MSM: Mature-sapling mosaic, Mat: Mature, ET: Early transition, MT: Mid-transition, LT: Late transition, SS: steady state.

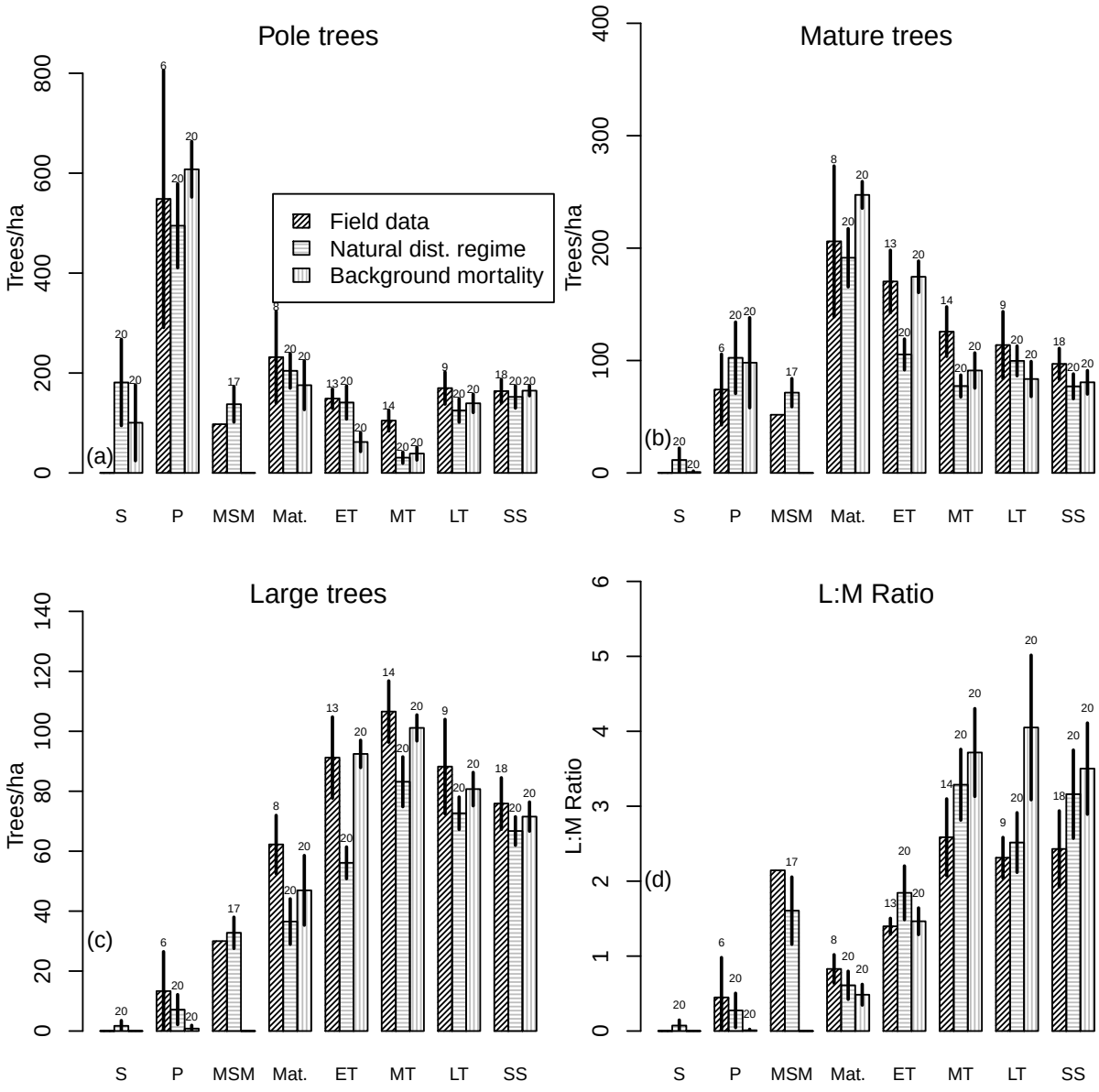

Figure 5: Comparison of the observed and simulated numbers of trees per hectare in broad size classes for each structural stage. Simulations started after a clearcut that removed all trees larger than 2 cm dbh. Bars give the range of variation and numbers above them reflect the sample size. Abbreviations were S: Sapling, P: Pole, others as in Fig. 4.

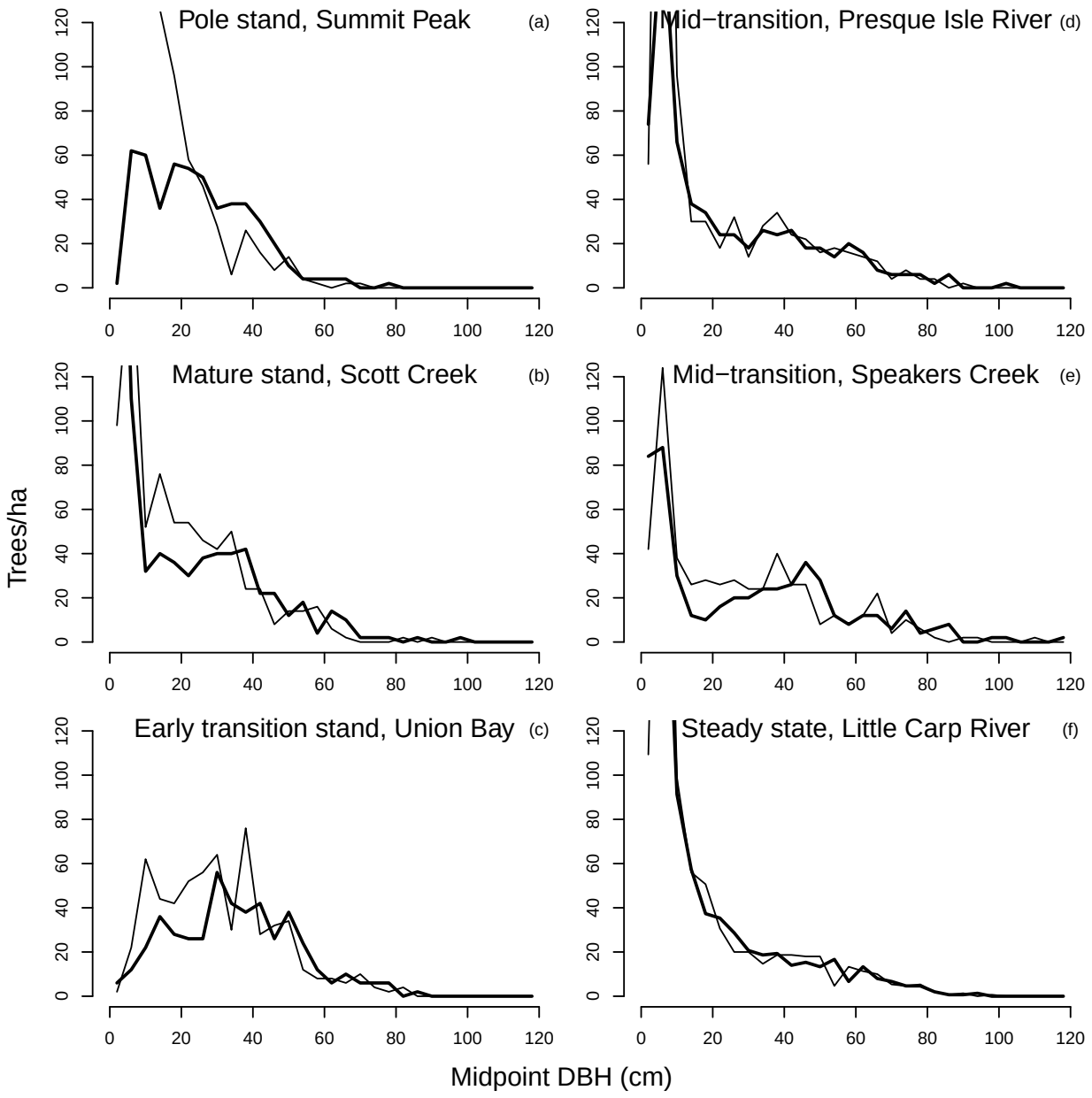

Figure 6: Changes in diameter distribution of the permanent plots over 30 years.

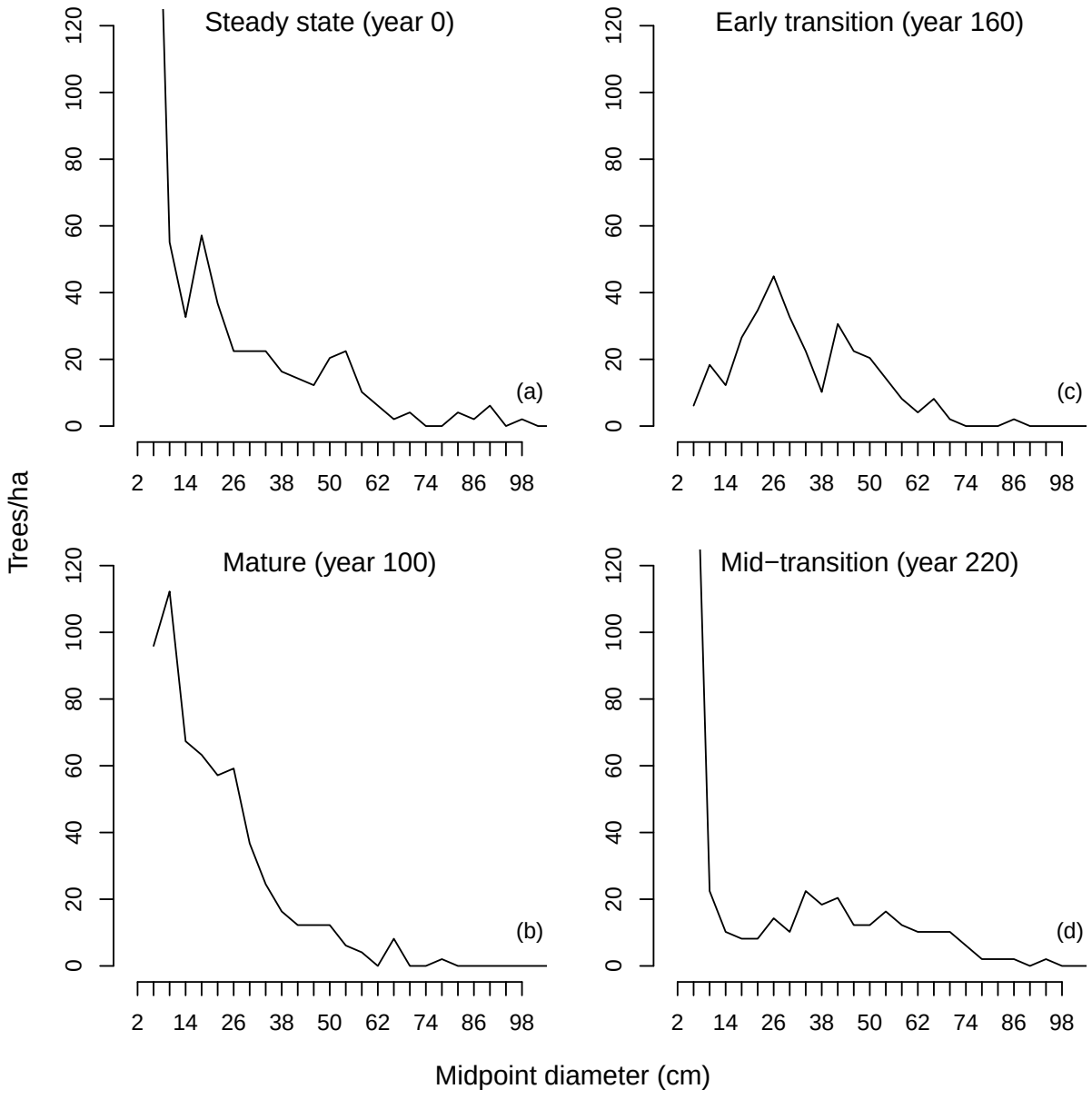

Figure 7: Simulation showing structural retrogression of a stand from steady state to transition stage after a series of four mild disturbances (20% crown area removal) at 30 year intervals in the first 90 years of simulation.

Table 1: Summary of key metrics by stand structural stage observed in the 70 upper Michigan field plots.

				Transition			
Structural feature	Pole	Mature	MSM	Early	Mid-	Late	Steady state
Total basal area	28	43	18	46	48	42	37
Average modal diameter	37	42	60	47	55	58	55
Basal of mature and large trees	10	35	14	41	44	36	31
Percent basal area of sapling	12.4	2.1	5.8	1.0	1.0	1.7	3.7
Percent basal area of pole	44.2	14.4	13.0	9.1	6.1	10.7	11.0
Percent basal area of mature	23.6	45.9	29.5	37.2	26.8	26.5	25.8
Percent basal area of large	11.7	35.4	48.8	51.1	65.0	59.6	58.0
Percent basal area of large (max)	20.8	46.6	56.7	56.5	76.8	63.7	72.3
Percent basal area of gap saplings	6.8	0.55	4.0	0.2	0.4	0.3	1.6
Large:Mature ratio	0.45	0.83	2.1	1.4	2.6	2.3	2.4
Large:Mature ratio (max)	1.29	1.08	3.2	1.7	4.5	2.7	4.7
Trees/ha larger than 50 cm dbh	7	43	26	63	85	66	58
Ages (post-clearcut simulations)	20-70	80-140	–	140-160	170-250	260-270+	280+
Percentage of study landscapes	9	11	3	19	20	13	26

APPENDIX B

CANOPY model equations

Table 1: Stand stage classification rules.

		Yes	No
1.	Mature + Large basal area $\geq$ 20 m^2/ha?	2	7
2.	Percent large basal area > 45 AND Large:Mature > 1.1?	3	*Mature*
3.	1.1 $\leq$ Large:Mature < 1.4?	*Early Transition*,	4
4.	Large:Mature > 1.75 AND %(Sapling + Pole basal area) < 10 ?	*Mid-transition*	5
5.	Large:Mature $\geq$ 1.7 AND 10 $\leq$ %(Sapling + Pole basal area) < 20 AND %(Gap sapling basal area) <0.6?	*Late Transition*	6
6.	Large:Mature > 1.4 AND 10 $\leq$ %(Sapling + Pole basal area) < 20 AND %(Gap sapling basal area) $\geq$ 0.6 ?	*Steady state*	*Early transition*
7.	Mature + Large basal area > 10 m^2/ha?	8	9
8.	% Pole basal area > 30 ?	*Pole*	*Mature-sapling mosaic*
9.	Pole + Mature + Large basal area $\geq$ 10 m^2/ha ?	*Pole*	*Sapling*

Sapling: 0-11 cm dbh; Pole: 11 - 26 cm dbh; Mature: 26 - 46 cm dbh; Large: 46+ cm dbh

Table 2: Growth equations.

Species	$\ln\Delta D$ (mm/yr)	RMSE	R^2
	Without even-aged variables		
Sugar maple	$3.77611 + 1.78561\cdot\ln(D) - 0.63133\cdot\sqrt{D} - 1.17279\cdot\ln(S) +$ $AOCa\cdot(-1.63801 + 0.38850\cdot\ln(S)) +$ $ATD\cdot(-2.39432 + 0.49766\cdot\ln(S))$	0.668	0.410
Hemlock	$3.55465 + 0.62131\cdot\ln(D) - 0.13625\cdot\sqrt{D} - 0.84355\cdot\ln(S)$	0.720	0.145
Yellow birch	$2.63158 + 2.12509\cdot\ln(D) - 0.84656\cdot\sqrt{D} - 0.104412\cdot\ln(S) +$ $AOCa\cdot(-1.89722 + 0.55303\cdot\ln(S) +$ $0.50702\cdot ATD$	0.746	0.328
Red maple	$-1.0088 + 3.1011\cdot\ln(D) - 1.0896\cdot\sqrt{D} - 0.5144\cdot\ln(S)$ $0.1856\cdot AOCa$	0.705	0.305
Ash	$-1.87360 + 1.58654\cdot\ln(D) - 0.50035\cdot\sqrt{D} + 0.05889\cdot\ln(S) +$ $AOCa\cdot(1.42342 + 0.33364\cdot\ln(D) - 0.44274\cdot\ln(S))$	0.536	0.333
Basswood	$-1.8954 + 2.4149\cdot\ln(D) - 0.7779\cdot\sqrt{D} - 0.3982\cdot\log(S) +$ $1.0692\cdot AOCa +$ $ATD\cdot(4.1372 - 0.6694\cdot\ln(S))$	0.663	0.274
Ironwood	$2.5292 - 0.5590\cdot\ln(S) - 0.3287\cdot AOCa$	0.821	0.065
	With even-aged variables		
Sugar maple	$3.70539 + 1.83246\cdot\ln(D) - 0.65283\cdot\sqrt{D} - 1.16630\cdot\ln(S) +$ $E\cdot(1.87563 - 0.41408\cdot\ln(S)) +$ $AOCa\cdot(-2.35133 + 0.55468\cdot\ln(S)) +$ $ATD\cdot(-2.42867 + 0.51098\cdot\ln(S))$	0.665	0.415
Yellow birch	$1.9621 + 2.2637\cdot\ln(D) - 0.8994\cdot\sqrt{D} - 0.9391\cdot\ln(S) +$ $E\cdot(3.1228 - 0.6708\cdot\ln(S)) +$ $AOCa\cdot(-1.9064 + 0.5638\cdot\ln(S)) +$ $0.5464\cdot ATD$	0.741	0.334
Red maple	$-1.0700 + 3.2407\cdot\ln(D) - 1.1444\cdot\sqrt{D} - 0.5290\cdot\ln(S) +$ $E\cdot(2.4947 - 0.5381\cdot\ln(S)) +$ $AOCa\cdot(-1.8137 + 0.4496\cdot\ln(S))$	0.698	0.314
Ash	$-1.80895 + 1.40155\cdot\ln(D) - 0.41957\cdot\sqrt{D} + 0.06857\cdot\ln(S)$ $0.14343\cdot E +$ $AOCa\cdot(1.38725 + 0.33290\cdot\ln(D) - 0.44339\cdot\ln(S))$	0.534	0.336
Basswood	$-0.46008 + 1.71851\cdot\ln(D) - 0.46575\cdot\sqrt{D} - 0.55461\cdot\ln(S) +$ $0.42968\cdot E +$ $0.70359\cdot AOCa +$ $ATD\cdot(2.77882 - 0.46826\cdot\ln(S))$	0.664	0.312
Ironwood	$2.3785 - 0.4263\cdot\ln(S) - 0.3653\cdot E$	0.816	0.075

D: diamter at breast height (cm).

S: northern hardwood stocking value

E: 1 for even-aged, 0 for uneven-aged.

ATM, AOCa, ATD: Categorical variables for habitat type.

Table 3: Mortality equations including habitat variables.

Species	X
	Without even-aged variables
Sugar maple	-1.0849·$\sqrt{D}$ + 0.1232·D -0.3608·ln(S) + AOCa·(-12.7668 -0.3108·$\sqrt{D}$ +2.7155·ln(S)) + ATD· (-2.6611 -0.4633·$\sqrt{D}$ + 0.8485·ln(S))
Hemlock	-1.4302·$\sqrt{D}$ + 0.1204·D -0.2207·ln(S) + ATM·(6.3953 -1.2475·ln(S))
Yellow birch	-2.3225·$\sqrt{D}$ + 0.2003·D + 0.2811·ln(S) + 0.5574·ATM
Red maple	-2.4951 -0.9903·$\sqrt{D}$ + 0.0942·D + 1.0553· ATM
Ash	-16.2836 -0.2974·$\sqrt{D}$ +2.6076·ln(S)
Basswood	-1.7014 -0.1641·D + AOCa·(-22.5704 + 4.5452·ln(S))
Ironwood	-10.2681 - 2.4489·$\sqrt{D}$ + 0.3303·D + 2.1448·ln(S) + 0.8931·AOCa
	Including even-aged variables
Sugar maple	-1.1435·$\sqrt{D}$ + 0.1294·D - 0.3372·ln(S) + 0.3399·E + AOCa·(-11.0919 -0.3034·$\sqrt{D}$ +2.3192·ln(S)) + ATD·(-3.1414 -0.4757·$\sqrt{D}$ + 0.9427·ln(S))
Yellow birch	-2.5734·$\sqrt{D}$ + 0.2262·D + 0.3391·ln(S) + 0.9938·E + 0.7639·ATM
Ash	-19.6718 -0.0278·D + 3.1378·ln(S) -1.1835·E + 1.0412·ATM
Basswood	-15.3054 -0.1768·D +2.6486·ln(S) + 0.9358·E
Ironwood	-8.7213 -2.4697·$\sqrt{D}$ + 0.3659·D + 1.9036·ln(S) + 0.1870·E

D: diamter at breast height (cm).
S: northern hardwood stocking value
E: 1 for even-aged, 0 for uneven-aged.
ATM, AOCa, ATD: Categorical variables for habitat type.
The value 'x' is used in logistic regressio nto calculate the annual probability of mortality: p(mort) = 1/(1+exp(-x)).

Table 4: Mortality equations without habitat variables.

Species	X
	Without even-aged variables
Sugar maple	-8.3470 -1.5961·$\sqrt{D}$ + 0.1377·D + 1.5969·ln(S)
Hemlock	-1.3568 -1.3066·$\sqrt{D}$ + 0.1105·D
Yellow birch	-2.3385·$\sqrt{D}$ + 0.1985·D + 0.3446·ln(S)
Red maple	-7.5231 -0.2003·$\sqrt{D}$ + 0.8547·ln(S)
Ash	-17.5287 -0.0329·D + 2.7363·ln(S)
Basswood	-14.4636 -1.3268·$\sqrt{D}$ + 3.1005·ln(S)
Ironwood	-9.7996 -2.2811·$\sqrt{D}$ + 0.3131·D + 2.0856·ln(S)
	Including even-aged variables
Yellow birch	-2.5096·$\sqrt{D}$ + 0.2155·D + 0.3991·ln(S) + 0.6380·E
Basswood	-12.3662 -1.3382·$\sqrt{D}$ + 2.5380·ln(S) +0.9340·E
Ironwood	-9.1134 -2.7375·$\sqrt{D}$ +0.3741·D +1.9989·ln(S) + 0.9585·E

D: diamter at breast height (cm).
S: northern hardwood stocking value
E: 1 for even-aged, 0 for uneven-aged.
The value 'x' is used in logistic regressio nto calculate the annual probability of mortality: p(mort) = 1/(1+exp(-x)).

Table 5: Sapling height growth equations (cm/yr; trees $\leq$ 17m tall)

Species	Equation[a]	R^2	MSE	n
AOCa and ATD habitat types				
Sugar maple	$\ln \Delta H = 1.77 + 0.566 \ln H - 0.0953H - 0.746CI_S + 0.270 \ln G_S + 0.0426 \ln G_S \times T_3$	0.48	0.70	509
Yellow birch	$\ln \Delta H = 1.87 + 0.644 \ln H - 0.0995H - 0.313CI_S + 0.240 \ln G_S$	0.35	0.88	111
Ash	$\ln \Delta H = 2.49 + 0.876 \ln H - 0.0916H - 0.545CI_S + 0.140 \ln G_S$	0.69	0.33	173
Basswood	$\ln \Delta H = 2.51 + 0.755 \ln H - 0.0930H - 0.323CI_S + 0.0926 \ln G_S$	0.66	0.30	133
Hop-hornbeam	$\ln \Delta H = 1.96 + 0.543 \ln H - 0.195H \times T_3 - 0.582CI_S \times T_3 + 0.195 \ln G_S$	0.39	0.62	144
ATM habitat type				
Sugar maple	$\ln \Delta H = 2.17 + 0.388 \ln H - 0.488CI_S + 0.200 \ln G_S - 0.0348G_A$	0.41	1.56	77
Yellow birch	$\ln \Delta H = 2.00 + 0.740 \ln H - 0.361CI_S + 0.164 \ln G_S - 0.0493G_A$	0.37	2.77	78
Eastern hemlock	$\ln \Delta H = 1.35 + 0.610 \ln H - 0.394CI_S + 0.142 \ln G_S$	0.62	0.54	103
Red maple	$\ln \Delta H = 2.13 + 0.0498H - 0.592CI_S + 0.244 \ln G_S - 0.0149G_A$	0.59	0.49	56

[a]Variables: ΔH, annual height increment (cm); H, initial height of sapling (m); CI_S, sum of competitor saplings' total crown area (m^2) divided by the plot size (m^2); G_S, gap size (m^2); G_A, gap age (years); T_3, binary variable for AOCa habitat (1 if tree on AOCa habitat, 0 otherwise).

Table 6: Equations to predict sapling (2-6 cm dbh) stem density per 100 m^2 and species composition.

	Equation[a]	R^2	MSE	n
Number of saplings				
Saplings/100 m^2	$\text{sqrt(sap)} = 14.5 - 2.04 \ln S_{800} - 0.533 \ln S_{100}$	0.47	1.89	151[b]
Species composition of saplings[c]				
Sugar maple	$x = -2.94T_1 + 0.539 \ln S_{900} - 2.27T_2 \times RB_{hem} \times \ln S_{900}$	0.49	6.84	154[d]
Eastern hemlock	$x = -4.54 + 1.25T_1 \times RB_{hem} \times \ln S_{900} + 1.88T_2 \times RB_{hem} \times \ln S_{900}$	0.59	3.18	154
Yellow birch	$x = 4.98 - 4.98T_2 - 1.96 \ln S_{900} + 0.341T_1 \times \ln S_{900} + 1.07T_2 \times \ln S_{900}$	0.30	2.41	154
Red maple	$x = -4.60 + 4.64T_1 - 0.809T_1 \times \ln S_{900}$	0.29	0.60	154
Ash	$x = 2.39 - 5.95T_1 - 6.99T_2 - 1.78 \ln S_{900} + 1.29T_1 \times \ln S_{900} + 1.74A_a$	0.24	0.67	154
Basswood	$x = -2.89 - 1.71T_2 - 0.254 \ln S_{900} + 0.177A_b$	0.07	0.18	154
Hop-hornbeam	$x = -7.11 + 0.835 \ln S_{900} + 0.820A_h$	0.06	4.89	154
Time required for seedlings to reach 2 cm DBH				
All species	$\text{years} = 10.31 + 0.11S_{900}$	0.45	4.42	36[e]

[a]Variables: sap, number of saplings; S_{100}, competition level (%) on 100 m^2 nested subplot; S_{800}, competition level (%) on 800 m^2 nested subplot; S_{900}, competition level (%) on 900 m^2 main plot; T_1, binary variable code for ATM habitat type (1 if habitat is ATM, 0 otherwise); T_2, binary code for ATD habitat type; RB_{hem}, relative eastern hemlock basal area (decimal fraction) on the 900 m^2 main plot; A_a, code for presence of ash species (1 if ash is present on 900 m^2 plot, 0 if absent); A_b, binary variable coding for presence of basswood species; A_h, code for presence of hop-hornbeam.
[b]The number of recruitment plots with 100 and 800 m^2 sample zones and that were usable in this analysis.
[c]x in these equations is the logit or exponent of the logistic regression equation for the probability that a given tree is of species i: $P_i = 1/(1 + \exp[-x])$.
[d]Some recruitment plots had no saplings and were not usable in this analysis.
[e]Number of cut sample trees.

APPENDIX C

CANOPY source code: C++ files

Source code listings use the "OCR-B" font, which is designed to facilitate optical character recognition.

C.1 COMMAND_LINE.CPP

This defines CANOPY's global variables which are used for option flags, and sets up the main simulation loop.

```
// This is the main CANOPY program for the command-line version, also
// currently the only version as we've not yet re-written the graphics
// portion to work on modern machines.

#include <unistd.h>
#include <string.h>
#include <sys/types.h>
#include <sys/stat.h>
#include <stdlib.h>
#include <assert.h>
#include <slist.h>

#include "util.h"
#include "Tree.h"
#include "TreeData.h"
#include "Stand.h"

// Global variables to tweak some parameters of the simulation
// Simulation mode
bool use_windstorms = false;

// Growth settings
bool use_branch_growth = false;
bool use_height_growth = false;
float lg_growth_nsd = 0;
float ht_growth_nsd = 0;
bool use_even_uneven = true;
float eu_switch_age = -1;
bool use_v2_equations = false;
bool use_old_hm_mort = false;
int mort_equation_type = 0;
```

```
bool use_mort = true;
bool gentle_birch_mort = false;

// Regeneration settings
bool regen_maintain_initial_saplings = false;
bool use_stem_exclusion = false;
bool only_sm_regen = false;
bool only_hm_regen = false;
bool forbid_hm_regen = false;
bool use_test_regen_props = false;
int regen_recovery_time = 0;
int n_saps_to_maintain = -1;

// Save settings
int save_interval   = 5; // how often to save, in years
double min_diameter = 2.0; // min dbh in cm of trees to save

// Windstorm settings:
int disturbance_threshold = 10;

void usage(){
  printf("Usage: canopy -i INPUT -o OUTPUT OPTIONS\n"
         "INPUT\tis the CANOPY treelist giving the conditions at t=0.\n"
         "OUTPUT\tgives the name of the result db. CANOPY creates this file.\n"
         "\tAn error will happen if you try to re-use an existing file.\n"
         "OPTIONS is any combination of:\n"
         "-l N\tsimulate for N years.\n"
         "-I N\tsave every N years (default is 5).\n"
         "-f N\tfirst save at Nth simulatin year.\n"
         "-c N\tsave treedata for trees larger than N cm (default is 2.0)\n"
         "-H\tUse height growth\n"
         "-b\tUse branch (crown) growth\n"
         "-s\tUse stochastic diameter growth\n"
         "-t\tUse stochastic height growth\n"
         "-w N\tSimulate windstorms\n"
         "-d N\tIgnore windstorms with %%eca removal less than N\n"
         "-B\tMaintain initial sapling population to test sustainability\n"
         "-r\tStart regen in 'recovery from stem exclsuion' mode\n"
         "-?\tDisplay this help\n"
         "The -i and -o flags are required.\n");
  exit(1);
}

int main(int argc, char* argv[]){
  int length=10; // simulation length in years
  slist<char*> harvests; // List of harvest lua files
  char* storm_spec = NULL; // Storm specification file (optional)

  // Stand expansion factors
  int multiply_x=1, multiply_y=1;

  // Temp variables for processing command line arguments.
  char *infile=NULL, *outfile=NULL, ch;
  extern char *optarg;

  while( (ch=getopt(argc,argv,"i:o:l:x:y:h:bHstUvV:2m:Yd:wW:BC:rR:XeNOL?")) != -1 ){
    switch(ch){
```

```
  // Operational options:
case 'i': infile = strdup(optarg);  break;
case 'o': outfile = strdup(optarg); break;
case 'l': length=strtol(optarg,NULL,10);           break;
case 'x': multiply_x = strtol(optarg,NULL,10);     break;
case 'y': multiply_y = strtol(optarg,NULL,10);     break;
  //Harvesting
case 'h': harvests.push_front(strdup(optarg));  break;
  // Options affecting growth
case 'b': use_branch_growth = true;              break;
case 'H': use_height_growth = true;              break;
case 's': lg_growth_nsd = 1.645;                 break;
case 't': ht_growth_nsd = 1.645;                 break;
case 'U': use_even_uneven = false;               break;
case 'v':
case 'V':
  if (optarg != NULL){
    eu_switch_age = atof(optarg);
  } else {
    eu_switch_age = 200;
  }
  break;

case '2': use_v2_equations = true;               break;
case 'm':
  mort_equation_type = strtol(optarg,NULL,10);
  break;
case 'Y':
  gentle_birch_mort = true;  break;
  //Options affecting disturbance
case 'd':
  disturbance_threshold = strtol(optarg,NULL,10);
  break;
case 'w':
case 'W':
  use_windstorms = true;
  use_branch_growth = true;

  if (optarg != NULL){
    storm_spec = strdup(optarg);
  }
  break;
  //Regeneration options
case 'B': regen_maintain_initial_saplings=true; break;
case 'C':
  n_saps_to_maintain = atoi(optarg);
  regen_maintain_initial_saplings=true;
  break;
case 'r':
case 'R':
  if (optarg!=NULL){
    regen_recovery_time = atof(optarg);
  } else {
    regen_recovery_time = 15;
  }
```

```
      break;
    case 'X': use_test_regen_props = true;          break;
    case 'e': use_stem_exclusion = true;            break;
    case 'N': only_hm_regen = true;                 break;
    case 'O': only_sm_regen = true;                 break;
    case 'L': forbid_hm_regen = true;               break;
      //Debugging options
    case '?':
      usage();                                  break;
    }
}

if( !use_height_growth && ht_growth_nsd!=0 ){
  printf("WARNING: Cannot use stochastic dH when dH is turned off\n");
}

if (infile!=NULL && outfile!=NULL) {
  Stand *s;

  s = new Stand(infile, outfile, multiply_x, multiply_y,
                argc, argv);

  for (slist<char*>::iterator it=harvests.begin();
       it != harvests.end(); it++){
    s->add_harvester( (*it) );
  }

  if (storm_spec != NULL){
    s->add_storm_model(storm_spec);
  }

  if (mort_equation_type == -1 ){
    use_mort = false;
  }

  s->calculate(); // Get initial stocking values
  if (!harvests.empty()) { s->harvest(); }
  s->calculate();
  s->save();
  s->inc_year();

  for (int i=1; i<=length; i++){
    printf("Year %i\n", i);

    s->grow();
    if (use_windstorms){ s->wind(); }
    if (use_mort)      { s->die();  }
    s->decay();
    if (!harvests.empty()) { s->harvest(); }
    s->regenerate();
    s->calculate();

    s->save();

    s->inc_year();

    fflush(NULL);
  }
  s->db_close();
} else {
```

```
    usage();
  }
}
```

C.2 Tree.cp

```
// -*- C++ -*-
/********** Tree.cp **************/

#include <math.h>
#include <stdlib.h>
#include <assert.h>
#include "Tree.h"
#include "CanopyGlobals.h"
#include "TreeDB.h"
#include "Random.h"
#include "util.h"

int Tree::last_treeno = 0;
extern int save_interval;
extern double min_diameter;

//define these as globals in the main program.
extern bool use_branch_growth;
extern bool use_height_growth;
extern float lg_growth_nsd;
extern float ht_growth_nsd;
extern bool use_even_uneven;
extern float eu_switch_age;
extern bool use_v2_equations;
extern int mort_equation_type;
extern bool gentle_birch_mort;

Tree::Tree(){ ; }

//! Set or auto-assign the treeno for this tree.
// tn >= 0 means 'use this treeno'
// tn <0 means 'auto-assign'
void Tree::set_treeno(int tn){
  if( tn<0 ){
    treeNo= ++last_treeno;
  } else {
    treeNo = tn;
    if( tn > last_treeno) { last_treeno = tn; }
  }
}

//! Initialize default values
void Tree::set(hdata_t* hd, int no, double x, double y, double d,
               species_t sp, double ncr, double ecr, double scr, double wcr ){
  treeCoordX=x; treeCoordZ=y;
  species=sp; dbh = d;

  born_in_sap_stand = false;
  set_treeno(no);

  status = st_live;
  gap_age=20;
```

```
  gap_area=1;
  // Estimate initial age from Singer and Lorimer equation
  age = lround(exp( (log(dbh)-0.123)/0.718 ));

  set_habitat(hd);

  double mcr = predict_mcr();

  crown_radius[0]=(ncr!=-1)?ncr:mcr;
  crown_radius[1]=(ecr!=-1)?ecr:mcr;
  crown_radius[2]=(scr!=-1)?scr:mcr;
  crown_radius[3]=(wcr!=-1)?wcr:mcr;

  for( int i=0; i<4; i++){
    exposed_crown_radius[i] = 0;
  }

  save_status.need_initial=true;
  save_status.need_final=true;
  save_status.need_forced=false;
  save_status.last = 0;

  if(last_treeno<=no){ last_treeno=no+1; }

  for(int i=0; i<4; i++){
    xy_neighbor[i].reset();
    rz_neighbor[i].reset();
    eca_neighbor[i].reset();
    facing_gap[i]=0;
  }

  for(int i=0; i<8; i++){
    gap_neighbor[i].reset();
  }

  stochastic_hg_modifier.value = 0;
  stochastic_hg_modifier.is_good = ht_growth_nsd==0;

  stochastic_lg_grow_modifier.value = 0;
  stochastic_lg_grow_modifier.is_good = lg_growth_nsd==0;

  stochastic_ht_dbh_modifier.value = 0;
  stochastic_ht_dbh_modifier.is_good = false;

  rel_ht = 0;
  rel_diam = 0;

  setStormCoefs(species);
  clear_caches();
}
//! Compute the stochastic modifier for the ht/diameter relation
double Tree::get_ht_dbh_modifier(){
  if( !stochastic_ht_dbh_modifier.is_good ){
    double rmse;
    switch( habitat_type ){
    case ht_ATM:
      switch(species){
      case sp_sugar_maple:  rmse = 0.2792848; break;
```

```
      case sp_hemlock:       rmse = 0.3193744; break;
      case sp_yellow_birch:  rmse = 0.3146427; break;
      case sp_red_maple:     rmse = 0.2213594; break;
      case sp_white_ash:     rmse = 0.2167948; break;
      default:               rmse = 0;         break;
      }
      break;

    case ht_AOCa:
      switch(species){
      case sp_sugar_maple:   rmse = 0.2236068; break;
      case sp_basswood:      rmse = 0.2024846; break;
      case sp_white_ash:     rmse = 0.2387467; break;
      default:               rmse = 0;         break;
      }
      break;

    case ht_ATD:
      switch(species){
      case sp_sugar_maple:   rmse = 0.2323790; break;
      case sp_hemlock:       rmse = 0.2932576; break;
      case sp_yellow_birch:  rmse = 0.2880972; break;
      case sp_red_maple:     rmse = 0.2549510; break;
      default:               rmse = 0;         break;
      }
      break;

    default:
      printf("Call to get_ht_dbh_modifier() on an unknown habitat type\n");
      exit(1);
    }

    stochastic_ht_dbh_modifier.value =
      rmse == 0 ? 0 : rnorm_trunc(0, rmse);
    stochastic_ht_dbh_modifier.is_good = true;
  }

  return stochastic_ht_dbh_modifier.value;
}

////// Constructors:

Tree::Tree(int no, double x, double y, double d, species_t sp,
           double ncr, double ecr, double scr, double wcr, hdata_t* hd){
  set(hd, no, x, y, d, sp, ncr, ecr, scr, wcr);
}

Tree::Tree(double x, double y, double d, species_t sp,
           double ncr, double ecr, double scr, double wcr, hdata_t* hd){
  set(hd, -1, x, y, d, sp, ncr, ecr, scr, wcr);
}

Tree::Tree(int no, double x, double y, double d, species_t sp, hdata_t* hd){
  set(hd, no, x, y, d, sp, -1,-1,-1,-1 );
}

Tree::Tree(double x, double y, double d, species_t sp, hdata_t* hd ){
  set(hd, -1, x, y, d, sp, -1,-1,-1,-1 );
}

// "Copy constructor" saying how to duplicate a tree
```

```
Tree& Tree::operator=(const Tree& t){
  if( this != &t ){
    set_treeno(-1);
    born_in_sap_stand = false;
    treeCoordX = t.treeCoordX;
    treeCoordZ = t.treeCoordZ;
    dbh = t.dbh;
    species = t.species;
    status = t.status;

    age = t.age;
    gap_age = t.gap_age;
    gap_area = t.gap_area;
    status   = t.status;

    habitat_type = t.habitat_type;

    save_status.need_initial=true;
    save_status.need_final=true;
    save_status.need_forced=false;
    save_status.last = 0;

    for(int i=0; i<4; i++){
      crown_radius[i] = t.crown_radius[i];
      exposed_crown_radius[i] = t.exposed_crown_radius[i];

      xy_neighbor[i].reset();
      rz_neighbor[i].reset();
      eca_neighbor[i].reset();
      facing_gap[i]=0;
    }

    for(int i=0; i<8; i++){
      gap_neighbor[i].reset();
    }

    stochastic_hg_modifier.value = 0;
    stochastic_hg_modifier.is_good = ht_growth_nsd==0;

    stochastic_lg_grow_modifier.value = 0;
    stochastic_lg_grow_modifier.is_good = lg_growth_nsd==0;

    stochastic_ht_dbh_modifier.value = 0;
    stochastic_ht_dbh_modifier.is_good = false;

    setStormCoefs(species);
    clear_caches();
  }

  return *this;
}
//! Destructor (deallocates a Tree object)
Tree::~Tree(){
  clear_competitor_saplings();

#ifndef __no_graphics__
  if (itsTreeModel != NULL)
    Q3Object_Dispose(itsTreeModel);
```

```
#endif
}

species_t Tree::GetSpecies(void) { return species; }
double Tree::GetBA(void){ return M_PI*pow( (dbh/2) /100 , 2); }
long Tree::GetTreeno(void){ return treeNo; }
double Tree::GetDBH(void){ return dbh; }
double Tree::GetRH(void){ return rel_ht; }
double Tree::GetRD(void){ return rel_diam; }

//! Get the mean crown radius of a tree
double Tree::GetMCR(){
  double rc=0;
  for (int i=0; i<4; i++){
    rc += crown_radius[i];
  }
  return rc;
}

//! Get the status code for a tree (live, dead, etc)
status_t Tree::GetStatus(void) { return (status); }

//! Get the height of the base of the crown
double  Tree::GetBaseHeight(){
  if( ! base_height.is_good ) {
    switch (species) {
    case sp_sugar_maple:
    default:
      base_height.value = -1.3946+0.61316*GetTotalHeight();
      break;
    case sp_hemlock:
    case sp_white_pine:
      base_height.value = pow(-0.9734+1.24609*log(GetTotalHeight()), 2);
      break;
    case sp_yellow_birch:
      base_height.value = -1.1878+0.65831*GetTotalHeight();
      break;
    case sp_red_maple:
      base_height.value = -1.8377+0.68909*GetTotalHeight();
      break;
    }
    // Ensure that base_height is between 0 and TH:
    base_height.value = fmax(0,fmin( base_height.value, GetTotalHeight()));
    base_height.is_good = true;
    assert(isfinite(base_height.value));
  }

  return base_height.value;
}

//! Height of the widest part of the crown
double  Tree::GetWidestHeight() {
  if( ! widest_height.is_good ){
    switch (species) {
    case sp_sugar_maple:
    default:
      widest_height.value = -1.6101+0.88096*GetTotalHeight();
      break;
```

```
    case sp_hemlock:
    case sp_white_pine:
      widest_height.value = exp(-1.33558+1.28089*log( GetTotalHeight() ));
      break;
    case sp_yellow_birch:
      widest_height.value = -1.1093+0.89031*GetTotalHeight();
      break;
    case sp_red_maple:
      widest_height.value = -2.1281+0.94456*GetTotalHeight();
      break;
    }
    assert(isfinite(widest_height.value));
    widest_height.value = fmax(0, widest_height.value);
    widest_height.is_good = true;
  }

  return widest_height.value;
}

//! Fill in an array with coefficients from the total height equations
void Tree::total_height_equation_coefs(double* bi){
  int sp;

  double coefs[][2] = {
    // CONST      ln(dbh)
    // AOCa:
    {  1.44818,  0.955864 }, // AOCa:SM   0
    {  1.72731,  0.953316 }, // AOCa:WA   1
    {  1.45300,  0.958920 }, // AOCa:BW   2
    //ATD:
    {  1.084297, 1.090011 }, // ATD:SM    3
    { -1.017610, 1.517670 }, // ATD:HM    4
    {  1.341310, 0.938250 }, // ATD:YB    5
    {  1.449740, 0.995870 }, // ATD:RM    6
    {  1.30240,  0.947100 }, // ATD:IW    7
    //ATM:
    {  1.140040, 1.050100 }, // ATM:SM    8
    {  0,        1.249200 }, // ATM:HM    9
    {  1.378800, 0.942980 }, // ATM:YB   10
    {  1.751530, 0.919210 }, // ATM:RM   11
    {  1.055300, 1.144900 }  // ATM:WA   12
  };

  if( habitat_type==ht_AOCa ){ //AOCa
    switch(species){
    case sp_hemlock:       sp= 4;  break;
    case sp_yellow_birch:  sp= 5;  break;
    case sp_red_maple:     sp= 6;  break;
    case sp_white_ash:     sp= 1;  break;
    case sp_basswood:      sp= 2;  break;
    case sp_ironwood:      sp= 7;  break;
    default:               sp= 0;  break;
    }
  } else if( habitat_type==ht_ATD){ //ATD
    switch(species){
    case sp_hemlock:       sp= 4;  break;
```

```
      case sp_yellow_birch: sp= 5;  break;
      case sp_red_maple:    sp= 6;  break;
      case sp_white_ash:    sp= 1;  break;
      case sp_basswood:     sp= 2;  break;
      case sp_ironwood:     sp= 7;  break;
      default:              sp= 3;  break;
      }
    } else if( habitat_type==ht_ATM){ //ATM
      switch(species){
      case sp_hemlock:      sp= 9;  break;
      case sp_yellow_birch: sp=10;  break;
      case sp_red_maple:    sp=11;  break;
      case sp_white_ash:    sp=12;  break;
      case sp_basswood:     sp= 2;  break;
      case sp_ironwood:     sp= 7;  break;
      default:              sp= 8;  break;
      }
    } else {
      printf("ERROR: unknown htype: %i %s:%i (this=%x)\n",
             habitat_type, __FILE__, __LINE__, (void*)this);
      exit(1);
    }
    bi[0] = coefs[sp][0];
    bi[1] = coefs[sp][1];
}

//! Allometrically determine total height
double  Tree::GetTotalHeight() {
    /// Allometrically determins total height
    /** This uses species/habitat type specific equations developed by
        Jake Hanson in 2005.
    */
    if( ! total_height.is_good ){
      double bi[2];
      double xi[2];

      xi[0]=1;
      xi[1]=log(dbh);

      //Preconditions:
      assert(isfinite(xi[1]));

      total_height_equation_coefs(bi);
      double rc = bi[0]*xi[0]+bi[1]*xi[1] + get_ht_dbh_modifier();
      rc = pow(rc, 2);
      assert(isfinite(rc));

      total_height.value = rc;
      total_height.is_good=true;
    }

    return total_height.value;
}

//! Update diameter based on a desired total height
void Tree::SetTotalHeight(double th) {
    double bi[2];

    total_height_equation_coefs(bi);
```

```
  double log_dbh = (sqrt(th)- bi[0] - get_ht_dbh_modifier() )/bi[1];
  double new_dbh = exp(log_dbh);

  if (new_dbh >= dbh){
    dbh = new_dbh; clear_caches();
  }
}

double Tree::GetXCoord(void) { return (treeCoordX); }
double Tree::GetYCoord(void) { return (treeCoordZ); }

void Tree::SetRH(double avg_ht900) { rel_ht = GetTotalHeight()/avg_ht900; }
void Tree::SetXCoord(double x){ treeCoordX=x; }
void Tree::SetYCoord(double y){ treeCoordZ=y; }

void Tree::SetTreeData(long TreeNo, double xCoord, double yCoord,
                       species_t Species, short CC,
                       double DBH, double TotalHeight, double BaseHeight,
                       double WidestHeight, double NorthTotal,
                       double NorthExp,
                       double EastTotal, double EastExp, double SouthTotal,
                       double SouthExp, double WestTotal, double WestExp,
                       double Distance, double Azimuth, status_t Status,
                       double BasalArea, double PreviousBA,
                       long YearsOnTreeList) {
  treeNo = TreeNo;
  treeCoordX = xCoord;
  treeCoordZ = yCoord;
  dbh = DBH;   clear_caches();
  species = Species;
  status = Status;
  if( treeNo > last_treeno ) { last_treeno = treeNo; }
}

void Tree::SetStatus(status_t st){ status = st; }

//! Crown growth
void Tree::grow_branches(double delta_dbh, double rh){
  // Preconditions:
  assert( isfinite(dbh) && 0<= dbh );
  assert( isfinite(delta_dbh) && 0<= delta_dbh );
  assert( isfinite(rh) && 0<= rh );

  for(int i = 0; i<4; i++){
    double bgrow;

    if( touching[i] == true ){
      bgrow=0;
    } else if ( gap_area == 0 || shaded[i]==true || dbh<11 ){
      //Grow at background rate.
      int ix;
      switch(species){
      default:
        if( habitat_type==ht_A0Ca) { ix=0; } else { ix=1; }
        break;
      case sp_hemlock:      ix=2; break;
      case sp_yellow_birch: ix=3; break;
```

```
      case sp_red_maple:     ix=4; break;
      case sp_white_ash:     ix=5; break;
      case sp_basswood:      ix=6; break;
      case sp_ironwood:      ix=7; break;
      }
      double coefs[][3] = {
        { /* SM A0Ca       */  8.13689, 0.03818, 18.56050 },
        { /* SM ATD/ATM    */  7.91233, 0.02746, 24.93295 },
        { /* Hemlock       */  4.51000, 0.04241, 10.10187 },
        { /* Yellow Birch */  7.76193, 0.04056, 25.38926 },
        { /* Red Maple     */  5.24100, 0.09100, 18.62089 },
        { /* White Ash     */  6.01723, 0.05012, 20.01200 },
        { /* Basswood      */  6.18326, 0.03983, 19.18628 },
        { /* Ironwood      */  3.24642, 0.30031,  5.00321 }};

      double A=coefs[ix][0];
      double B=coefs[ix][1];
      double C=coefs[ix][2];

      double dcr_ddbh= -A*-B*exp(-exp(-B*(dbh-C)))*exp(-B*(dbh-C));

      bgrow=dcr_ddbh*delta_dbh;
    } else {
      //These equations come from Page 166 of Chris Webster's PhD Thesis.
      int yellow_birch = (species==sp_yellow_birch)?1:0;
      int dominant     = (rh>=0.66)?1:0;

      if(species==sp_hemlock || species==sp_white_pine ){
        bgrow = 14.836-2.66*log(dbh);
      } else if( is_facing_gap(i) == true ){
        bgrow = pow(2.99+0.607*yellow_birch-0.0091*dbh-0.434*dominant,2);
      } else {
        bgrow = pow(0.65+0.755*yellow_birch+0.0652*dbh
                    -0.000799*pow(dbh,2),2);
      }

      /* Apply a per-habitat-type growth-rate difference to sugar
         maple, which is the only species for which we can detect a
         significant effect of habitat type in crown-growth rate.
      */
      if      (species==sp_sugar_maple && habitat_type==ht_ATD ) {bgrow*=1.06;}
      else if (species==sp_sugar_maple && habitat_type==ht_A0Ca ){bgrow*=1.17;}
      bgrow /= 100; //convert from cm to meters.
    }
    bgrow = fmax(0, bgrow); // Require non-negative

    //Postconditions:
    assert( isfinite(bgrow) && 0<= bgrow );

    crown_radius[i]+= bgrow;
  }
}

//! Size increase for trees, uses dD where appropriate, and dH otherwise
double Tree::grow(double stock, double rd, stage_t stage){
  double th=GetTotalHeight();
  double growth;

  rel_diam = rd;
```

```
  if (use_height_growth &&
      ( (species != sp_ironwood && species != sp_hemlock && GetTotalHeight() <= 17) ||
        (species == sp_hemlock && GetTotalHeight() <= 8.25) ) ){
    growth = grow_ht();
  } else {
    if (use_v2_equations){
      growth = grow_dbh(stock, stage);
    } else {
      growth = grow_dbh_new(stock, stage);
    }
  }

  if (use_branch_growth) {
    grow_branches(growth,this->rel_ht);
  } else {
    //Predict the MCR and set all radii equal to it.
    double mcr = predict_mcr();
    for( int i=0; i<4; i++){ crown_radius[i]=mcr; }
  }

  age++;

  return growth;
}

//! Compute the TCA of competitor saplings
double Tree::get_comp_sap_tca(){
  if ( !comp_sap_tca.is_good ){
    double rc=0;

    for (int i=0; i< competitor_saplings.size(); i++) {
      if ( competitor_saplings[i]->GetTotalHeight() > this->GetTotalHeight() &&
           competitor_distances[i] <= 5.05 ) {
        rc += competitor_saplings[i]->get_tca();
      }
    }
    comp_sap_tca.value = rc;
    comp_sap_tca.is_good = true;
  }
  return comp_sap_tca.value;
}

//! Compute a competition index based on competitor saplings
double Tree::get_comp_sap_ci(){
  if( ! comp_sap_ci.is_good ){
    double mcr = GetMCR();

    double rc = 0;
    for (int i=0; i<competitor_saplings.size(); i++) {
      if (competitor_distances[i] <= 2.5 * mcr) {
        rc += competitor_saplings[i]->get_tca();
      }
    }

    rc /= ( M_PI*pow(2.5*mcr,2) );

    comp_sap_ci.value = rc;
    comp_sap_ci.is_good = true;
  }
  return comp_sap_ci.value;
}
```

```
//! Height growth
double Tree::grow_ht( ){
  //// Regression coefficients:
  double coefs[][7]={
    //Const,    H,       lnH,     lnGA,     SapTCA,   CI,      gAge
    // ATD/AOCa
    { 1.8669, -0.0995, 0.6439, 0.2404,   -0.3130, 0,        0      }, //YB  0
    { 2.4871, -0.0916, 0.8761, 0.1398,   -0.5450, 0,        0      }, //WA  1
    //AOCa
    { 1.7715, -0.0953, 0.5656, 0.3128, -0.7457,  0,        0      }, //SM  2
    { 2.5063, -0.0930, 0.7551, 0.0926, -0.3226,  0,        0      }, //BW  3
    { 2.5047, -0.1953, 0.6969, 0.1952, -0.5816,  0,        0      }, //IW  4
    // ATD
    { 1.7715, -0.0953, 0.5656, 0.2702, -0.7457,  0,        0      }, //SM  5
    { 2.5063, -0.0930, 0.7551, 0,       -0.3226,  0,        0      }, //BW  6
    { 1.9618,  0,       0.6969, 0,        0,        0,        0      }, //IW  7
    // ATM
    { 2.1665,  0,       0.3883, 0.1998,  0,       -0.4875, -0.0348}, //SM  8
    { 2.0031,  0,       0.7398, 0.1639,  0,       -0.3609, -0.0493}, //YB  9
    { 1.3463,  0,       0.6103, 0.1415,  0,       -0.3935,  0      }, //HM 10
    { 2.1299,  0.0498, 0,       0.2444,  0,       -0.5924, -0.0149}, //RM 11
  };

  double rmse[] = {
    /* AOCa/ATD: YB */ 0.9364828,
    /* AOCa/ATD: WA */ 0.5775812,
    /* AOCa:     SM */ 0.8377947,
    /* AOCa:     BW */ 0.5515433,
    /* AOCa:     IW */ 0.7866384,
    /* ATD:      SM */ 0.8377947,
    /* ATD:      BW */ 0.5515433,
    /* ATD:      IW */ 0.7866384,
    /* ATM:      SM */ 1.2498000,
    /* ATM:      YB */ 1.6649920,
    /* ATM:      Hm */ 0.7342343,
    /* ATM:      RM */ 0.7014984,
  };

  double bias_correction[] = {
    /* AOCa/ATD: YB */ 1.20,
    /* AOCa/ATD: WA */ 1.11,
    /* AOCa:     SM */ 1.23,
    /* AOCa:     BW */ 1.14,
    /* AOCa:     IW */ 1.21,
    /* ATD:      SM */ 1.23,
    /* ATD:      BW */ 1.14,
    /* ATD:      IW */ 1.21,
    /* ATM:      SM */ 1.24,
    /* ATM:      YB */ 1.22,
    /* ATM:      Hm */ 1.13,
    /* ATM:      RM */ 1.11,
   };

  int species_code;
  // Determine a species code which is used to index into
```

```
//  the table of coefficients
switch (habitat_type){
case ht_AOCa:
  switch(species){
  case sp_yellow_birch: species_code=0;  break;
  case sp_white_ash:    species_code=1;  break;
  case sp_basswood:     species_code=3;  break;
  case sp_ironwood:     species_code=4;  break;
  case sp_hemlock:      species_code=10; break;
  case sp_red_maple:    species_code=11; break;
  default: species_code=2; break;
  }
  break;

case ht_ATD:
  switch(species){
  case sp_yellow_birch: species_code=0;  break;
  case sp_white_ash:    species_code=1;  break;
  case sp_basswood:     species_code=6;  break;
  case sp_ironwood:     species_code=7;  break;
  case sp_hemlock:      species_code=10; break;
  case sp_red_maple:    species_code=11; break;
  default: species_code=5; break;
  }
  break;

case ht_ATM:
  switch(species){
  case sp_yellow_birch: species_code=9;  break;
  case sp_white_ash:    species_code=1;  break;
  case sp_basswood:     species_code=6;  break;
  case sp_ironwood:     species_code=7;  break;
  case sp_hemlock:      species_code=10; break;
  case sp_red_maple:    species_code=11; break;
  default: species_code=8; break;
  }
  break;

default:
  printf("Call to grow_ht() on a habitat type for which "
         "no growth equations are available\n");
  exit(1);
}

double th = GetTotalHeight();
double xi[7];
xi[0]=1;
xi[1]=th;
xi[2]=log(th);
xi[3]=log(fmin(1000,fmax(1,gap_area)));
xi[4]=get_comp_sap_tca()/80;
xi[5]=get_comp_sap_ci();
xi[6]=gap_age;

//Preconditions:
assert( isfinite(xi[1]) && 0< xi[1] ); // Height
```

```
  assert( isfinite(xi[2])              ); // ln(Height)
  assert( isfinite(xi[3])              ); // ln(GA)
  assert( isfinite(xi[4]) && 0<=xi[4] ); // SapTCA
  assert( isfinite(xi[5]) && 0<=xi[5] ); // CI
  assert( isfinite(xi[6]) && 0<=xi[6] ); // gAge

  double result=0;
  for(int i=0; i<7; i++) {
    result += xi[i]*coefs[species_code][i];
  }

  double bc = 1; // bias correction factor for this tree

  if ( ht_growth_nsd!=0 ){
    if ( !stochastic_hg_modifier.is_good ){
      stochastic_hg_modifier.value = rnorm_trunc(0, rmse[species_code], ht_growth_nsd);
      stochastic_hg_modifier.is_good = true;
    }

    result += stochastic_hg_modifier.value;
  } else {
    //bc = bias_correction[species_code];
  }

  double hgrow_m = fmax(0, bc*exp(result)/100);

  //Adjust the white-ash growth:
  if ( habitat_type==ht_ATM && species==sp_white_ash ){
    hgrow_m *= 0.925;
  }

  //Postconditions:
  assert( isfinite(hgrow_m) && 0 <= hgrow_m );

  // For SM on ATM, limit dH to less than the max dH observed:
  if (species_code==8){
    double pct_lim=0;
    if      (th <  4.013) { pct_lim = -0.0424 * th + 0.3500; }
    else if (th <  8.200) { pct_lim = -0.0215 * th + 0.2663; }
    else if (th < 15.000) { pct_lim = -0.0059 * th + 0.1382; }
    else if (th < 18.300) { pct_lim = -0.0055 * th + 0.1318; }
    else                  { pct_lim = -0.0043 * th + 0.1101; }

    hgrow_m = fmin( pct_lim * th, hgrow_m);
  }

  // For YB on ATM, limit dH to less than the max dH observed:
  if (species_code==9){
    double pct_lim=0;
    if (th < 9.51) { pct_lim = -0.0284 * th + 0.3500; }
    else           { pct_lim = -0.0048 * th + 0.1255; }

    hgrow_m = fmin( pct_lim * th, hgrow_m);
  }

  double prev_dbh = dbh;
  SetTotalHeight( th + hgrow_m );

  double dgrow = fmax(0, dbh-prev_dbh); //use fmax to account for round-off

  return dgrow;
}
```

```
//! Diameter growth
// Predicts diameter growth in terms of stocking and current diameter,
// with separate equations for each species and habitat type.  Also
// has alternate equations for even v. uneven aged conditions.
double Tree::grow_dbh(double stock, stage_t stage){
  const int n_species = 7;
  const int n_params = 4;

  double coefs[3][n_species][n_params]={
    //                      _CONST_    _ln(DBH)_ _DBH_      _STOCK__
    /************/
    /*** AOCa ***/
    /************/
    {
      //// "Standard" MS-thesis equation set:
      /* Sugarmaple   */ {-0.24524, 0.90394, -0.02842, -0.00788},
      /* Hemlock      */ { 0,       0.60313, -0.01260, -0.00617},
      /* Yellow birch */ { 0,       0.80757, -0.03100, -0.00841},
      /* Red maple    */ {-1.34701, 1.25414, -0.03392, -0.00560},
      /* White ash    */ {-1.37627, 1.17654, -0.03423, -0.00213},
      /* Basswood     */ { 0,       0.61999, -0.01514, -0.00451},
      /* Ironwood     */ { 0,      -0.18438,  0,        0       },
    },
    /************/
    /*** ATD ***/
    /************/
    {
      ///// "Standard" MS-thesis equation set:
      /* Sugarmaple   */ {-0.24524, 0.88668, -0.02842, -0.00788},
      /* Hemlock      */ { 0,       0.60313, -0.01260, -0.00499},
      /* Yellow birch */ { 0,       0.80514, -0.03100, -0.00841},
      /* Red maple    */ {-1.34701, 1.25414, -0.03392, -0.00560},
      /* White ash    */ {-1.37627, 1.17654, -0.03423, -0.00213},
      /* Basswood     */ { 0,       0.61999, -0.01514, -0.00451},
      /* Ironwood     */ { 0,      -0.18438,  0,        0       },
    },
    /************/
    /*** ATM ***/
    /************/
    {
      //// "Standard" MS-thesis equation set:
      /* Sugarmaple   */ {-0.24524, 0.85316, -0.02842, -0.00788},
      /* Hemlock      */ { 0,       0.60313, -0.01260, -0.00617},
      /* Yellow birch */ { 0,       0.66074, -0.03100, -0.00841},
      /* Red maple    */ {-1.34701, 1.25414, -0.03392, -0.00560},
      /* White ash    */ {-1.37627, 1.17654, -0.03423, -0.00213},
      /* Basswood     */ { 0,       0.61999, -0.01514, -0.00451},
      /* Ironwood     */ { 0,      -0.18438,  0,        0       },
    }
  };

  double rmse[n_species] = {
    /* Sugarmaple   */ 0.6282993,
    /* Hemlock      */ 0.7400068,
    /* Yellow birch */ 0.8055433,
    /* Red Maple    */ 0.7555859,
    /* White ash    */ 0.5085764,
    /* Basswood     */ 0.7027019,
```

```
  /* Ironwood     */ 0.9074194,
};

double bias_correction[n_species] = {
  /* Sugarmaple   */ 1.120,
  /* Hemlock      */ 1.199,
  /* Yellow Birch */ 1.225,
  /* Red Maple    */ 1.170,
  /* White Ash    */ 1.125,
  /* Basswood     */ 1.169,
  /* Ironwood     */ 1.227,
};

int species_code;
//Determine the species code, used to index the tables of coefficients
switch (species){
default:
case sp_sugar_maple:
  species_code = 0; break;
case sp_hemlock:
  species_code = 1; break;
case sp_yellow_birch:
  species_code = 2; break;
case sp_red_maple:
  species_code = 3; break;
case sp_american_elm:
case sp_northern_red_oak:
case sp_green_ash:
case sp_paper_birch:
case sp_black_cherry:
case sp_white_ash:
  species_code = 4; break;
case sp_basswood:
  species_code = 5; break;
case sp_mountain_maple:
case sp_balsam_fir:
case sp_ironwood:
  species_code = 6; break;
}

double xi[n_params];
xi[0]=1;         //constant term
xi[1]=log(dbh);
xi[2]=dbh;
xi[3]=stock;

// Preconditions:
assert( isfinite(xi[1]) );                 // log(dbh)
assert( isfinite(xi[2]) && 0 <= xi[2] ); // dbh
assert( isfinite(xi[3]) && 0 <= xi[3] ); // stock

double result=0;
for(int i=0; i<n_params; i++) {
  result += xi[i] * coefs[habitat_type][species_code][i];
}
```

```
  double bc = 1; // bias correction factor for this tree

  //Apply the stochastic effect, when required
  if ( lg_growth_nsd!=0 ) {
    if ( !stochastic_lg_grow_modifier.is_good ){
      stochastic_lg_grow_modifier.value = rnorm_trunc(0, rmse[species_code],
          lg_growth_nsd);
      stochastic_lg_grow_modifier.is_good = true ;
    }

    result += stochastic_lg_grow_modifier.value ;
  } else {
    bc = bias_correction[species_code];
  }

  // back-transform the result of the regression, applying bias correction;
  double growth = bc * exp(result)/10;

  //Postconditions:
  assert(isfinite(growth) && 0 <= growth );

  dbh += growth; clear_caches();

  return growth;
}

//! Diameter growth
// Predicts diameter growth in terms of stocking and current diameter,
// with separate equations for each species and habitat type.  Also
// has alternate equations for even v. uneven aged conditions.
double Tree::grow_dbh_new(double stock, stage_t stage){
  const int n_params = 4;
  const int n_species = 7;

  double coefs[3][3][n_species][n_params]={
    { //AOCa:
      { // Split -- Uneven-aged:
        { 1.354065,    1.832456,  -0.65283,   -0.6116123}, //SM
        { 3.554651,    0.6213118, -0.1362489, -0.8433515}, //Hm
        { 0.05568177,  2.263826,  -0.8993678, -0.3768509}, //Yb
        {-2.8837185,   3.240746,  -1.144422,  -0.07943475},//Rm
        { 0.26988830,  1.23624,   -0.2888182, -0.3020435}, //Ash
        { 0.24351410,  1.718514,  -0.4657525, -0.5546132}, //BW
        { 2.517114,    0,          0,         -0.5539939}, //Iw
      }, { // Split -- Even-aged:
        { 3.229695,    1.832456,  -0.65283,   -1.0256956}, //SM
        { 3.554651,    0.6213118, -0.1362489, -0.8433515}, //Hm
        { 3.17845763,  2.263826,  -0.8993678, -1.0476214}, //Yb
        {-0.3890647,   3.240746,  -1.144422,  -0.61756261},//Rm
        { 0.26988830,  1.23624,   -0.2888182, -0.3020435}, //Ash
        { 0.67319572,  1.718514,  -0.4657525, -0.5546132}, //BW
        { 2.159450,    0,          0,         -0.5539939}, //Iw
      },{ // Unified
        { 2.138098,    1.785612,  -0.6313298, -0.7842926}, //SM
        { 3.554651,    0.6213118, -0.1362489, -0.8433515}, //Hm
        { 0.7343615,   2.125094,  -0.8465606, -0.4910896}, //Yb
        {-0.823174,    3.1011,    -1.08964,   -0.5144198}, //Rm
        { 0.26988830,  1.23624,   -0.2888182, -0.3020435}, //Ash
```

```
    {-0.8262529,   2.41491,   -0.7779329, -0.3982128}, //BW
    { 2.340304,    0,          0,         -0.5857937}, //Iw
  }
}, { // ATD:
  { // Split -- Uneven-aged:
    { 1.276725,    1.832456,  -0.65283,   -0.6553158}, //SM
    { 3.554651,    0.6213118, -0.1362489, -0.8433515}, //Hm
    { 2.50847061,  2.263826,  -0.8993678, -0.9391122}, //Yb
    {-1.0700185,   3.240746,  -1.144422,  -0.52898924}, //Rm
    {-0.09283665,  1.23624,   -0.2888182, -0.3020435}, //Ash
    { 2.31874145,  1.718514,  -0.4657525, -1.0228735}, //BW
    { 2.517114,    0,          0,         -0.5539939}, //Iw
  }, { // Split -- Even-aged:
    { 3.152355,    1.832456,  -0.65283,   -1.0693991}, //SM
    { 3.554651,    0.6213118, -0.1362489, -0.8433515}, //Hm
    { 5.63124647,  2.263826,  -0.8993678, -1.6098827}, //Yb
    { 1.4246352,   3.240746,  -1.144422,  -1.06711710}, //Rm
    {-0.09283665,  1.23624,   -0.2888182, -0.3020435}, //Ash
    { 2.74842307,  1.718514,  -0.4657525, -1.0228735}, //BW
    { 2.159450,    0,          0,         -0.5539939}, //Iw
  }, { // Unified
    { 1.381788,    1.785612,  -0.6313298, -0.6751264}, //SM
    { 3.554651,    0.6213118, -0.1362489, -0.8433515}, //Hm
    { 3.1386007,   2.125094,  -0.8465606, -1.0441164}, //Yb
    {-1.008771,    3.1011,    -1.08964,   -0.5144198}, //Rm
    {-0.09283665,  1.23624,   -0.2888182, -0.3020435}, //Ash
    { 2.2417766,   2.41491,   -0.7779329, -1.0676043}, //BW
    { 2.664816,    0,          0,         -0.5857937}, //Iw
  }
}, { // ATM:
  { // Split -- Uneven-aged:
    { 3.705394,    1.832456,  -0.65283,    -1.1662953}, //SM
    { 3.554651,    0.6213118, -0.1362489,  -0.8433515}, //Hm
    { 1.96208177,  2.263826,  -0.8993678,  -0.9391122}, //Yb
    {-1.0700185,   3.240746,  -1.144422,   -0.52898924},//Rm
    {-0.09283665,  1.23624,   -0.2888182,  -0.3020435}, //Ash
    {-0.46008074,  1.718514,  -0.4657525,  -0.5546132}, //BW
    { 2.517114,    0,          0,         -0.5539939}, //Iw
  }, { // Split -- Even-aged:
    { 5.581024,    1.832456,  -0.65283,    -1.5803786}, //SM
    { 3.554651,    0.6213118, -0.1362489,  -0.8433515}, //Hm
    { 5.08485763,  2.263826,  -0.8993678,  -1.6098827}, //Yb
    { 1.4246352,   3.240746,  -1.144422,   -1.06711710},//Rm
    {-0.09283665,  1.23624,   -0.2888182,  -0.3020435}, //Ash
    {-0.03039912,  1.718514,  -0.4657525,  -0.5546132}, //BW
    { 2.159450,    0,          0,         -0.5539939}, //Iw
  }, { // Unified
    { 3.776112,    1.785612,  -0.6313298,  -1.1727903}, //SM
    { 3.554651,    0.6213118, -0.1362489,  -0.8433515}, //Hm
    { 2.6315772,   2.125094,  -0.8465606,  -1.0441164}, //Yb
    {-1.008771,    3.1011,    -1.08964,    -0.5144198}, //Rm
    {-0.09283665,  1.23624,   -0.2888182,  -0.3020435}, //Ash
    {-1.8954150,   2.41491,   -0.7779329,  -0.3982128}, //BW
```

```
      { 2.664816,     0,           0,           -0.5857937}, //Iw
    }
  }
};

double bias_correction[2][n_species] = {
  { // Split:
    /*     Sm          Hm          Yb          Rm         Ash          Bw          Iw */
    1.154889, 1.221601, 1.209713, 1.172483, 1.126267, 1.151046, 1.327567,
  }, { // Unified:
    1.153982, 1.221601, 1.215973, 1.175144, 1.126267, 1.165245, 1.332544,
  }
};

double rmse[2][n_species] = {
  { //Split:
    /*      Sm          Hm          Yb          Rm          Ash          Bw          Iw */
    0.6650471, 0.7204375, 0.7410988, 0.6976488, 0.5477915, 0.6443756, 0.8188313,
  }, { //Unified
    0.6678621, 0.7204375, 0.7456844, 0.704972,  0.5477915, 0.6626826, 0.8229926,
  }
};

//Determine the species code, used to index the tables of coefficients
int species_ix;
switch (species){
default:
case sp_sugar_maple:
  species_ix = 0; break;
case sp_hemlock:
  species_ix = 1; break;
case sp_yellow_birch:
  species_ix = 2; break;
case sp_red_maple:
  species_ix = 3; break;
case sp_american_elm:
case sp_northern_red_oak:
case sp_green_ash:
case sp_paper_birch:
case sp_black_cherry:
case sp_white_ash:
  species_ix = 4 ; break;
case sp_basswood:
  species_ix = 5; break;
case sp_mountain_maple:
case sp_balsam_fir:
case sp_ironwood:
  species_ix = 6; break;
}

int ht_ix;
switch(habitat_type){
  case ht_AOCa: ht_ix = 0; break;
  case ht_ATD:  ht_ix = 1; break;
  case ht_ATM:  ht_ix = 2; break;
  default:
    printf("ERROR: Unknown habitat type %i in grow_dbh_new() at %s:%i\n",
```

```
            habitat_type, __FILE__, __LINE__ );
    exit(1);
    break;
}

int structure_ix;
int rmse_bc_ix;
if (use_even_uneven){
  rmse_bc_ix = 0;

  if (eu_switch_age < 0){
    if (born_in_sap_stand &&
        (stage == stage_sapling ||
         stage == stage_pole ||
         stage == stage_mature) ) {
      structure_ix = 1;
    } else {
      structure_ix = 0;
    }
  } else {
    if (born_in_sap_stand && age < eu_switch_age ){
      structure_ix = 1;
    } else {
      structure_ix = 0;
    }
  }
} else {
  rmse_bc_ix = 1;
  structure_ix = 2;
}

double xi[n_params];
xi[0] = 1;
xi[1] = log(dbh);
xi[2] = sqrt(dbh);
xi[3] = log(fmax(1,stock));

double result=0;
for (int i=0; i<n_params; i++) {
  result += xi[i] * coefs[ht_ix][structure_ix][species_ix][i];
}

double bc = 1; // bias correction factor for this tree

//Apply the stochastic effect, when required
if ( lg_growth_nsd!=0 ){
  if ( !stochastic_lg_grow_modifier.is_good ){
    if ( species_ix == 4) { // Species that use the WA equations:
      stochastic_lg_grow_modifier.value = rnorm_trunc(0, rmse[rmse_bc_ix][species_ix],
          1.0);
    } else {
      stochastic_lg_grow_modifier.value = rnorm_trunc(0, rmse[rmse_bc_ix][species_ix],
          lg_growth_nsd);
    }
    stochastic_lg_grow_modifier.is_good = true ;
  }

  result += stochastic_lg_grow_modifier.value ;
```

```
  } else {
    bc = bias_correction[rmse_bc_ix][species_ix];
  }

  // back-transform the result of the regression, applying bias correction;
  double growth = bc * exp(result) / 10;

  //Postconditions:
  assert(isfinite(growth) && 0 <= growth );

  dbh += growth; clear_caches();

  return growth;
}

double Tree::pdie(double stock, double rd, stage_t stage, double pct_hemlock){
  double rc;
  if (use_v2_equations){
    rc = Tree::pdie_old( stock, stage );
  } else {
    // On maple-dominated ATM, use the ht-insensitive equations for lack of data.
    if (habitat_type == ht_ATM && pct_hemlock < 0.35 ) {
      rc = Tree::pdie(species, dbh, rd, stock, stage);
    } else {
      switch (mort_equation_type){
      case 0:
        rc = Tree::pdie(species, dbh, rd, stock, stage);
        break;
      case 1:
        rc = Tree::pdie_ht_simple(habitat_type, species, dbh, rd, stock, stage);
        break;
      case 2:
        rc = Tree::pdie_ht_complex(habitat_type, species, dbh, rd, stock, stage);
        break;
      default:
        printf("Unknown mort equation type (%i) requested at %s:%i\n",
               mort_equation_type, __FILE__, __LINE__);
        exit(1);
      }
    }

    if (gentle_birch_mort && species==sp_yellow_birch && dbh <= 6){
      rc = min(rc, 0.114);
    }

  }
  return rc;
}

double Tree::pdie_ht_complex(htype_t ht, species_t species,
                  double dbh, double rd, double stock, stage_t stage){
  const int n_params = 4;
  const int n_species = 8;

  double rc;

  // For those species where pdie doesn't increase past 50 cm, sub in the SM mort rates
  // but with an offset:
  const int dbh_thresh = 30;
  if (dbh > dbh_thresh &&
```

```
    (species==sp_white_ash || species==sp_green_ash ||
     species==sp_basswood  || species==sp_red_maple) ){
  double offset = Tree::pdie_ht_complex( ht, species, dbh_thresh, rd, stock, stage ) -
    Tree::pdie_ht_complex( ht, sp_sugar_maple, dbh_thresh, rd, stock, stage );

  rc = Tree::pdie_ht_complex( ht, sp_sugar_maple, dbh, rd, stock, stage ) + offset;
  rc = fmin(1, fmax(0, rc)); // Limit to 0-1 range.
} else {
  // These coefficients refer to the "full" mortality model
  //  which includes dbh and stocking interacting with htype.
  // Coefficients here
  double coefs[3][3][n_species][n_params] = {
    { //  AOCa
      { //  Split -- Uneven-aged
        { -11.0919493729187, -1.44682983222491, 0.129409954392803, 1.98199673918403 },
            // sugarmaple
        { 0, -1.4301692508473, 0.120464391752789, -0.220688513111923 }, // hemlock
        { 0, -2.57342617882839, 0.22618393330138, 0.339104105662145 }, // yellowbirch
        { -2.49513019835464, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
        { -18.6306014938655, 0, -0.0277554385091683, 3.13781080527705 }, // ash
        { -15.3054249627291, 0, -0.17683340161845, 2.63859636790818 }, // basswood
        { -8.72132178035137, -2.67969507054725, 0.365898550012711, 1.9035955071559 },
            // ironwood
      },
      { //  Split -- Even-aged
        { -10.7520325224063, -1.44682983222491, 0.129409954392803, 1.98199673918403 },
            // sugarmaple
        { 0, -1.4301692508473, 0.120464391752789, -0.220688513111923 }, // hemlock
        { 0.99382925346374, -2.57342617882839, 0.22618393330138, 0.339104105662145 },
            // yellowbirch
        { -2.49513019835464, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
        { -19.8141143959989, 0, -0.0277554385091683, 3.13781080527705 }, // ash
        { -14.3695805207333, 0, -0.17683340161845, 2.63859636790818 }, // basswood
        { -8.72132178035137, -2.67969507054725, 0.365898550012711, 2.0905923002985 },
            // ironwood
      },
      { //  Unified
        { -12.7668011091723, -1.39566495591577, 0.123195309542577, 2.3547048284027 },
            // sugarmaple
        { 0, -1.4301692508473, 0.120464391752789, -0.220688513111923 }, // hemlock
        { 0, -2.32248168368333, 0.200322309158145, 0.281145919306438 }, // yellowbirch
        { -2.49513019835464, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
        { -16.283566947238, -0.29742856726233, 0, 2.60756571300774 }, // ash
        { -24.2717926203571, 0, -0.164081113113589, 4.53518694015095 }, // basswood
        { -9.37506083639396, -2.44890683572759, 0.3302915397907, 2.14476792592674 }, //
            ironwood
      },
    },
    { //  ATD
      { //  Split -- Uneven-aged
        { -3.14141631758702, -1.61918506190018, 0.129409954392803, 0.605462908173403 },
            // sugarmaple
        { 0, -1.4301692508473, 0.120464391752789, -0.220688513111923 }, // hemlock
        { 0, -2.57342617882839, 0.22618393330138, 0.339104105662145 }, // yellowbirch
        { -2.49513019835464, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
        { -19.6718444022543, 0, -0.0277554385091683, 3.13781080527705 }, // ash
        { -15.3054249627291, 0, -0.17683340161845, 2.63859636790818 }, // basswood
```

```
      { -8.72132178035137, -2.67969507054725, 0.365898550012711, 1.9035955071559 },
          // ironwood
    },
    { //  Split -- Even-aged
      { -2.80149946707461, -1.61918506190018, 0.129409954392803, 0.605462908173403 },
          // sugarmaple
      { 0, -1.4301692508473, 0.120464391752789, -0.220688513111923 }, // hemlock
      { 0.99382925346374, -2.57342617882839, 0.22618393330138, 0.339104105662145 },
          // yellowbirch
      { -2.49513019835464, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
      { -20.8553573043877, 0, -0.0277554385091683, 3.13781080527705 }, // ash
      { -14.3695805207333, 0, -0.17683340161845, 2.63859636790818 }, // basswood
      { -8.72132178035137, -2.67969507054725, 0.365898550012711, 2.0905923002985 },
          // ironwood
    },
    { //  Unified
      { -2.66112451052846, -1.5482278645825, 0.123195309542577, 0.487725394472917 },
          // sugarmaple
      { 0, -1.4301692508473, 0.120464391752789, -0.220688513111923 }, // hemlock
      { 0, -2.32248168368333, 0.200322309158145, 0.281145919306438 }, // yellowbirch
      { -2.49513019835464, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
      { -16.283566947238, -0.29742856726233, 0, 2.60756571300774 }, // ash
      { -1.70141900871589, 0, -0.164081113113589, 0 }, // basswood
      { -10.268141219877, -2.44890683572759, 0.3302915397907, 2.14476792592674 }, //
          ironwood
    },
  },
  { //  ATM
    { //  Split -- Uneven-aged
      { 0, -1.14345741439302, 0.129409954392803, -0.337238409813987 }, // sugarmaple
      { 6.39531972029322, -1.4301692508473, 0.120464391752789, -1.46819654504647 },
          // hemlock
      { 0.763885265257845, -2.57342617882839, 0.22618393330138, 0.339104105662145 },
          // yellowbirch
      { -1.43982736043819, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
      { -19.6718444022543, 0, -0.0277554385091683, 3.13781080527705 }, // ash
      { -15.3054249627291, 0, -0.17683340161845, 2.63859636790818 }, // basswood
      { -8.72132178035137, -2.67969507054725, 0.365898550012711, 1.9035955071559 },
          // ironwood
    },
    { //  Split -- Even-aged
      { 0.339916850512411, -1.14345741439302, 0.129409954392803, -0.337238409813987
          }, // sugarmaple
      { 6.39531972029322, -1.4301692508473, 0.120464391752789, -1.46819654504647 },
          // hemlock
      { 1.75771451872159, -2.57342617882839, 0.22618393330138, 0.339104105662145 },
          // yellowbirch
      { -1.43982736043819, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
      { -20.8553573043877, 0, -0.0277554385091683, 3.13781080527705 }, // ash
      { -14.3695805207333, 0, -0.17683340161845, 2.63859636790818 }, // basswood
      { -8.72132178035137, -2.67969507054725, 0.365898550012711, 2.0905923002985 },
          // ironwood
    },
    { //  Unified
      { 0, -1.08491416621476, 0.123195309542577, -0.360754423770616 }, // sugarmaple
      { 6.39531972029322, -1.4301692508473, 0.120464391752789, -1.46819654504647 },
          // hemlock
```

```
      { 0.557421061207682, -2.32248168368333, 0.200322309158145, 0.281145919306438 },
          // yellowbirch
      { -1.43982736043819, -0.990349631587634, 0.0942201593341137, 0 }, // redmaple
      { -16.283566947238, -0.29742856726233, 0, 2.60756571300774 }, // ash
      { -1.70141900871589, 0, -0.164081113113589, 0 }, // basswood
      { -10.268141219877, -2.44890683572759, 0.3302915397907, 2.14476792592674 }, //
          ironwood
    },
  },
} ;

// Determine a species code, used to index the tables of coefficients
int species_ix;
switch (species){
default:
case sp_sugar_maple:     species_ix = 0; break;
case sp_hemlock:         species_ix = 1; break;
case sp_yellow_birch:    species_ix = 2; break;
case sp_red_maple:       species_ix = 3; break;
case sp_green_ash:
case sp_white_ash:       species_ix = 4; break;
case sp_basswood:        species_ix = 5; break;
case sp_black_cherry:
case sp_mountain_maple:
case sp_ironwood:        species_ix = 6; break;
}

// Determine which set of equations to use:
int structure_ix;
if (use_even_uneven){
  if (stage==stage_sapling || stage==stage_pole || stage==stage_mature) {
    structure_ix = 1;
  } else {
    structure_ix = 0;
  }
} else {
  structure_ix = 2;
}

int ht_ix;
switch(ht){
case ht_AOCa: ht_ix = 0; break;
case ht_ATD:  ht_ix = 1; break;
case ht_ATM:  ht_ix = 2; break;
default:
  printf("ERROR: Unknown habitat type %i in pdie() at %s:%i\n",
         ht, __FILE__, __LINE__ );
  exit(1);
  break;
}

double xi[n_params];
xi[0]= 1; //constant term
xi[1]= sqrt( dbh );
xi[2]= dbh;
xi[3]= log( fmax(stock,1) );
```

```
    double result=0;
    for(int i=0; i<n_params; i++) {
      result += xi[i] * coefs[ht_ix][structure_ix][species_ix][i];
    }

    rc=1/(1+exp(-result));
  }

  return rc;
}

double Tree::pdie_ht_simple(htype_t ht, species_t species,
                  double dbh, double rd, double stock, stage_t stage){
  const int n_params = 4;
  const int n_species = 8;

  double rc;

  // For those species where pdie doesn't increase past 50 cm, sub in the SM mort rates
  // but with an offset:
  const int dbh_thresh = 30;
  if (dbh > dbh_thresh &&
      (species==sp_white_ash || species==sp_green_ash ||
       species==sp_basswood  || species==sp_red_maple) ){
    double offset = Tree::pdie_ht_simple( ht, species, dbh_thresh, rd, stock, stage ) -
      Tree::pdie_ht_simple( ht, sp_sugar_maple, dbh_thresh, rd, stock, stage );

    rc = Tree::pdie_ht_simple( ht, sp_sugar_maple, dbh, rd, stock, stage ) + offset;
    rc = fmin(1, fmax(0, rc)); // Limit to 0-1 range.
  } else {

    double coefs[3][3][n_species][n_params] = {
      { //  AOCa
        { //  Split -- Uneven-aged
          { -5.0796541118625, -1.62094525443602, 0.142138481132722, 0.7880894771428 }, //
              sugarmaple
          { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
              // hemlock
          { 1.37885584299291, -2.55442339867376, 0.224187881831436, 0 }, // yellowbirch
          { -2.67257079861686, -1.3164873723464, 0.131902867460731, 0 }, // redmaple
          { -18.6345529240607, 0, -0.0277524745464901, 3.13862570172295 }, // ash
          { -12.3679215015006, -1.33825313908912, 0, 2.53836270278231 }, // basswood
          { -8.71641647085716, -2.67892649455328, 0.365790256401635, 1.90237575371674 },
              // ironwood
        },
        { //  Split -- Even-aged
          { -8.96746373906252, -1.62094525443602, 0.142138481132722, 1.69542461425505 },
              // sugarmaple
          { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
              // hemlock
          { 2.35302959275649, -2.55442339867376, 0.224187881831436, 0 }, // yellowbirch
          { -2.67257079861686, -1.3164873723464, 0.131902867460731, 0.302848181338032 },
              // redmaple
          { -19.8182208575547, 0, -0.0277524745464901, 3.13862570172295 }, // ash
          { -11.4339250159705, -1.33825313908912, 0, 2.53836270278231 }, // basswood
          { -8.71641647085716, -2.67892649455328, 0.365790256401635, 2.08935677466145 },
              // ironwood
        },
        { //  Unified
```

```
      { -7.48811240415304, -1.49626852130552, 0.128446143768754, 1.32053989937788 },
          // sugarmaple
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { -0.82453232706546, -2.13477287961369, 0.184419816268809, 0.310024962821592 },
          // yellowbirch
      { -2.49825279896328, -0.989439662250345, 0.0941470570771009, 0 }, // redmaple
      { -16.2828478940774, -0.29739997287526, 0, 2.60740852422047 }, // ash
      { -14.4611554231974, -1.32692820512033, 0, 3.1000785071855 }, // basswood
      { -9.3762688828569, -2.44873080195632, 0.330257414756533, 2.14497622317327 },
          // ironwood
    },
  },
  { //  ATD
    { //  Split -- Uneven-aged
      { -4.22742887337506, -1.62094525443602, 0.142138481132722, 0.7880894771428 },
          // sugarmaple
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { 1.7042421216495, -2.55442339867376, 0.224187881831436, 0 }, // yellowbirch
      { -2.67257079861686, -1.3164873723464, 0.131902867460731, 0 }, // redmaple
      { -19.6761896681554, 0, -0.0277524745464901, 3.13862570172295 }, // ash
      { -12.3679215015006, -1.33825313908912, 0, 2.53836270278231 }, // basswood
      { -8.71641647085716, -2.67892649455328, 0.365790256401635, 1.90237575371674 },
          // ironwood
    },
    { //  Split -- Even-aged
      { -8.11523850057508, -1.62094525443602, 0.142138481132722, 1.69542461425505 },
          // sugarmaple
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { 2.67841587141307, -2.55442339867376, 0.224187881831436, 0 }, // yellowbirch
      { -2.67257079861686, -1.3164873723464, 0.131902867460731, 0.302848181338032 },
          // redmaple
      { -20.8598576016494, 0, -0.0277524745464901, 3.13862570172295 }, // ash
      { -11.4339250159705, -1.33825313908912, 0, 2.53836270278231 }, // basswood
      { -8.71641647085716, -2.67892649455328, 0.365790256401635, 2.08935677466145 },
          // ironwood
    },
    { //  Unified
      { -7.082225615309, -1.49626852130552, 0.128446143768754, 1.32053989937788 }, //
          sugarmaple
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { -0.468224282272373, -2.13477287961369, 0.184419816268809, 0.310024962821592
          }, // yellowbirch
      { -2.49825279896328, -0.989439662250345, 0.0941470570771009, 0 }, // redmaple
      { -16.2828478940774, -0.29739997287526, 0, 2.60740852422047 }, // ash
      { -14.4611554231974, -1.32692820512033, 0, 3.1000785071855 }, // basswood
      { -10.2692858126261, -2.44873080195632, 0.330257414756533, 2.14497622317327 },
          // ironwood
    },
  },
  { //  ATM
    { //  Split -- Uneven-aged
      { -3.94309614577309, -1.62094525443602, 0.142138481132722, 0.7880894771428 },
          // sugarmaple
```

```
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { 2.39009821350074, -2.55442339867376, 0.224187881831436, 0 }, // yellowbirch
      { -0.789775377365177, -1.3164873723464, 0.131902867460731, 0 }, // redmaple
      { -19.6761896681554, 0, -0.0277524745464901, 3.13862570172295 }, // ash
      { -12.3679215015006, -1.33825313908912, 0, 2.53836270278231 }, // basswood
      { -8.71641647085716, -2.67892649455328, 0.365790256401635, 1.90237575371674 },
          // ironwood
    },
    { //  Split -- Even-aged
      { -7.8309057729731, -1.62094525443602, 0.142138481132722, 1.69542461425505 },
          // sugarmaple
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { 3.36427196326431, -2.55442339867376, 0.224187881831436, 0 }, // yellowbirch
      { -0.789775377365177, -1.3164873723464, 0.131902867460731, 0.302848181338032 },
          // redmaple
      { -20.8598576016494, 0, -0.0277524745464901, 3.13862570172295 }, // ash
      { -11.4339250159705, -1.33825313908912, 0, 2.53836270278231 }, // basswood
      { -8.71641647085716, -2.67892649455328, 0.365790256401635, 2.08935677466145 },
          // ironwood
    },
    { //  Unified
      { -6.84445233108611, -1.49626852130552, 0.128446143768754, 1.32053989937788 },
          // sugarmaple
      { 5.39293399204969, -1.48632826981557, 0.123484324190217, -1.22918503828234 },
          // hemlock
      { 0, -2.13477287961369, 0.184419816268809, 0.310024962821592 }, // yellowbirch
      { -1.44225710036038, -0.989439662250345, 0.0941470570771009, 0 }, // redmaple
      { -16.2828478940774, -0.29739997287526, 0, 2.60740852422047 }, // ash
      { -14.4611554231974, -1.32692820512033, 0, 3.1000785071855 }, // basswood
      { -10.2692858126261, -2.44873080195632, 0.330257414756533, 2.14497622317327 },
          // ironwood
    },
  },
} ;

// Determine a species code, used to index the tables of coefficients
int species_ix;
switch (species){
default:
case sp_sugar_maple:     species_ix = 0; break;
case sp_hemlock:         species_ix = 1; break;
case sp_yellow_birch:    species_ix = 2; break;
case sp_red_maple:       species_ix = 3; break;
case sp_green_ash:
case sp_white_ash:       species_ix = 4; break;
case sp_basswood:        species_ix = 5; break;
case sp_black_cherry:
case sp_mountain_maple:
case sp_ironwood:        species_ix = 6; break;
}
// Determine which set of equations to use:
int structure_ix;
if (use_even_uneven){
```

```
      if (stage==stage_sapling || stage==stage_pole || stage==stage_mature) {
        structure_ix = 1;
      } else {
        structure_ix = 0;
      }
    } else {
      structure_ix = 2;
    }

    int ht_ix;
    switch(ht){
    case ht_AOCa: ht_ix = 0; break;
    case ht_ATD:  ht_ix = 1; break;
    case ht_ATM:  ht_ix = 2; break;
    default:
      printf("ERROR: Unknown habitat type %i in pdie() at %s:%i\n",
             ht, __FILE__, __LINE__ );
      exit(1);
      break;
    }

    double xi[n_params];
    xi[0]= 1; //constant term
    xi[1]= sqrt( dbh );
    xi[2]= dbh;
    xi[3]= log( fmax(stock,1) );

    double result=0;
    for(int i=0; i<n_params; i++) {
      result += xi[i] * coefs[ht_ix][structure_ix][species_ix][i];
    }

    rc=1/(1+exp(-result));
  }

  return rc;
}
//! Habitat insensitive mortality equations
double Tree::pdie(species_t species, double dbh, double rd, double stock, stage_t stage){
  const int n_params = 4;
  const int n_species = 8;

  double rc;

  // For those species where pdie doesn't increase past 30 cm, sub in the SM mort rates
  // but with an offset:
  const int dbh_thresh = 30;
  if (dbh > dbh_thresh &&
      (species==sp_white_ash || species==sp_green_ash ||
       species==sp_basswood  || species==sp_red_maple) ){
    double offset = Tree::pdie(species, dbh_thresh, rd, stock, stage) -
      Tree::pdie(sp_sugar_maple, dbh_thresh, rd, stock, stage);

    rc = Tree::pdie(sp_sugar_maple, dbh, rd, stock, stage) + offset;
    rc = fmin(1, fmax(0, rc)); // Limit to 0-1 range.
  } else {
    double coefs[3][n_species][n_params] = {
      {// Split -- Uneven-aged
```

```
    {-8.347049,    -1.59610100,  0.1377459,    1.596875 }, //Sm
    { 0,           -1.8212,      0.1524,       0,       }, //Hm
    {0.0000000,    -2.50960000,  0.2154519,    0.3990607}, //Yb
    {-7.523106,    -0.20032720,  0,            0.8546875}, //Rm
    {-17.52870,     0,           -0.03292095,  2.736349 }, //Ash
    {-12.36621,    -1.3381990,   0,            2.53799  }, //BW
    {-9.113437,    -2.7374680,   0.3740731,    1.998865 }  //IW
  }, { // Split -- even-aged
    {-8.347049,    -1.59610100,  0.1377459,    1.596875 }, //Sm
    { 0,           -1.8212,      0.1524,       0,       }, //Hm
    {0.6379802,    -2.50960000,  0.2154519,    0.3990607}, //Yb
    {-7.523106,    -0.20032720,  0,            0.8546875}, //Rm
    {-17.52870,     0,           -0.03292095,  2.736349 }, //Ash
    {-11.43219,    -1.3381990,   0,            2.53799  }, //BW
    {-8.154951,    -2.7374680,   0.3740731,    1.998865 }  //IW
  }, { // Unified
    {-8.347049,    -1.5961010,   0.13774590,   1.596875 }, //Sm
    { 0,           -1.8212,      0.1524,       0,       }, //Hm
    { 0,           -2.3386410,   0.19854060,   0.3446231}, //Yb
    {-7.523106,    -0.2003272,   0,            0.8546875}, //Rm
    {-17.52870,     0,           -0.03292095,  2.736349 }, //Ash
    {-14.46361,    -1.3268370,   0,            3.100503 }, //BW
    {-9.799602,    -2.2811360,   0.31313870,   2.085597 }  //IW
  }
};
// Determine a species code, used to index the tables of coefficients
int species_ix;
switch (species){
default:
case sp_sugar_maple:
  species_ix = 0; break;
case sp_hemlock:
  species_ix = 1; break;
case sp_yellow_birch:
  species_ix = 2; break;
case sp_red_maple:
  species_ix = 3; break;
case sp_green_ash:
case sp_white_ash:
  species_ix = 4; break;
case sp_basswood:
  species_ix = 5; break;
case sp_black_cherry:
case sp_mountain_maple:
case sp_ironwood:
  species_ix = 6; break;
}
// Determine which set of equations to use:
int structure_ix;
if (use_even_uneven){
  if (stage == stage_sapling ||
      stage == stage_pole ||
```

```
          stage == stage_mature ) {
        structure_ix = 1;
      } else {
        structure_ix = 0;
      }
    } else {
      structure_ix = 2;
    }

    double xi[n_params];
    xi[0]= 1; //constant term
    xi[1]= sqrt( dbh );
    xi[2]= dbh;
    xi[3]= log( fmax(stock,1) );

    double result=0;
    for(int i=0; i<n_params; i++) {
      result += xi[i] * coefs[structure_ix][species_ix][i];
    }

    rc=1/(1+exp(-result));
  }

  return rc;
}

//! Probability of mortality
double Tree::pdie_old(double stock, stage_t stage){
  const int n_params = 5;

  //Coefficients here for the hemlock and yellow birch species refer
  //to the no drought condition.
  double coefs[][n_params]={
    //_const_ _(dbh)^.5_ _dbh_    _stock_  _ln(stock)_
    { -1.7169,  -1.7679,   0.15120,  0.011190, 0       }, //  0 SM
    {  0,       -1.8212,   0.1524,   0,        0       }, //  1 Hm
    { -4.9166,  -1.7105,   0.1368,   0,        1.0756  }, //  2 YB
    {  0,       -1.6860,   0.15450,  0,        0       }, //  3 RM
    {  0,       -1.9345,   0.18650,  0.007032, 0       }, //  4 BW
    {  0,       -1.6391,   0.20210,  0,        0       }, //  5 IW
    {  0,       -1.5007,   0.13520,  0,        0       }, //  6 WA
    {  0,       -1.3728,   0.1204,   0,       -0.2369  }, //  7 Hm-drought
    {  0,       -1.1050,   0.1029,   0.012220,-0.5900  }, //  8 YB-drought
  };

  // Determine a species code, used to index the tables of coefficients
  int species_code;
  switch (species){
  default:
  case sp_sugar_maple:
    species_code = 0; break;
  case sp_hemlock:
    species_code = 1; break;
  case sp_white_ash:
    species_code = 6; break;
  case sp_yellow_birch:
    species_code = 2; break;
```

```
  case sp_red_maple:
    species_code = 3; break;
  case sp_basswood:
    species_code = 4; break;
  case sp_black_cherry:
  case sp_mountain_maple:
  case sp_ironwood:
    species_code = 5; break;
  }

  double xi[n_params];
  xi[0]=1; //constant term
  xi[1]=sqrt( dbh );
  xi[2]=dbh;
  xi[3]=stock;
  xi[4]=log( fmax(stock,1) );

  // Preconditions:
  assert( isfinite(xi[1]) && 0<= xi[1] ); //sqrt(dbh)
  assert( isfinite(xi[2]) && 0<= xi[2] ); //dbh
  assert( isfinite(xi[3]) && 0<= xi[3] ); //stock
  assert( isfinite(xi[4]) );              //log(stock)

  double result=0;
  for(int i=0; i<n_params; i++) {
    result += xi[i] * coefs[species_code][i];
  }

  double pdie=1/(1+exp(-result));

  // Postconditions:
  assert( isfinite(pdie) && 0<= pdie && pdie <= 1);

  return pdie;
}

/// Set storm coefficients a, b.
void Tree::setStormCoefs(species_t _species){
  switch (_species) {
  case sp_hemlock:       storm.a = -3.529; storm.b =  0.486; break;
  case sp_red_maple:     storm.a = -0.968; storm.b = -0.021; break;
  case sp_yellow_birch: storm.a = -2.935; storm.b =  0.369; break;
  default:               storm.a = -4.108; storm.b =  0.525; break;
  }
}

/*************************************************************************
Blowdown
*************************************************************************/

//! Probability of blowdown
double  Tree::blowDown(double _si) {
  double probBlowDown;
  probBlowDown = exp(storm.a + storm.b*(sqrt(dbh))+_si )/
    (1 + exp(storm.a + storm.b*(sqrt(dbh)) + _si ));
  return probBlowDown;
}

void Tree::set_dbh(double d){
  dbh=d; clear_caches();
}
```

```
void Tree::set_gap_area(double a){
  if( gap_area==0 && a>0 ){
    gap_age=0;
  } else {
    gap_age++;
  }

  gap_area=a;
}

void Tree::set_age(int a){ gap_age = a; age = a; }
void Tree::set_born_in_sap_stand(){ born_in_sap_stand = true; }

void Tree::set_touching(int i, bool v){ touching[i]=v; }
void Tree::set_shaded(int i, bool v){ shaded[i]=v; }
void Tree::set_facing_gap(int i, bool v){facing_gap[i]=v?(facing_gap[i]+1):0; }

double Tree::get_gap_area(){ return gap_area; }

bool Tree::is_facing_gap(int i) {
  return  0 < facing_gap[i];
}

double Tree::get_cr(int i){ return crown_radius[i]; }

//!Reset cached data calculated based on DBH
void Tree::clear_caches(){
  total_height.is_good = false;
  widest_height.is_good = false;
  base_height.is_good = false;
  eca.is_good = false;
  tca.is_good = false;
  comp_sap_tca.is_good = false;
  comp_sap_ci.is_good = false;
}
//! Store a copy of the tree's habitat in the tree.
void Tree::set_habitat( hdata_t* hd ){
  int px = (int) floor(treeCoordX/10);
  int py = (int) floor(treeCoordZ/10);

  habitat_type = hd->hdata[py][px];
  assert( habitat_type == ht_AOCa ||
          habitat_type == ht_ATD  ||
          habitat_type == ht_ATM );
}

TreePtr Tree::get_xy_neighbor( int i){ return  xy_neighbor[i]; }
TreePtr Tree::get_rz_neighbor( int i){ return  rz_neighbor[i]; }
TreePtr Tree::get_eca_neighbor(int i){ return eca_neighbor[i]; }
TreePtr Tree::get_gap_neighbor(int i){ return gap_neighbor[i]; }

void Tree::set_xy_neighbor( int i, TreePtr t) {  xy_neighbor[i]=t; }
void Tree::set_rz_neighbor( int i, TreePtr t) {  rz_neighbor[i]=t; }
void Tree::set_gap_neighbor(int i, TreePtr t) { gap_neighbor[i]=t; }
void Tree::set_eca_neighbor(int i, TreePtr t) { eca_neighbor[i]=t; }

void Tree::set_ecr(int ix, double r){ exposed_crown_radius[ix]=r; }
```

```
double Tree::get_ecr(int ix){ return exposed_crown_radius[ix]; }

double Tree::get_eca(){
  if( ! eca.is_good ){
    if(status == st_live ){
      double tmp=0;
      for(int i=0; i<4; i++){
        tmp += exposed_crown_radius[i]*exposed_crown_radius[(i+1)%4];
      }
      eca.value = M_PI/4*tmp;
    } else {
      eca.value = 0;
    }
    eca.is_good = true;
  }
  return eca.value;
}

double Tree::get_tca(){
  if( ! tca.is_good ){
    if(status == st_live){
      double tmp=0;
      for(int i=0; i<4; i++){
        tmp += crown_radius[i]*crown_radius[(i+1)%4];
      }
      tca.value = M_PI/4*tmp;
    } else {
      tca.value = 0;
    }
    tca.is_good = true;
  }
  return tca.value;
}

void Tree::add_competitor_sapling(TreePtr comp, double dist){
  competitor_saplings.push_back(comp);
  competitor_distances.push_back(dist);
}

void Tree::clear_competitor_saplings(){
  competitor_saplings.clear();
  competitor_distances.clear();
}

double Tree::predict_mcr(){
  int ix;
  switch(species){
  default:
    if( habitat_type == ht_A0Ca ) { ix=0; } else { ix=1; }
    break;
  case sp_hemlock:       ix=2; break;
  case sp_yellow_birch: ix=3; break;
  case sp_red_maple:     ix=4; break;
  case sp_white_ash:     ix=5; break;
  case sp_basswood:      ix=6; break;
  case sp_ironwood:      ix=7; break;
  }
  double coefs[][3] = {
```

```
    { /* SM A0Ca       */  8.13689, 0.03818, 18.56050 },
    { /* SM ATD/ATM    */  7.91233, 0.02746, 24.93295 },
    { /* Hemlock       */  4.51000, 0.04241, 10.10187 },
    { /* Yellow Birch */  7.76193, 0.04056, 25.38926 },
    { /* Red Maple     */  5.24100, 0.09100, 18.62089 },
    { /* White Ash     */  6.01723, 0.05012, 20.01200 },
    { /* Basswood      */  6.18326, 0.03983, 19.18628 },
    { /* Ironwood      */  3.24642, 0.30031,  5.00321 }};

  double B=coefs[ix][0];
  double C=coefs[ix][1];
  double D=coefs[ix][2];

  double mcr = B * exp(-exp(-C * (dbh - D)));

  return mcr;
}

int Tree::get_age(){ return age; }
int Tree::get_gap_age() { return gap_age; }

//! Determine if a tree shoud be saved
/**
 * Trees are saved
 * 1. as soon as they cross the min_save size threshold.
 * 2. Every save_interval years
 * 3. When they die
 * Note: trees that don't live long enough to cross min_save won't ever
 * show up in the database.
 */
int Tree::need_save(int year){
  int rc = 0;

  // rc = 0 -> no save needed
  // rc = 1 -> non-intial save needed
  // rc = 2 -> initial save needed.
  // rc = 3 -> initial and final save needed.

  if (big_enough_to_save()){
    if (status == st_live){
      if (save_status.need_initial){
        rc = 2;
        save_status.need_initial = false;
      } else if (save_status.need_forced){
        rc = 1;
        save_status.need_forced = false;
      } else {
        rc = (year%save_interval == 0)?1:0;
      }
    } else if (save_status.need_final){
      if (save_status.need_initial){
        rc=3;
        save_status.need_initial = false;
      } else {
        rc=1;
      }
      save_status.need_final = false;
    }
  }

  if (rc>0){ save_status.last = year; }
```

```
  return rc;
}

bool Tree::big_enough_to_save(){ return dbh >= min_diameter; }

void Tree::force_save(){ save_status.need_forced = true; }
```

C.3 Stand.cp

```
//-*- C++ -*-
#include <algo.h>
#include <math.h>
#include <assert.h>
#include <sys/types.h>
#include <sys/stat.h>
#include "Stand.h"
#include "Random.h"
#include "Tree.h"
#include "TreeDB.h"
#include "TreeData.h"
#include "util.h"
#include "StormModel.h"
#include "Harvest.h"
#include "TreeGrid.h"
#include "CanopyGlobals.h"

extern bool use_branch_growth;
extern bool use_height_growth;
extern bool regen_maintain_initial_saplings;
extern bool only_hm_regen;
extern bool only_sm_regen;
extern bool use_test_regen_props;
extern bool forbid_hm_regen;
extern bool use_stem_exclusion;
extern int regen_recovery_time;
extern int disturbance_threshold;
extern int n_saps_to_maintain;

//! Initialize the stand:
void Stand::stand_init(TreeList* tl, hdata_t* hd, char* db_fname,
                       int argc, char* argv[]){
  year=0;
  storm_clock = 0;

  // Set up the treelists, harest lists, and list of group centers:
  babies = new vector<TreePtr>();
  living = new TreeList();
  dead = new TreeList();
  blown_down = new TreeList();
  harvested = new TreeList();
  harvested_left = new TreeList();
  harvesters = new slist<Harvest*>();
  gcs_group_centers = new list<point_t*>();
  storm_queue = new list<double>();

  // Set up the lists of snags / logs / "into the earth" trees
```

```
for( int i=0; i<5; i++){
  cwd_log[i] = new TreeList();
  cwd_snag[i] = new TreeList();
}
cwd_out    = new TreeList();

// Take the habitat type and live tree list passed in:
hdata = hd;
*living = *tl;

// Go through the living list and throw trees onto status-specific lists:
TreeList::iterator it = living->begin();
while ( it != living->end() ){

  status_t tgt_st = (*it)->GetStatus();

  if ( tgt_st != st_live ){
    // Cut trees go on 'harvested', everything else is 'dead',
    // And may have some other CWD attributes as well
    if ( tgt_st == st_cut) { harvested->push_back( *it ); }
    else                   { dead->push_front( *it );     }

    // Check for a CWD status:
    switch (tgt_st){
    case st_log_1:   cwd_log[0]->push_front( *it ); break;
    case st_log_2:   cwd_log[1]->push_front( *it ); break;
    case st_log_3:   cwd_log[2]->push_front( *it ); break;
    case st_log_4:   cwd_log[3]->push_front( *it ); break;
    case st_log_5:   cwd_log[4]->push_front( *it ); break;
    case st_snag_1: cwd_snag[0]->push_front( *it ); break;
    case st_snag_2: cwd_snag[1]->push_front( *it ); break;
    case st_snag_3: cwd_snag[2]->push_front( *it ); break;
    case st_snag_4: cwd_snag[3]->push_front( *it ); break;
    case st_snag_5: cwd_snag[4]->push_front( *it ); break;
    case st_dead:
      break;
    default:
      printf("ERROR: Unknown tree status: %i\n", tgt_st);
      exit(1);
      break;
    }

    it = living->erase(it);
  } else {
    it++;
  }
}

if( db_fname ){
  tdb = new TreeDB();
  tdb->open(db_fname);
}

xmax = hdata->ncol;
ymax = hdata->nrow;

xmax_meters = 10.0 * xmax;
ymax_meters = 10.0 * ymax;

//Insert some check to be sure that the trees on the tree list
```

```
  //fall inside the xmax,ymax that the user gave me.

  if (regen_maintain_initial_saplings){
    if (n_saps_to_maintain < 0){
      // Count up the initial population of saplings, to used in the
      //fixed-regeneration mode
      target_sapling_population = 0;
      for (it = living->begin(); it != living->end(); it++){
        if ( (*it)->GetDBH() <= 6 ){
          target_sapling_population++;
        }
      }
    } else {
      // Presume that the saps to maintain is in per ha, conver to per plot:
      target_sapling_population = n_saps_to_maintain * (xmax*ymax) / 100 ;
    }
  }

  // Set up the counter for stem exclusion / regen delay
  regen_delay_counter = (int**)malloc(xmax*sizeof(int*)); CHK_MEM( regen_delay_counter ) ;

  for (int i=0; i<xmax; i++){
    regen_delay_counter[i] = (int*)malloc( ymax*sizeof(int));  CHK_MEM(
        regen_delay_counter[i] );
  }

  for (int i=0; i<xmax; i++){
    for (int j=0; j<ymax; j++){
      regen_delay_counter[i][j] = regen_recovery_time;
    }
  }

  subplot_100=init_subplot();
  subplot_800=init_subplot();
  subplot_900=init_subplot();
  subplot_trees_100=init_subplot_trees();
  subplot_trees_800=init_subplot_trees();
  subplot_trees_900=init_subplot_trees();

  subplot_stages_2500 = (stage_data_t**)malloc(xmax*sizeof(stage_data_t*)); CHK_MEM(
      subplot_stages_2500 );

  for(int i=0; i<xmax; i++){
    subplot_stages_2500[i]=(stage_data_t*)malloc(ymax*sizeof(stage_data_t)); CHK_MEM(
        subplot_stages_2500[i] );
  }

  the_grid = new TreeGrid(this);
  the_grid->add_trees(living);

  if( db_fname ){
    tdb->init(xmax_meters,ymax_meters);
    tdb->argv_ins( argc, argv );
  }

  my_storm_model = NULL;
}

Stand::subplot_t** Stand::init_subplot(){
  subplot_t** rc = (subplot_t**)malloc(xmax*sizeof(subplot_t*)); CHK_MEM( rc );

  for(int i=0; i<xmax; i++){
```

```
    rc[i]=(subplot_t*)malloc(ymax*sizeof(subplot_t)); CHK_MEM( rc[i] );
  }
  return(rc);
}

TreeList*** Stand::init_subplot_trees(){
  TreeList*** rc = (TreeList***)malloc(xmax*sizeof(TreeList**)); CHK_MEM( rc );
  for(int i=0; i<xmax; i++){
    rc[i]=(TreeList**)malloc(ymax*sizeof(TreeList*));  CHK_MEM( rc[i] );
    for(int j=0; j<ymax; j++){
      rc[i][j] = new TreeList();
    }
  }
  return rc;
}

/// Stand constructor
/** fname     - filename for the treelist
    db_fname - filename for the output database
    mx - number of x replicates
    my - number of y replicates
*/
Stand::Stand(char* fname, char* db_fname, int mx, int my,
             int argc, char* argv[]){
  hdata_t *hd = NULL;
  TreeData *td = new TreeData();
  TreeList* tl = new TreeList();

  td->ReadTreeList( fname, tl, hd, mx, my);
  stand_init(tl, hd, db_fname, argc, argv);

  delete td;
  delete tl;
}

/// Erases all the instance varaibles.
Stand::~Stand(){
  TreeList::iterator it;

  delete living;
  delete dead;
  delete blown_down;
  delete harvested;
  delete harvested_left;
}

void Stand::db_close(){ if(tdb){ tdb->close(); } }

void Stand::save(){ tdb->insert(this); }

void Stand::inc_year(){
  the_grid->remove_dead();
  year++;
}

// Perform necessary calculations to simulate growth/mort/etc:
void Stand::calculate(){
  xoff=10*urand01()-5;
  yoff=10*urand01()-5;

  calculate_stocking_grid( living );
```

```
  if (use_branch_growth || use_height_growth){
    calculate_crown_status();
  }

  if (use_height_growth){
    find_gaps();
  }
}

/** Sums up the relevant basal areas and dbh sums so that computing
    stocking on the 100m^2 plots and the 900m^2 donuts is fast/easy.
    This could be extended to any plot size that is a multiple of
    100m^2.  This function does not compute stocking, but
    FixedWidthStocking() in Tree.cp uses the data that this computes.
    Unfortunately, this needs to be re-run every year because of our
    random offset procedure. Otherwise, we could just be updating the
    subplot_trees lists and calculating all the rest based on those.
    */
void Stand::calculate_stocking_grid( TreeList *aTreeList){
  TreeList::iterator iter;
  int x,y;

  sp_prop.px = -1;
  sp_prop.py = -1;

  //clear the data entry for this subplot.
  for(x=0; x<xmax; x++){
    bzero( (void*)subplot_100[x], ymax*sizeof(subplot_t) );
    bzero( (void*)subplot_800[x], ymax*sizeof(subplot_t) );
    bzero( (void*)subplot_900[x], ymax*sizeof(subplot_t) );
    bzero( (void*)subplot_stages_2500[x], ymax*sizeof(stage_data_t) );

    for(y=0; y<ymax; y++){
      subplot_trees_100[x][y]->clear();
      subplot_trees_800[x][y]->clear();
      subplot_trees_900[x][y]->clear();
    }
  }

  double yoff_800[8] = { +10, +10, +10,   0,   0, -10, -10, -10 };
  double xoff_800[8] = { +10,   0, -10, +10, -10, +10,   0, -10 };

  double yoff_900[9] = { +10, +10, +10,   0,   0,   0, -10, -10, -10 };
  double xoff_900[9] = { +10,   0, -10, +10,   0, -10, +10,   0, -10 };

  double yoff_2500[25] = {+20, +20, +20, +20, +20,
                          +10, +10, +10, +10, +10,
                            0,   0,   0,   0,   0,
                          -10, -10, -10, -10, -10,
                          -20, -20, -20, -20, -20 };
  double xoff_2500[25] = {+20, +10,   0, -10, -20,
                          +20, +10,   0, -10, -20,
                          +20, +10,   0, -10, -20,
                          +20, +10,   0, -10, -20,
                          +20, +10,   0, -10, -20 };

  // Step One: Assign each living tree to a subplot.
  for( iter = aTreeList->begin(); iter != aTreeList->end(); iter++ ){
    double tree_x = (*iter)->GetXCoord();
    double tree_y = (*iter)->GetYCoord();
```

```
  // Calculate x/y plot coordinatees
  x = x_index(tree_x);
  y = y_index(tree_y);

  // 100 m2 subplot s
  subplot_trees_100[x][y]->push_front(*iter);

  //800m2 'donut' subplot:
  for( int i=0; i<8; i++ ){
    int xx= x_index(tree_x+xoff_800[i]);
    int yy= y_index(tree_y+yoff_800[i]);

    subplot_trees_800[xx][yy]->push_front(*iter);
  }

  //900 subplot:
  for( int i=0; i<9; i++ ){
    int xx= x_index(tree_x+xoff_900[i]);
    int yy= y_index(tree_y+yoff_900[i]);

    subplot_trees_900[xx][yy]->push_front(*iter);
  }
}

//Step two: sort each subplot by height
//          calculate ref_height for each subplot
for( int i=0; i<xmax; i++){
  for( int j=0; j<ymax; j++){
    subplot_100[i][j].ref_ht=ref_ht(subplot_trees_100[i][j]);
    subplot_800[i][j].ref_ht=ref_ht(subplot_trees_800[i][j]);
    subplot_900[i][j].ref_ht=ref_ht(subplot_trees_900[i][j]);
  }
}

// Iteration 2, Now, knowing who the dom/codom trees are,
for( iter = aTreeList->begin(); iter != aTreeList->end(); iter++ ){
  // Calculate x/y plot coordinatees
  double tree_x = (*iter)->GetXCoord();
  double tree_y = (*iter)->GetYCoord();

  x = x_index( tree_x );
  y = y_index( tree_y );

  // 100 m2 subplot:
  subplot_t* cur = &(subplot_100[x][y]);
  process_subplot(cur, iter);

  //800 m2 'donut' subplot:
  for( int i=0; i<8; i++ ){
    int xx= x_index(tree_x+xoff_800[i]);
    int yy= y_index(tree_y+yoff_800[i]);

    subplot_t* cur = &(subplot_800[xx][yy]);
    process_subplot(cur, iter);
  }

  //900 m2 subplot:
  for( int i=0; i<9; i++ ){
    int xx= x_index(tree_x+xoff_900[i]);
    int yy= y_index(tree_y+yoff_900[i]);

    subplot_t* cur = &(subplot_900[xx][yy]);
```

```
      process_subplot(cur, iter);
    }

    // 2500 m2 (50x50m) subplot for stand stages:
    for (int i=0; i<25; i++ ){
      int xx=x_index(tree_x+xoff_2500[i]);
      int yy=y_index(tree_y+yoff_2500[i]);

      if ( (*iter)->GetSpecies() != sp_ironwood ){
        double dbh = (*iter)->GetDBH();
        double ba = M_PI*pow(dbh/200,2);

        double lrg_cutoff =
          (hdata->hdata[yy][xx] == ht_ATM)?
          44 : 46;

        if (dbh<12) {
          subplot_stages_2500[xx][yy].sap_ba += ba;
          if ((*iter)->get_eca() >= 0.25 &&
              (*iter)->get_gap_area() >= 2.75) {
            subplot_stages_2500[xx][yy].gap_sap_ba += ba;
          }
        }
        else if (dbh<26)          { subplot_stages_2500[xx][yy].pol_ba += ba; }
        else if (dbh<lrg_cutoff) { subplot_stages_2500[xx][yy].mat_ba += ba; }
        else                      { subplot_stages_2500[xx][yy].lrg_ba += ba; }
      }
    }
  }
}

/** Updates the per-plot summary statistics.  Called one per tree on a
    subplot.  Sorts each tree into a size class (for the q-ratio
    calculation in the regeneration model), and totals up subplot ba,
    subplot dominant ba, number of saplings, per-species sapling ba,
    and per-species counts of dominants.
 */
void Stand::process_subplot(subplot_t* cur, TreeList::iterator tree){
  int sp = subplot_idx( (*tree)->GetSpecies() );
  double dbh = (*tree)->GetDBH();
  double ba = M_PI*pow(dbh/200,2);
  double height = (*tree)->GetTotalHeight();

  cur->sum_ba+=ba;
  cur->ba[sp]+=ba;

  if( height >= 0.66*cur->ref_ht ){
    //Dom/codom
    cur->N++;
    cur->n[sp]++;
    cur->sum_dbh += dbh;
    cur->sum_ht  += height;
  }

  if( dbh <= 6 ){
    cur->n_sap++;
    cur->sap_sum_ba+=ba;
    cur->sap_ba[sp]+=ba;
  }
}

/** Compute reference height for a given treelist, expects to be used
```

```
    on an appropriate 'subplot_trees' entry.
 */
double Stand::ref_ht(TreeList* cur){
  double rc=0;

  Tree::height_comp_t ht;
  cur->sort(ht);

  int n=lround(fmax(1,0.10*cur->size() ));

  if( cur->size() > 0 ){

    TreeList::iterator it=cur->begin();

    for(int i=0; i<n; i++){
      rc+=(*it++)->GetTotalHeight();
    }
  }

  rc/=n;
  return(rc);
}

double Stand::ref_ba(double phemlock, double pbasswood, double avg_dbh){
  double rc;
  avg_dbh = fmin(62, avg_dbh);
  if( (phemlock < 0.2 && pbasswood < 0.2 )) {
    rc = 6.7697+0.618*avg_dbh-0.005*pow(avg_dbh,2);
  } else if( 0.20 <= phemlock && phemlock < 0.5 ) {
    rc = 8.819 + 0.6583*avg_dbh-0.0055*pow(avg_dbh,2);
  } else if( 0.50 <= phemlock ){
    rc = 12.609 + 0.7498*avg_dbh - 0.0064*pow(avg_dbh,2);
  } else if( 0.20 <= pbasswood && pbasswood < 0.5 ){
    rc = 6.5779 + 0.7217*avg_dbh - 0.006*pow(avg_dbh,2);
  } else if ( 0.50 <= pbasswood ) {
    rc = 9.377+0.798*avg_dbh - 0.0069*pow(avg_dbh,2);
  } else {
    printf("Error at %s:%i - unexpected species proportion\n",
           __FILE__, __LINE__);
  }
  return fmax(0.001,rc); // always return nonzero
}
/// RefBA
/** Calculate reference basal area for use in other stocking calculations.
 */
double Stand::ref_ba(subplot_t* cur ){
  double phemlock =cur->sum_ba>0?cur->ba[subplot_idx(sp_hemlock)]/cur->sum_ba:0;
  double pbasswood=cur->sum_ba>0?cur->ba[subplot_idx(sp_basswood)]/cur->sum_ba:0;
  double avg_dbh = cur->sum_dbh / fmax(1,cur->N);
  return ref_ba( phemlock, pbasswood, avg_dbh);
}

double Stand::stocking(subplot_t *cur, double psize, double subj_ba ){
  double my_ref_ba = ref_ba(cur);
  double obs_ba = 10000*( cur->sum_ba - subj_ba )/psize;
  double stock=100 * obs_ba / fmax(0.001, my_ref_ba );
  return fmax(0,stock);
}
```

```
//The truly lazy programmer will write a program which then writes his
//program :

#define DFN_SINGLE_PLOT_STAT(FNAME,PSIZE,FCN)                          \
  double Stand::FNAME##_##PSIZE(int xi, int yi){                       \
    double ps = PSIZE;                                                 \
    subplot_t* cur = &( subplot_##PSIZE[xi][yi] );                     \
    return FCN;                                                        \
  }

#define DFN_SINGLE_TREE_STAT(FNAME,PSIZE,FCN)                          \
  double Stand::FNAME##_##PSIZE(TreePtr t){                            \
    double ps = PSIZE;                                                 \
    subplot_t* cur = &( subplot_##PSIZE[x_index(t->GetXCoord())]       \
                        [y_index(t->GetYCoord())] );                   \
    return FCN;                                                       \
  }

#define DFN_PLOT_STAT(FNAME,FCN)      \
  DFN_SINGLE_PLOT_STAT(FNAME,100,FCN) \
  DFN_SINGLE_PLOT_STAT(FNAME,800,FCN) \
  DFN_SINGLE_PLOT_STAT(FNAME,900,FCN)
#define DFN_TREE_STAT(FNAME,FCN)      \
  DFN_SINGLE_TREE_STAT(FNAME,100,FCN) \
  DFN_SINGLE_TREE_STAT(FNAME,800,FCN) \
  DFN_SINGLE_TREE_STAT(FNAME,900,FCN)

DFN_PLOT_STAT(stocking,stocking(cur,ps,0) )
DFN_PLOT_STAT(average_height,(cur->N>0)?(cur->sum_ht/cur->N):0 )

DFN_TREE_STAT(stocking,stocking(cur,ps,t->GetBA()) )
DFN_TREE_STAT(average_height,(cur->N>0)?(cur->sum_ht/cur->N):0 )
DFN_TREE_STAT(average_diam, (cur->N>0)?(cur->sum_dbh/cur->N):0 )

#undef DFN_SINGLE_PLOT_STAT
#undef DFN_SINGLE_TREE_STAT
#undef DFN_PLOT_STAT
#undef DFN_TREE_STAT

// Sometimes, I *really* wish that C++ had actual meta-programming
// features so that I wouldn't find myself pulling pre-processor
// tricks like this:
#define DFN_BA_UPDATE(MTYPE)                                           \
  void Stand::update_##MTYPE##_ba( TreePtr t ){                        \
    double tree_x = t->GetXCoord();                                    \
    double tree_y = t->GetYCoord();                                    \
    double ba = t->GetBA();                                            \
    double yoff_900[9] = {+10,+10,+10,  0,  0,  0,-10,-10,-10 };      \
    double xoff_900[9] = {+10,  0,-10,+10,  0,-10,+10,  0,-10 };      \
    for( int i=0; i<9; i++ ){                                          \
      int xx= x_index(tree_x+xoff_900[i]);                             \
      int yy= y_index(tree_y+yoff_900[i]);                             \
      subplot_t* cur = &(subplot_900[xx][yy]);                         \
      cur->MTYPE##_ba += ba;                                           \
    }                                                                  \
  }

DFN_BA_UPDATE(harv)
```

```
#undef DFN_BA_UPDATE

///PercentBA
/** Determine the percent basal area of a given species on a 900m^2 plot
    centered at (x,y).  The species index sp must be a species index,
    not a species code.  If you have a species code, use subplot_idx to
    convert it to an index.
*/
double Stand::PercentBA(species_t sp, int xi, int yi){
  int sp_ix = subplot_idx(sp);
  subplot_t* cur = &(subplot_900[xi][yi]) ;
  return cur->sum_ba>0 ? cur->ba[sp_ix] / cur->sum_ba : 0 ;
}

int Stand::x_index(double x ){
  double xp = x+xoff;
  double xpp= xp - xmax_meters*floor(xp/xmax_meters);
  return (int)floor( xpp/10 );
}

int Stand::y_index(double y ){
  double yp = y+yoff;
  double ypp= yp - ymax_meters*floor(yp/ymax_meters);
  return (int)floor( ypp/10 );
}

///Convert from a species code to a species index.
int Stand::subplot_idx(species_t s){
  int rc;
  switch(s){
  case sp_basswood:
    rc=0; break;
  case sp_sugar_maple:
    rc=1; break;
  case sp_white_ash:
  case sp_american_elm:
  case sp_northern_red_oak:
  case sp_green_ash:
  case sp_paper_birch:
  case sp_black_cherry:
    rc=2; break;
  case sp_mountain_maple:
  case sp_balsam_fir:
  case sp_ironwood:
    rc=3; break;
  case sp_hemlock:
    rc=4; break;
  case sp_yellow_birch:
    rc=5; break;
  case sp_red_maple:
    rc=6; break;
  case sp_white_pine:
    rc=7; break;
  default:
    rc=1; break; // Unknown -> SM
  }
```

```
  return rc;
}

/** Determine the stage of development for a specified 2500 m2 area.
 */
stage_t Stand::get_stage(int x_ix, int y_ix){
  stage_t rc;

  double sap_ba = subplot_stages_2500[x_ix][y_ix].sap_ba;
  double pol_ba = subplot_stages_2500[x_ix][y_ix].pol_ba;
  double mat_ba = subplot_stages_2500[x_ix][y_ix].mat_ba;
  double lrg_ba = subplot_stages_2500[x_ix][y_ix].lrg_ba;
  double all_ba = sap_ba + pol_ba + mat_ba + lrg_ba;

  double gap_sap_ba = round( 1000 * subplot_stages_2500[x_ix][y_ix].gap_sap_ba) / 1000;

  double et_thr = (PercentBA(sp_hemlock,x_ix,y_ix) >= 0.25) ? 0.10 : 0.08;

  // Note: ba thresholds are divided by 4 because we're using a 1/4th HA plot
  if (mat_ba + lrg_ba >= 20/4) {
    if (lrg_ba/all_ba > 0.45 &&
        lrg_ba/mat_ba > 1.1) {
      if (lrg_ba/mat_ba > 1.4 &&
          (sap_ba+pol_ba)/all_ba >= 0.10 &&
          gap_sap_ba >= 0.006 &&
          (sap_ba+pol_ba)/all_ba < 0.20 )         { rc = stage_og_ss;   }
      else {
        if (lrg_ba/mat_ba > 1.75 &&
            (sap_ba+pol_ba)/all_ba < et_thr)      { rc = stage_og_et;   }
        else {
          if (lrg_ba/mat_ba >= 1.7 &&
              (sap_ba+pol_ba)/all_ba<0.20)        { rc = stage_og_lt;   }
          else                                    { rc = stage_og_la;   }
        }
      }
    } else                                        { rc = stage_mature;  }
  } else {
    if (mat_ba + lrg_ba > 10/4) {
      if (pol_ba/all_ba > 0.30)                   { rc = stage_pole;    }
      else                                        { rc = stage_msm;     }
    } else {
      if (pol_ba+mat_ba+lrg_ba >= 10/4 &&
          pol_ba / all_ba > 0.30 )                { rc = stage_pole;    }
      else                                        { rc = stage_sapling; }
    }
  }

  return rc;
}

stage_t Stand::get_stage(TreePtr t){
  return get_stage( x_index( t->GetXCoord() ),
                    y_index( t->GetYCoord() ) );
}

/** Generate a new tree with given coordinates and stochastically determined
    initial size and species.
 */
TreePtr Stand::make_baby(double tree_x, double tree_y, species_t sp,
                         double st900, bool is_sap_stand ){
```

```
  //Distribution of initial sapling sizes is distributed lognormally
  // with ln(mean)=1.213, ln(var)=0.04052
  double dbh = fmax(2,fmin(6,exp( rnorm(1.213, sqrt(0.04052)) )));
  int sap_age = lround(10.31451 + 0.11295*st900 );

  TreePtr baby (new Tree(tree_x,tree_y,dbh,sp,hdata) );
  baby->set_gap_area(0);
  baby->set_age(sap_age);
  if (is_sap_stand){
    baby->set_born_in_sap_stand();
  }

  return baby;
}

TreePtr Stand::make_baby( double tree_x, double tree_y, species_t sp){
  int xi = x_index(tree_x);
  int yi = y_index(tree_y);

  double st900 = stocking_900(xi,yi);
  bool is_sap_stand = get_stage(xi,yi) == stage_sapling;

  return make_baby( tree_x, tree_y, sp, st900, is_sap_stand);
}


TreePtr Stand::make_baby(int xi, int yi, species_t sp){
  double st900 = stocking_900(xi,yi);

  double tree_x = 10.0*(xi+urand01())+xoff;
  double tree_y = 10.0*(yi+urand01())+yoff;
  tree_x = tree_x - xmax_meters*floor(tree_x/xmax_meters);
  tree_y = tree_y - ymax_meters*floor(tree_y/ymax_meters);

  bool is_sap_stand = get_stage(xi,yi) == stage_sapling;

  return make_baby( tree_x, tree_y, sp, st900, is_sap_stand );
}

int Stand::number_of_saplings(int xi, int yi){
  double st100 = stocking_100(xi,yi);
  double st800 = stocking_800(xi,yi);
  double st900 = stocking_900(xi,yi);
  int cur_saps =  subplot_100[xi][yi].n_sap;

  return number_of_saplings(xi, yi, st100, st800, st900, cur_saps);
}

int Stand::number_of_saplings(int xi, int yi, double st100, double st800, double st900,
    int cur_saps){
  int rc = 0;

  //explosions are bad.
  assert(isfinite(st100));
  assert(isfinite(st800));
  st100 = fmax(st100, 1);
  st800 = fmax(st800, 1);

  double deficit;
  if (regen_delay_counter[xi][yi] > 0){
```

```
    deficit = 0;
  } else {
    double sqsap = 14.4564 - 2.0407*log(st800) - 0.5332*log(st100);
    double expected_saps = pow( fmax(0,sqsap), 2);

    deficit = fmax(0, expected_saps - cur_saps);
    if (deficit > 3){
      deficit /= sapling_time_to_2cm(xi, yi);
    }

    // Add in whole numbers of trees
    rc      += floor(deficit);
    deficit -= floor(deficit);
  }

  // Stochastically add fractional trees:
  if (deficit>0){
    if (urand01()<deficit){ rc+=1; }
  }

  return rc;
}

//! Length of time for a seedling to reach 2cm.
/** Calculate how many years it should take for a seedling to reach
    2.5cm based on pre-removal stocking and including stochastic
    variation.
*/
int Stand::sapling_time_to_2cm(int subplot_i, int subplot_j){
  return 10.31451 + 0.11295 * stocking_900(subplot_i, subplot_j);
}

//! Remove sapling layer inside a harvest unit
void Stand::clean_around_and_delay(TreePtr t){
  int xi = x_index(t->GetXCoord()),
    yi = y_index(t->GetYCoord());
  regen_delay_counter[xi][yi] =
    max( regen_delay_counter[xi][yi],
         sapling_time_to_2cm( xi, yi ) );
}

//! Plant
void Stand::plant_tree(double x, double y, species_t sp, double dbh){
  TreePtr baby = make_baby(x, y, sp);
  baby->set_dbh(dbh);

  // If we planted a sapling, update the counts of saplings so that CANOPY sees this one
  // when deciding how many saplings should recruit.
  if (dbh <= 6){
    int px = x_index( x );
    int py = y_index( y );
    subplot_t* cur =&(subplot_100[px][py]);
    cur->n_sap++;
  }

  living->push_front( baby );
  if (the_grid) { the_grid -> add_tree( baby ); }
}

//! Regenerate
/** The regeneration routine.
    Adds saplings (random spatial pattern) to 100m^2 plots based on the
    stocking and surrounding donut.
    Assumes that our plots go from (0,0) to some (xmax,ymax).
```

```
*/
void Stand::regenerate_grid(){
  for (int xi=0; xi<xmax; xi++){
    for (int yi=0; yi<ymax; yi++){
      int nsap = number_of_saplings(xi, yi);
      for (int s=0; s<nsap; s++){
        species_t sp = RandomSpecies(xi,yi);
        TreePtr baby = make_baby(xi,yi, sp);
        babies->push_back(baby);
      }
    }
  }
}

/** Perfom regeneration based on group centered stocking assessments.
 */
void Stand::regenerate_gcs(){
  double r100 = sqrt(100/M_PI);
  double r900 = sqrt(900/M_PI);

  // Grab the new centers from the harvesters
  for (slist<Harvest*>::iterator it = harvesters->begin();
       it != harvesters->end(); it++){
    point_t* pt;
    while ( (pt=(*it)->get_group()) != NULL ){
      gcs_group_centers->push_back( pt );
    }
  }

  for (list<point_t*>::iterator it = gcs_group_centers->begin();
       it != gcs_group_centers->end(); it++ ){
    point_t* pt = (*it);

    TreeList trees_100, trees_800, trees_900;

    TreePtr t;
    int current_saps = 0;

    the_grid->set_search_center( pt->x, pt->y );

    while ( (t=the_grid->get_next_tree()) != NULL ){
      double gx = (10.0*xmax)/2,           gy = (10.0*ymax)/2;
      double dx = gx - pt->x,              dy = gy - pt->y;
      double tx = t->GetXCoord() + dx,     ty = t->GetYCoord() + dy;

      // Warp the tre back in if it was shifted out of the plot
      tx = tx - 10.0*xmax*floor( tx/(10.0*xmax) );
      ty = ty - 10.0*ymax*floor( ty/(10.0*ymax) );

      double d = hypot( gx-tx, gy-ty );

      if ( d<=r100 ) {
        trees_100.push_front(t);
        if (t->GetDBH()<6) current_saps++;
      }
      if (    r100<d && d<=r900 ){ trees_800.push_front(t); }
      if (              d<=r900 ){ trees_900.push_front(t); }
    }

    double stock_100, stock_800, stock_900,
                   phm_900, pwa_900, pbw_900, piw_900;
```

```
    compute_circular_plot( &trees_100, 100, stock_100);
    compute_circular_plot( &trees_800, 800, stock_800);
    compute_circular_plot( &trees_900, 900, stock_900,
                           phm_900, pwa_900, pbw_900, piw_900);

    int nsap = number_of_saplings(x_index(pt->x), y_index(pt->y),
                                  stock_100, stock_800, stock_900, current_saps);

    for (int s=0; s<nsap; s++){
      double tree_r     = r100*urand01();
      double tree_theta = 2*M_PI*urand01();

      double tree_x= pt->x + tree_r*cos(tree_theta);
      double tree_y= pt->y + tree_r*sin(tree_theta);

      tree_x = tree_x - 10.0*xmax*floor(tree_x/(10.0*xmax));
      tree_y = tree_y - 10.0*ymax*floor(tree_y/(10.0*ymax));

      species_t sp = RandomSpecies(x_index(pt->x),
                                   y_index(pt->y), stock_900,
                                   phm_900, pwa_900, pbw_900, piw_900);

      TreePtr baby = make_baby( tree_x, tree_y, sp, stock_900, true);

      babies->push_back(baby);
    }
  } // foreach group center

  // Now to clean up non-delayed groups
  list<point_t*>::iterator it  = gcs_group_centers->begin();
  while (it != gcs_group_centers->end() ){
    int xi = x_index( (*it)->x );
    int yi = y_index( (*it)->y );

    if (regen_delay_counter[xi][yi] <= 0){
      free(*it);
      it = gcs_group_centers->erase(it);
    } else {
      it++;
    }
  }
}

//! Compute circular plot stocking returning stocking and species proportions.
void Stand::compute_circular_plot( TreeList *tl, double plot_size,
                                   double &stock,
                                   double &pct_hm, double &pct_wa,
                                   double &pct_bw, double &pct_iw) {
  double sum_ba=0, sum_d=0;
  double hm_ba=0, bw_ba=0, wa_ba=0, iw_ba=0;
  int n=0;

  for (TreeList::iterator it=tl->begin();
       it != tl->end(); it++){
    double   dbh = (*it)->GetDBH();
    double    ba = (*it)->GetBA();
    species_t sp = (*it)->GetSpecies();
    sum_ba += ba;
```

```
      sum_d  += dbh;
      n++;
      switch(sp) {
      case sp_hemlock:   hm_ba += ba; break;
      case sp_basswood:  bw_ba += ba; break;
      case sp_white_ash: wa_ba += ba; break;
      case sp_ironwood:  iw_ba += ba; break;
      default: break;
      }
    }

    pct_hm = n>0 ? hm_ba / sum_ba : 0;
    pct_bw = n>0 ? bw_ba / sum_ba : 0;
    pct_wa = n>0 ? wa_ba / sum_ba : 0;
    pct_iw = n>0 ? iw_ba / sum_ba : 0;

    double obs_ba = 10000*sum_ba/plot_size;
    stock = 100* obs_ba / fmax(0.001, ref_ba(pct_hm, pct_bw, n>0?sum_d/n:0));
}

//! Compute circular plot, for just the stocking values
void Stand::compute_circular_plot( TreeList *tl, double plot_size, double &stock)
{
    double pct_hm, pct_wa, pct_bw, pct_iw;
    compute_circular_plot( tl, plot_size, stock, pct_hm, pct_wa, pct_bw, pct_iw);
}

void Stand::regenerate(){
    // Update the regen delay counts
    for (int i=0; i<xmax; i++){
      for (int j=0; j<ymax; j++){
        if (regen_delay_counter[i][j]>0) {
          regen_delay_counter[i][j]--;
        }
      }
    }

    // For pole stands, turn on our "stem exclusion" mode
    if (use_stem_exclusion){
      for (int xi=0; xi<xmax; xi++){
        for (int yi=0; yi<ymax; yi++){
          // Next determine if we're in stem exclusion
          stage_t my_stage = get_stage(xi,yi);
          double sap_ba = subplot_stages_2500[xi][yi].sap_ba;
          double pol_ba = subplot_stages_2500[xi][yi].pol_ba;
          double mat_ba = subplot_stages_2500[xi][yi].mat_ba;
          double lrg_ba = subplot_stages_2500[xi][yi].lrg_ba;
          double all_ba = sap_ba + pol_ba + mat_ba + lrg_ba;

          // Use 1/4 because the 2500 m2 plot is 1/4th ha.
          if (all_ba > 20/4 &&
              (pol_ba/all_ba >= 0.20 || mat_ba/all_ba>=0.35)) {
            regen_delay_counter[xi][yi] =
              max( regen_delay_counter[xi][yi], sapling_time_to_2cm(xi, yi));
          }
        }
      }
    }

    // Before adding anything, take a census of how many babies we're
    // working with:
    int current_sapling_population = 0;
```

```
if (regen_maintain_initial_saplings){
  for (int xi=0; xi<xmax; xi++){
    for (int yi=0; yi<ymax; yi++){
      current_sapling_population += subplot_100[xi][yi].n_sap;
    }
  }
}

// Do the group-centered stocking evaluation.
regenerate_gcs();

vector<TreePtr>::iterator it;
// Update the counts of saplings on each subplot, as if the babies had all been added
// We'll need to do this so that the grid-based regen sees the stand as it would be
//  if all the trees get added:
for (it=babies->begin(); it!=babies->end(); it++){
  int px = x_index( (*it)->GetXCoord() );
  int py = y_index( (*it)->GetYCoord() );
  subplot_t* cur = &(subplot_100[px][py]);
  cur->n_sap++;
}

// Keep track of how many babies were added this way.
int n_to_skip = babies->size();

// Next, do the grid-based regeneration.
regenerate_grid();

// Advance our iterator so that it's pointing at the grid-based
// babies
it = babies->begin();
for (int i=0; i<n_to_skip; i++){ it++; }

//then update our count of saplings per subplot:
while (it != babies->end()){
  int px = x_index( (*it)->GetXCoord() );
  int py = y_index( (*it)->GetYCoord() );
  subplot_t* cur = &(subplot_100[px][py]);
  cur->n_sap++;

  it++;
}

// Our per-sapling counts are now accurate assuming we will add all the babies
if (regen_maintain_initial_saplings){
  // If we're maintaining a fixed population, select a random subset
  // to add.  Shuffle the list of babies, then whittle it down
  // from the end until we've got the right number of them.
  random_shuffle(babies->begin(), babies->end());

  while ( (babies->size() + current_sapling_population)
            > target_sapling_population && babies->size() > 0 ){
    TreePtr t = babies->back();

    // Again, twiddle the sapling counts appropriately
    //  to account for the babies we won't be adding:
    int px = x_index( t->GetXCoord() );
    int py = y_index( t->GetYCoord() );
    subplot_t* cur = &(subplot_100[px][py]);
    cur->n_sap--;

    babies->pop_back();
  }
```

```
  }
  // Add the selected babies to the live tree list:
  for (it=babies->begin(); it!=babies->end(); it++){
    living->push_front( *it );
    if (the_grid) { the_grid->add_tree( *it ); }
  }
  // And clear the 'babies' list
  babies->clear();
}
///Select a species for a new recruit.
species_t Stand::RandomSpecies(int px, int py){
  double pct_hm = PercentBA(sp_hemlock,px,py);
  double pct_wa = PercentBA(sp_white_ash,px,py);
  double pct_bw = PercentBA(sp_basswood,px,py);
  double pct_iw = PercentBA(sp_ironwood,px,py);
  double st900 = stocking_900(px,py);

  return RandomSpecies(px, py, st900,
                       pct_hm, pct_wa, pct_bw, pct_iw);
}

species_t Stand::RandomSpecies(int px, int py, double st900,
                               double pct_hm, double pct_wa, double pct_bw, double
                                   pct_iw){
  species_t rc;
  const int n_species = 7;
  const int n_params  = 9;

  if (only_sm_regen){
    return sp_sugar_maple;
  }

  if (only_hm_regen){
    return sp_hemlock;
  }

  if (! (sp_prop.px==px && sp_prop.py==py) ){
    double pi[n_species], pip[n_species], sum_pi=0;

    //For now, the coefficients below for Hemlock refer to the
    //not-browsed condition.

    double coefs[3][n_species][n_params] = {
      //A0Ca      _INT_     _lnSt_    _lSt*hb_ _ash_    _bass_    _iron_   _StGr80_ _HmGr60_
          _LnPctHm_
      {/*BW */ {-2.89174, -0.25357, 0,        0,        0.17670, 0,        0,      0,      0
               },
       /*SM */ { 0,         0.53854, 0,        0,        0,        0,        0,      0,      0
               },
       /*WA */ { 2.39251, -1.77969, 0,        1.73752, 0,        0,        0,      0,      0
               },
       /*IW */ {-7.10564,  0.83489, 0,        0,        0,        0.81957, 0,      0,      0
               },
       /*YB */ { 4.97688, -1.95941, 0,        0,        0,        0,        0,      0,      0
               },
       /*Hm */ {-4.725,     0,        0,        0,        0,        0,        1.021, 4.105,
           0.759 },
```

```
    /*RM */ {-5.50000,  0,        0,        0,        0,        0,        0,      0,      0
            }},
  //ATD
  {/*BW */ {-5.50000,  0,        0,        0,        0.17670, 0,        0,      0,      0
          },
    /*SM */ { 0,        0.53854,-2.26847, 0,        0,        0,        0,      0,      0
            },
    /*WA */ {-5.50000,  0,        0,        0,        0,        0,        0,      0,      0
            },
    /*IW */ {-7.10564,  0.83489, 0,        0,        0,        0.81957, 0,      0,      0
            },
    /*YB */ { 0,        -0.89252, 0,        0,        0,        0,        0,      0,      0
            },
    /*HM */ {-4.725,    0,        0,        0,        0,        0,        1.021, 4.105,
        0.759 },
    /*RM */ {-5.50000,  0,        0,        0,        0,        0,        0,      0,      0
            }},
  //ATM
  {/*BW */ {-2.89174, -0.25728, 0,        0,        0.17670, 0,        0,      0,      0
          },
    /*SM */ {-2.94369,  0.53854, 0,        0,        0,        0,        0,      0,      0
            },
    /*WA */ {-3.56466, -0.49267, 0,        1.73752, 0,        0,        0,      0,      0
            },
    /*IW */ {-7.10564,  0.83489, 0,        0,        0,        0.81957, 0,      0,      0
            },
    /*YB */ { 4.97688, -1.61851, 0,        0,        0,        0,        0,      0,      0
            },
    /*HM */ {-4.725,    0,        0,        0,        0,        0,        1.021, 4.105,
        0.759 },
    /*RM */ { 0,        -0.80890, 0,        0,        0,        0,        0,      0,      0
            }}
};

int ht = hdata->hdata[py][px];

double xi[n_params];
xi[0] = 1;
xi[1] = log(st900+1);
xi[2] = xi[1]*fmin(0.5,pct_hm);
xi[3] = (pct_wa > 0) ? 1 : 0 ;
xi[4] = (pct_bw > 0) ? 1 : 0 ;
xi[5] = (pct_iw > 0) ? 1 : 0 ;
xi[6] = (st900>80) ? 1 : 0;
xi[7] = (pct_hm>=0.6) ? 1 : 0;
xi[8] = (pct_hm< 0.6)*log(fmax(1,100*pct_hm));

// Check preconditions
for (int i=0; i<n_params; i++){
  assert(isfinite(xi[i]));
}

for (int i=0; i<n_species; i++){
  double X=0;
  for (int j=0; j<n_params; j++){ X+= xi[j]*coefs[ht][i][j]; }
```

```
    pi[i]=1/(1+exp(-X));
    sum_pi += pi[i];
  }

  if (use_test_regen_props && year < 20){
    if (ht==ht_AOCa){
      double balance[] = { 20, 60, 15, 0,  5,  0,  0 };
      for (int i=0; i<n_species; i++){ pi[i] = balance[i]; }
    } else if (ht==ht_ATD){
      double balance[] = {  5, 80,  3, 0,  5,  2,  5 };
      for (int i=0; i<n_species; i++){ pi[i] = balance[i]; }
    } else {
      double balance[] = {  5, 40,  5, 0, 30,  5, 15 };
      for (int i=0; i<n_species; i++){ pi[i] = balance[i]; }
    }
    sum_pi = 100;
  }

  if (forbid_hm_regen){
    sum_pi -= pi[5];
    pi[5] = 0;
  }

  // Normalize evertyhing so that it sums to 1.
  for (int i=0; i<n_species; i++){ pip[i]=pi[i]/sum_pi; }

  // Compute the cumulative probabilities for each species:
  sp_prop.cum_pi[0]=pip[0];
  for (int i=1; i<n_species; i++){
    sp_prop.cum_pi[i]=sp_prop.cum_pi[i-1]+pip[i];
  }

  // Save the coordinates for this patch, so that we can re-use
  //  these cumulative probabilities for the next tree on our patch.
  sp_prop.px=px;
  sp_prop.py=py;
}

int sp_final=-1;
double rn = urand01();
for (int i=0; i<n_species; i++){
  if (rn < sp_prop.cum_pi[i] ){ sp_final=i; break; }
}

switch(sp_final){
case 0: rc = sp_basswood;      break;
case 1: rc = sp_sugar_maple;   break;
case 2: rc = sp_white_ash;     break;
case 3: rc = sp_ironwood;      break;
case 4: rc = sp_yellow_birch;  break;
case 5: rc = sp_hemlock;       break;
case 6: rc = sp_red_maple;     break;
default:
  printf("ERROR: non-existent species "
         "in RandomSpecies() %s:%i\n",
         __FILE__, __LINE__);
  exit(1);
  break;
}
```

```
  return rc;
}

void Stand::grow(){
  for(TreeList::iterator iter=living->begin();
      iter!=living->end(); iter++){
    assert( (*iter)->GetStatus() == st_live );
    (*iter)->grow( stocking_900(*iter),
                   (*iter)->GetDBH() / average_diam_900(*iter),
                   get_stage(*iter) );
  }
}

void Stand::die(){
  if (storm_clock > 0){
    storm_clock--;
  } else {
    TreeList::iterator it = living->begin();
    while ( it != living->end() ){
      double pct_hm = PercentBA( sp_hemlock,
                                 x_index((*it)->GetXCoord()),
                                 y_index((*it)->GetYCoord()) );
      double pdie = (*it)->pdie( stocking_900(*it),
                                 (*it)->GetDBH() / average_diam_900(*it),
                                 get_stage(*it), pct_hm);
      if( urand01() < pdie ){
        dead->push_front( *it );
        (*it)->SetStatus( st_dead );
        it = living->erase(it);
      } else {
        it++;
      }
    }
  }
}

void Stand::add_harvester(char* fname){
  Harvest *h = new Harvest(this, fname);
  harvesters->push_front(h);
}

void Stand::add_storm_model(char* fname){
  if (my_storm_model != NULL){
    printf("ERROR: Attempt to add more than one storm model.\n");
    exit(1);
  }

  my_storm_model = new StormModel(fname);
}

void Stand::harvest(){
  // Run each harvester, having all of them mark trees toasted.
  for (slist<Harvest*>::iterator it = harvesters->begin();
       it != harvesters->end(); it++){
    (*it)->do_harvest();
  }

  // Now that all the harvesters have run, remove the cut trees from
  // the living list.
```

```
  TreeList::iterator it = living->begin();
  while (it != living->end() ){
    status_t this_st = (*it)->GetStatus();
    if ( this_st == st_cut ||
         this_st == st_cut_leave ||
         this_st == st_cut_girdle ) {
      update_harv_ba( *it );
      if ( this_st == st_cut ) { harvested->push_front( *it ); }
      if ( this_st == st_cut_leave ) {
        harvested_left -> push_front( *it );
        cwd_log[0] -> push_front( *it );
      }
      if ( this_st == st_cut_girdle ) {
        harvested_left -> push_front( *it );
        cwd_snag[0] -> push_front( *it );
      }

      it = living->erase(it);
    } else {
      it++;
    }
  }
}

/// Find the border trees for a sapling
/** Once the border trees are identified, calculate the area of the gap.
    Also, as the border trees are being scanned, add up the TCA of
    spalings within the 80m^2 competition plot.
 */
void Stand::find_gaps(){
  double distance[8];
  double angles[8];

  // Walk the treelist
  for (TreeList::iterator cur_tree=living->begin();
       cur_tree != living->end(); cur_tree++ ){

    // For every tree shorter than 17m:
    if ((*cur_tree)->GetTotalHeight()<17){

      // Determine if we're on a sapling/pole patch, and therefore
      // want to use 'strict gaps' which require an angle to gap
      // border trees of at least 20 degrees.
      bool strict_gaps = false;

      stage_t this_stage = get_stage( x_index( (*cur_tree)->GetXCoord() ),
                                      y_index( (*cur_tree)->GetYCoord() ) );

      if (this_stage == stage_sapling || this_stage == stage_pole ) {
        strict_gaps = true;
      }

      // Check to see if last year's set of gap neighbors are still
      // alive and still meet the criteria to be gap neighbors.
      bool cache_is_good = true;
      for (int i=0; i<8; i++){
        TreePtr neighbor=(*cur_tree)->get_gap_neighbor(i);
        if (neighbor == NULL || neighbor->GetStatus() != st_live) {
          cache_is_good = false; break;
        } else {
          double TH_subj = (*cur_tree)->GetTotalHeight();
```

```
    double WH_comp = neighbor->GetWidestHeight();
    double Rh900_comp =
      neighbor->GetTotalHeight()/average_height_900(neighbor);
    double this_distance = tree_distance_xy( *cur_tree, neighbor)-
      calculate_cr_toward_competitor_xy( neighbor, *cur_tree );
    double gap_depth = neighbor->GetWidestHeight() - (*cur_tree)->GetTotalHeight();
    double tree_angle = 180/M_PI * atan2(gap_depth, fmax(0,this_distance) );

    if ( !(TH_subj <= WH_comp && Rh900_comp >= 0.66 && WH_comp>=12  &&
           (!strict_gaps || tree_angle > 20)) ) {
      cache_is_good = false; break;
    }
  }
}

// If the cached neighbors are still valid, just re-use them and
// keep the list of competitor saplings.
if (cache_is_good){
  for( int i=0; i<8; i++){
    TreePtr neighbor=(*cur_tree)->get_gap_neighbor(i);

    distance[i]=tree_distance_xy( *cur_tree, neighbor) -
      calculate_cr_toward_competitor_xy(neighbor, *cur_tree );
    angles[i]=tree_angle( *cur_tree, neighbor );
  }
} else {
  // Otherwise, search for a new set of neighbors and competitor saplings.
  (*cur_tree)->clear_competitor_saplings();

  for (int i=0; i<8; i++){
    (*cur_tree)->set_gap_neighbor(i, TreePtr() );
    distance[i]= 30; // because it's greater than max search
    angles[i] = ((float)i)/8*(2*M_PI);
  }

  int found=0; // Bits 0-8 are flag values for the eight neighbors
  //A value of 0xFF indicates that they've all been found.

  the_grid->set_search_center( (*cur_tree)->GetXCoord(),
                               (*cur_tree)->GetYCoord() );

  // Note: the TreeGrid limits the search to a 50x50 meter area
  // surrounding the current search center
  TreePtr neighbor;
  while ( (neighbor=the_grid->get_next_tree()) != NULL &&
          found != 0xFF ){

    // Skip over ourselves
    if (neighbor == *cur_tree) { continue; }

    // Determine which quadrant the neighbor is in:
    double angle=tree_angle( *cur_tree, neighbor);
    int ix=0;
    if      (  22.5 < angle && angle <= 67.5  ){ ix=1; }
    else if(  67.5 < angle && angle <= 112.5 ){ ix=2; }
    else if( 112.5 < angle && angle <= 157.5 ){ ix=3; }
    else if( 157.5 < angle && angle <= 202.5 ){ ix=4; }
    else if( 202.5 < angle && angle <= 247.5 ){ ix=5; }
    else if( 247.5 < angle && angle <= 292.5 ){ ix=6; }
    else if( 292.5 < angle && angle <= 337.5 ){ ix=7; }
```

```
    // Border trees:
    //   If this tree is closer than current border, take this tree
    //   If this tree is the same distance as current border,
    //      but this tree is closer to the midline of the search area,
    //      take this tree.
    double this_distance = tree_distance_xy( *cur_tree, neighbor)-
      calculate_cr_toward_competitor_xy( neighbor, *cur_tree );
    double gap_depth = neighbor->GetWidestHeight() - (*cur_tree)->GetTotalHeight();
    double tree_angle = 180/M_PI * atan2(gap_depth, fmax(0,this_distance) );
    double TH_subj = (*cur_tree)->GetTotalHeight();
    double WH_comp = neighbor->GetWidestHeight();
    double Rh900_comp =
      neighbor->GetTotalHeight()/average_height_900(neighbor);

    if ( (TH_subj <= WH_comp && Rh900_comp >= 0.66 && WH_comp>=12 &&
          (!strict_gaps || tree_angle > 20) ) &&
         (this_distance < distance[ix]) ) {
      found |= (1<<ix);
      distance[ix]=this_distance;
      angles[ix]=angle;
      (*cur_tree)->set_gap_neighbor(ix, neighbor);
    }

    // Competitor saplings:
    // One of our crown-based competiton metrics is on a fixed
    // r=5.05m plot, and the other is on a 2.5*mcr
    // variable-radius plot.  However, we only look for new
    // competitor saplings when one of the former gap borders
    // dies.  Therefore, when we're setting our plot radius
    // based on MCR, we'll include slightly more than
    // necessary to accomodate subject-tree crown growth.
    double mcr=(*cur_tree)->GetMCR();
    double xy_dist = tree_distance_xy( *cur_tree, neighbor);
    if( xy_dist <= max(5.05, 3.5*mcr) &&
        neighbor->GetTotalHeight() <= 12){
      (*cur_tree)->add_competitor_sapling( neighbor, xy_dist );
    }
  } //Foreach neighbor
}

// Neighbors either found or pulled from cache at this point.

double area=0;
for (int i=0; i<8; i++){
  double a0 = angles[i];
  double a1 = angles[(i+1)%8];

  double theta = a1-a0;
  theta = theta - 90*floor(theta/90); // wrap to 0<=theta<90

  // If another tree covers us completely, we're overtopped and
  // have zero gap area.
  if (distance[i]<=0){
    area = 0;
    break;
  } else {
    // Area of each triangle is A = 1/2 A*B*sin(theta)
    double ai = 0.5*distance[i]*distance[(i+1)%8] * sin( M_PI/180*theta );
    area += ai;
  }
}
```

```
      (*cur_tree)->set_gap_area(area);
    } else {
      // For trees taller than 17m, clear the comp sap list
      (*cur_tree)->clear_competitor_saplings();
    }
  }
}

void Stand::transform(TreePtr subj, TreePtr comp, coords_t& rc){
  transform( subj->GetXCoord(), subj->GetYCoord(), comp, rc );
}

void Stand::transform(double s_x, double s_y, TreePtr comp, coords_t& rc){
  //Transform the subject and competitor tree into a coordinate system
  //where the subject is at the center of the plot.  If the competitor
  //is outside the plot boundary, warp it back in appropriately.
  double dx = (10*xmax)/2 - s_x;
  double dy = (10*ymax)/2 - s_y;

  rc.subj.x = (10*xmax)/2; // raw_sx +dx = raw_sx +center -raw_sx = center
  rc.subj.y = (10*ymax)/2; // same here.

  double x_trans = comp->GetXCoord()+dx;
  double y_trans = comp->GetYCoord()+dy;

  rc.comp.x = x_trans - 10*xmax*floor( x_trans / (10*xmax) );
  rc.comp.y = y_trans - 10*ymax*floor( y_trans / (10*ymax) );
}

double Stand::tree_distance_xy( TreePtr subj, TreePtr comp ){
  double rc;
  // we are one plot-width away from ourself (or rather, the us in the mirror).
  if( subj == comp){
    rc = 5*xmax+5*ymax;
  } else {
    coords_t coords;
    transform(subj,comp,coords);

    double dx = coords.comp.x - coords.subj.x;
    double dy = coords.comp.y - coords.subj.y;

    rc  = hypot(dx, dy);
  }

  return rc;
}

double Stand::tree_distance_rz( TreePtr t1, TreePtr t2){
  double dr = tree_distance_xy(t1,t2);
  double dz = t1->GetWidestHeight() - t2->GetWidestHeight();
  double rc = hypot(dr,dz);
  return rc;
}

double Stand::tree_angle ( TreePtr subj, TreePtr comp) {
  coords_t coords;

  transform(subj,comp,coords);

  double dx = coords.comp.x - coords.subj.x;
  double dy = coords.comp.y - coords.subj.y;
```

```
  double angle = 180/M_PI*atan2( dx,dy ) ;
  if( angle<0) { angle = 360+angle; }
  return angle;
}

double Stand::calculate_cr_toward_competitor_xy( TreePtr subj,
                                                 TreePtr comp ){
  coords_t coords;

  transform(subj,comp,coords);

  double dx = coords.comp.x - coords.subj.x;
  double dy = coords.comp.y - coords.subj.y;

  double a,b;

  if(dy>=0) { a = subj->get_cr(0); } //North CR
  else      { a = subj->get_cr(2); } //South CR

  if(dx>=0) { b = subj->get_cr(1); } //East CR
  else      { b = subj->get_cr(3); } //West CR

  double theta = tree_angle(subj,comp) * M_PI/180 ;

  double xr = a * cos (theta);
  double yr = b * sin (theta);

  double r = hypot(xr,yr);

  return r;
}

double Stand::calculate_cr_toward_competitor_rz( TreePtr subj,
                                                 TreePtr comp ){
  double dr = tree_distance_xy(subj,comp);
  double dz = comp->GetWidestHeight()-subj->GetWidestHeight();
  double theta = atan2(dz,dr);

  double a = calculate_cr_toward_competitor_xy(subj,comp);
  double b;

  if(dz>=0){ b=subj->GetTotalHeight()-subj->GetWidestHeight();}
  else     { b=subj->GetBaseHeight()-subj->GetWidestHeight(); }

  double xr = a*cos(theta);
  double yr = b*sin(theta);
  double r = hypot(xr,yr);
  return r;
}

///Determine (for each crown lobe) of a tree if it is in or facing a gap
/** If the maximum percent radial overlap on a given lobe of crown is
    50% or greater, then that lobe is not in a gap
    If the minimum distance from a given lobe of crown to the next
    nearest competitor crown lobe is less than 2.5 m, then that lobe is not
    facing a gap.
    If the minimum distance from a given lobe of crown to the next
    nearest competitor crown lobe is less than 10cm, then those lobes
    are touching.
*/
void Stand::calculate_crown_status(){
```

```
TreeList::iterator subj;

for (subj=living->begin(); subj!=living->end(); subj++){
  // First, compute the relative height:
  (*subj)->SetRH( average_height_900( *subj ) );

  //Next, work out the crown competitors:
  TreePtr comp;

  double margin_distance[4];
  double crown_distance[4];
  double pct_overlap[4];

  double cthresh;
  if ( (*subj)->GetSpecies() == sp_hemlock ){
    cthresh = 0.85*(*subj)->GetTotalHeight();
  } else {
    cthresh = (*subj)->GetWidestHeight();
  }
  // Check to see that the stored crown neighbors are still alive
  // and still tall enough to be crown neighbors
  bool cache_is_good = true;
  for (int i=0; i<4; i++){
    comp = (*subj)->get_xy_neighbor(i);
    if ( comp == NULL || comp->GetStatus() != st_live ||
         comp->GetTotalHeight() < cthresh ) {
      cache_is_good = false; break;
    }

    comp = (*subj)->get_rz_neighbor(i);
    if ( comp == NULL || comp->GetStatus() != st_live ||
         comp->GetTotalHeight() < cthresh ) {
      cache_is_good = false; break;
    }

    comp = (*subj)->get_eca_neighbor(i);
    if ( comp == NULL || comp->GetStatus() != st_live ||
         comp->GetWidestHeight() < (*subj)->GetWidestHeight() ){
      cache_is_good = false; break;
    }
  }

  if (cache_is_good) {
    for (int i=0; i<4; i++){
      // Determine crown margin distance based on XY neighbor
      comp = (*subj)->get_xy_neighbor(i);
      margin_distance[i] =
        tree_distance_xy(*subj, comp) -
        calculate_cr_toward_competitor_xy(*subj, comp) -
        calculate_cr_toward_competitor_xy(comp, *subj);

      //Determine distance to nearest crown based on RZ neighbor
      comp = (*subj)->get_rz_neighbor(i);
      crown_distance[i] =
        tree_distance_rz(*subj, comp) -
        calculate_cr_toward_competitor_rz(*subj, comp) -
        calculate_cr_toward_competitor_rz(comp, *subj);

      // Determine crown overlap based on ECA neighbor
```

```
    comp = (*subj)->get_eca_neighbor(i);
    double subj_cr = calculate_cr_toward_competitor_xy(*subj, comp);
    double overlap =
      tree_distance_xy(*subj, comp) -
      subj_cr -
      calculate_cr_toward_competitor_xy(comp, *subj);
    pct_overlap[i] = fmin( 1, fmax(0, -overlap)/subj_cr);
  }
} else {
  for (int i=0; i<4; i++){
    double d = hypot(10.0*xmax, 10.0*ymax); // one plot away
    margin_distance[i]=d;
    crown_distance[i]=d;
    pct_overlap[i]=0;
    (*subj)->set_xy_neighbor(i, TreePtr() );
    (*subj)->set_rz_neighbor(i, TreePtr() );
    (*subj)->set_eca_neighbor(i, TreePtr() );
  }

  int found=0; // 12 bits of flag values.
  // 0xFFF indicates everything is found
  // Bit 0-3: xy neighbors
  // Bit 4-7: rz neighbors
  // Bit 8-12: eca neighbors

  the_grid->set_search_center( (*subj)->GetXCoord(),
                               (*subj)->GetYCoord() );

  while( (comp=the_grid->get_next_tree() ) != NULL &&
         found != 0xFFF ) {

    // Don't consider competing with ourself.
    if (*subj == comp) { continue; }

    // Determine which quadrant the competitor is in, set an ndex value
    double angle=tree_angle(*subj,comp);
    int ix=0;
    if      ( 45  <= angle && angle < 135 ) { ix = 1; }
    else if ( 135 <= angle && angle < 225 ) { ix = 2; }
    else if ( 225 <= angle && angle < 315 ) { ix = 3; }

    if ( comp->GetTotalHeight() >= cthresh ){
      // It's a potential XY or RZ neibhor
      double this_margin_distance =
        tree_distance_xy(*subj,comp) -
        calculate_cr_toward_competitor_xy(*subj,comp) -
        calculate_cr_toward_competitor_xy(comp,*subj);

      if ( this_margin_distance < margin_distance[ix] ) {
        found |= (1<<ix);
        (*subj)->set_xy_neighbor(ix, comp);
        margin_distance[ix] = this_margin_distance;
      }

      double this_crown_distance =
        tree_distance_rz(*subj,comp) -
        calculate_cr_toward_competitor_rz(*subj,comp) -
        calculate_cr_toward_competitor_rz(comp,*subj);
```

```
          if (this_crown_distance < crown_distance[ix] ){
            found |= (16<<ix);
            (*subj)->set_rz_neighbor(ix, comp);
            crown_distance[ix] = this_crown_distance;
          }
        }

        if ( comp->GetWidestHeight() >= (*subj)->GetWidestHeight() ){
          // It's a potential ECA neighbor
          double this_subj_cr = calculate_cr_toward_competitor_xy(*subj,comp);

          double this_overlap =
            tree_distance_xy(*subj, comp) -
            this_subj_cr -
            calculate_cr_toward_competitor_xy(comp, *subj);

          double this_pct_overlap =
            fmin(1, fmax(0,-this_overlap)/this_subj_cr );

          if( pct_overlap[ix] < this_pct_overlap ){
            found |= (256<<ix);
            (*subj)->set_eca_neighbor(ix, comp);
              pct_overlap[ix] = this_pct_overlap;
          }
        }
      } // For each competitor
    }

    for (int i=0; i<4; i++){
      double touching_threshold;
      if( (*subj)->GetSpecies() == sp_hemlock ){
        touching_threshold = -0.1* (*subj)->get_cr(i);
      } else {
        touching_threshold = 0;
      }

      (*subj)->set_facing_gap(i, crown_distance[i] > 1 );
      (*subj)->set_touching(i, crown_distance[i] < touching_threshold );

      (*subj)->set_shaded(i, margin_distance[i]/(*subj)->get_cr(i) <= -0.5 );

      double ecr = (*subj)->get_cr(i) * (1-pct_overlap[i]);
      (*subj)->set_ecr(i,  ecr );
    }
  } // For each tree
}

bool Stand::wind(){
  bool rc = false;

  // Check for a storm this year, queue it if necessary
  double tgt = (my_storm_model != NULL) ?
    my_storm_model->get_storm_specification(year) :
    StormModel::storm_eca_removal();
  if (tgt>0) { storm_queue->push_back(tgt); }

  // Check the queue, do a storm if necessary
  double target_eca_removal = -1;
  if (storm_clock > 0 || storm_queue->empty() ) {
    target_eca_removal = 0;
  } else {
```

```
  target_eca_removal = storm_queue->front();
  storm_queue->pop_front();
}

if (target_eca_removal >= disturbance_threshold){
  rc = true;
  TreeList::iterator it;

  // Convenience function for checking set membership
  struct {
    bool operator()(TreeSet *s, long it) {
      return s->find(it) != s->end();
    }
  } in_set ;

  // Compute the total ECA of the stand
  double total_eca = 0;
  for (it = living->begin(); it != living->end(); it++){
    total_eca += (*it)->get_eca();
  }

  double si = StormModel::severity_index(target_eca_removal);

  double removed_eca=0, min_error=1000, error;

  TreeSet *best_blown_down = new TreeSet();
  TreeSet *tmp_blown_down = new TreeSet();

  int iterations = 0;

  printf("Storm Simulation:\n");
  do {
    // Do a trial storm
    double tmp_removed_eca = StormModel::storm(si, living, tmp_blown_down);

    // Compute the %removal, and %erorr
    double trial_eca_removal = 100*tmp_removed_eca / total_eca ;
    error = (trial_eca_removal-target_eca_removal) / target_eca_removal;

    // Check to see if this was our best storm yet.
    if ( fabs(error) < fabs(min_error) ){
      best_blown_down->clear();
      swap( best_blown_down, tmp_blown_down );

      min_error = error;
      removed_eca = trial_eca_removal;
    }

    printf("  iter=%i, tgt=%f, trial=%f, si=%f\n",
           iterations, target_eca_removal, trial_eca_removal, si);

    // If we were 5% high, decrease the si by 5%
    // If we were 5% low (neg. error), this will increase si by 5%
    si = fmin(1,fmax(0, si - si*error));
    iterations++;
  } while ( fabs(error) >= 0.05 && iterations < 100 );


  printf("  Post-Iteration step, tgt=%f, removed=%f\n",
         target_eca_removal, removed_eca );

  // Now, if we removed too much ECA, through the treelist
  //  Searching for those trees that blew down.  For each windthrown tree,
```

```
    //  with p=0.05, remove it from the windthrown list.  Keep doing this until
    //  we're no longer over the target.
    it = living->begin();
    while ( removed_eca > target_eca_removal ) {
      if ( in_set( best_blown_down, (*it)->GetTreeno() ) &&
          urand01() <= 0.05 ){
        best_blown_down->erase( (*it)->GetTreeno() );
        removed_eca -= 100 * (*it)->get_eca() / total_eca;
      }

      it++;
      if (it == living->end()) {
        it = living->begin();
      }
    }

    // Now, we've arrived at our final removal:
    printf("  FINAL, tgt=%f, removed=%f\n",
           target_eca_removal, removed_eca );
    tdb->storm_insert(year, lround(target_eca_removal), removed_eca );

    // Find those trees which blew down in the best storm,
    //  And move them off of the living list.
    it = living->begin();
    while ( it != living->end() ){
      if ( in_set( best_blown_down, (*it)->GetTreeno()) ) {
        blown_down->push_front( *it );
        (*it)->SetStatus( st_dead );

        it = living->erase(it);
      } else {
        it++;
      }
    }

    delete tmp_blown_down;
    delete best_blown_down;

    storm_clock = 10;
  }

  return rc;
}

void Stand::decay(){
  // There are separate lists for trees in each decay class

  // Snag Decay
  cwd_transition( cwd_snag[3], cwd_snag[4], 0.861, 0.861, st_snag_5 );
  cwd_transition( cwd_snag[2], cwd_snag[3], 0.917, 0.917, st_snag_4 );
  cwd_transition( cwd_snag[1], cwd_snag[2], 0.965, 0.965, st_snag_3 );
  cwd_transition( cwd_snag[0], cwd_snag[1], 0.991, 0.991, st_snag_2 );

  // Log Decay
  cwd_transition( cwd_log[4], cwd_out,    0.200000, 0.090909, st_out   );
  cwd_transition( cwd_log[3], cwd_log[4], 0.125000, 0.040816, st_log_5 );
  cwd_transition( cwd_log[2], cwd_log[3], 0.333333, 0.074074, st_log_4 );
  cwd_transition( cwd_log[1], cwd_log[2], 0.076923, 0.071429, st_log_3 );
  cwd_transition( cwd_log[0], cwd_log[1], 0.2,      1.0,      st_log_2 );

  // Next, snag fall
```

```
double coefs[] = { 5.691, -3.777, 0.531, 0.157 };
double xi[4];
for( int i=0; i<5; i++){
  TreeList::iterator it = cwd_snag[i]->begin();
  while (it != cwd_snag[i]->end() ){
    int px = (int)floor( (*it)->GetXCoord()/10 );
    int py = (int)floor( (*it)->GetYCoord()/10 );
    assert( 0<= px && px <= xmax );
    assert( 0<= py && py <= ymax );

    subplot_t* cur = &(subplot_900[px][py]);

    xi[0] = 1;
    xi[1] = log( (*it)->GetDBH() );
    xi[2] = pow(xi[1],2);
    xi[3] = cur->harv_ba * (10000)/ 900;

    double X=0;
    for( int j=0; j<4; j++ ){ X += xi[j]*coefs[j]; }

    switch( (*it)->GetSpecies() ){
    case sp_yellow_birch: X += -0.380; break;
    case sp_red_maple:    X +=  0.348; break;
    case sp_white_ash:    X +=  0.092; break;
    case sp_basswood:     X +=  1.163; break;
    case sp_ironwood:     X +=  0.837; break;
    case sp_hemlock:      X += -0.293; break;
    default: break;
    }

    switch( i ){
    case 1: X += 0.177; break;
    case 2: X += 0.542; break;
    case 3: X += 0.702; break;
    case 4: X += 0.528; break;
    }

    double p5 = exp(X)/(1+exp(X));
    double p1 = 1-pow(1-p5,0.2);

    if( urand01() < p1 ){
      cwd_log[1]->push_front( *it );
      (*it)->SetStatus( st_log_2 );
      it = cwd_snag[i]->erase(it);
    } else {
      it++;
    }
  }
}

// The mortality routine just marks trees as 'dead', so we need to
// go through dead tree list and move trees into a decay class.
for( TreeList::iterator it = dead->begin();
     it != dead->end(); it++){
  // For all freshly dead trees, determine if they are a snag or a log
  // Push them onto the appropriate cwd list, and set their status
  if( (*it)->GetStatus() == st_dead ){
    double coefs[5] = { -0.805, -0.016, -0.026, 3.389, -0.084 };
```

```
      double xi[5];

      // Determine harvest status by looking at %mortality on the 900m2
      int px = (int)floor( (*it)->GetXCoord()/10 );
      int py = (int)floor( (*it)->GetYCoord()/10 );
      assert( 0<= px && px <= xmax );
      assert( 0<= py && py <= ymax );

      subplot_t* cur = &(subplot_900[px][py]);

      xi[0] = 1;
      xi[1] = (*it)->GetDBH();
      xi[2] = M_PI*pow(xi[1]/2,2);
      xi[3] = (cur->harv_ba>0) ? 0 : 1 ;
      xi[4] = xi[1]*xi[3];

      double X=0;
      for( int i=0; i<5; i++){ X += xi[i]*coefs[i]; }

      switch( (*it)->GetSpecies() ){
      case sp_yellow_birch: X += -0.454; break;
      case sp_red_maple:    X +=  0.336; break;
      case sp_white_ash:    X +=  0.057; break;
      case sp_basswood:     X +=  1.149; break;
      case sp_ironwood:     X +=  1.056; break;
      case sp_hemlock:      X +=  1.151; break;
      default: break;
      }

      double p5 = exp(X)/(1+exp(X));
      double p1 = 1-pow(1-p5,0.2);

      if( urand01() < p1 ){
        cwd_log[0]->push_front( *it );
        (*it)->SetStatus( st_log_1 );
      } else {
        cwd_snag[0]->push_front( *it);
        (*it)->SetStatus( st_snag_1 );
      }
    }
  }

  // Move all the storm-killed trees into log_1
  for( TreeList::iterator it = blown_down->begin();
       it != blown_down->end(); it++){
    cwd_log[0]->push_front(*it);
    (*it)->SetStatus( st_log_1 );
  }

  // Flip the harvested trees into either log_1 or snag_1.
  for( TreeList::iterator it = harvested->begin();
       it != harvested->end(); it++){
    if( (*it)->GetStatus() == st_log_1  ){
      cwd_log[0]->push_front( *it );
    }
    if( (*it)->GetStatus() == st_snag_1 ){
      cwd_snag[0]->push_front( *it );
    }
  }
}
```

```
void Stand::cwd_transition( TreeList* from, TreeList* to,
                            double sm_prob, double hm_prob,
                            status_t status ){

  TreeList::iterator it = from->begin();
  while ( it != from->end() ){
    double prob = ((*it)->GetSpecies()==sp_hemlock) ? hm_prob : sm_prob ;
    if( urand01() < prob ){
      (*it)->SetStatus(status);
      to->push_front( *it );
      it = from->erase(it);
    } else {
      it++;
    }
  }
}
```

C.4 Harvest.cp

```
// -*- C++ -*-
extern "C" {
#include <lua.h>
#include <lauxlib.h>
#include <lualib.h>
}

#include <assert.h>
#include <algo.h>
#include "util.h"
#include "CanopyGlobals.h"
#include "Harvest.h"
#include "Tree.h"
#include "Stand.h"

Harvest::Harvest( Stand* s, char* fname ){
  my_stand = s;

  //Set the plot_size lua global
  l_setglobal("plot_size", 100.0*s->xmax*s->ymax );

  l_setglobal("xmax", 10.0 * s->xmax );
  l_setglobal("ymax", 10.0 * s->ymax );

  // Tell the harvester about our status codes.
  l_setglobal("st_live",       (double)st_live       );
  l_setglobal("st_dead",       (double)st_dead       );
  l_setglobal("st_cut",        (double)st_cut        );
  l_setglobal("st_cut_leave",  (double)st_cut_leave  );
  l_setglobal("st_cut_girdle", (double)st_cut_girdle);
  l_setglobal("st_log_1",      (double)st_log_1      );
  l_setglobal("st_log_2",      (double)st_log_2      );
  l_setglobal("st_log_3",      (double)st_log_3      );
  l_setglobal("st_log_4",      (double)st_log_4      );
  l_setglobal("st_log_5",      (double)st_log_5      );
  l_setglobal("st_snag_1",     (double)st_snag_1     );
  l_setglobal("st_snag_2",     (double)st_snag_2     );
  l_setglobal("st_snag_3",     (double)st_snag_3     );
```

```
  l_setglobal("st_snag_4",      (double)st_snag_4     );
  l_setglobal("st_snag_5",      (double)st_snag_5     );

  // Tell the harvester about our species codes.
  l_setglobal("sp_basswood",          (double)  2 );
  l_setglobal("sp_sugar_maple",       (double)  4 );
  l_setglobal("sp_white_ash",         (double) 13 );
  l_setglobal("sp_ironwood",          (double) 24 );
  l_setglobal("sp_hemlock",           (double)  5 );
  l_setglobal("sp_yellow_birch",      (double)  6 );
  l_setglobal("sp_red_maple",         (double)  9 );
  l_setglobal("sp_white_pine",        (double)  7 );
  l_setglobal("sp_paper_birch",       (double)  1 );
  l_setglobal("sp_northern_red_oak", (double) 10 );
  l_setglobal("sp_green_ash",         (double) 14 );
  l_setglobal("sp_american_elm",      (double) 21 );
  l_setglobal("sp_black_cherry",      (double) 22 );
  l_setglobal("sp_mountain_maple",    (double) 39 );
  l_setglobal("sp_balsam_fir",        (double) 41 );
  l_setglobal("sp_white_spruce",      (double) 43 );
  l_setglobal("sp_white_cedar",       (double) 46 );

  harvestable_trees = new vector<TreePtr>();
  cwd_trees         = new vector<TreePtr>();

  int error = luaL_loadfile(Lua, fname) || lua_pcall(Lua, 0, 0, 0);
  if (error) {
    fprintf(stderr, "%s", lua_tostring(Lua, -1));
    lua_pop(Lua, 1);
    exit(1);
  }

  // Code to verify functions should probably all go here.
  // Need to check and make sure the lua file that was just loaded
  // actually includes all of the functions we need.

}

/// Convert a TreePtr to a Lua table sitting on TOS.
void Harvest::tree_to_lua(TreePtr t){
  double th = t->GetTotalHeight();
  double rh = th/ my_stand->average_height_900(t);
  double st900 = my_stand->stocking_900(t);

  // Crown radii
  double n_cr = t->get_cr(0);
  double e_cr = t->get_cr(1);
  double s_cr = t->get_cr(2);
  double w_cr = t->get_cr(3);

  // exposed crown radii
  double n_ecr = t->get_ecr(0);
  double e_ecr = t->get_ecr(1);
  double s_ecr = t->get_ecr(2);
  double w_ecr = t->get_ecr(3);

  // Crown summaries.
  double tca = M_PI/4*( n_cr*e_cr + e_cr*s_cr + s_cr*w_cr + w_cr*n_cr);
  double eca = M_PI/4*( n_ecr*e_ecr + e_ecr*s_ecr + s_ecr*w_ecr + w_ecr*n_ecr);
```

```
  double mcr = (n_cr + e_cr + s_cr + w_cr)/4;

  lua_newtable(Lua);
  l_setfield("treeno",  (double) t->GetTreeno()  );
  l_setfield("x",                t->GetXCoord()  );
  l_setfield("y",                t->GetYCoord() );
  l_setfield("species", (double) t->GetSpecies() );
  l_setfield("dbh",              t->GetDBH() );
  l_setfield("total_height",     th);
  l_setfield("rh_900",           rh);
  l_setfield("tca",              tca);
  l_setfield("eca",              eca);
  l_setfield("mcr",              mcr);
  l_setfield("stock",            st900);
}

void Harvest::tree_vector_to_lua( vector<TreePtr> *trees){
  int tn=0;
  lua_newtable(Lua);
  for( vector<TreePtr>::iterator it = trees->begin();
       it != trees->end(); it++){
    lua_pushnumber(Lua, (double)tn++);
    tree_to_lua( *it );
    lua_settable( Lua, -3 );
  }
}

void Harvest::status_vector_to_lua( vector<double> *status){
  int ix=0;
  lua_newtable(Lua);
  for( vector<double>::iterator it = status->begin();
       it != status->end(); it++){
    lua_pushnumber(Lua, (double)ix++ );
    lua_pushnumber(Lua, *it );
    lua_settable( Lua, -3 );
  }
}

bool Harvest::is_it_time(int year){
  bool rc;
  lua_getglobal(Lua, "is_it_time");
  lua_pushnumber(Lua, (double)year );
  lua_pcall(Lua, 1, 1, 0);
  assert( lua_isboolean(Lua, -1) );
  rc = lua_toboolean(Lua, -1);
  lua_pop(Lua, 1);
  return rc;
}

bool Harvest::is_it_inside(TreePtr t){
  bool rc;
  lua_getglobal(Lua, "is_it_inside");
  lua_pushnumber(Lua, t->GetXCoord());
  lua_pushnumber(Lua, t->GetYCoord());
  lua_pcall(Lua,2,1,0);
  assert( lua_isboolean(Lua, -1) );
```

```
  rc = lua_toboolean(Lua, -1);
  lua_pop(Lua, 1);
  return rc;
}

/*************************
  rc=0 -> leave it be
  rc=1 -> thwack
  rc=2 -> girdle
  rc=3 -> thwack and leave
*/
int Harvest::cut_this_tree(TreePtr t){
  int rc;
  lua_getglobal(Lua, "cut_this_tree");
  tree_to_lua(t);
  lua_pcall(Lua,1,1,0);

  assert( lua_isnumber(Lua, -1));
  rc = (int)lua_tonumber(Lua, -1);
  lua_pop(Lua, 1);
  assert( 0 <= rc && rc <= 3);
  return rc;
}

void Harvest::prepare(){
  // First, clear out the group centers
  while( ! group_centers.empty() ){
    free( group_centers.top() );
    group_centers.pop();
  }

  // See if there's a cwd_prepare, call it if so.
  lua_getglobal(Lua, "cwd_prepare");
  if( lua_isfunction(Lua, -1) == 1 ) {
    vector<double> cwd_status;

    cwd_trees->clear();

    TreeList **cwd_log, **cwd_snag;
    cwd_log = my_stand->cwd_log;
    cwd_snag = my_stand->cwd_snag;
    double log_status[] =
      {(double)st_log_1, (double)st_log_2, (double)st_log_3,
       (double)st_log_4, (double)st_log_5 };
    double snag_status[] =
      {(double)st_snag_1, (double)st_snag_2, (double)st_snag_3,
       (double)st_snag_4, (double)st_snag_5 };

    for( int i=0; i<5; i++){
      for( TreeList::iterator it = cwd_log[i]->begin();
           it != cwd_log[i]->end(); it++){
        if( is_it_inside( *it ) ){
          cwd_trees->push_back( *it );
          cwd_status.push_back( log_status[i] );
        }
      }
      for( TreeList::iterator it = cwd_snag[i]->begin();
           it != cwd_snag[i]->end(); it++){
        if( is_it_inside( *it ) ){
```

```
            cwd_trees->push_back( *it );
            cwd_status.push_back( snag_status[i] );
          }
        }
      }

      tree_vector_to_lua( cwd_trees );
      status_vector_to_lua( &cwd_status );
      lua_pcall(Lua, 2, 0, 0);
    } else {
      lua_pop(Lua, 1);
    }

    TreeList *living = my_stand->living;

    harvestable_trees->clear();
    //Build the list of harvestable trees.
    for( TreeList::iterator it = living->begin();
         it != living->end(); it++){
      if( is_it_inside( *it ) ){
        harvestable_trees->push_back( *it );
        (*it)->force_save();
      }
    }

    random_shuffle( harvestable_trees->begin(), harvestable_trees->end() );

    //Assemble the args for prepare, and call it.
    lua_getglobal(Lua, "prepare");
    tree_vector_to_lua( harvestable_trees );
    lua_pcall(Lua,1,0,0);
}

void Harvest::perform(){
    for( vector<TreePtr>::iterator it = harvestable_trees->begin();
         it != harvestable_trees->end(); it++){
      int c = cut_this_tree( *it );

      if (c>20) {
        c -= 20;
        my_stand->clean_around_and_delay(*it);
      }

      switch(c){
      case  1: (*it)->SetStatus(st_cut);         break;
      case  2: (*it)->SetStatus(st_cut_girdle); break;
      case  3: (*it)->SetStatus(st_cut_leave);  break;
      }

    }
    harvestable_trees->clear();
}

void Harvest::do_harvest(){
    if( is_it_time( my_stand->year ) ){
      prepare();
      perform();
      l_fetch_groups();
      do_planting();
    }
}
```

```
void Harvest::l_fetch_groups(){
  point_t *pt;

  lua_getglobal(Lua, "gcs_group_centers");
  if( lua_istable(Lua, 1) ){
    lua_pushnil(Lua);
    while (lua_next(Lua, 1) != 0) {
      lua_getfield(Lua, -1, "x");
      lua_getfield(Lua, -2, "y");
      pt = (point_t*)malloc(sizeof(point_t)); CHK_MEM(pt);
      pt->y = lua_tonumber(Lua, -1);
      pt->x = lua_tonumber(Lua, -2);
      group_centers.push(pt);
      lua_pop( Lua, 3 );
    }

    lua_newtable(Lua);
    lua_setglobal(Lua, "gcs_group_centers");
  }

  lua_pop(Lua, 1);
}

void Harvest::do_planting(){

  lua_getglobal(Lua, "trees_to_plant");
  if (lua_istable(Lua,1) ){
    lua_pushnil(Lua);
    while (lua_next(Lua, 1) != 0){
      lua_getfield(Lua, -1, "x");
      lua_getfield(Lua, -2, "y");
      lua_getfield(Lua, -3, "sp");
      lua_getfield(Lua, -4, "dbh");
      double dbh = lua_tonumber(Lua, -1);
      int     sp = lua_tointeger(Lua, -2);
      double   y = lua_tonumber(Lua, -3);
      double   x = lua_tonumber(Lua, -4);
      lua_pop( Lua, 5);

      my_stand->plant_tree( x, y, (species_t) sp, dbh);
    }

    lua_newtable(Lua);
    lua_setglobal(Lua, "trees_to_plant");
  }

  lua_pop(Lua, 1);
}

/** Gets the next group center, or NULL if none exists.
    It's the caller's job to free the point_t returned.
 */
point_t* Harvest::get_group(){
  point_t* rc = NULL;
  if( ! group_centers.empty() ){
    rc = group_centers.top();
    group_centers.pop();
  }
  return rc;
}
```

C.5 TreeDB.cp

```
// -*- C++ -*-
#include <assert.h>
#include <stdlib.h>
#include <math.h>
#include "version.h"
#include "Tree.h"
#include "TreeDB.h"
#include "Stand.h"
#include "util.h"

extern int save_interval;

class sqlite_exception {
public:
  const char* error_message;
  int error_code;
  sqlite_exception(sqlite3* db){
    error_message = sqlite3_errmsg(db);
    error_code = sqlite3_errcode(db);
    printf("sqlite error %i: %s\n", error_code, error_message);
  }
};

void TreeDB::db_prepare(sqlite3_stmt** st, const char* cmd){
  int rc;
  if( (rc=sqlite3_prepare(db,cmd,-1,st,0)) != SQLITE_OK)
    throw sqlite_exception(db);
}

void TreeDB::db_finalize(sqlite3_stmt* st){
  int rc;
  if( (rc=sqlite3_finalize(st))!=SQLITE_OK) throw sqlite_exception(db);
}

void TreeDB::db_step(sqlite3_stmt* st){
  int rc;
  if( (rc=sqlite3_step(st))!=SQLITE_DONE) throw sqlite_exception(db);
  if( (rc=sqlite3_reset(st))!=SQLITE_OK) throw sqlite_exception(db);
}

void TreeDB::db_bind_double(sqlite3_stmt* st, int pos, double d){
  if(sqlite3_bind_double(st, pos, d)!=SQLITE_OK) throw sqlite_exception(db);
}

void TreeDB::db_bind_int(sqlite3_stmt* st, int pos, int d){
  if(sqlite3_bind_int(st, pos, d)!=SQLITE_OK) throw sqlite_exception(db);
}

void TreeDB::db_bind_text(sqlite3_stmt* st, int pos, char* d){
  if(sqlite3_bind_text(st, pos, d, strlen(d), SQLITE_STATIC)!=SQLITE_OK)
    throw sqlite_exception(db);
}

void TreeDB::open(char* fname){
  if( sqlite3_open(fname, &db) != SQLITE_OK) throw sqlite_exception(db);
}

void TreeDB::close(){
```

```
  db_finalize(stmt.begin_tx);
  db_finalize(stmt.end_tx);
  db_finalize(stmt.trees_ins);
  db_finalize(stmt.size_ins);
  db_finalize(stmt.gap_ins);
  db_finalize(stmt.birth_ins);
  db_finalize(stmt.live_ins);
  db_finalize(stmt.dead_ins);
  db_finalize(stmt.cut_taken_ins);
  db_finalize(stmt.cut_left_ins);
  db_finalize(stmt.wind_ins);
  db_finalize(stmt.storm_ins);
  db_finalize(stmt.cwd_out_ins);
  db_finalize(stmt.stocking_ins);

  for( int i=0; i<5; i++){
    db_finalize(stmt.cwd_log_ins[i]);
    db_finalize(stmt.cwd_snag_ins[i]);
  }

  if(sqlite3_close(db)!=SQLITE_OK) throw sqlite_exception(db);
}

void TreeDB::sql(const char *cmd){
  if( sqlite3_exec(db, cmd, 0, 0, 0) != SQLITE_OK)
    throw sqlite_exception(db);
}

void TreeDB::init(int xmax, int ymax){
  sql("begin transaction;\n"
      // CANOPY information
      "create table schema_version (value int);\n"
      "insert into schema_version values(22);\n"
      "create table canopy_version (value text);\n"
      "create table canopy_changes (value text);\n"

      // Simulation information
      "create table sim_args (value text);\n"

      // Tree-level information
      "create table trees (treeno int, x float, y float, species int,\n"
      "    primary key(treeno));\n"

      "create table sizes (treeno int, year int,\n"
      "    n float, e float, s float, w float, \n"
      "    nexp float, eexp float, sexp float, wexp float, \n"
      "    dbh float, ht float, hbc float, hwc float,  \n"
      "    relht float, reld float, \n"
      "    age int, \n"
      "    primary key(treeno,year));\n"

      // Gap information:
      "create table gaps (treeno int, year int,\n"
      "    gap_age int, gap_area float, comp_sap_tca float, comp_sap_ci float, \n"
      "    primary key(treeno,year));\n"

      "create table birth(treeno int, year int, primary key(treeno));\n"
      "create table live (treeno int, year int, primary key(treeno,year));\n"
```

```
"create table dead (treeno int, year int, primary key(treeno));\n"
"create table cut_taken  (treeno int, year int, primary key(treeno));\n"
"create table cut_left   (treeno int, year int, primary key(treeno));\n"
"create table wind (treeno int, year int, primary key(treeno));\n"
//CWD information
"create table cwd_log_1(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_log_2(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_log_3(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_log_4(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_log_5(treeno int,year int,primary key(treeno,year));\n"

"create table cwd_snag_1(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_snag_2(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_snag_3(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_snag_4(treeno int,year int,primary key(treeno,year));\n"
"create table cwd_snag_5(treeno int,year int,primary key(treeno,year));\n"

"create table cwd_out(treeno int, year int, primary key(treeno));\n"
// Plot-level information
"create table plotinfo (xmax int, ymax int);\n"
"create table stocking (year int, x int, y int,\n"
"  type int, stocking_level float);\n"
"create table storms (year int, target int, severity double, primary key(year));\n"
// Views, to make extracting stuff user-friendly
"create view cut as select * from cut_taken union \n"
"  select * from cut_left;\n"

"create view live_trees as select * from \n"
"   (select * from live left outer join trees using(treeno)) \n"
"   join sizes using(treeno,year);\n"

"create view new_trees as select * from \n"
"  birth join live_trees using(treeno,year);\n"

"create view dead_trees as select * from \n"
"   (select * from dead left outer join trees using(treeno)) \n"
"   join sizes using(treeno,year);\n"

"create view cut_taken_trees as select * from \n"
"   (select * from cut_taken left outer join trees using(treeno)) \n"
"   join sizes using(treeno,year);\n"

"create view cut_left_trees as select * from \n"
"   (select * from cut_left left outer join trees using(treeno)) \n"
"   join sizes using(treeno,year);\n"

"create view cut_trees as select * from \n"
"   (select * from cut left outer join trees using(treeno)) \n"
"   join sizes using(treeno,year);\n"

"create view wind_trees as select * from \n"
"   (select * from wind left outer join trees using(treeno)) \n"
"   join sizes using(treeno,year);\n"

"commit;\n");
```

```
  char* tmp = (char*)malloc(50+strlen(full_version_string)); CHK_MEM(tmp);
  sprintf(tmp, "insert into canopy_version values (\'%s\');",
          full_version_string);
  sql(tmp);
  free(tmp);

  tmp = (char*)malloc(50+strlen(local_changes_string)); CHK_MEM(tmp);
  sprintf(tmp, "insert into canopy_changes values (\'%s\');",
          local_changes_string);
  sql(tmp);
  free(tmp);

  char cmd[1024];
  snprintf(cmd,1024,"insert into plotinfo values (%i,%i);\n",xmax,ymax);
  sql(cmd);

  sql("pragma synchronous = off;");

  db_prepare( &stmt.begin_tx, "begin transaction");
  db_prepare( &stmt.end_tx, "end transaction");
  db_prepare( &stmt.trees_ins,
              "insert or ignore into trees (x,y,species,treeno) "
              "values (?,?,?,?)");
  db_prepare( &stmt.size_ins,
              "insert into sizes (year,treeno,dbh,n,e,s,w,"
              "    nexp,eexp,sexp,wexp,ht,hbc,hwc,age,relht,reld)"
              "  values (?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?,?)");
  db_prepare( &stmt.gap_ins,
              "insert into gaps (year,treeno,gap_age,gap_area,comp_sap_tca,comp_sap_ci) "
              " values(?,?,?,?,?,?)");
  db_prepare( &stmt.birth_ins,"insert or ignore into birth values(?,?)");
  db_prepare( &stmt.live_ins, "insert or ignore into live values(?,?)");
  db_prepare( &stmt.dead_ins, "insert or ignore into dead values(?,?)" );
  db_prepare( &stmt.cut_taken_ins,  "insert or ignore into cut_taken values(?,?)");
  db_prepare( &stmt.cut_left_ins,   "insert or ignore into cut_left values(?,?)");
  db_prepare( &stmt.wind_ins, "insert or ignore into wind values (?,?)");
  db_prepare( &stmt.storm_ins,"insert or ignore into storms values(?,?,?)");
  db_prepare( &stmt.cwd_out_ins, "insert or ignore into cwd_out values(?,?)");
  db_prepare( &stmt.stocking_ins, "insert or ignore into stocking values(?,?,?,?,?)");

  for( int i=0; i<5; i++){
    char cmd[200];
    sprintf(cmd, "insert or ignore into cwd_log_%i values(?,?)", i+1);
    db_prepare(&stmt.cwd_log_ins[i], cmd);
    sprintf(cmd, "insert or ignore into cwd_snag_%i values(?,?)", i+1);
    db_prepare(&stmt.cwd_snag_ins[i], cmd);
  }
}

void TreeDB::size_ins(TreePtr t, int year){
  assert(year>=0);

  int ns_status = t->need_save(year);

  if (ns_status==2 || ns_status==3){
```

```
  // Check data before insertion.
  assert( isfinite( t->GetXCoord() ));
  assert( isfinite( t->GetYCoord() ));
  assert( isfinite( t->GetSpecies() ));
  assert( isfinite( t->GetTreeno() ));

  db_bind_double( stmt.trees_ins, 1, t->GetXCoord()  );
  db_bind_double( stmt.trees_ins, 2, t->GetYCoord()  );
  db_bind_int   ( stmt.trees_ins, 3, t->GetSpecies() );
  db_bind_int   ( stmt.trees_ins, 4, t->GetTreeno()  );
  db_step( stmt.trees_ins );

  if (ns_status==2) {
    db_bind_int( stmt.birth_ins, 1, t->GetTreeno() );
    db_bind_int( stmt.birth_ins, 2, year);
    db_step( stmt.birth_ins );
  }
}

if (ns_status>0){
  // Extract data from the tree obj:
  int    treeno = t->GetTreeno();
  double dbh    = t->GetDBH();
  double n      = t->get_cr(0);
  double e      = t->get_cr(1);
  double s      = t->get_cr(2);
  double w      = t->get_cr(3);
  double nexp   = t->get_ecr(0);
  double eexp   = t->get_ecr(1);
  double sexp   = t->get_ecr(2);
  double wexp   = t->get_ecr(3);
  double th     = t->GetTotalHeight();
  double bh     = t->GetBaseHeight();
  double wh     = t->GetWidestHeight();
  double age    = t->get_age();
  double relht  = t->GetRH();
  double reld   = t->GetRD();

  // Check the data. If it's become corrupted, we want to explode.
  assert( 0<= treeno );
  assert( isfinite( dbh )    && 2 <= dbh );
  assert( isfinite( n )      && 0<= n    );
  assert( isfinite( e )      && 0<= e    );
  assert( isfinite( s )      && 0<= s    );
  assert( isfinite( w )      && 0<= w    );
  assert( isfinite( nexp )   && 0<= nexp );
  assert( isfinite( eexp )   && 0<= eexp );
  assert( isfinite( sexp )   && 0<= sexp );
  assert( isfinite( wexp )   && 0<= wexp );
  assert( isfinite( th)      && 0 <= th  );
  assert( isfinite( wh )     && 0 <= wh  );
  assert( isfinite( age )    && 0 <= age );
  assert( isfinite( relht )  && 0 <= relht );

  db_bind_int   ( stmt.size_ins,  1, year);
```

```
    db_bind_int    ( stmt.size_ins,  2, treeno );
    db_bind_double( stmt.size_ins,  3, dbh );
    db_bind_double( stmt.size_ins,  4, n );
    db_bind_double( stmt.size_ins,  5, e );
    db_bind_double( stmt.size_ins,  6, s );
    db_bind_double( stmt.size_ins,  7, w );
    db_bind_double( stmt.size_ins,  8, nexp );
    db_bind_double( stmt.size_ins,  9, eexp );
    db_bind_double( stmt.size_ins, 10, sexp );
    db_bind_double( stmt.size_ins, 11, wexp );
    db_bind_double( stmt.size_ins, 12, th  );
    db_bind_double( stmt.size_ins, 13, bh  );
    db_bind_double( stmt.size_ins, 14, wh  );
    db_bind_double( stmt.size_ins, 15, age );
    db_bind_double( stmt.size_ins, 16, relht );
    db_bind_double( stmt.size_ins, 17, reld );

    if ( th<=17 && t->GetSpecies() != sp_ironwood){
      double g_area = t->get_gap_area();
      double g_age  = t->get_gap_age();
      double comp_sap_tca = t->get_comp_sap_tca();
      double comp_sap_ci  = t->get_comp_sap_ci();

      db_bind_int( stmt.gap_ins, 1, year);
      db_bind_int( stmt.gap_ins, 2, treeno);
      db_bind_double( stmt.gap_ins, 3, g_age);
      db_bind_double( stmt.gap_ins, 4, g_area);
      db_bind_double( stmt.gap_ins, 5, comp_sap_tca);
      db_bind_double( stmt.gap_ins, 6, comp_sap_ci);

      db_step( stmt.gap_ins );
    }

    db_step( stmt.size_ins );
  }
}

void TreeDB::insert(Stand* st){
  assert( st->year >= 0 );

  db_step(stmt.begin_tx);

  TreeList::iterator iter, stop;

  stop=st->living->end();
  for(iter=st->living->begin(); iter!=stop; iter++){
    if( (*iter)->big_enough_to_save() ){
      db_bind_int( stmt.live_ins, 1, (*iter)->GetTreeno());
      db_bind_int( stmt.live_ins, 2, st->year );
      db_step( stmt.live_ins);

      size_ins( *iter, st->year);
    }
  }

  stop=st->dead->end(); //haha!
  for(iter=st->dead->begin(); iter!=stop; iter++){
    if( *iter == NULL ) {
```

```
    printf ("WARNING: null tree on the dead tree list\n");
  } else {
    if( (*iter)->big_enough_to_save() ){
      db_bind_int(stmt.dead_ins, 1, (*iter)->GetTreeno());
      db_bind_int(stmt.dead_ins, 2, st->year );
      db_step(stmt.dead_ins);

      size_ins( *iter, st->year );
    }
  }
}

stop=st->blown_down->end();
for(iter=st->blown_down->begin(); iter!=stop; iter++){
  if( (*iter)->big_enough_to_save() ){
    db_bind_int(stmt.wind_ins, 1, (*iter)->GetTreeno());
    db_bind_int(stmt.wind_ins, 2, st->year);
    db_step(stmt.wind_ins);

    size_ins( *iter, st->year );
  }
}

stop=st->harvested->end();
for(iter=st->harvested->begin(); iter!=stop; iter++){
  if( (*iter)->big_enough_to_save() ){
    db_bind_int(stmt.cut_taken_ins, 1, (*iter)->GetTreeno());
    db_bind_int(stmt.cut_taken_ins, 2, st->year);
    db_step(stmt.cut_taken_ins);
  }

  size_ins(*iter, st->year);
}

stop=st->harvested_left->end();
for(iter=st->harvested_left->begin(); iter!=stop; iter++){
  if( (*iter)->big_enough_to_save() ){
    db_bind_int(stmt.cut_left_ins, 1, (*iter)->GetTreeno());
    db_bind_int(stmt.cut_left_ins, 2, st->year);
    db_step(stmt.cut_left_ins);
  }

  size_ins(*iter, st->year);
}

if (st->year%save_interval==0) {
  for( int i=0; i<5; i++){
    stop=st->cwd_log[i]->end();
    for( iter= st->cwd_log[i]->begin(); iter!= stop; iter++ ){
      db_bind_int(stmt.cwd_log_ins[i], 1, (*iter)->GetTreeno());
      db_bind_int(stmt.cwd_log_ins[i], 2, st->year );
      db_step(stmt.cwd_log_ins[i] );
    }

    stop=st->cwd_snag[i]->end();
    for( iter= st->cwd_snag[i]->begin(); iter!= stop; iter++ ){

      db_bind_int(stmt.cwd_snag_ins[i], 1, (*iter)->GetTreeno());
      db_bind_int(stmt.cwd_snag_ins[i], 2, st->year );
      db_step(stmt.cwd_snag_ins[i]);
```

```
      }
    }

    stop=st->cwd_out->end();
    for( iter=st->cwd_out->begin(); iter != stop; iter++){
      db_bind_int(stmt.cwd_out_ins, 1, (*iter)->GetTreeno());
      db_bind_int(stmt.cwd_out_ins, 2, st->year);
      db_step(stmt.cwd_out_ins);
    }

    // Insert 900m2 stocking values:
    for( int i=0; i<st->xmax; i++ ){
      for( int j=0; j<st->ymax; j++ ){

        db_bind_int( stmt.stocking_ins, 1, st->year);
        db_bind_int( stmt.stocking_ins, 2, i   );
        db_bind_int( stmt.stocking_ins, 3, j   );
        db_bind_int( stmt.stocking_ins, 4, 900 );
        db_bind_double( stmt.stocking_ins, 5, st->stocking_900(i,j) );
        db_step( stmt.stocking_ins );
      }
    }
  }

  db_step(stmt.end_tx);
}

void TreeDB::storm_insert( int year, int target, double severity){
  db_bind_int (stmt.storm_ins, 1, year);
  db_bind_int (stmt.storm_ins, 2, target);
  db_bind_double(stmt.storm_ins, 3, severity);
  db_step(stmt.storm_ins);
}

void TreeDB::begin_tx(){ db_step(stmt.begin_tx); }
void TreeDB::end_tx  (){ db_step(stmt.end_tx  ); }

void TreeDB::argv_ins(int argc, char* argv[]){

  int length = 0;
  for( int i=0; i<argc; i++){
    length += strlen(argv[i]);
    length += 1; // for a space or a null termination
  }

  char* all_args = (char*)calloc(length, sizeof(char)); CHK_MEM(all_args);

  for( int i=0; i<argc; i++){
    strcat(all_args, argv[i]);
    if (i< (argc-1) ) { strcat( all_args, " "); }
  }

  sqlite3_stmt *stmt;

  db_prepare( &stmt, "insert into sim_args values (?)");
  db_bind_text( stmt, 1, all_args );
  db_step(stmt);
  db_finalize(stmt);

  free( all_args );
}
```

C.6 LuaHelper.cp

```
// -*- C++ -*-

#include "LuaHelper.h"

extern "C" {
#include <lua.h>
#include <lauxlib.h>
#include <lualib.h>
}

LuaHelper::LuaHelper( void ){
  Lua = lua_open();

  luaL_openlibs(Lua);
}

//! Set a global value in this helper's environment
void LuaHelper::l_setglobal(const char* name, double value){
  lua_pushnumber(Lua, value );
  lua_setglobal(Lua, name);
}

//! Set a field in the Lua table sitting at TOS.
void LuaHelper::l_setfield(const char* key, double value){
  lua_pushstring(Lua, key);
  lua_pushnumber(Lua, value);
  lua_settable(Lua, -3);
}
```

C.7 Random.cp

```
//-*- C++ -*-

#include <limits.h>
#include <stdlib.h>
#include <stdint.h>
#include <math.h>
#include <assert.h>
#include "util.h"
#include "Random.h"

#ifndef UINT32_MAX
#warning No definition found for UINT32_MAX... Defining it myself
#define UINT32_MAX 0xffffffffU
#endif

double urand01(){
  double rc = (double)arc4random()/UINT32_MAX;
  assert(isfinite(rc) && 0< rc && rc < 1);
  return rc;
}

double rnorm(double mean, double sd){
  double rc;
  do {
    double u1= urand01();
    double u2= urand01();
    double z = sqrt(-2*log( u1 ))*cos(2*M_PI*u2);
```

```
    rc = mean + sd * z;
  } while ( ! isfinite(rc)  );

  return rc;
}

double rnorm_trunc(double mean, double sd, double nsd){
  double rc;
  do {
    rc = rnorm(mean,sd);
  } while( fabs(rc-mean) > nsd*sd );
  assert(isfinite(rc));
  return rc;
}

double rnorm_trunc(double mean, double sd){
  return rnorm_trunc(mean,sd,1.96);
}
```

C.8 StormModel.cp

```
// -*- C++ -*-
extern "C" {
#include <lua.h>
#include <lauxlib.h>
#include <lualib.h>
}

#include <assert.h>

#include "StormModel.h"
#include "Tree.h"
#include "Random.h"

/// This must determine storm severity.
/** If the severity is <10% ECA, there is no storm. */
double StormModel::storm_eca_removal(void){
  double RN, percentECA;
  RN = urand01();

  // Roll a random number (RN) between 0 and 1:
  // Like the old STORM simulation, work down from the most intense
  // disturbance:

  if (RN > 0.999732191)  {percentECA = 70; return percentECA;}
  if (RN > 0.999479167)  {percentECA = 60; return percentECA;}
  if (RN > 0.998986829)  {percentECA = 50; return percentECA;}
  if (RN > 0.998031496)  {percentECA = 40; return percentECA;}
  if (RN > 0.996168582)  {percentECA = 30; return percentECA;}
  if (RN > 0.992537313)  {percentECA = 20; return percentECA;}
  if (RN > 0.985507246)  {percentECA = 10; return percentECA;}
  return 0;
}

double StormModel::severity_index(double percent_eca){
  double X = (-0.00482864898067599) + ( 0.00770638698811676) * percent_eca;
  return fmin(1,fmax(0,X));
}

/// determine if there is a storm.
/* If so, determine its severity and convert to severity index (Si)
   for each tree in the list living
```

```
    if rand() < Tree::blowdown(Si) {
    add to blown_down */
double StormModel::storm(double si, TreeList* living, TreeSet *blown_down){
  double eca_removed = 0;
  blown_down->clear();

  for(TreeList::iterator it = living->begin();
      it != living->end(); it++){
    if ( urand01() < (*it)->blowDown(si) ){
      blown_down->insert( (*it)->GetTreeno() );
      eca_removed += (*it)->get_eca();
    }
  }

  return eca_removed;
}

StormModel::StormModel(char* fname){
  l_setglobal("storm_stochastic", -1 );
  l_setglobal("storm_none", 0 );

  int error = luaL_loadfile(Lua, fname) || lua_pcall(Lua, 0, 0, 0);

  if (error) {
    fprintf(stderr, "%s", lua_tostring(Lua, -1));
    lua_pop(Lua, 1);
    exit(1);
  }
}

int StormModel::get_storm_specification(int year){
  // -1 : stochastic storm simulation
  // 0 : no storm in this year
  // Other: Specified removal given by return value

  int rc;
  lua_getglobal(Lua, "get_storm_specification");
  lua_pushnumber(Lua, (double)year );
  lua_pcall(Lua,1,1,0);

  assert( lua_isnumber(Lua, -1));
  rc = (int)lua_tonumber(Lua, -1);
  lua_pop(Lua, 1);

  return rc;
}
```

C.9 TreeData.cp

```
// -*- c++ -*-
/***************** CTreeData.cp ***************/
#include <assert.h>
#include <unistd.h>
#include "util.h"
#include "TreeData.h"
#include "Tree.h"
#include "CanopyGlobals.h"
#include "Random.h"

/// Convert a string to an htype_t
htype_t TreeData::string_to_htype(char* string){
```

```
  htype_t rc;
  if      ( strcmp(string,"AOCa")==0 ){ rc=ht_AOCa; }
  else if( strcmp(string,"ATD" )==0 ){ rc=ht_ATD;  }
  else if( strcmp(string,"ATM" )==0 ){ rc=ht_ATM;  }
  // Equivalents:
  else if( strcmp(string, "TD" )==0 ){ rc=ht_ATD;  }
  else if( strcmp(string, "TMC")==0 ){ rc=ht_ATM;  }
  else if( strcmp(string, "FI" )==0 ){ rc=ht_AOCa; }
  else {
    printf("ERROR (%s:%i): Unknown habitat type: %s\n",
           __FILE__, __LINE__, string );
    exit(1);
  }
  return rc;
}

///ReadHabSpec
/** Reads a habitat specification from a file. */
hdata_t* TreeData::ReadHabSpec(char* file){
  FILE* fp = fopen(file,"r");
  int nrow=0;
  int ncol=1;
  char c;

  hdata_t* rc = (hdata_t*)malloc( sizeof(hdata_t) ); CHK_MEM(rc);

  //Count the number of columns in the first row.
  while( (c=fgetc(fp)) != '\n' ) {
    if( c==',' ) ncol++;
  }

  fseek(fp,0,SEEK_SET);

  while( (c=fgetc(fp)) != EOF ){
    if( c=='\n' ) nrow++;
    //Should assert() that all ncol are equal.
  }

  fseek(fp,0,SEEK_SET);

  rc->hdata=(htype_t**)malloc( (nrow)*sizeof(htype_t*)); CHK_MEM(rc->hdata);

  for(int i=0; i<nrow; i++){
    rc->hdata[i]=(htype_t*)malloc( ncol*sizeof(htype_t)); CHK_MEM(rc->hdata[i]);
  }

  rc->ncol = ncol;
  rc->nrow = nrow;

  int row=0;
  int col=0;
  int bufpos=0;
  char buf[20];

  while( (c=fgetc(fp)) != EOF ){
    // if c ~= m/[A-Za-z]/
    if( ('A'<=c && c<='Z') || ('a'<=c && c<='z') ){
      buf[bufpos++]=c;
    }

    if( c==',' || c=='\n' ){
```

```
      buf[bufpos]='\0';
      rc->hdata[row][col]=string_to_htype(buf);
      if(c==','){ col++; }
      if(c=='\n'){
        col=0;
        row++;
      }
      bufpos = 0;
    }
  }
  fclose(fp);
  // 0,0 is the soutwest corner.
  // Because I read in row-ordering, my habitat types are upside down.
  // fix this:
  for(row = 0; row<=floor( (nrow-1)/2 ); row++){
    htype_t* tmp;
    int first = row;
    int second = nrow-row-1;
    tmp = rc->hdata[second];
    rc->hdata[second]=rc->hdata[first];
    rc->hdata[first]=tmp;
  }

  return rc;
}

hdata_t* TreeData::GenerateHabitatSpec(char* type,
                                       double width, double height){
  int nrow = floor( height/10 );
  int ncol = floor( width/10 );

  htype_t tgt_type = string_to_htype(type);

  hdata_t* rc = (hdata_t*)malloc( sizeof(hdata_t) ); CHK_MEM(rc);

  rc->nrow = nrow;
  rc->ncol = ncol;

  rc->hdata=(htype_t**)malloc( (nrow)*sizeof(htype_t*)); CHK_MEM( rc->hdata);

  for(int i=0; i<nrow; i++){
    rc->hdata[i]=(htype_t*)malloc( ncol*sizeof(htype_t)); CHK_MEM( rc->hdata[i] )
  }

  for(int i=0; i<nrow; i++){
    for( int j=0; j<ncol; j++){
      rc->hdata[i][j] = tgt_type;
    }
  }

  return rc;
}

void TreeData::ReadTreeList(char* file, TreeList *aTreeList, hdata_t* &hd,
                            int nx, int ny){
  int rc;
  // First, check to see if there is an hdata file
  char hdata_fname[1024]; // PATH_MAX, maybe ?

  rc = snprintf(hdata_fname, 1024, "%s.ht", file);
```

```
if (rc>1024) { exit(1); }

if ( access( hdata_fname, R_OK) == 0 ){
  hd = ReadHabSpec(hdata_fname);
} else {
  hd = NULL;
}

FILE* fp = fopen(file,"r");

int line=1;

// Read in comments in the header, ditch them.
char c;
while( (c=getc(fp)) == '#' ){
  while( getc(fp) != '\n') { ; }
  line++;
}
ungetc(c, fp); // We went one too far.  put it back.

char file_type[512];
rc = fscanf(fp, "%[A-z0-9 -]\n", file_type); line++;
assert(rc==1);

if ( strcmp(file_type, "CANOPY spatial TreeList rev 1")==0 ){
  this->read_spatial_t1( aTreeList, fp, line, hd );
} else if ( strcmp(file_type, "CANOPY non-spatial TreeList rev 1")==0){
  rc=fscanf(fp, "Treeno Species dbh_cm ht_m\n"); line++;
  assert(rc==0);
  this->read_nonspatial_t1(aTreeList, fp, line, hd);
} else if ( strcmp(file_type, "CANOPY non-spatial TreeList rev 2")==0){
  rc=fscanf(fp, "Plot Treeno Species dbh_cm ht_m\n"); line++;
  assert(rc==0);
  this->read_nonspatial_t2(aTreeList, fp, line, hd);
} else if ( strcmp(file_type, "CANOPY non-spatial TreeList rev 3")==0){
  int nx, ny;
  double width, height;
  char habitat_type[512];

  // Read the header specifying the format.
  rc = fscanf(fp, "Plot Width  (m,ncells): %lf, %i\n", &width,  &nx); line++;
  assert(rc==2);
  rc = fscanf(fp, "Plot Height (m,ncells): %lf, %i\n", &height, &ny); line++;
  assert(rc==2);
  rc = fscanf(fp, "Habitat type: %s\n", habitat_type); line++;
  assert(rc==1);
  rc = fscanf(fp, "Px Py Treeno Species dbh_cm\n"); line++;
  assert(rc==0);

  hd = GenerateHabitatSpec( habitat_type, width, height);

  double cell_width = width / nx;
  double cell_height = height / ny;

  this->read_nonspatial_t3(aTreeList, fp, line, hd,
                           cell_width, cell_height);
} else {
```

```
    printf("Unknown file format\n");
    exit(1);
  }

  if( nx > 1 || ny > 1 ) {
    this->multiply_tl(aTreeList, hd, nx, ny);
    this->multiply_ht(hd, nx, ny);
  }

  fclose(fp);
}

void TreeData::read_spatial_t1(TreeList* tl, FILE* fp,
                               int& line, hdata_t* hd){
  int TreeNo;
  double xCoord, yCoord, DBH, TotalHeight;
  double BaseHeight, WidestHeight;
  double NorthTotal, NorthExp, EastTotal, EastExp;
  double SouthTotal, SouthExp, WestTotal, WestExp;
  double Distance, Azimuth;
  int Species, CC, Status;
  double BasalArea, PreviousBA;
  int count;
  while( (count=fscanf(fp,
                       "%li\t%lf\t%lf\t%i\t%i\t%lf\t%lf\t%lf\t"
                       "%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t%lf\t"
                       "%lf\t%lf\t%lf\t%i\t%lf\t%lf\n",
                       &TreeNo, &xCoord, &yCoord, &Species, &CC, &DBH,
                       &TotalHeight, &BaseHeight, &WidestHeight,
                       &NorthTotal, &NorthExp, &EastTotal, &EastExp,
                       &SouthTotal, &SouthExp, &WestTotal, &WestExp,
                       &Distance, &Azimuth, &Status,
                        &BasalArea, &PreviousBA )) >0 ){
    if(count==22) {
      xCoord = fmin(10.0*hd->ncol-0.01, fmax(0, xCoord));
      yCoord = fmin(10.0*hd->nrow-0.01, fmax(0, yCoord));
      TreePtr tree ( new Tree(TreeNo, xCoord, yCoord, DBH, (species_t) Species,
                              NorthTotal, EastTotal, SouthTotal, WestTotal, hd) );
      tree->SetStatus( (status_t)Status );
      tl->push_front(tree);
    } else {
      printf("ERROR: %i data items (need 22) on line %i\n",
             count, line);
      exit(1);
    }
    line++;
  }
}

void TreeData::read_nonspatial_t1(TreeList* tl, FILE* fp,
                                  int& line, hdata_t* hd){
  int treeno, sp;
  double dbh, ht;
  int count;
  while( (count=fscanf(fp, "%i %i %lf %lf\n", &treeno, &sp, &dbh, &ht)) >0 ){
    if(count==4){
```

```
      TreePtr tree (new Tree(5, 5, dbh, (species_t)sp, -1, -1, -1, -1, hd) );
      tree->SetStatus( st_live );
      tl->push_front(tree);
    } else {
      printf("ERROR: file format seems wrong (line %i)\n",line);
      exit(1);
    }
    line++;
  }
}

void TreeData::read_nonspatial_t2(TreeList* tl, FILE* fp,
                                  int& line, hdata_t* hd){
  int plot, treeno, sp, count;
  double dbh, ht;
  while( (count=fscanf(fp,"%i %i %i %lf %lf\n",
                       &plot,&treeno,&sp,&dbh,&ht)) >0 ){
    if( count==5){
      TreePtr tree (new Tree(10*(plot-1)+5,5,dbh,(species_t)sp,-1,-1,-1,-1,hd) );
      tree->SetStatus( st_live );
      tl->push_front(tree);
    } else {
      printf("ERROR: file format seems wrong (line %i)\n", line);
      exit(1);
    }
    line++;
  }
}

void TreeData::read_nonspatial_t3(TreeList* tl, FILE* fp,
                                  int& line, hdata_t* hd,
                                  double cell_width, double cell_height){
  int px, py, treeno, species, count;
  double dbh;
  while ( (count = fscanf(fp, "%i %i %i %i %lf\n",
                          &px, &py, &treeno, &species, &dbh)) >0 ){
    if (count==5){
      if ( dbh >= 2.0 ) {
        double x =  cell_width*urand01() + (px-1)*cell_width  ;
        double y = cell_height*urand01() + (py-1)*cell_height ;
        TreePtr tree (new Tree( x, y, dbh, (species_t)species,
                        -1, -1, -1, -1,
                                hd) );
        tree->SetStatus(st_live);
        tl->push_front(tree);
      }
    } else {
      printf("ERROR: file format seems wrong (line %i)\n", line);
      exit(1);
    }
    line++;
  }
}
/** Multiply a habitat type grid.
 htype - the habitat type to be multiplied
 nx    - the number of replicates to make in the x direction
 ny    - the number of replicates to make in the Y direction
```

```
 */
void TreeData::multiply_ht(hdata_t* hd, int nx, int ny){

  check_init_flips(nx,ny);

  int old_ncol = hd->ncol;
  int old_nrow = hd->nrow;

  int ncol = nx*old_ncol;
  int nrow = ny*old_nrow;

  //Expand this htype to be big enough:
  hd->hdata=(htype_t**)realloc(hd->hdata,nrow*sizeof(htype_t*)); CHK_MEM(hd->hdata);

  // Expand the existing rows:
  for( int i=0; i< old_nrow; i++){
    hd->hdata[i]=(htype_t*)realloc(hd->hdata[i],ncol*sizeof(htype_t));
        CHK_MEM(hd->hdata[i]);
  }

  // Create new rows as needed:
  for( int i=old_nrow; i<nrow; i++){
    hd->hdata[i]=(htype_t*)malloc( ncol*sizeof(htype_t)); CHK_MEM( hd->hdata[i] );
  }

  hd->ncol = ncol;
  hd->nrow = nrow;

  // Now, iterate through the area, scattering copies as required:
  for(int i=0; i<nrow; i++){
    for( int j=0; j<ncol; j++){
      // If we're in a new area (eg. outside the old bounds):
      if( old_nrow <= i || old_ncol <= j ){
        int old_i = i%old_nrow;
        int old_j = j%old_ncol;

        int fx = floor(j/old_ncol);
        int fy = floor(i/old_nrow);

        if( flips[fx][fy] & 0x1 ){ old_i = (old_ncol-1)-old_i; }
        if( flips[fx][fy] & 0x2 ){ old_j = (old_nrow-1)-old_j; }

        hd->hdata[i][j]=hd->hdata[ old_i ][ old_j ];
      }
    }
  }
}

/** Multiply a treelist.
    tl - the source treelist
    ht - the source habitat type grid (matching the source tl).
    nx - the number of replicates to make in the x direction
    ny - the number of replicates to make in the y direction
*/

void TreeData::multiply_tl(TreeList* tl, hdata_t* hd, int nx, int ny){
  double xmax = 10.0*hd->ncol;
  double ymax = 10.0*hd->nrow;

  TreeList *tmp = new TreeList();

  check_init_flips(nx,ny);

  for( TreeList::iterator iter = tl->begin();
```

```
        iter != tl->end(); iter++){

    for( int i=0; i<nx; i++ ){
      for( int j=0; j<ny; j++ ){
        if( i>0 || j>0 ){
          TreePtr t(new Tree());

          // Do a member-wise copy of the current tree.
          *t = *(*iter);

          double xi = (*iter)->GetXCoord();
          double yi = (*iter)->GetYCoord();

          if( flips[i][j] & 0x1 ) { xi = xmax-xi; }
          if( flips[i][j] & 0x2 ) { yi = ymax-yi; }

          //Do coord transform.
          t->SetXCoord( xmax*i + xi );
          t->SetYCoord( ymax*j + yi );

          // Push the transformed tree onto the new treelist.
          tmp -> push_front(t);
        }
      }
    }
  }

  tl->splice(tl->end(), *tmp);
  delete tmp;
}

void TreeData::check_init_flips(int nx, int ny){
  if( flips == NULL) {
    flip_x = nx;
    flip_y = ny;

    flips = (short**)malloc(nx*sizeof(short*)); CHK_MEM(flips);
    for( int i=0; i<nx; i++){
      flips[i] = (short*)malloc(ny*sizeof(short)); CHK_MEM(flips[i]);
    }

    for( int i=0; i<nx; i++) {
      for( int j=0; j<ny; j++){
        if( i==0 && j == 0 ) {
          flips[i][j]=0;
        } else {
          flips[i][j] = 1*round(urand01())+2*round(urand01());
        }
      }
    }
  } else {
    assert( nx == flip_x );
    assert( ny == flip_y );
  }
}

TreeData::TreeData(){
  flips=NULL;
}

TreeData::~TreeData(){
  if (flips != NULL) {
```

```
    for (int i=0; i<flip_x; i++){
      free(flips[i]);
    }
    free(flips);
  }
}
```

C.10 TreeGrid.cp

```
// -*- c++ -*-

#include <list.h>
#include <assert.h>
#include "util.h"
#include "CanopyGlobals.h"
#include "Tree.h"
#include "TreeGrid.h"
#include "Stand.h"

TreeGrid::TreeGrid(Stand* st_in){
  //Initialize the grid, start empty.

  st = st_in;
  grid = (TreeList***) malloc( st->xmax*sizeof(TreeList**)); CHK_MEM( grid );

  for (int i=0; i<st->xmax; i++){
    grid[i]=(TreeList**)malloc( st->ymax*sizeof(TreeList*)); CHK_MEM(grid[i]);
    for( int j=0; j<st->ymax; j++){
      grid[i][j] = new TreeList();
    }
  }
}

///Add trees in to the TreeGrid
void TreeGrid::add_trees(TreeList* trees){

  for( TreeList::iterator it = trees->begin(); it != trees->end(); it++){
    add_tree( *it );
  }
}

void TreeGrid::add_tree( TreePtr t){
  int i = floor( t->GetXCoord() / 10 );
  int j = floor( t->GetYCoord() / 10 );

  grid[i][j]->push_front( t );
}

/// Start a new search and specify where it will be
void TreeGrid::set_search_center(double x, double y){
  cur_i = floor(x/10);
  cur_j = floor(y/10);

  cur_i = cur_i - st->xmax*floor(cur_i/st->xmax);
  cur_j = cur_j - st->ymax*floor(cur_j/st->ymax);

  center_x = x;
  center_y = y;

  cur_n=0;

  cur_list = grid[cur_i][cur_j];
```

```
  // Need to sort?
  Stand::fixed_dist_comp_t cmp;
  cmp.xc = center_x;
  cmp.yc = center_y;
  cmp.st = st;
  cur_list->sort(cmp);

  cur_tree = cur_list->begin();
}

/// When we've finished in one bucket, determine the next bucket
/** The math in this one is a little hairy. It will produce a sequence
    of x/y offsets that when added to the current search center will
    form a grid like this, with center at (0):
    [ 9] [10] [11] [12] [16]
    [24] [ 1] [ 2] [ 4] [15]
    [23] [ 8] ( 0) [ 3] [14]
    [22] [ 7] [ 5] [ 6] [13]
    [21] [17] [18] [19] [20]

    int* o is a pointer to an int[2] which will receive the offset
    int  n is the  index number of the requested tile.
*/
void TreeGrid::perform_offset(int* o, int n){
  if( n>0 ){
    int r=0;
    int np = n;
    while( np > 8*r ){
      np -= 8*r;
      r++;
    }

    np--;

    int w=2*r+1;

    if     (   0 <= np && np < 2*r ) { o[0]= np%w-r;         o[1]= r; }
    else if( 2*r <= np && np < 4*r ) { o[1]=(np-2*r)%w-r+1; o[0]= r; }
    else if( 4*r <= np && np < 6*r ) { o[0]=(np-4*r)%w-r+1; o[1]=-r; }
    else if( 6*r <= np && np < 8*r ) { o[1]=(np-6*r)%w-r;   o[0]=-r; }
    else {
      assert(0);
    }
  } else {
    o[0]=0;
    o[1]=0;
  }
}

/// Pull a tree out in distance order
/**
   Starts at the search center, presents trees in order if their
   distance to the search center. Goes one bucket at a time outward
   from there in the order defined by perform_offset()
   Note: this stops after the 24th bucket, as that's enough for a
   50x50 meter search area, 2500m^2, which is larger than the largest
   gap in any of our data sources.
 */
TreePtr TreeGrid::get_next_tree(){
  TreePtr rc;

  while( cur_tree == cur_list->end() && cur_n <= 24 ){
    int offset[2];
```

```
    perform_offset(offset, ++cur_n);
    int xp = cur_i + offset[0];
    int yp = cur_j + offset[1];

    xp = xp - st->xmax * floor( (float)xp / st->xmax);
    yp = yp - st->ymax * floor( (float)yp / st->ymax);

    cur_list = grid[xp][yp];

    Stand::fixed_dist_comp_t cmp;
    cmp.xc = center_x;
    cmp.yc = center_y;
    cmp.st = st;
    cur_list->sort(cmp);

    cur_tree = cur_list->begin();
  }

  if( cur_n <= 24 ){
    rc = *cur_tree;
    cur_tree++;
  } else {
    rc.reset();
  }

  return rc;
}
//Pull dead trees out of the grid
void TreeGrid::remove_dead(){
  for (int i=0; i<st->xmax; i++){
    for (int j=0; j<st->ymax; j++){
      TreeList::iterator it = grid[i][j]->begin();
      while (it != grid[i][j]->end() ){
        if ( (*it)->GetStatus() != st_live ){
          it = grid[i][j]->erase(it);
        } else {
          it++;
        }
      }
    }
  }
}
```

APPENDIX D

CANOPY source code: C++ header files

D.1 CanopyGlobals.h

```
#ifndef __CanopyGlobals_h__
#define __CanopyGlobals_h__

#include <list>
#include <set>
#include <tr1/memory>

// Constant definitions

typedef enum {
  ht_AOCa=0,
  ht_ATD=1,
  ht_ATM=2
} htype_t;

typedef struct {
  int nrow, ncol;
  htype_t** hdata;
} hdata_t;

typedef enum {
  // Statuses that occur in datafiles.
  // For these we need to define particular status codes.
  st_live=0, st_dead=1, st_cut=2,
   st_log_1=11,  st_log_2=12,  st_log_3=13,  st_log_4=14,  st_log_5=15,
  st_snag_1=21, st_snag_2=22, st_snag_3=23, st_snag_4=24, st_snag_5=25,
  // Status codes used internally.
  // Particular values don't matter.
  st_cut_leave, st_cut_girdle, st_out
} status_t;

typedef enum {
  sp_basswood          = 2,
  sp_sugar_maple       = 4,
  sp_white_ash         = 13,
  sp_ironwood          = 24,
  sp_hemlock           = 5,
  sp_yellow_birch      = 6,
```

```
  sp_red_maple         = 9,
  sp_white_pine        = 7,
  sp_paper_birch       = 1,
  sp_northern_red_oak  = 10,
  sp_green_ash         = 14,
  sp_american_elm      = 21,
  sp_black_cherry      = 22,
  sp_mountain_maple    = 39,
  sp_balsam_fir        = 41,
  sp_white_spruce      = 43,
  sp_white_cedar       = 46
} species_t;

typedef enum {
  stage_sapling,
  stage_pole,
  stage_msm,
  stage_mature,
  stage_og_la,
  stage_og_et,
  stage_og_lt,
  stage_og_ss
} stage_t;

// Type definitions needed nearly everywhere

class Tree;
typedef std::tr1::shared_ptr<Tree> TreePtr;
typedef std::list<TreePtr> TreeList;

struct ltlong {
  bool operator()(const long a, const long b) const {
    return a < b;
  }
};

typedef std::set<long, ltlong> TreeSet;

#endif
```

D.2 Harvest.h

```
// -*- C++ -*-

#ifndef __harvest_h__
#define __harvest_h__

#include "LuaHelper.h"

#include <vector.h>
#include <stack.h>

class Stand;
class Tree;

typedef struct {
  double x,y;
} point_t;
```

```
class Harvest : LuaHelper {

public:
  Harvest(Stand*, char*);
  void do_harvest();
  point_t* get_group();

private:
  Stand *my_stand;
  vector<TreePtr>* harvestable_trees;
  vector<TreePtr>* cwd_trees;
  stack<point_t*> group_centers;

  // Interfaces to Lua functions.
  bool is_it_time(int);
  bool is_it_inside(TreePtr);
  int cut_this_tree(TreePtr);
  void prepare();

  // Utility functions.
  void tree_to_lua(TreePtr);
  void tree_vector_to_lua(vector<TreePtr>*);
  void status_vector_to_lua( vector<double>*);
  void perform();

  void l_fetch_groups(void);
  void do_planting(void);
};

#endif
```

D.3 LuaHelper.h

```
// -*- C++ -*-

#ifndef __lua_helper_h__
#define __lua_helper_h__

extern "C"{
#include <lua.h>
}

class LuaHelper {

public:
  LuaHelper(void);

protected:
  lua_State *Lua;

  // Utility functions.
  void l_setfield(const char*, double);
  void l_setglobal(const char*, double);
};

#endif
```

D.4 Random.h

```
// -*- C++ -*-
```

```
#ifndef __Random_h__
#define __Random_h__

double urand01();
double rnorm(double,double);
double rnorm_trunc(double,double);
double rnorm_trunc(double,double,double);

#endif // __Random_h__
```

D.5 Stand.h

```
// -*- C++ -*-
#ifndef __Stand_h__
#define __Stand_h__

#include <list.h>
#include <slist.h>
#include "CanopyGlobals.h"
#include "Harvest.h"
#include "Tree.h"

class TreeDB;
class TreeGrid;
class StormModel;

typedef enum {
  regen_normal,
  regen_stem_exclusion,
  regen_recovery
} regen_mode;

#define DFN_PLOT_STAT(FNAME)                    \
  double FNAME##_##100(int,int);                \
  double FNAME##_##800(int,int);                \
  double FNAME##_##900(int,int)

#define DFN_TREE_STAT(FNAME)    \
  double FNAME##_##100(TreePtr); \
  double FNAME##_##800(TreePtr); \
  double FNAME##_##900(TreePtr)

class Stand {
  friend class Harvest;
  friend class TreeDB;
  friend class TreeGrid;

public:
  Stand(char*,char*,int,int, int, char**);

  ~Stand();
  void calculate(); // Calculates stocking, gap sizes, etc.
  void regenerate();
  void grow();
  void die();
  void decay();
  bool wind(); // windstorm simulation
```

```
  void add_harvester(char*);
  void add_storm_model(char*);
  void harvest();
  void plant_tree(double, double, species_t, double);

  void inc_year();
  void save();
  void db_close();

private:
  // Constant defs:
  static const int n_subplot_species = 8;

  // Type definitions:
  typedef struct {
    int          n[n_subplot_species];
    double      ba[n_subplot_species];
    double sap_ba[n_subplot_species];
    int N, n_sap;
    double sum_dbh, sum_ba, harv_ba;
    double sap_sum_ba;
    double sum_ht, ref_ht;
  } subplot_t;

  typedef struct {
    double sap_ba, pol_ba, mat_ba, lrg_ba, gap_sap_ba;
  } stage_data_t;

  typedef struct {
    struct { double x, y; } subj;
    struct { double x, y; } comp;
  } coords_t;

  typedef struct {
    double xc, yc;
    Stand* st;
    bool operator()( const TreePtr T1, const TreePtr T2) const {
      double dist1, dist2;

      coords_t T1_c, T2_c;

      T1_c.subj.x = xc;
      T1_c.subj.y = yc;
      T1_c.comp.x = T1->GetXCoord();
      T1_c.comp.y = T1->GetYCoord();

      T2_c.subj.x = xc;
      T2_c.subj.y = yc;
      T2_c.comp.x = T2->GetXCoord();
      T2_c.comp.y = T2->GetYCoord();

      st->transform( xc, yc, T1, T1_c );
      st->transform( xc, yc, T2, T2_c );

      dist1 = hypot( T1_c.comp.x - T1_c.subj.x,
                     T1_c.comp.y - T1_c.subj.y );
```

```
    dist2 = hypot( T2_c.comp.x - T2_c.subj.x,
                   T2_c.comp.y - T2_c.comp.y );

    return dist1 < dist2;
  }
} fixed_dist_comp_t;

// Instance variables:

TreeList *living;
TreeList *dead;
TreeList *blown_down;
TreeList *harvested;
TreeList *harvested_left;
TreeList *cwd_log[5], *cwd_snag[5], *cwd_out;
slist<Harvest*> *harvesters;
list<point_t*> *gcs_group_centers;
list<double> *storm_queue;
vector<TreePtr> *babies;

TreeGrid *the_grid;

StormModel *my_storm_model;
int storm_clock;

int year;
double xoff, yoff;
TreeDB *tdb;

// The grid of subplots:
subplot_t **subplot_100, **subplot_800, **subplot_900;
TreeList ***subplot_trees_100, ***subplot_trees_800, ***subplot_trees_900;

stage_data_t **subplot_stages_2500;

// For the "maintain smallest" mode:
unsigned int target_sapling_population;

hdata_t *hdata;
int xmax, ymax;
double xmax_meters, ymax_meters;
int **regen_delay_counter;

struct {
  int px,py;
  double cum_pi[7];
} sp_prop;

int SaplingCount(double x, double y);
int TreeCount(double x, double y);

int subplot_idx(species_t s);
int x_index(double x);
int y_index(double y);

void stand_init(TreeList*, hdata_t*, char*, int, char**);
subplot_t** init_subplot();
TreeList*** init_subplot_trees();
double stocking(subplot_t*,double,double);
```

```
  double stocking_sm(subplot_t*,double,double);
  double stocking(subplot_t*,double,double,bool);
  DFN_TREE_STAT(average_height);
  DFN_TREE_STAT(average_diam);
  DFN_TREE_STAT(stocking);
  DFN_PLOT_STAT(stocking);
  DFN_PLOT_STAT(stocking_sm);
  DFN_PLOT_STAT(ref_ba);
  DFN_PLOT_STAT(average_height);
  double ref_ba(double,double,double);
  double ref_ba(subplot_t*);
  double ref_ht(TreeList*);
  stage_t get_stage(int,int);
  stage_t get_stage(TreePtr);
  void calculate_stocking_grid(TreeList*);
  void compute_fixed_regen();
  void calculate_whole_plot();
  void calculate_delay();
  void regenerate_grid();
  void regenerate_gcs();
  void update_harv_ba( TreePtr );
  species_t RandomSpecies(int,int);
  species_t RandomSpecies(int,int,double,double,double,double,double);
  double PercentBA(species_t, int, int);
  void find_gaps(void);
  double triangle_area(double,double,double);
  double tree_distance_xy(TreePtr, TreePtr);
  double tree_distance_rz(TreePtr, TreePtr);
  double tree_angle(TreePtr, TreePtr);
  double calculate_cr_toward_competitor_xy(TreePtr, TreePtr);
  double calculate_cr_toward_competitor_rz(TreePtr, TreePtr);
  void check_proximity_to_competitors();
  bool is_tree_harvestable(TreeList::iterator);
  void process_subplot(subplot_t*, TreeList::iterator);
  void calculate_crown_status();
  void transform(double,double,TreePtr,coords_t&);
  void transform(TreePtr,TreePtr,coords_t&);
  void cwd_transition(TreeList*, TreeList*, double, double, status_t);
  void clean_treelist(TreeList*);
  void clean_treelist_rm(TreeList*);
  int number_of_saplings(int,int);
  int number_of_saplings(int,int,double,double,double,int);
  TreePtr make_baby(int,int,species_t);
  TreePtr make_baby(double,double,species_t);
  TreePtr make_baby(double,double,species_t, double, bool);
  void compute_circular_plot( TreeList*, double, double&);
  void compute_circular_plot( TreeList*, double,double&,double&,double&,double&,double&);
  int sapling_time_to_2cm(int,int);
  void clean_around_and_delay(TreePtr);
};

#undef DFN_PLOT_STAT
```

```
#undef DFN_TREE_STAT

#endif //__Stand_h__
```

D.6 StormModel.h

```
// -*- C++ -*-
#ifndef __StormModel_h__
#define __StormModel_h__

#include "CanopyGlobals.h"
#include "LuaHelper.h"


class StormModel : LuaHelper{
 public:
  StormModel(char*);
  int get_storm_specification(int);

  static double storm_eca_removal();
  static double severity_index(double);
  static double storm(double, TreeList*, TreeSet*);

};

#endif
```

D.7 Tree.h

```
//-*- C++ -*-
#ifndef __Tree_h__
#define __Tree_h__

class TreeDB;

#include <vector.h>
#include <math.h>
#include "CanopyGlobals.h"
#include "Random.h"
#include "TreeData.h"

//! Functions and data for dealing with individual trees
/**
 * Data contained:
 *  diamter, location, crown area
 * Functions contained:
 *  growth, mortality prob, windthrow prob, allometric size relations.
 */
class Tree {
  // Allow related classes to access data from this class:
  friend class Tester;
  friend class TreeGrid;
  friend class TreeData;
public:
  // Constructors:
  Tree();
  Tree(double x, double y, double d, species_t sp,
       double ncr, double ecr, double scr, double wcr, hdata_t*);
  Tree(int no, double x, double y, double d, species_t sp,
       double ncr, double ecr, double scr, double wcr, hdata_t*);
  Tree(int no, double x, double y, double d, species_t sp, hdata_t*);
```

```
Tree(double x, double y, double d, species_t sp, hdata_t*);
~Tree();
Tree& operator=(const Tree&);

void SetTreeData(long TreeNo, double xCoord, double yCoord,
                 species_t Species, short CC,
                 double DBH, double TotalHeight,
                 double BaseHeight,
                 double WidestHeight, double NorthTotal,
                 double NorthExp,
                 double EastTotal, double EastExp,
                 double SouthTotal,
                 double SouthExp, double WestTotal, double WestExp,
                 double Distance, double Azimuth, status_t Status,
                 double BasalArea, double PreviousBA,
                 long YearsOnTreeList);
long GetTreeno(void);
long GetYearsOnTreeList(void);
species_t GetSpecies(void);
double GetBA(void);
double GetDBH(void);
double GetRH(void);
double GetRD(void);
double GetMCR();
status_t GetStatus(void);

double get_cr(int);
double GetBaseHeight();
double GetWidestHeight();
double GetTotalHeight();
double GetXCoord(void);
double GetYCoord(void);

void SetRH(double);
void  SetStatus(status_t);

void SetTotalHeight(double);

double blowDown(double _si);

void set_dbh(double d);

double grow(double, double, stage_t);

void set_age(int);
void set_born_in_sap_stand();
void set_gap_area(double);
double get_gap_area();
void set_comp_sap_tca(double);

double get_comp_sap_tca();
double get_comp_sap_ci();

void set_shaded(int, bool);
void set_facing_gap(int, bool);
void set_touching(int, bool);
```

```
  TreePtr get_xy_neighbor(int);
  TreePtr get_rz_neighbor(int);
  TreePtr get_eca_neighbor(int);

  TreePtr get_gap_neighbor(int);
  void set_gap_neighbor(int, TreePtr);

  void set_xy_neighbor(int, TreePtr);
  void set_rz_neighbor(int, TreePtr);
  void set_eca_neighbor(int, TreePtr);

  void set_ecr(int,double);
  double get_ecr(int);

  double get_eca(void);
  double get_tca(void);

  void add_competitor_sapling(TreePtr, double);
  void clear_competitor_saplings();

  double pdie(double, double, stage_t, double);
  double pdie_old(double, stage_t);
  int get_age();
  int get_gap_age();

  int need_save(int);
  bool big_enough_to_save();
  void force_save();

private:
  typedef struct {
    double value;
    bool is_good;
  } cache_var_t;

  long   treeNo;
  double treeCoordX;
  double treeCoordZ;
  species_t  species;
  double dbh;
  double rel_ht, rel_diam;

  cache_var_t total_height;
  cache_var_t base_height;
  cache_var_t widest_height;
  cache_var_t eca;
  cache_var_t tca;
  cache_var_t comp_sap_tca;
  cache_var_t comp_sap_ci;

  htype_t habitat_type;

  double crown_radius[4];
  double exposed_crown_radius[4];

  status_t status;
```

```
  int age, gap_age;
  bool born_in_sap_stand;

  bool touching[4], shaded[4];
  int facing_gap[4];

  TreePtr xy_neighbor[4];
  TreePtr rz_neighbor[4];
  TreePtr eca_neighbor[4];

  TreePtr gap_neighbor[8];

  vector<TreePtr> competitor_saplings;
  vector<double>  competitor_distances;

  double gap_area;

  struct { double a,b; } storm;

  cache_var_t stochastic_lg_grow_modifier, stochastic_hg_modifier,
    stochastic_ht_dbh_modifier;

public:

  typedef struct {
    bool operator()( const TreePtr T1, const TreePtr T2) const {
      return ( T2->GetTotalHeight() < T1->GetTotalHeight() );
    }
  } height_comp_t;

 private:
  static int last_treeno;
  struct { int last; bool need_initial, need_final, need_forced; } save_status;

  static double pdie(species_t, double, double, double, stage_t);
  static double pdie_ht_complex(htype_t, species_t, double, double, double, stage_t);
  static double pdie_ht_simple(htype_t, species_t, double, double, double, stage_t);
  void setStormCoefs(species_t _species);
  void set(hdata_t*, int no, double x, double y, double d, species_t sp,
           double ncr, double ecr, double scr, double wcr );
  void clear_caches();
  double predict_mcr();
  void set_habitat(hdata_t*);
  double get_hgrow_modifier();
  double get_ht_dbh_modifier();
  bool is_facing_gap(int i);
  void total_height_equation_coefs(double*);
  void SetXCoord(double);
  void SetYCoord(double);
  void set_treeno(int);
  double grow_dbh(double, stage_t);
  double grow_dbh_new(double, stage_t);
  double grow_ht();
  void grow_branches(double,double);

};
```

```
#endif //__Tree_h__
```

D.8 TreeDB.h

```
// -*- C++ -*-
#ifndef __TreeDB_h__
#define __TreeDB_h__

#include <sqlite3.h>
#include "CanopyGlobals.h"

class Stand;

class TreeDB {
private:
  sqlite3 *db;
  void sql(const char*);
  void db_prepare(sqlite3_stmt** st, const char* cmd);
  void db_finalize(sqlite3_stmt* st);
  void db_step(sqlite3_stmt* st);
  void db_bind_double(sqlite3_stmt* st, int pos, double d);
  void db_bind_int(sqlite3_stmt* st, int pos, int d);
  void db_bind_text(sqlite3_stmt*st, int pos, char* d);

  struct {
    sqlite3_stmt *begin_tx;
    sqlite3_stmt *end_tx;
    sqlite3_stmt *trees_ins;
    sqlite3_stmt *size_ins;
    sqlite3_stmt *gap_ins;
    sqlite3_stmt *birth_ins;
    sqlite3_stmt *live_ins;
    sqlite3_stmt *cut_taken_ins;
    sqlite3_stmt *cut_left_ins;
    sqlite3_stmt *wind_ins;
    sqlite3_stmt *dead_ins;
    sqlite3_stmt *storm_ins;
    sqlite3_stmt *cwd_log_ins[5];
    sqlite3_stmt *cwd_snag_ins[5];
    sqlite3_stmt *cwd_out_ins;
    sqlite3_stmt *stocking_ins;
  } stmt;

public:
  void open(char*);
  void close();
  void begin_tx();
  void end_tx();
  void init(int,int);
  void insert(Stand*);
  void storm_insert(int, int, double);
  void argv_ins(int, char**);

private:
  bool debug_insert_check(int,int);
```

```
  void size_ins(TreePtr,int);
};

#endif // __TreeDB_h__
```

D.9 TreeData.h

```
// -*- c++ -*-=
/************************ CTreeData.h **************/

#ifndef __TreeData_h__
#define __TreeData_h__

#include "CanopyGlobals.h"

class TreeData {
private:
  short** flips;
  int flip_x, flip_y;

  void check_init_flips(int,int);
  static htype_t string_to_htype(char*);
  void read_spatial_t1(TreeList*, FILE*, int&, hdata_t*);
  void read_nonspatial_t1(TreeList*, FILE*, int&, hdata_t*);
  void read_nonspatial_t2(TreeList*, FILE*, int&, hdata_t*);
  void read_nonspatial_t3(TreeList*, FILE*, int&, hdata_t*, double, double);
  static hdata_t* ReadHabSpec(char*);
  hdata_t* GenerateHabitatSpec(char*, double, double);
  void multiply_ht(hdata_t*, int, int);
  void multiply_tl(TreeList*, hdata_t*, int, int);

public:
  void ReadTreeList(char*, TreeList*, hdata_t*&, int, int);
  TreeData(void);
  ~TreeData(void);
};

#endif
```

D.10 TreeGrid.h

```
// -*- c++ -*-
#ifndef __TreeGrid_h__
#define __TreeGrid_h__

#include "CanopyGlobals.h"
#include "Stand.h"

class TreeGrid{
public:
  TreeGrid(Stand*);
  void add_trees(TreeList*);
  void add_tree( TreePtr );
  TreePtr get_next_tree();
  void set_search_center(double,double);
  void remove_dead();

private:
```

```
  void perform_offset(int*, int);

  TreeList*** grid;
  Stand* st;

  TreeList* cur_list;
  TreeList::iterator cur_tree;
  int cur_n, cur_i, cur_j;
  double center_x, center_y;

};

#endif // __TreeGrid_h__
```

D.11 UTIL.H

```
#ifndef __util_h__
#define __util_h__

#ifdef OSX
#  ifndef isfinite
#    define isfinite(x) __isfinited(x)
#  endif
#endif // OSX

#define CHK_MEM(x)                                      \
  if (x==NULL){                                         \
    printf("Memory allocation failure at %s:%i\n", \
           __FILE__,__LINE__);                          \
    exit(1);                                            \
  }

#endif //__util_h__
```

APPENDIX E

CANOPY source code: LUA modules

E.1 DSI_CONST_REGEN.LUA

```
trees_to_plant = {}

function is_it_time(year) return true end
function is_it_inside(x,y) return true end
function cut_this_tree(tree) return 0 end

function prepare(treedata)
  local n_small = 0
  local scale = (100*100) / (xmax*ymax)

  for index,tree in ipairs(treedata) do
    if (tree.dbh <= 6 ) then
      n_small = n_small + scale
    end
  end

  if (n_small < tgt_pop ) then
    local to_add = math.ceil( (tgt_pop - n_small) / scale)

    for i=1,to_add do
      table.insert( trees_to_plant,
        {x=xmax*math.random(), y=ymax*math.random(), sp=4, dbh=2+4*math.random() } )
    end
  end
end
```

E.2 DSI_SMALL_Q.LUA

```
function is_it_time(t) return true end
function is_it_inside(x,y) return true end

do
  -- Strangely, Lua doesn't provide a math.round.
  -- Provide our own, with probabilistic rounding of numbers == 0.5
  local function round(x)
    local fp = x - math.floor(x)  -- fp: fractional part
    if (fp < 0.5 or (fp==0.5 and math.random()<0.5) ) then
```

```
    return math.floor(x)
  else
    return math.ceil(x)
  end
end

-- Size classes are 2-7, 7-12, 12-17, 17-22
local tgt_n = { 0, 0, 0, 0}
local act_n = { 0, 0, 0, 0}

function prepare(treedata)

  -- Clear the counts of current trees:
  for i=1,4 do act_n[i] = 0 end

  -- And re-compute the target distribution:

  -- Count the number of 17-22 cm trees
  -- This will be used as the basis for our residual distribution
  tgt_n[4] = 0
  for index,tree in ipairs(treedata) do
    if ( 17 <= tree.dbh and tree.dbh < 22 ) then
      tgt_n[4] = tgt_n[4] + 1
    end
  end

  -- Compute targets in other size classes based on 17-22
  for i=3,1,-1 do
    tgt_n[i] = tgt_q * tgt_n[i+1]
  end

  -- Now round to the nearest whole number of trees:
  for i=1,4 do
    tgt_n[i] = round(tgt_n[i])
  end
end

function cut_this_tree(tree)
  local rc=0

  if (tree.dbh > 17) then
    rc=0
  else
    local ix
    if     (tree.dbh < 7 ) then ix=1
    elseif (tree.dbh < 12) then ix=2
    else                        ix=3
    end

    if (act_n[ix] >= tgt_n[ix]) then
      rc = 1
    else
      act_n[ix] = act_n[ix] + 1
      rc = 0
    end
  end
```

```
    return rc
  end
end
```

APPENDIX F

CANOPY source code: post-processing

F.1 MK_STD_REPORT.PL

```
#!/usr/bin/perl

use POSIX       qw(ceil);
use List::Util qw(min);
use File::Glob ':glob';

sub get_cmd_name {
    my ($basename) = @_;

    my $cmd_name = $basename;
    $cmd_name =~ s/_(.)/\u\1/g ;
    $cmd_name =~ s/-(.)/\u\1/g ;
    $cmd_name =~ s/0/Zero/g;
    $cmd_name =~ s/1/One/g;
    $cmd_name =~ s/2/Two/g;
    $cmd_name =~ s/3/Three/g;
    $cmd_name =~ s/4/Four/g;
    $cmd_name =~ s/5/Five/g;
    $cmd_name =~ s/6/Six/g;
    $cmd_name =~ s/7/Seven/g;
    $cmd_name =~ s/8/Eight/g;
    $cmd_name =~ s/9/Nine/g;

    return $cmd_name;
}


sub begin_doc {
    print <<'END';
    \documentclass[english,10pt]{article}
    \usepackage[T1]{fontenc}
    \usepackage{babel}
    \usepackage{graphicx}
    \usepackage{fancyhdr}
    \usepackage{hyperref}
    \usepackage{longtable}
    \usepackage[landscape]{geometry}
```

```
    \geometry{verbose,tmargin=0.75in,bmargin=1in,lmargin=1in,rmargin=0.75in}
    \pagestyle{fancy}
    \setlength{\parindent}{0pt}
    \begin{document}
END

    my $cover_page_exists = -e 'cover_page.tex';
    if ( !$cover_page_exists ){
       open($fh, ">", "cover_page.tex")
    }

    for $sim (@sims) {
        my $basename = $sim;
        $basename =~ s/.r01.db//;
        chomp($basename);

        $cmd_name = get_cmd_name($basename);

        $body = $basename;
        $body =~ s/_/\\_/g;

        print "\\newcommand{\\$cmd_name}{$body}\n";
        if ( !$cover_page_exists ){
          print $fh "\\renewcommand{\\$cmd_name}{$body}\n";
        }
    }

    if ( !$cover_page_exists ){
        close($fh);
    }

    print `cat cover_page.tex`;
    print "\n\n\\vspace{3em}\n";
    print "Plots in this document which refer to 'canopy trees' " .
          "include trees >11 cm dbh with >= 20 \\% ECA\n";
    print "\\newpage\n";

    print <<'END'
        \tableofcontents
        \newpage
END
}

sub label {
  my ($title) = @_;
  print('\subsubsection{'.$title."}\n\n");
}

sub mk_fig {
    my ($fname, $width) = @_;
    -e $fname or  die "$fname does not exist";
    print '\includegraphics[width='.(($width)/100).'\textwidth]{'.$fname.'}' ;
}

sub mk_table {
    my ($fname, $is_rba) = @_;
    my @rows = ();

    my $sk;
```

```
my $label="";
if ($fname =~ /ddist/ ||
    $fname =~ /dbh/ ){
    $sk = 1;
    $label = "DBH";
} else {
    $sk=1;
    $label="Year";
}

open (FD,$fname) or die "could not open $fname";
my $year_offset = 0;
my $i=0;
while(<FD>){
    my ($n, $year,
        $x_max, $x_min, $x_bar, $x_sd,
       $y_max, $y_min, $y_bar, $y_sd,
        $z_max, $z_min, $z_bar, $z_sd,
        $a_max, $a_min, $a_bar, $a_sd,
        $b_max, $b_min, $b_bar, $b_sd,
        $c_max, $c_min, $c_bar, $c_sd
        ) = split(/,/);
    if( $i==1 and $year==10 ) { $year_offset = -10 ; }

    if( $i > 0 ) {
        $rows[$i-1] = [ $year + $year_offset,
                        $x_max, $x_min, $x_bar, $x_sd,
                        $y_max, $y_min, $y_bar, $y_sd,
                        $z_max, $z_min, $z_bar, $z_sd,
                        $a_max, $a_min, $a_bar, $a_sd,
                        $b_max, $b_min, $b_bar, $b_sd,
                        $c_max, $c_min, $c_bar, $c_sd ];
    }

    $i++;
}
close(FD);

sub table_cell {
    my ($text,$is_endr) = @_;

    print $text ;
    if ($is_endr){
        print "\\\\\n" ;
    } else {
        print ' & ';
    }
}

sub table_start{
    my ($local_cols, $local_rows) = @_;

    print '\begin{tabular}{ c | ';
    for $k ( 2 .. $local_cols ) { print "c"; }
    print "}\n\\hline\n";
}
```

```
    sub table_end { print '\end{tabular}'."\n\n" ; }

    my $cpT     = 18;
    my $ntables = ceil( $i/($sk*$cpT) );

    for $T ( 0 .. ($ntables-1) ) {
        my $E = min( $cpT, ceil($i/$sk) - $cpT*$T - 1);
        my $max_row = ( $is_rba eq 1 ) ? 24 : 4 ;

        table_start($E+1, $max_row + 1);

        for $j ( 0 .. $max_row ) {
            my $sp_label = "";
            if( $is_rba eq 1 ) {
                if    ($j <= 4 ){ $sp_label = " Sugar maple"; }
                elsif ($j <= 8 ){ $sp_label = " Hemlock"; }
                elsif ($j <= 12){ $sp_label = " Yellow birch"; }
                elsif ($j <= 16){ $sp_label = " Ash"; }
                elsif ($j <= 20){ $sp_label = " Red maple"; }
                elsif ($j <= 24){ $sp_label = " Basswood"; }
            }

            if    ( $j   == 0){ table_cell("$label", 0 ); }
            elsif (($j%4)== 1){ table_cell("Max${sp_label}",  0 ); }
            elsif (($j%4)== 2){ table_cell("Min${sp_label}",  0 ); }
            elsif (($j%4)== 3){ table_cell("Mean${sp_label}", 0 ); }
            elsif (($j%4)== 0){ table_cell("Sd${sp_label}",   0 ); }

            for $k ( 0 .. ($E-1) ){
                my $kn = $sk*($k + $cpT*$T);
                if( $j==0 ) {
                    table_cell(sprintf("%i",   $rows[$kn][$j]),$k==($E-1));
                } else {
                    $cell_number = $rows[$kn][$j];
                    if    ( $cell_number > 100) { $cell_text = sprintf("%0.0f",
                        $rows[$kn][$j]); }
                    elsif ( $cell_number > 10 ) { $cell_text = sprintf("%0.1f",
                        $rows[$kn][$j]); }
                    else                        { $cell_text = sprintf("%0.2f",
                        $rows[$kn][$j]); }

                    table_cell($cell_text,$k==($E-1)); }
                }
                if ($j==0) { print("\\hline\n"); }
            }
            table_end();
        }
        print("\\vspace{1em}\n\n");
    }

    sub mk_dbh_table {
        my ($fname) = @_;

        my $header_done=0;

        open (FD,$fname) or die "could not open $fname";

        print("\\begin{longtable}");
```

```
    while (<FD>){
        chomp;
        s/,/&/g;
        s/"//g;

        if ($header_done==0){
            my $offset = 0;
            my $result = index($_, '&', $offset);
            my $count = 0;

            while ($result != -1 ){
                $count++;
                $offset = $result+1;
                $result = index($_, '&', $offset);
            }
            $count++; # For last column

            print("{");
            for (my $i=0; $i<$count; $i++){
                print("c");
                if ($i==0){ print("|"); }
            }
            print("}\n");
            print("Year & \\multicolumn{".($count-1)."}{c}{Midpoint diameter class (cm)}
                \\\\\n");
            print;
            print("\\\\\n");
            print("\\hline\n");
            print("\\endfirsthead\n");
            print("Year & \\multicolumn{".($count-1)."}{c}{Midpoint diameter class (cm)}
                \\\\\n");
            print;
            print("\\\\\n");
            print("\\hline\n");
            print("\\endhead\n");
            $header_done=1;
        } else {
            print ;
            print("\\\\\n");
        }
    }
    print("\\end{longtable}\n\n");
    close(FD);
}

##### Document starts here ########
@sims = bsd_glob('*.r01.db');

begin_doc();

for $sim (@sims) {
  my $basename = $sim;
  $basename =~ s/.r01.db//;
  chomp($basename);

  $cmd_name = get_cmd_name($basename);
```

```
print "\\rhead{\\".$cmd_name."}\n\\section{\\".$cmd_name."}\n";

print("\\subsection{Snapshot DBH and ECA distributions, crown maps}\n");
@files = bsd_glob("${basename}_r01_[0-9][0-9][0-9][0-9]_dbh.png");
for $file (@files){
  my ($dbh_dist, $eca_dist, $stock_dist, $top_view, $dgr_plot) =
      ($file,$file,$file,$file,$file);
  $eca_dist =~ s/dbh/eca/;
  $stock_dist =~ s/dbh/stock/;
  $top_view  =~ s/dbh/top/;
  $dgr_plot  =~ s/dbh/dgr_vs_dbh/;

  mk_fig($top_view,"20");
  mk_fig($stock_dist,"20");
  mk_fig($dbh_dist,"20");
  mk_fig($eca_dist,"20");
  -e $dgr_plot && mk_fig($dgr_plot, "20");
  print("\n\n");
}

print("\\subsection{DBH distribution table (T/ha, avg of all reps)}\n");
print("\\tiny{\n");
mk_dbh_table("${basename}_dbh_sequence.csv");
print("}\n");

print("\\subsection{DBH, CR, and Age relations, beginning of rep 01}\n");
mk_fig("${basename}_r01_diam_vs_age_start.png","45");
mk_fig("${basename}_r01_cr_vs_age_start.png","45");
print("\n\n");
mk_fig("${basename}_r01_cr_vs_diam_start.png","45");
print("\n\n");


print("\\subsection{DBH, CR, and Age relations, end of rep 01}\n");
mk_fig("${basename}_r01_diam_vs_age.png","45");
mk_fig("${basename}_r01_cr_vs_age.png","45");
print("\n\n");
mk_fig("${basename}_r01_cr_vs_diam.png","45");
print("\n\n");

print("\\subsection{Ages of trees > 25 cm at time of death, rep 01}\n");
mk_fig("${basename}_r01_all_age_dist.png","45");
print("\n\n");
mk_fig("${basename}_r01_sm_age_dist.png","45");
mk_fig("${basename}_r01_hm_age_dist.png","45");
print("\n\n");
mk_fig("${basename}_r01_yb_age_dist.png","45");
mk_fig("${basename}_r01_wa_age_dist.png","45");
print("\n\n");
mk_fig("${basename}_r01_bw_age_dist.png","45");
mk_fig("${basename}_r01_rm_age_dist.png","45");

if (-e "${basename}_all_age_dist_over.png" ){
    mk_fig("${basename}_r01_all_age_dist_over.png","45");
}
```

```
print("\\subsection{Stand-level summaries, averaged across replicates}\n");

label("Stocking (\\%)");
mk_fig("${basename}_stock.png","45"); print("\n\n");
mk_table("${basename}_stock.csv");

label("Density (T/Ha)");
mk_fig("${basename}_all_trees.png","45"); print("\n\n");
mk_table("${basename}_all_trees.csv");

label("Basal Area (\$m^2/Ha\$)");
mk_fig("${basename}_ba.png","45"); print("\n\n");
mk_table("${basename}_ba.csv");

if ( -e "${basename}_ba_over.csv"){
     label("Initial Cohort Basal Area (\$m^2/Ha\$)");
     mk_fig("${basename}_ba_over.png","45"); print("\n\n");
     mk_table("${basename}_ba_over.csv");
}

label("Standing Live Volume (\$m^3/ha\$)");
mk_fig("${basename}_vol.png","45"); print("\n\n");
mk_table("${basename}_vol.csv");

if ( -e "${basename}_vol_over.csv"){
     label("Initial Cohort Standing Live Volume (\$m^3/ha\$)");
     mk_fig("${basename}_vol_over.png","45"); print("\n\n");
     mk_table("${basename}_vol_over.csv");
}

label("Standing Live Biomass (\$mg/ha\$)");
mk_fig("${basename}_mg.png","45"); print("\n\n");
mk_table("${basename}_mg.csv");

if ( -e "${basename}_mg_over.csv" ){
     label("Initial Cohort Standing Live Biomass (\$mg/ha\$)");
     mk_fig("${basename}_mg_over.png","45"); print("\n\n");
     mk_table("${basename}_mg_over.csv");
}

label("Ending DBH distribution (Trees/Ha)");
mk_fig("${basename}_dbh.png",    "45");
mk_fig("${basename}_log_dbh.png","45");
print("\n\n");
mk_table("${basename}_dbh.csv");

label("QMSD");
mk_fig("${basename}_qmsd.png", "45"); print("\n\n");
mk_table("${basename}_qmsd.csv");

label("Canopy tree mean diameter");
mk_fig("${basename}_ct_dbar.png","45"); print("\n\n");
mk_table("${basename}_ct_dbar.csv");

label("Canopy tree mean height");
mk_fig("${basename}_ct_hbar.png","45"); print("\n\n");
mk_table("${basename}_ct_hbar.csv");
```

```
label("Canopy tree mean MCR");
mk_fig("${basename}_ct_mcrbar.png","45"); print("\n\n");
mk_table("${basename}_ct_mcrbar.csv");

label("Canopy tree mean age");
mk_fig("${basename}_ct_agebar.png","45"); print("\n\n");
mk_table("${basename}_ct_agebar.csv");

if ( -e "${basename}_ctg.csv" ) {
label("Canopy Tree Mean Radial Increment (mm/yr)");
mk_fig("${basename}_ctg.png", "45"); print("\n\n");
mk_table("${basename}_ctg.csv");
}

label("Canopy Tree Mean Radial Crown Increment (cm/yr)");
mk_fig("${basename}_ctcg.png","45"); print("\n\n");
mk_table("${basename}_ctcg.csv");

label("Harvested Volume (\$m^3/ha\$)");
mk_fig("${basename}_harv.png","45"); print("\n\n");
mk_table("${basename}_harv.csv");

label("Releative basal area per species (\\%)");
mk_fig("${basename}_rba.png","45"); print("\n\n");
mk_table("${basename}_rba.csv", 1);

label("ECA as a percentage of plot area");
mk_fig("${basename}_peca.png","45"); print("\n\n");
mk_table("${basename}_peca.csv");

label("TCA as a percentage of plot area");
mk_fig("${basename}_ptca.png","45"); print("\n\n");
mk_table("${basename}_ptca.csv");

label("Average Developmental Stage");
mk_fig("${basename}_st_stage.png","45"); print("\n\n");
mk_table("${basename}_st_stage.csv");

label ("Developmental stages per replication");
print `cat ${basename}_stage_data.tex`;

label ("Residence times for each stage and time to OG");
print `cat ${basename}_residence_times.tex`;

print("\\subsection{Growth by component (ingrowth, etc)}\n");

# BA growth (>4.6 in)
label("BA ingrowth across a 4.6in (11.68cm) threshold (\$m^2/ha/yr\$)");
mk_fig("${basename}_ig.png","45");  print("\n\n");
mk_table("${basename}_ig.csv");

label("BA survivor growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_sg.png","45"); print("\n\n");
mk_table("${basename}_sg.csv");

label("BA mortality above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_mt.png","45"); print("\n\n");
```

```
mk_table("${basename}_mt.csv");

label("BA gross growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_gg.png","45"); print("\n\n");
mk_table("${basename}_gg.csv");

label("BA net growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_ng.png",  "45");  print("\n\n");
mk_table("${basename}_ng.csv");

if (-e "${basename}_s_ig.csv"){
# BA growth (> 9.6 in)
label("BA ingrowth across a 9.6in (24.38cm) threshold (\$m^2/ha/yr\$)");
mk_fig("${basename}_s_ig.png","45");  print("\n\n");
mk_table("${basename}_s_ig.csv");

label("BA survivor Growth above 9.6in (24.38cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_s_sg.png","45"); print("\n\n");
mk_table("${basename}_s_sg.csv");

label("BA mortality above 9.6in (24.38cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_s_mt.png","45");  print("\n\n");
mk_table("${basename}_s_mt.csv");

label("BA gross growth above 9.6in (24.38cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_s_gg.png","45"); print("\n\n");
mk_table("${basename}_s_gg.csv");

label("BA net growth above 9.6in (24.38cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_s_ng.png","45");  print("\n\n");
mk_table("${basename}_s_ng.csv");
}

# Volume growth (>4.6 in)
label("Volume ingrowth across a 4.6in (11.68cm) threshold (\$m^3/ha/yr\$)");
mk_fig("${basename}_m3_ig.png","45");  print("\n\n");
mk_table("${basename}_m3_ig.csv");

label("Volume survivor growth across a 4.6in (11.68cm) threshold (\$m^3/ha/yr\$)");
mk_fig("${basename}_m3_sg.png","45"); print("\n\n");
mk_table("${basename}_m3_sg.csv");

label("Volume mortality across a 4.6in (11.68cm) threshold (\$m^3/ha/yr\$)");
mk_fig("${basename}_m3_mt.png","45"); print("\n\n");
mk_table("${basename}_m3_mt.csv");

label("Volume gross growth across a 4.6in (11.68cm) threshold (\$m^3/ha/yr\$)");
mk_fig("${basename}_m3_gg.png","45"); print("\n\n");
mk_table("${basename}_m3_gg.csv");

label("Volume net growth across a 4.6in (11.68cm) threshold (\$m^3/ha/yr\$)");
mk_fig("${basename}_m3_ng.png","45" );  print("\n\n");
mk_table("${basename}_m3_ng.csv");

if (-e "${basename}_kg_all_ig.csv"){
# Biomass growth (>3 cm)
label("Biomass ingrowth across a 3 cm threshold (\$kg/ha/yr\$)");
```

```
mk_fig("${basename}_kg_all_ig.png","45");  print("\n\n");
mk_table("${basename}_kg_all_ig.csv");

label("Biomass survivor growth across a 3 cm threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_all_sg.png","45"); print("\n\n");
mk_table("${basename}_kg_all_sg.csv");

label("Biomass mortality across a 3 cm threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_all_mt.png","45"); print("\n\n");
mk_table("${basename}_kg_all_mt.csv");

label("Biomass gross growth across a 3 cm threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_all_gg.png","45" );  print("\n\n");
mk_table("${basename}_kg_all_gg.csv");

label("Biomass net growth across a 3 cm threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_all_ng.png","45" );  print("\n\n");
mk_table("${basename}_kg_all_ng.csv");
}

if (-e "${basename}_kg_ig.csv"){
# Biomass growth (>4.6in)
label("Biomass ingrowth across a 4.6in (11.68cm) threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_ig.png","45");  print("\n\n");
mk_table("${basename}_kg_ig.csv");

label("Biomass survivor growth across a 4.6in (11.68cm) threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_sg.png","45"); print("\n\n");
mk_table("${basename}_kg_sg.csv");

label("Biomass mortality across a 4.6in (11.68cm) threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_mt.png","45"); print("\n\n");
mk_table("${basename}_kg_mt.csv");

label("Biomass gross growth across a 4.6in (11.68cm) threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_gg.png","45" );  print("\n\n");
mk_table("${basename}_kg_gg.csv");

label("Biomass net growth across a 4.6in (11.68cm) threshold (\$kg/ha/yr\$)");
mk_fig("${basename}_kg_ng.png","45" );  print("\n\n");
mk_table("${basename}_kg_ng.csv");
}

## Over/under analysis:
if (-e "${basename}_over_ig.csv" ){
   ## Overstory BA growth:
   label("Initial cohort BA ingrowth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
   mk_fig("${basename}_over_ig.png",  "45");  print("\n\n");
   mk_table("${basename}_over_ig.csv");

   label("Initial Cohort BA survivor growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
   mk_fig("${basename}_over_sg.png",  "45");  print("\n\n");
   mk_table("${basename}_over_sg.csv");

   label("Initial Cohort BA mortality above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
   mk_fig("${basename}_over_mt.png",  "45");  print("\n\n");
   mk_table("${basename}_over_mt.csv");
```

```
label("Initial Cohort BA net growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_over_ng.png",  "45");  print("\n\n");
mk_table("${basename}_over_ng.csv");

## Understory BA growth:
label("Later cohorts BA ingrowth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_under_ig.png",  "45");  print("\n\n");
mk_table("${basename}_under_ig.csv");

label("Later cohorts BA survivor growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_under_sg.png",  "45");  print("\n\n");
mk_table("${basename}_under_sg.csv");

label("Later cohorts BA mortality above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_under_mt.png",  "45");  print("\n\n");
mk_table("${basename}_under_mt.csv");

label("Later cohorts BA net growth above 4.6in (11.68cm) (\$m^2/ha/yr\$)");
mk_fig("${basename}_under_ng.png",  "45");  print("\n\n");
mk_table("${basename}_under_ng.csv");


## Overstory Volume growth:
label("Initial cohort volume ingrowth above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_over_ig_m3.png",  "45");  print("\n\n");
mk_table("${basename}_over_ig_m3.csv");

label("Initial cohort volume survivor growth above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_over_sg_m3.png",  "45");  print("\n\n");
mk_table("${basename}_over_sg_m3.csv");

label("Initial cohort volume mortality above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_over_mt_m3.png",  "45");  print("\n\n");
mk_table("${basename}_over_mt_m3.csv");

label("Initial cohort volume net growth above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_over_ng_m3.png",  "45");  print("\n\n");
mk_table("${basename}_over_ng_m3.csv");

## Understory Volume growth:
label("Later cohorts volume ingrowth above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_under_ig_m3.png",  "45");  print("\n\n");
mk_table("${basename}_under_ig_m3.csv");

label("Later cohorts volume survivor growth above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_under_sg_m3.png",  "45");  print("\n\n");
mk_table("${basename}_under_sg_m3.csv");

label("Later cohorts volume mortality above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_under_mt_m3.png",  "45");  print("\n\n");
mk_table("${basename}_under_mt_m3.csv");

label("Later cohorts volume net growth above 4.6in (11.68cm) (\$m^3/ha/yr\$)");
mk_fig("${basename}_under_ng_m3.png",  "45");  print("\n\n");
mk_table("${basename}_under_ng_m3.csv");


## Overstory Biomass growth:
```

```
    label("Initial cohort biomass ingrowth above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_over_ig_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_over_ig_kg.csv");

    label("Initial cohort biomass survivor growth above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_over_sg_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_over_sg_kg.csv");

    label("Initial cohort biomass mortality above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_over_mt_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_over_mt_kg.csv");

    label("Initial cohort biomass net growth above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_over_ng_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_over_ng_kg.csv");

  ## Understory Biomass growth:
    label("Later cohorts biomass ingrowth above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_under_ig_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_under_ig_kg.csv");

    label("Later cohorts biomass survivor growth above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_under_sg_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_under_sg_kg.csv");

    label("Later cohorts biomass mortality above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_under_mt_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_under_mt_kg.csv");

    label("Later cohorts biomass net growth above 4.6in (11.68cm) (\$kg/ha/yr\$)");
    mk_fig("${basename}_under_ng_kg.png",  "45");  print("\n\n");
    mk_table("${basename}_under_ng_kg.csv");
}

## CWD analysis:
if ( -e "${basename}_cwd_all.csv") {
  print("\\subsection{CWD Summary}\n");

  label("CWD Overall Volume (\$m^3/Ha\$)");
  mk_fig("${basename}_cwd_all.png",       "45" );   print("\n\n");
  mk_table("${basename}_cwd_all.csv");

  label("CWD Log Volume (\$m^3/Ha\$)");
  mk_fig("${basename}_cwd_logs.png",  "45");  print("\n\n");
  mk_table("${basename}_cwd_logs.csv");

  label("CWD Snag Volume (\$m^3/Ha\$)");
  mk_fig("${basename}_cwd_snags.png", "45");  print("\n\n");
  mk_table("${basename}_cwd_snags.csv");

  label("CWD Overall Diameter distribution (Trees/Ha), end of simulations");
  mk_fig("${basename}_cwd_ddist_all.png", "45" );  print("\n\n");
  mk_table("${basename}_cwd_ddist_all.csv");

  label("Diameter distribution of decayed logs (logs/Ha), end of simulations");
  mk_fig("${basename}_cwd_ddist_logs.png", "45");  print("\n\n");
  mk_table("${basename}_cwd_ddist_logs.csv");
```

```
    label("Diameter distribution of snags (snags/Ha), end of simulations");
    mk_fig("${basename}_cwd_ddist_snags.png","45");  print("\n\n");
    mk_table("${basename}_cwd_ddist_snags.csv");
  }

  print("\\subsection{BA,ECA,etc summary by size category}\n");

  if (-e $basename . "_ba_summary.tex") {
      label ("Per-size BA summary (m2/ha)");
      print `cat ${basename}_ba_summary.tex`;
  }

  if (-e $basename . "_ba_summary.tex") {
      label ("Per-size relative BA summary");
      print `cat ${basename}_rel_ba_summary.tex`;
  }

  if (-e $basename . "_eca_summary.tex") {
     label ("Per-size ECA summary (m2/ha)");
     print `cat ${basename}_eca_summary.tex`;
 }

  if (-e $basename . "_eca_summary.tex") {
      label ("Per-size relative ECA summary");
      print `cat ${basename}_rel_eca_summary.tex`;
  }

  @var_suffix = ("", "_ba", "_pba", "_eca","_peca","_rg","_mt");
  @var_name = ("Density (T/ha)", "Basal area (\$m^2/ha\$)",
               "Percent basal area (\\% of total BA)",
               "Exposed crown area (\$m^2/ha\$)",
               "Percent exposed crown area (\\% of total ECA)",
               "Mean radial growth (mm/yr)",
               "BA mortality (m2/yr)");

  for ($i=0; $i< scalar @var_suffix; $i++){
    for $sc ("seed", "sap", "pole", "mat", "large", "xlarge", "g50"){
      my $desc="";
      if    ( $sc eq "seed"   ) { $desc="Seedlings (0-6 cm)"; }
      elsif( $sc eq "sap"    ) { $desc="Saplings (0-11 cm)"; }
      elsif( $sc eq "pole"   ) { $desc="Poles (11-26 cm)";   }
      elsif( $sc eq "mat"    ) { $desc="Mature (26-46 cm)";  }
      elsif( $sc eq "large"  ) { $desc="Large (46+ cm)";     }
      elsif( $sc eq "xlarge" ) { $desc="X-Large (66+ cm)";   }
      elsif( $sc eq "g50"    ) { $desc="Large (>50 cm)";     }

      if ( -e  $basename . "_" . $sc . $var_suffix[$i] .".csv" ){
      label($var_name[$i]." : ".$desc);
      mk_fig($basename . "_" . $sc . $var_suffix[$i] .".png",  "45");
      print("\n\n");
      mk_table($basename . "_" . $sc . $var_suffix[$i] .".csv");
      }
    }
  }
  print("\\newpage\n");
}

print '\end{document}'."\n";
```

F.2 MK_STD_3D.PL

```
#!/usr/bin/perl

use File::Glob ':glob';
use File::stat;

if( $^O =~ /MSWin32/ ){
  $ENV{"PATH"} .= ";C:\\Program Files\\POV-Ray for Windows v3.6\\bin";
  $frame="perl frame.pl";
  $povray="pvengine.exe /EXIT";
} else {
  $frame="~/canopy/util/frame.pl";
  $povray="povray36 -DO";
}

# Get a list of all the dbh dists we have.
# Make top-view crown maps for each.
@files = bsd_glob('*_r01_*_dbh.png');

for $file (@files) {
  $file =~ /.*_r01_([0-9]+)_dbh\.png/;
  $yr = $1;

  $dbfile = $file;
  $dbfile =~ s/_[0-9]+_dbh\.png//;
  $dbfile =~ s/_r01/.r01/;
  $dbfile .= ".db";

  $outfile = $file; $outfile =~ s/dbh/top/;
  print("$outfile\n");
  if( ! stat($outfile) ) {
    `$frame $dbfile $yr 0.001 90 1.5 > $$.pov`;
    `$povray +W800 +H800 +Q11 +UL +UV +I$$.pov +O$outfile`;
    unlink("$$.pov");
  }
}
```

F.3 FRAME.PL

```
#!/usr/bin/perl
use DBI;
$db=DBI->connect("dbi:SQLite:dbname=$ARGV[0]","","");

if( $#ARGV==6 ){
$sel=$db->prepare("select treeno,x,y,species,year,n,e,s,w,dbh,ht,hbc,hwc " .
                  "from live_trees where year==$ARGV[1] and " .
                  " $ARGV[5] <= dbh and dbh < $ARGV[6]");

} else {
$sel=$db->prepare("select treeno,x,y,species,year,n,e,s,w,dbh,ht,hbc,hwc " .
                  "from live_trees where year==$ARGV[1]");
}
$sel->execute();

print("#include \"/home/crhalpin/canopy/util/tree.ini\"\n");
while( ($treeno,$x,$z,$species,$year,$north_total,$east_total,
        $south_total,$west_total,$dbh,$total_height,$base_height,
        $widest_height)=
          $sel->fetchrow_array){
```

```
    if   ($species == 2  ) { $col="Blue";        } # Basswood
    elsif($species == 5  ) { $col="Med_Purple"; } # Hemlock
    elsif($species == 6  ) { $col="Yellow";      } # Yellow Birch
    elsif($species == 7  ) { $col="Black";       } # White Pine
    elsif($species == 9  ) { $col="Red";         } # Red Maple
    elsif($species == 13 ) { $col="Orange";      } # White Ash
    elsif($species == 14 ) { $col="Orange";      } # Green ash
    elsif($species == 24 ) { $col="Cyan";        } # Ironwood
    elsif($species ==  4 ) { $col="Green";       } # Sugar maple
    else                   { $col="Grey";        } # Other

    if($base_height   ==0) { $base_height   = 0.50*$total_height; }
    if($widest_height ==0) { $widest_height = 0.75*$total_height; }

    if($species == 7 || $species == 5 ){ print("treept"); }
    else                               { print("tree");   }

    print("($x,$z,$dbh/100,$total_height,$base_height,$widest_height,");
    print("$north_total,$east_total,$south_total,$west_total, $col)\n");
}

$sel=$db->prepare("select * from plotinfo");
$sel->execute();
($xmax,$ymax)=$sel->fetchrow_array; $sel->fetchrow_array;

if($#ARGV==4 || $#ARGV==6 ){
    $cx=($xmax/2)+$xmax*$ARGV[2]*cos(3.14159*$ARGV[3]/180);
    $cy=$xmax*$ARGV[4];
    $cz=($ymax/2)+$ymax*$ARGV[2]*sin(3.14159*$ARGV[3]/180);
} else {
    ($cx,$cy,$cz)=($xmax/2, $dist, $xmax);
}

print("light_source { <$xmax/2,300,$ymax/2> color White shadowless}\n");
print("camera { right 1*x location <$cx,$cy,$cz> look_at <$xmax/2,10,$ymax/2> }\n");
print("cylinder { <0,0,0>, <$xmax,0,0>,0.1 pigment { color Red } }\n");
print("cylinder { <0,0,0>, <0,0,$ymax>,0.1 pigment { color Red } }\n");


$db->disconnect();
```

F.4 TREE.INI

```
#include "colors.inc"    // The include files contain
#include "shapes.inc"    // pre-defined scene elements

#macro tree(xc,zc,dbh,top,bot,mid,ni,ei,si,wi,col)
  #local U=max(0.001,top-mid);
  #local L=max(0.001,mid-bot);
  #local n=max(dbh,ni);
  #local s=max(dbh,si);
  #local e=max(dbh,ei);
  #local w=max(dbh,wi);
  cylinder { <xc,0,zc>, <xc,mid,zc>, dbh/2 pigment {color Brown} }
  union {
    intersection{ sphere{ <0,0,0>,1 scale <e,U,n> } box{<0,0,0>,< e, U, n>}}
    intersection{ sphere{ <0,0,0>,1 scale <e,L,n> } box{<0,0,0>,< e,-L, n>}}
```

```
    intersection{ sphere{ <0,0,0>,1 scale <e,U,s> } box{<0,0,0>,< e, U,-s>}}
    intersection{ sphere{ <0,0,0>,1 scale <e,L,s> } box{<0,0,0>,< e,-L,-s>}}

    intersection{ sphere{ <0,0,0>,1 scale <w,U,s> } box{<0,0,0>,<-w, U,-s>}}
    intersection{ sphere{ <0,0,0>,1 scale <w,L,s> } box{<0,0,0>,<-w,-L,-s>}}

    intersection{ sphere{ <0,0,0>,1 scale <w,U,n> } box{<0,0,0>,<-w, U, n>}}
    intersection{ sphere{ <0,0,0>,1 scale <w,L,n> } box{<0,0,0>,<-w,-L, n>}}
    translate <xc, mid, zc>
    pigment { color col }
  }
#end

#macro treept(xc,zc,dbh,top,bot,mid,ni,ei,si,wi,col)
  #local ch=max(0.002,top-bot);  //height
  #local cm=max(0.001,mid-bot);  //height to the middle
  #local cr=ch/2;
  #local n=max(dbh,ni);
  #local s=max(dbh,si);
  #local e=max(dbh,ei);
  #local w=max(dbh,wi);
  cylinder { <xc,0,zc>, <xc,mid,zc>, dbh/2 pigment {color Brown} }
  union {
    intersection { //ne
      union {
        cone{<0,00,0>,00,<0,cm,0>,cr}
        cone{<0,cm,0>,cr,<0,ch,0>,00}
      }
      box{<0,0,0>,<+cr,ch,+cr>}
      scale <e/cr,1,n/cr>
    }
    intersection { //se
      union {
        cone{<0,00,0>,00,<0,cm,0>,cr}
        cone{<0,cm,0>,cr,<0,ch,0>,00}
      }
      box{<0,0,0>,<+cr,ch,-cr>}
      scale <e/cr,1,s/cr>
    }
    intersection { //nw
      union {
        cone{<0,00,0>,00,<0,cm,0>,cr}
        cone{<0,cm,0>,cr,<0,ch,0>,00}
      }
      box{<0,0,0>,<-cr,ch,+cr>}
      scale <w/cr,1,n/cr>
    }
    intersection { //sw
      union {
        cone{<0,00,0>,00,<0,cm,0>,cr}
        cone{<0,cm,0>,cr,<0,ch,0>,00}
      }
      box{<0,0,0>,<-cr,ch,-cr>}
      scale <w/cr,1,s/cr>
    }
  pigment { color col }
```

```
  translate <xc, bot, zc>
  }
#end

plane { y, 0 pigment { color White } finish {ambient 1} }
sky_sphere {
  pigment {
    gradient y
    color_map {
      [0 color Red]
      [1 color Blue]
    }
    scale 2
    translate -1
  }
}
```

F.5 STANDARD_ANALYSIS.R

```
# standard_analysis.r
#   Routines to analyze a batch of simulations.

##################################

source("~/canopy/util/util.r");
library(xtable)

plot_sim= function(base, label, n, all_details=T){
  clear_caches()

  limit_lines = function(bottom, top,managed=F){
    if( managed ){ pc = 1 } else { pc = 3 } # 1 = circle, 3 = x
    lines(c(0,max_year), rep(bottom,2), type="b", pch=pc, lwd=2);
    lines(c(0,max_year), rep(top,2),    type="b", pch=pc, lwd=2);
  }

  plot_save( base, 'stock', function(){
    plot_job(base, stock_data, 300, 'Stocking (%)',n) } )

  plot_save( base, 'ba', function(){
    plot_job(base,ba_data,70,'BA (m2/ha)',n) })

  plot_save( base, 'peca', function(){
    plot_job(base,peca_data, 125, 'ECA as a percentage of plot area',n) })

  plot_save( base, 'ptca', function(){
    plot_job(base, ptca_data, 225, 'TCA as a percentage of plot area',n) })

  plot_save( base, 'ct_dbar', function(){
    plot_job(base, canopy_tree_mean_diam, 50, 'Canopy tree mean diameter',n) })

  plot_save( base, 'ct_hbar', function(){
    plot_job(base, canopy_tree_mean_th, 25, 'Canopy tree mean height',n) })

  plot_save( base, 'ct_agebar', function(){
    plot_job(base, canopy_tree_mean_age, 200, 'Canopy tree mean age',n) })

  plot_save( base, 'ct_mcrbar', function(){
    plot_job(base, canopy_tree_mean_mcr, 5, 'Canopy tree mean MCR',n) })
```

```
plot_save( base, 'qmsd', function(){
  plot_job(base, qm_stand_diameter, 50, 'QM stand diameter',n) })

## BA Growth summaries for trees >4.6in
plot_save( base, 'ig', function(){
  plot_job(base,ingrowth,0.5,'Ingrowth >= 11.68cm (m2/ha)',n) })
plot_save( base, 'sg', function(){
  plot_job(base, survivor_growth, 1.5, 'Survivor growth >= 11.68 cm (m2/ha)', n ) })
plot_save( base, 'mt', function(){
  plot_job(base, mortality,3,'Mortality >= 11.68 cm (m2/ha)',n) })
plot_save( base, 'ng', function(){
  plot_job(base, net_growth,c(-3.0, 3.0), 'Net growth >= 11.68 cm (m2/ha)', n) })
plot_save( base, 'gg', function(){
  plot_job(base, gross_growth, 4, 'Gross growth >= 11.68 cm (m2/ha)', n) })

if (all_details){
## BA Growth summaries for trees >9.6in
plot_save( base, 's_ig', function(){
  plot_job(base,ingrowth_saw,0.5,'Ingrowth >= 24.38 cm (m2/ha)',n) })
plot_save( base, 's_sg', function(){
  plot_job(base, survivor_growth_saw, 1.5, 'Survivor hrowth >= 24.38 cm (m2/ha)', n ) })
plot_save( base, 's_mt', function(){
  plot_job(base, mortality_saw,1.5,'Mortality >= 24.38 cm (m2/ha)',n)  })
plot_save( base, 's_ng', function(){
  plot_job(base, net_growth_saw, c(-1.5,1.5), 'Net Growth >= 24.38 cm (m2/ha)', n) })
plot_save( base, 's_gg', function(){
  plot_job(base, gross_growth_saw, 1.5, 'Gross Growth >= 24.38 cm (m2/ha)', n) })
}

## Volume growth summaries for trees >4.6in
plot_save(base, 'm3_ig', function(){
  plot_job(base, ingrowth_m3, 2, 'Ingrowth >= 11.68cm (m3/ha)', n) })
plot_save(base, 'm3_sg', function(){
  plot_job(base, survivor_growth_m3, 8, 'Survivor growth >= 11.68cm (m3/ha)', n) })
plot_save(base, 'm3_mt', function(){
  plot_job(base, mortality_m3, 8, 'Mortality >= 11.68cm (m3/ha)', n) })
plot_save(base, 'm3_ng', function(){
  plot_job(base, net_growth_m3, c(-5,5), 'Net growth >= 11.68cm (m3/ha)', n) })
plot_save(base, 'm3_gg', function(){
  plot_job(base, gross_growth_m3, 8, 'Gross growth >= 11.68cm (m3/ha)', n) })


## Cohort analysis for clearcut simulations.
if (grepl('ccut|ctl|empty', base) ) {
  if (grepl('ccut|empty',base)){
    is_over <<- function(d){
      tn_good = sql('select distinct treeno as tn from live_trees where year<=20')$tn
      d$treeno %in% tn_good
    }

    cohort_over <<- function(gtype_fcn, dlim, metric_fcn){
      gtype_fcn(
        dlim,
        metric_fcn,
        function(d){
```

```
        tn_good = sql('select distinct treeno as tn from live_trees where
            year<=20')$tn
        d[d$treeno %in% tn_good,]
      } )
  }

  cohort_under <<- function(gtype_fcn, dlim, metric_fcn){
    gtype_fcn(
      dlim,
      metric_fcn,
      function(d){
        tn_bad = sql('select distinct treeno as tn from live_trees where year<=20')$tn
        d[ !(d$treeno %in% tn_bad), ]
      } )
  }

} else {
  ## Use RD from year=5 because we're not setting/saving rd in year=0.
  ##  it gets saved when a tree grows.  But, we kind of need a complete census,
  ##  which happens every 5th year.
  is_over <<- function(d){
    tn_good = sql('select distinct treeno as tn from live_trees where year=5 and
        reld>=0.7')$tn
    d$treeno %in% tn_good
  }

  cohort_over <<- function(gtype_fcn, dlim, metric_fcn){
    gtype_fcn(
      dlim,
      metric_fcn,
      function(d){
        tn_good = sql('select distinct treeno as tn from live_trees where year=5 and
            reld>=0.7')$tn
        d[d$treeno %in% tn_good,]
      } )
  }

  cohort_under <<- function(gtype_fcn, dlim, metric_fcn){
    gtype_fcn(
      dlim,
      metric_fcn,
      function(d){
        tn_bad = sql('select distinct treeno as tn from live_trees where year=5 and
            reld>=0.7')$tn
        d[ !(d$treeno %in% tn_bad), ]
      } )
  }
}

plot_save( base, 'ba_over', function(){
  plot_job(base,ba_over,70,'Initial cohort BA (m2/ha)',n) })

plot_save(base, 'vol_over', function(){
  plot_job(base,volume_over,700, 'Initial cohort volume (m3/ha)', n ); })

plot_save(base, 'mg_over', function(){
  plot_job(base,biomass_over,500, 'Initial cohort biomass (Mg/ha)', n ); })
```

```
plot_save(base, 'over_ig', function(){
  plot_job(base, ig_over_cohort, c(0,4), 'Initial cohort ingrowth >= 11.68cm
      (m2/ha)',n)})
plot_save(base, 'over_sg', function(){
  plot_job(base, sg_over_cohort, c(0,4), 'Initial cohort survivor growth >= 11.68 cm
      (m2/ha)',n)})
plot_save(base, 'over_mt', function(){
  plot_job(base, mt_over_cohort, c(0,4), 'Initial cohort mortality >= 11.68 cm
      (m2/ha)',n)})
plot_save(base, 'over_ng', function(){
  plot_job(base, ng_over_cohort, c(-3,3), 'Initial cohort net growth >= 11.68cm
      (m2/ha)',n)})

plot_save(base, 'under_ig', function(){
  plot_job(base, ig_under_cohort, c(0,4), 'Later cohorts ingrowth >= 11.68 cm
      (m2/ha)',n)})
plot_save(base, 'under_sg', function(){
  plot_job(base, sg_under_cohort, c(0,4), 'Later cohorts survivor growth >= 11.68 cm
      (m2/ha)',n)})
plot_save(base, 'under_mt', function(){
  plot_job(base, mt_under_cohort, c(0,4), 'Later cohorts mortality >= 11.68 cm
      (m2/ha)',n)})
plot_save(base, 'under_ng', function(){
  plot_job(base, ng_under_cohort, c(-3,3), 'Later cohorts net growth >= 11.68cm
      (m2/ha)',n)})

plot_save(base, 'over_ig_kg', function(){
  plot_job(base, ig_kg_over_cohort, c(0,4000), 'Initial cohort ingrowth >= 11.68cm
      (kg/ha)',n)})
plot_save(base, 'over_sg_kg', function(){
  plot_job(base, sg_kg_over_cohort, c(0,4000), 'Initial cohort survivor growth >=
      11.68 cm (kg/ha)',n)})
plot_save(base, 'over_mt_kg', function(){
  plot_job(base, mt_kg_over_cohort, c(0,4000), 'Initial cohort mortality >= 11.68 cm
      (kg/ha)',n)})
plot_save(base, 'over_ng_kg', function(){
  plot_job(base, ng_kg_over_cohort, c(-3000,3000), 'Initial cohort net growth >=
      11.68cm (kg/ha)',n)})

plot_save(base, 'under_ig_kg', function(){
  plot_job(base, ig_kg_under_cohort, c(0,4000), 'Later cohorts ingrowth >= 11.68 cm
      (kg/ha)',n)})
plot_save(base, 'under_sg_kg', function(){
  plot_job(base, sg_kg_under_cohort, c(0,4000), 'Later cohorts survivor growth >=
      11.68 cm (kg/ha)',n)})
plot_save(base, 'under_mt_kg', function(){
  plot_job(base, mt_kg_under_cohort, c(0,4000), 'Later cohorts mortality >= 11.68 cm
      (kg/ha)',n)})
plot_save(base, 'under_ng_kg', function(){
  plot_job(base, ng_kg_under_cohort, c(-3000,3000), 'Later cohorts net growth >=
      11.68cm (kg/ha)',n)})
```

```
  plot_save(base, 'over_ig_m3', function(){
    plot_job(base, ig_m3_over_cohort, c(0,2), 'Initial cohort ingrowth >= 11.68cm
        (m3/ha)',n)})
  plot_save(base, 'over_sg_m3', function(){
    plot_job(base, sg_m3_over_cohort, c(0,8), 'Initial cohort survivor growth >= 11.68
        cm (m3/ha)',n)})
  plot_save(base, 'over_mt_m3', function(){
    plot_job(base, mt_m3_over_cohort, c(0,8), 'Initial cohort mortality >= 11.68 cm
        (m3/ha)',n)})
  plot_save(base, 'over_ng_m3', function(){
    plot_job(base, ng_m3_over_cohort, c(-3000,3000), 'Initial cohort net growth >=
        11.68cm (m3/ha)',n)})

  plot_save(base, 'under_ig_m3', function(){
    plot_job(base, ig_m3_under_cohort, c(0,2), 'Later cohorts ingrowth >= 11.68 cm
        (m3/ha)',n)})
  plot_save(base, 'under_sg_m3', function(){
    plot_job(base, sg_m3_under_cohort, c(0,8), 'Later cohorts survivor growth >= 11.68
        cm (m3/ha)',n)})
  plot_save(base, 'under_mt_m3', function(){
    plot_job(base, mt_m3_under_cohort, c(0,8), 'Later cohorts mortality >= 11.68 cm
        (m3/ha)',n)})
  plot_save(base, 'under_ng_m3', function(){
    plot_job(base, ng_m3_under_cohort, c(-5,5), 'Later cohorts net growth >= 11.68cm
        (m3/ha)',n)})
}

if (all_details){
## Biomass growth summaries for trees >4.6in
plot_save(base, 'kg_ig', function(){
  plot_job(base, ingrowth_kg, 300, 'Ingrowth  >= 11.68cm (kg/ha)', n) })
plot_save(base, 'kg_sg', function(){
  plot_job(base, survivor_growth_kg, 5000, 'Survivor growth  >= 11.68cm (kg/ha)', n) })
plot_save(base, 'kg_mt', function(){
  plot_job(base, mortality_kg, 5000, 'Mortality  >= 11.68cm (kg/ha)', n) })
plot_save(base, 'kg_ng', function(){
  plot_job(base, net_growth_kg, c(-2500,2500), 'Net growth  >= 11.68cm (kg/ha)', n) })
plot_save(base, 'kg_gg', function(){
  plot_job(base, gross_growth_kg, 5000, 'Gross growth  >= 11.68cm (kg/ha)', n) })
}

if (all_details){
## Biomass growth summaries for trees > 3 cm
plot_save(base, 'kg_all_ig', function(){
  plot_job(base, ingrowth_all_kg, 300, 'Ingrowth  >= 3 cm (kg/ha)', n) })
plot_save(base, 'kg_all_sg', function(){
  plot_job(base, survivor_growth_all_kg, 5000, 'Survivor growth  >= 3 cm (kg/ha)', n) })
plot_save(base, 'kg_all_mt', function(){
  plot_job(base, mortality_all_kg, 5000, 'Mortality  >= 3 cm (kg/ha)', n) })
plot_save(base, 'kg_all_ng', function(){
  plot_job(base, net_growth_all_kg, c(-2500,2500), 'Net growth  >= 3 cm (kg/ha)', n) })
plot_save(base, 'kg_all_gg', function(){
  plot_job(base, gross_growth_all_kg, 5000, 'Gross growth  >= 3 cm (kg/ha)', n) })
}
```

```
if (all_details){
## Canopy tree radial increment
plot_save( base, 'ctg', function(){
  plot_job(base,canopy_tree_growth, 3, 'Canopy Tree Radial Increment (mm/yr)',n) })
}

if (all_details){
## Radial growth summary by size class:
plot_save( base, 'seed_rg', function(){
  plot_job(base,seed_radial_growth,3,'Seedling (0-6cm) Radial Increment (mm/yr)',n) })
plot_save( base, 'sap_rg', function(){
  plot_job(base,sap_radial_growth,3,'Sapling (0-11cm) Radial Increment (mm/yr)',n) })
plot_save( base, 'pole_rg', function(){
  plot_job(base,pol_radial_growth, 3, 'Pole (11-26cm) Radial Increment (mm/yr)',n)  })
plot_save( base, 'mat_rg', function(){
  plot_job(base,mat_radial_growth, 3, 'Mature (26-46cm) Radial Increment (mm/yr)',n) })
plot_save( base, 'large_rg', function(){
  plot_job(base,lrg_radial_growth, 3, 'Large (46+cm) Radial Increment (mm/yr)',n) })
plot_save( base, 'xlarge_rg', function(){
  plot_job(base,xlg_radial_growth, 3, 'V. Large (66+cm) Radial Increment (mm/yr)',n) })
}

## Canopy tree crown radial increment
plot_save( base, 'ctcg', function(){
  plot_job(base,canopy_tree_crown_growth, 25, 'Canopy Tree Crown Radial Increment
      (cm/yr)',n)  })

if (all_details){
##  Mortality summary by size class:
plot_save( base, 'seed_mt', function(){
  plot_job(base,seed_mort, 0.5, 'Seedling (0-6cm) mortality (m2/yr)',n) })
plot_save( base, 'sap_mt', function(){
  plot_job(base,sap_mort, 0.5, 'Sapling (0-11cm) mortality (m2/yr)',n) })
plot_save( base, 'pole_mt', function(){
  plot_job(base,pol_mort, 0.5, 'Pole (11-26cm) mortality (m2/yr)',n) })
plot_save( base, 'mat_mt', function(){
  plot_job(base,mat_mort, 0.5, 'Mature (26-46cm) mortality (m2/yr)',n) })
plot_save( base, 'large_mt', function(){
  plot_job(base,lrg_mort, 0.5, 'Large (46+cm) mortality (m2/yr)',n) })
plot_save( base, 'xlarge_mt', function(){
  plot_job(base,xlg_mort, 0.5,'V. Large (66+cm) mortality (m2/yr)',n) })
}

if (all_details){
## Counts by size category
plot_save( base, 'seed', function(){
  plot_job(base,seed_trees,800,'Seedlings (0-6cm)/ha',n) })
plot_save( base, 'sap', function(){
  plot_job(base,sapling_trees,1500,'Saplings (11-26cm)/ha',n) })
plot_save( base, 'pole', function(){
  plot_job(base,pole_trees,400,'Pole trees (11-26cm)/ha',n) })
plot_save( base, 'mat', function(){
  plot_job(base,mature_trees,300,'Mature trees (26-46cm)/ha', n) })
plot_save( base, 'large', function(){
  plot_job(base,large_trees,200, 'Large trees (46+cm)/ha',n) })
```

```
plot_save( base, 'xlarge', function(){
  plot_job(base, xlarge_trees, 100, 'X-large trees (>66cm)/ha', n)  })
plot_save( base, 'g50', function(){
  plot_job(base, g50_trees, 100, 'Large trees (>50cm)/ha', n) })
plot_save (base, 'all_trees', function(){
  plot_job(base, all_trees, 5000, 'All trees (#/ha)', n)})
}

## Basal area by size category
plot_save( base, 'seed_ba', function(){
  plot_job(base,seed_trees_ba,50, 'Seedlings (0-6cm) m^2/ha',n) })
plot_save( base, 'sap_ba', function(){
  plot_job(base,sapling_trees_ba,50,'Saplings (0-11cm) m^2/ha',n)  })
plot_save( base, 'pole_ba', function(){
  plot_job(base,pole_trees_ba,50,'Pole trees (11-26cm) m^2/ha',n) })
plot_save( base, 'mat_ba', function(){
  plot_job(base,mature_trees_ba, 50,'Mature trees (26-46cm) m^2/ha', n) })
plot_save( base, 'large_ba', function(){
  plot_job(base,large_trees_ba, 50, 'Large trees (46+cm) m^2/ha',n) })
plot_save( base, 'xlarge_ba', function(){
  plot_job(base, xlarge_trees_ba, 50,'X-large trees (>66cm) m^2/ha', n) })

## Basal area by size category
plot_save( base, 'seed_pba', function(){
  plot_job(base,seed_trees_pba,50, 'Seedlings (0-6cm) %BA',n) })
plot_save( base, 'sap_pba', function(){
  plot_job(base,sapling_trees_pba,50,'Saplings (0-11cm) %BA',n)  })
plot_save( base, 'pole_pba', function(){
  plot_job(base,pole_trees_pba,50,'Pole trees (11-26cm) %BA',n) })
plot_save( base, 'mat_pba', function(){
  plot_job(base,mature_trees_pba, 50,'Mature trees (26-46cm) %BA', n) })
plot_save( base, 'large_pba', function(){
  plot_job(base,large_trees_pba, 50, 'Large trees (46+cm) %BA',n) })
plot_save( base, 'xlarge_pba', function(){
  plot_job(base, xlarge_trees_pba, 50,'X-large trees (>66cm) %BA', n) })

if (all_details){
## ECA by size category
plot_save( base, 'seed_eca', function(){
  plot_job(base,seed_trees_eca,100*100, 'Seedling (0-6cm) ECA (m^2/ha)',n) })
plot_save( base, 'sap_eca', function(){
  plot_job(base,sapling_trees_eca,100*100, 'Sapling (0-11cm) ECA (m^2/ha)',n) })
plot_save( base, 'pole_eca', function(){
  plot_job(base,pole_trees_eca,100*100, 'Pole trees (11-26cm) ECA (m^2/ha)',n) })
plot_save( base, 'mat_eca', function(){
  plot_job(base,mature_trees_eca, 100*100, 'Mature trees (26-46cm) ECA (m^2/ha)', n)})
plot_save( base, 'large_eca', function(){
  plot_job(base,large_trees_eca, 100*100, 'Large trees (46+cm) ECA (m^2/ha)',n) })
plot_save( base, 'xlarge_eca', function(){
  plot_job(base, xlarge_trees_eca, 100*100, 'X-large trees (>66cm) ECA (m2/ha)',n)})
}

## %ECA by size category
plot_save( base, 'seed_peca', function(){
```

```
  plot_job(base,seed_trees_peca,100,'Seedling (0-6cm) %ECA',n); })
plot_save( base, 'sap_peca', function(){
  plot_job(base,sapling_trees_peca,100,'Sapling (0-11cm) %ECA',n) })
plot_save( base, 'pole_peca', function(){
  plot_job(base,pole_trees_peca,100,'Pole trees (11-26cm) %ECA',n) })
plot_save( base, 'mat_peca', function(){
  plot_job(base,mature_trees_peca, 100,'Mature trees (26-46cm) %ECA', n)})
plot_save( base, 'large_peca', function(){
  plot_job(base,large_trees_peca, 100,'Large trees (46+cm) %ECA',n) })
plot_save( base, 'xlarge_peca', function(){
  plot_job(base, xlarge_trees_peca, 100,'X-large trees (>66cm) %ECA',n)})

plot_save(base, 'harv', function(){
  plot_job(base,harvest_volume,150,'Harvest volume (m3/ha)',n) })

plot_save(base, 'vol', function(){
  plot_job(base,standing_volume,700, 'Standing live volume (m3/ha)', n ); })

plot_save(base, 'mg', function(){
  plot_job(base,standing_biomass,500, 'Standing live biomass (Mg/ha)', n ); })

plot_save(base, 'mvol', function(){
  plot_job(base,mort_volume,150,'Dead volume (m3/ha)',n);})

plot_save(base, 'dbh', function(){
  plot_job(base,dbh_dist,200,'Trees/Ha',n, xlab='DBH (cm)',
           X=seq(2,118,by=4), xlim=c(11,100) ); })

plot_save(base, 'log_dbh', function(){
  plot_job(base,dbh_dist,c(0.1,200),'Trees/Ha',n, xlab='DBH (cm)',
           X=seq(2,118,by=4), xlim=c(11,100),log="y") })

plot_save(base, 'rba', function(){ rba_plot(base, n, label); })

plot_save(base, 'st_stage', function(){
  x=plot_job(base, stand_dev, 10, 'Developmental stage', n )
  legend('top',
         c('1: Sapling',
           '2: Pole',
           '3: Mat/Sap Mosaic',
           '4: Mature',
           '5: Late Agg.',
           '6: Early Tx',
           '7: Late Tx',
           '8: Steady State'), ncol=2, cex=0.75, y.intersp=1.6)
  x })

if (all_details){
  plot_save(base, 'cwd_logs', function(){
    plot_job(base, vol_logs, 200, 'Log volume (m3/ha)', n); })
  plot_save(base, 'cwd_log1s', function(){
    plot_job(base, vol_log1s, 100, 'Log 1 Volume', n); })
  plot_save(base, 'cwd_log2s', function(){
    plot_job(base, vol_log2s, 100, 'Log 2 Volume', n); })
  plot_save(base, 'cwd_log3s', function(){
```

```
  plot_job(base, vol_log3s, 100, 'Log 3 Volume', n); })
plot_save(base, 'cwd_log4s', function(){
  plot_job(base, vol_log4s, 100, 'Log 4 Volume', n); })
plot_save(base, 'cwd_log5s', function(){
  plot_job(base, vol_log5s, 100, 'Log 5 Volume', n); })

plot_save(base, 'cwd_snags', function(){
  plot_job(base, vol_snags, 100, 'Snag volume (m3/ha)', n); } )
plot_save(base, 'cwd_snag1s', function(){
  plot_job(base, vol_snag1s, 50, 'Snag 1 Volume', n); } )
plot_save(base, 'cwd_snag2s', function(){
  plot_job(base, vol_snag2s, 50, 'Snag 2 Volume', n); } )
plot_save(base, 'cwd_snag3s', function(){
  plot_job(base, vol_snag3s, 50, 'Snag 3 Volume', n); } )
plot_save(base, 'cwd_snag4s', function(){
  plot_job(base, vol_snag4s, 50, 'Snag 4 Volume', n); } )
plot_save(base, 'cwd_snag5s', function(){
  plot_job(base, vol_snag5s, 50, 'Snag 5 Volume', n); } )

plot_save( base, 'cwd_all', function(){
  plot_job(base, vol_all, 300, 'CWD volume (m3/ha)', n); } )

plot_save( base, 'cwd_ddist_log1', function(){
  plot_job(base, cwd_ddist_log1, 50, 'Log 1 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_log2', function(){
  plot_job(base, cwd_ddist_log2, 50, 'Log 2 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_log3', function(){
  plot_job(base, cwd_ddist_log3, 50, 'Log 3 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_log4', function(){
  plot_job(base, cwd_ddist_log4, 50, 'Log 4 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_log5', function(){
  plot_job(base, cwd_ddist_log5, 50, 'Log 5 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4),  xlim=c(11,100) ) })

plot_save( base, 'cwd_ddist_snag1', function(){
  plot_job(base, cwd_ddist_snag1, 50, 'Snag 1 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_snag2', function(){
  plot_job(base, cwd_ddist_snag2, 50, 'Snag 2 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_snag3', function(){
  plot_job(base, cwd_ddist_snag3, 50, 'Snag 3 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_snag4', function(){
  plot_job(base, cwd_ddist_snag4, 50, 'Snag 4 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
plot_save( base, 'cwd_ddist_snag5', function(){
  plot_job(base, cwd_ddist_snag5, 50, 'Snag 5 DBH Distribution', n,
           xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
```

```
    plot_save( base, 'cwd_ddist_snags', function(){
      plot_job(base, cwd_ddist_snags, 50, 'Snag DBH Distribution', n,
               xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
    plot_save( base, 'cwd_ddist_logs', function(){
      plot_job(base, cwd_ddist_logs, 50, 'Log DBH Distribution', n,
               xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })
    plot_save( base, 'cwd_ddist_all', function(){
      plot_job(base, cwd_ddist_all, 50, 'CWD DBH Distribution', n,
               xlab='DBH (cm)', X=seq(2,118,by=4), xlim=c(11,100) ) })

    plot_save( base, 'cwd_mass_all', function(){
      plot_job(base, mass_all, 300, 'CWD biomass (Mg/ha)', n); } )
    plot_save(base, 'cwd_mass_snags', function(){
      plot_job(base, mass_snags, 100, 'Snag biomass (Mg/ha)', n); } )
    plot_save(base, 'cwd_mass_logs', function(){
      plot_job(base, mass_logs, 200, 'Log biomass (Mg/ha)', n); })
  }

  generate_ddist_table(base,n)

  mk_std_2d_plots(base, n)

  mk_summary_tables(base)
}

## 2d plots for the handout, from replicate 01:
mk_std_2d_plots = function(base, n){
  db_open(sprintf('%s.r01.db',base))
  fig_basename = sprintf('%s_r01',base);

  ## Set interval for snapshot plots here:
  for (yr in seq(0,max_year,by=10) ){
    out=sprintf('%s_%04d_dbh.png',fig_basename,yr)
    if( length(list.files(pattern='^' %.% out))==0){
      png(out,width=600,height=600)
      par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
      dbh_dist_one(yr)
      dev.off()
    }

    out=sprintf('%s_%04d_eca.png',fig_basename,yr)
    if (length(list.files(pattern='^' %.% out))==0){
      png(out,width=600,height=600)
      par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
      eca_dist_one(yr)
      dev.off()
    }

    out=sprintf('%s_%04d_stock.png',fig_basename,yr)
    if (length(list.files(pattern='^' %.% out))==0){
      png(out,width=600,height=600)
      par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
      stocking_dist_one(yr)
      dev.off()
    }

    out=sprintf('%s_%04d_dgr_vs_dbh.png',fig_basename,yr)
```

```
  if (length(list.files(pattern='^' %.% out))==0 &&
      yr != max_year ){
    png(out,width=600,height=600)
    par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
    dgr_vs_dbh_plot_one(yr)
    dev.off()
  }
}

crh_mode = function(x){
  tx = sort(table(x), decreasing=T)
  mx = as.numeric( names(tx)[1] )
  ifelse (sum(tx==tx[1])>1, NA, mx)
}

mk_age_dist = function(sp){
  sp_lut = list()
  sp_lut[["sm"]] = 4
  sp_lut[["hm"]] = 5
  sp_lut[["yb"]] = 6
  sp_lut[["wa"]] = 13
  sp_lut[["bw"]] = 2
  sp_lut[["rm"]] = 9

  sp_names = list()
  sp_names[["sm"]] = "Sugar maple"
  sp_names[["hm"]] = "Hemlock"
  sp_names[["yb"]] = "Yellow birch"
  sp_names[["wa"]] = "Ash"
  sp_names[["bw"]] = "Basswood"
  sp_names[["rm"]] = "Red maple"

  out=sprintf('%s_%s_age_dist.png',fig_basename, sp)
  if (is.na(file.info(out)$size)){
    d_sm=sql(sprintf(
      'select dbh,age from dead_trees where species=%i and dbh>=25',
      sp_lut[[sp]] ) )

    png(out,width=600,height=600)
    par(ps=24, mai=c(1.35, 1.25, 1.25, 0.25 ))
    if (length(d_sm$age)>0){
      hist( d_sm$age, freq=F, right=F,
            breaks=seq(0,10*ceiling(max(d_sm$age/10)),by=10),
            col='grey', main=sp_names[[sp]],
            xlab='Age at time of death (yr)')
      mtext( sprintf("Mean: %.1f  Median: %.1f  Mode: %.1f",
                     mean(d_sm$age),
                     median(d_sm$age),
                     crh_mode( 10*floor(d_sm$age/10)) ), side=3, line=-1 )
    } else {
      plot( c(0,1), c(0,1), type="n")
      text(0.5, 0.5, "No data")
    }
    dev.off()
  }
}
```

```
for (sp in c("sm","hm","yb","wa","bw","rm")){
  mk_age_dist(sp)
}

out=sprintf('%s_all_age_dist.png',fig_basename)
if (is.na(file.info(out)$size)){
  d_all = sql('select dbh,age from dead_trees where dbh>=25')
  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 1.25, 0.25 ))
  if (length(d_all$age)>0){
    hist( d_all$age, freq=F, right=F,
          breaks=seq(0,10*ceiling(max(d_all$age/10)),by=10),
          col='grey', main='All species',
          xlab='Age at time of death (yr)')
    mtext( sprintf("Mean: %.1f  Median: %.1f  Mode: %.1f",
                   mean(d_all$age),
                   median(d_all$age),
                   crh_mode( 10*floor(d_all$age/10)) ), side=3, line=-1 )
  } else {
    plot( c(0,1), c(0,1), type="n")
    text(0.5, 0.5, "No data")
  }
  dev.off()
}

## Age at TOD distribution for overstory trees:
out = sprintf("%s_all_age_dist_over.png", fig_basename)
if (is.na(file.info(out)$size)){
  d_all = sql('select dbh,age from dead_trees where dbh >= 25 and ' %.%
    'treeno in (select treeno from live_trees where year<=20)')
  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 1.25, 0.25 ))
  if (length(d_all$age)>0){
    hist( d_all$age,freq=F, right=F,
          breaks=seq(0,10*ceiling(max(d_all$age/10)),by=10),
          col='grey', main='Overstory, all species',
          xlab='Age at time of death (yr)')
    mtext( sprintf("Mean: %.1f  Median: %.1f  Mode: %.1f",
                   mean(d_all$age),
                   median(d_all$age),
                   crh_mode( 10*floor(d_all$age/10)) ), side=3, line=-1 )
  } else {
    plot( c(0,1), c(0,1), type="n")
    text(0.5, 0.5, "No data")
  }
  dev.off()
}

gen_pch = function(d){
  pch = rep(0, length(d$dbh))

  pch[d$species ==  2] = 1
  pch[d$species ==  4] = 2
  pch[d$species == 13] = 15
  pch[d$species ==  5] = 4
```

```
  pch[d$species ==  6] = 19
  pch[d$species ==  9] = 6

  pch
}

gen_col = function(d){
  col = rep('black', length(d$dbh) )
  col[ d$species ==  2] = 'blue'
  col[ d$species ==  4] = 'green'
  col[ d$species == 13] = 'orange'
  col[ d$species ==  5] = 'purple'
  col[ d$species ==  6] = 'yellow'
  col[ d$species ==  9] = 'red'

  col
}

cr_plot = function(d){
  predict_mcr = function( species, habitat, dbh ){
    if (habitat=="AOCa" & species=="sugarmaple"){
      coefs = c( 8.13689, 0.03818, 18.56050 )
    } else if (species=="sugarmaple"){
      coefs = c( 7.91233, 0.02746, 24.93295 )
    } else if (species=="hemlock"){
      coefs = c( 4.51000, 0.04241, 10.10187 )
    } else {
      stop('Unknown species')
    }
    B = coefs[1]
    C = coefs[2]
    D = coefs[3]

    B * exp(-exp(-C * (dbh - D)))
  }

  plot( d$dbh, d$mcr, xlab="DBH (cm)", ylab="MCR (m)",
       pch=gen_pch(d), col=gen_col(d))
  dx = seq(min(d$dbh), max(d$dbh))
  lines( dx, predict_mcr("sugarmaple", "ATD", dx), col='green')
  lines( dx, predict_mcr("hemlock", "ATD", dx),    col='purple')

  legend('topleft',
         c('Basswood', 'Sugar maple', 'White ash',
           'Hemlock', 'Yellow birch', 'Red maple', 'Other'),
         col=c('blue','green','orange','purple','yellow','red','black'),
         pch=c(1,2,15,4,19,6,0))
}

age_plot = function(d) {
  plot( d$age, d$mcr, xlab="Age (yr)", ylab="MCR (m)",
       pch=gen_pch(d), col=gen_col(d))
  legend('topleft',
         c('Basswood', 'Sugar maple', 'White ash',
           'Hemlock', 'Yellow birch', 'Red maple', 'Other'),
         col=c('blue','green','orange','purple','yellow','red','black'),
         pch=c(1,2,15,4,19,6,0))
```

```
}

dbh_age_plot = function(d) {
  plot( d$age, d$dbh, xlab="Age (yr)", ylab="DBH (cm)",
        pch=gen_pch(d), col=gen_col(d))
  legend('topleft',
         c('Basswood', 'Sugar maple', 'White ash',
           'Hemlock', 'Yellow birch', 'Red maple', 'Other'),
         col=c('blue','green','orange','purple','yellow','red','black'),
         pch=c(1,2,15,4,19,6,0))
}

out=sprintf('%s_cr_vs_diam_start.png',fig_basename)
if (length(list.files(pattern='^' %.% out))==0){
  d <- get_trees(0)
  d <- d[d$species!=24,]

  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
  if (length(d$dbh>0)) { cr_plot(d) } else { plot(0,0) }
  dev.off()
}

out=sprintf('%s_cr_vs_diam.png',fig_basename)
if (length(list.files(pattern='^' %.% out))==0){
  d <- get_trees(max_year)
  d <- d[d$species!=24,]

  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
  if (length(d$dbh)>0) { cr_plot(d) } else { plot(0,0) }
  dev.off()
}

out=sprintf('%s_cr_vs_age_start.png',fig_basename)
if (length(list.files(pattern='^' %.% out))==0){
  d <- get_trees(0)
  d <- d[d$species!=24,]

  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
  if (length(d$dbh)>0) { age_plot(d) } else { plot(0,0) }
  dev.off()
}

out=sprintf('%s_cr_vs_age.png',fig_basename)
if (length(list.files(pattern='^' %.% out))==0){
  d <- get_trees(max_year)
  d <- d[d$species!=24,]

  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
  if (length(d$dbh)>0) { age_plot(d) } else { plot(0,0) }
  dev.off()
}

out=sprintf('%s_diam_vs_age_start.png',fig_basename)
if (length(list.files(pattern='^' %.% out))==0){
```

```
  d <- get_trees(0)
  d <- d[d$species!=24,]

  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
  if (length(d$dbh)>0) { dbh_age_plot(d) } else { plot(0,0) }
  dev.off()
}

out=sprintf('%s_diam_vs_age.png',fig_basename)
if (length(list.files(pattern='^' %.% out))==0){
  d <- get_trees(max_year)
  d <- d[d$species!=24,]

  png(out,width=600,height=600)
  par(ps=24, mai=c(1.35, 1.25, 0.25, 0.25 ))
  if (length(d$dbh)>0) { dbh_age_plot(d) } else { plot(0,0) }
  dev.off()
}

db_close()

## Residence times
stages = c()
for (i in 1:n){
  db_open(sprintf('%s.r%02i.db',base, i))
  stages = rbind ( stages, stand_dev() )
  db_close()
}

colnames(stages)=years

sink( sprintf('%s_stage_data.tex',base) )
for (i in 1:ceiling(length(years)/18)){
  ix = ( 18*(i-1) + 1): min(18*i, length(years) )
  print(xtable(stages[,ix]), floating=F)
  cat('\n\n')
}
sink()

## Now we have a matrix giving the stand stage at various times.
tot_times = matrix( 0, nrow=n, ncol=9 )
entries   = matrix( 0, nrow=n, ncol=9 )

entries[,9] = 1

for (i in 1:n) {
  hit_og = F
  for (j in 1:length(stages[i,]) ){
    stage_i = stages[i,j]
    tot_times[i, stage_i] = tot_times[i, stage_i] + meas_interval

    if (stage_i >= 5 ){ hit_og = T }

    if (! hit_og ) {
      tot_times[i, 9] = tot_times[i,9] + meas_interval
    }

    if (j==1 || stages[i,j-1] != stage_i){
```

```
        entries[i, stage_i] = entries[i, stage_i] + 1
      }
    }
  }

  res_times = matrix(0, nrow=n, ncol=9)

  for (i in 1:n) {
    res_times[i,] = tot_times[i,] / entries[i,]
  }

  colnames( res_times ) = c("Sapling", "Pole", "MSM", "Mature",
            "0G: LA", "0G: ET", "0G: LT", "0G: SS", "Time to 0G")

  sink( sprintf('%s_residence_times.tex', base) )
  print( xtable( res_times), floating=F)
  cat('\n\n')
  sink()
}

generate_ddist_table = function(base,n){
  # To set 'years':
  db_open(sprintf('%s.r01.db',base)); db_close()

  out=sprintf('%s_dbh_sequence.csv',base)
  if (length(list.files(pattern='^' %.% out))==0){
    rc = c()

    for (yr in years){
      row_i = envelope_engine(base, n, function(){dbh_dist(yr)} );
      rc = rbind( rc, row_i$Mu )
    }

    rc = round(rc[,2:30],1) # Trim the half-class w/ NA, round diameters to 1/10 cm
    colnames(rc) = seq(6,118,by=4)
    rownames(rc) = years

    write.csv(rc, out)
  }
}

mk_summary_tables = function(base){
  ## Pull out the BA summary data
  d_sap_ba = read.csv(sprintf('%s_sap_ba.csv', base))
  d_pol_ba = read.csv(sprintf('%s_pole_ba.csv', base))
  d_mat_ba = read.csv(sprintf('%s_mat_ba.csv', base))
  d_lrg_ba = read.csv(sprintf('%s_large_ba.csv', base))

  n_items = length(d_sap_ba[,5])

  ## Construct a summary of BA by size category:
  ba = rbind(
    d_sap_ba[,5],
    d_pol_ba[,5],
    d_mat_ba[,5],
    d_lrg_ba[,5] )
  rownames(ba) = c("Sapling", "Pole", "Mature", "Large")
  colnames(ba) = d_sap_ba[,2]

  ## In the perl standard handout, I'm using tables that are 18 cells wide
```

```
## So, replicate that here and chop up the data if it is getting too long.
sink( sprintf('%s_ba_summary.tex',base) )
for (i in 1:ceiling(n_items/18)){
  ix = ( 18*(i-1) + 1): min(18*i, n_items)
  print(xtable(ba[,ix]), floating=F)
  cat('\n\n')
}
sink()

tot_ba = apply( ba, 2, sum)

rel_ba = rbind(
  100 * d_sap_ba[,5] / tot_ba,
  100 * d_pol_ba[,5] / tot_ba,
  100 * d_mat_ba[,5] / tot_ba,
  100 * d_lrg_ba[,5] / tot_ba )

rownames(rel_ba) = rownames(ba)
colnames(rel_ba) = colnames(ba)

sink( sprintf('%s_rel_ba_summary.tex',base))
for (i in 1:ceiling(n_items/18)){
  ix = ( 18*(i-1) + 1): min( 18*i, n_items)
  print(xtable(rel_ba[,ix]), floating=F)
  cat('\n\n')
}
sink()

## Pull out the ECA summary data:
d_sap_eca = read.csv(sprintf('%s_sap_eca.csv', base))
d_pol_eca = read.csv(sprintf('%s_pole_eca.csv', base))
d_mat_eca = read.csv(sprintf('%s_mat_eca.csv', base))
d_lrg_eca = read.csv(sprintf('%s_large_eca.csv', base))

## Construct a summary of BA by size category:
eca = rbind(
  d_sap_eca[,5],
  d_pol_eca[,5],
  d_mat_eca[,5],
  d_lrg_eca[,5])
rownames(eca) = rownames(ba)
colnames(eca) = colnames(ba)

sink( sprintf('%s_eca_summary.tex',base) )
for (i in 1:ceiling(n_items/18)){
  ix = ( 18*(i-1) + 1): min(18*i, n_items)
  print(xtable(eca[,ix]), floating=F)
  cat('\n\n')
}
sink()

tot_eca = apply( eca, 2, sum)
rel_eca = rbind(
  100 * d_sap_eca[,5] / tot_eca,
  100 * d_pol_eca[,5] / tot_eca,
  100 * d_mat_eca[,5] / tot_eca,
```

```
    100 * d_lrg_eca[,5] / tot_eca )

  rownames(rel_eca) = rownames(ba)
  colnames(rel_eca) = colnames(ba)

  sink( sprintf('%s_rel_eca_summary.tex',base))
  for (i in 1:ceiling(n_items/18)){
    ix = ( 18*(i-1) + 1): min( 18*i, n_items)
    print(xtable(rel_eca[,ix]), floating=F)
    cat('\n\n')
  }
  sink()
}

compute_productivity = function(){
  sim_prod = function( dbase, tgt_yr) {
    cut_vol = 0;
    std_vol = 0;

    nrep = 3;
    for( sr in 1:nrep ){
      fname = sprintf('%s.r%02i.db', dbase, sr);

      db_open(fname)
      cut      = sql(sprintf('select treeno,dbh,ht ' %.%
                             'from cut_taken_trees where year<=%i', tgt_yr ));
      standing = sql(sprintf('select treeno,dbh,ht ' %.%
                             'from live_trees where year==%i', tgt_yr));
      cut_vol = cut_vol + sum(estimate_volume_go(      cut$dbh,      cut$ht ));
      std_vol = std_vol + sum(estimate_volume_go( standing$dbh, standing$ht ));
      db_close()
    }
    c( cut_vol / nrep, std_vol / nrep );
  }

  chvol = c(); chvol50 = c();
  fvol  = c();  fvol50 = c();
  sname = c();

  sims=list.files(pattern='.*.r01.db$')
  sims=sub('.r01.db','',sims,fixed=T)

  for( fname_base in sims ){
    x = sim_prod( fname_base, 300 );
    chvol = c(chvol, x[1])
    fvol  = c(fvol,  x[2])

    x = sim_prod( fname_base, 45);
    chvol50 = c(chvol50, x[1])
    fvol50  = c(fvol50,  x[2])

    sname = c(sname, fname_base)
  }

  print( data.frame(sname, fv45 =fvol50/9, hv45 =chvol50/(9*45 ),
                           fv300=fvol  /9, hv300=chvol  /(9*300) ))
}
```

F.6 UTIL.R

```
library("RSQLite")
library("zoo")

if (!exists('meas_interval')){ meas_interval=10 }

####################
# Utility functions

'%.%' = function(s1,s2){ paste(s1,s2,sep='') }

my_merge = function(x,y, ... ){
  if( length(x)>0 & length(y) >0 ){ rc = merge(x,y,...); }
  else if (length(x)>0 )           { rc = x;             }
  else if (length(y)>0 )           { rc = y;             }
  else                             { rc = c();           }
  rc;
}

#####################################
# SQL interface

## Provide a caching layer to make repeat queries less unpleasant
sql_cache <<- list()

dbi=SQLite();  dbi_file=''
db_open  = function(fname) {
  if (is.na(file.info(fname)$size)) {
    stop('no such file: ', fname)
  } else {
    db<<-dbConnect(dbi,fname)
    dbi_file<<-fname

    if (grepl('_atm[._]',fname)){
      db_site_quality <<- "F"
    } else {
      db_site_quality <<- "G"
    }

    # Provide a generalized result cache
    cache_fname = sprintf('%s_cache.rd', fname)
    if (is.na(file.info(cache_fname)$size)) {
      result_cache <<- list()
    } else {
      load(cache_fname, .GlobalEnv)
    }
    result_cache[["__dirty__"]] <<- F

    # And also a query cache along with a cache for annoyingly
    # complex growth analyses.
    if (is.null(sql_cache[[dbi_file]]) ){
      sql_cache_file = sprintf('%s_sql_cache.rd', fname)
      if (is.na(file.info(sql_cache_file)$size) ) {
        sql_cache[[dbi_file]] <<- list()
      } else {
        load( sql_cache_file )
        sql_cache[[dbi_file]] <<- this_sql_cache_data
      }
    }
```

```
    sql_cache[[dbi_file]][["__dirty__"]] <<- F

    d=sql('select max(year) as max_year from sizes')
    max_year <<- d$max_year
    years <<- seq(0,max_year,by=meas_interval)
  }
}

db_close = function() {
  if (dbi_file != ''){
    dbDisconnect(db)

    # This guy is quick to save, so just do so.
    if (result_cache[["__dirty__"]])
      save( result_cache,
           file=sprintf('%s_cache.rd',dbi_file) )
    result_cache <<- list()

    ## Depend on caller(s) to appropriately save the query caches.
    dbi_file<<-''
  }
}

save_caches = function() {
  for (db_fn in names(sql_cache) ){
    if (sql_cache[[db_fn]][["__dirty__"]] ){
      sql_cache[[db_fn]][["__dirty__"]] <<- F
      this_sql_cache_data = sql_cache[[db_fn]]
      save( this_sql_cache_data, file=sprintf('%s_sql_cache.rd', db_fn) )
    }
  }
}

sql = function(x){
  if (dbi_file == '') { stop('no db open') }
  if (is.null(sql_cache[[dbi_file]][[x]]) ){
    cat(sprintf('SQL: %s\n', x))
    rs = dbSendQuery(db,x)
    rc = fetch(rs, n=-1)
    dbClearResult(rs)
    sql_cache[[dbi_file]][[x]] <<- rc
    sql_cache[[dbi_file]][["__dirty__"]] <<- T
  }
  sql_cache[[dbi_file]][[x]]
}

clear_caches=function(){
  save_caches()
  sql_cache <<- list()
}

## Persistent results cache:
res_cache = function(fcn) {
  ## To generate indices from a list of parameters
  ##  Note that the parameters must be unique, which might necessitate
  ##  some use of eval(substitute() )
  mk_ix = function(x, ... ) {
    rc = gsub("  +", " ", paste0( deparse( x ), collapse='') )
```

```
    if (length(list(...))>0) {
      paste( rc, mk_ix(...) )
    } else {
      rc
    }
  }

  ix =  mk_ix(fcn)
  if (is.null(result_cache[[ix]]) ||
      (is.matrix(result_cache[[ix]]) &&
       dim(result_cache[[ix]])[2] != length(years)) ||
      (is.vector(result_cache[[ix]]) &&
       length(result_cache[[ix]]) != length(years)) ){
    result_cache[[ix]] <<- fcn()
    result_cache[["__dirty__"]] <<- T
  }
  result_cache[[ix]]
}

## Also try to run as much of the data analsys as possible through a standard set of
## query functions, to maximize the efficiacy of the cache:
get_scale=function(){
  r <- sql("select xmax*ymax as size from plotinfo")
  (100*100)/r$size
}

get_trees=function(year){
  d <- sql(sprintf('select x,y,year,treeno,species,age,dbh,relht,ht,hbc,hwc,'  %.%
                   '3.14/4*(nexp*eexp+eexp*sexp+sexp*wexp+wexp*nexp) as eca, ' %.%
                   '3.14/4*(n*e+e*s+s*w+w*n) as tca, '                         %.%
                   '(n+e+s+w)/4 as mcr, '                                      %.%
                   'n,e,s,w,nexp,eexp,sexp,wexp '                              %.%
                   'from live_trees where year%%%i=0', meas_interval))
  d[d$year==year,]
}

get_cwd_dbh=function(year, cl){
  cwd_dbh <- sql('select * from dead_trees union ' %.%
                 'select * from cut_left_trees')
  treenos <- sql(sprintf('select year,treeno from cwd_%s',cl))
  treenos <- treenos[treenos$year == year,]
  cwd_dbh[ cwd_dbh$treeno %in% treenos$treeno, ]
}

#### Data manipulation:
# An engine to compute envelopes for a family of simulations.
envelope_engine = function(base_name, n, summary_fcn){
  get_from_db = function(base_name, i){
    dbfi= sprintf('%s.r%02i.db',base_name,i)
    cat(sprintf('%s\n', dbfi))

    db_open(dbfi)
    rc=summary_fcn()
    db_close()
    if (is.matrix(rc)) { rc = apply(rc, 2, sum) }
    rc
  }
```

```
  my_sd = function(x,...){
    if(sum(is.na(x))==length(x) ){ NA;         }
    else                          { sd(x, ...) }
  }

  XX=get_from_db(base_name,1)
  rc = matrix(NA, nrow=n, ncol=length(XX))
  rc[1,] = XX

  if (n>=2) {
    for (i in 2:n){ rc[i,]=get_from_db(base_name, i) }
  }

  Up=apply(rc, 2, max,   na.rm=T)
  Dn=apply(rc, 2, min,   na.rm=T)
  Mu=apply(rc, 2, mean,  na.rm=T)
  Sd=apply(rc, 2, my_sd, na.rm=T)

  data.frame(Up, Dn, Mu, Sd);
}

##### Plotting functions
# Takes envelope data and generates a plot from it:
plot_job = function(
  base, func, ylims, ylab, n,
  xlab='Simulation year', X=years, xlim=c(0,max_year), ... ) {
  cat(sprintf('== %s ==\n', ylab));
  a = envelope_engine(base, n, func )

  if (length(ylims)==2){
    ymin=ylims[1]; ymax=ylims[2]
  } else {
    ymin=0; ymax=ylims;
  }

  plot( X, a$Mu, type='l', cex=2,
        lwd=3,ylim=c(ymin,ymax),ylab=ylab,xlim=xlim,xlab=xlab, ... );
  lines(X, a$Up, lty = 1, lwd=1);
  lines(X, a$Dn, lty = 1, lwd=1);
  cbind(X,a)
}

# Checks for the existence of a csv for a given plot.
# If one is not found, plot the data with the given plot_fcn
#  This function should return the data, which is dumped into a csv.
plot_save = function( base, tag, plot_fcn ){
  if (length(list.files(pattern='^' %.% sprintf('%s_%s.csv',base,tag)))==0){
    png(sprintf('%s_%s.png',base,tag), width=600, height=600)
    par(ps=24, mai=c(1.25, 1.25, 0.25, 0.25) )
    d=plot_fcn()
    dev.off()
    write.csv(d, file=sprintf('%s_%s.csv',base,tag))
  }
}

# ECA and Diameter distributions for a single replication:
eca_dist_one = function(year, plt_key=T) {
```

```
sum_eca = function(sp, others=F){
  if (others){
    ix = which( !data$species %in% sp )
  } else {
    ix = which(data$species %in% sp)
  }

  rc = rep(0, 30)

  if(length(ix)>0){
    size_class = 1+pmin(floor(data$dbh[ix]/4),29)
    for (i in 1:length(ix) ){
      rc[size_class[i]] = rc[size_class[i]] + data$eca[ix[i]]
    }
  }
  rc
}

data=sql("select species,dbh," %.%
  "(3.14/4)*(nexp*eexp+eexp*sexp+sexp*wexp+wexp*nexp) as eca " %.%
  "from live_trees where dbh > 3 and year=" %.% year )
sclass=seq(0,120,4) - 1

eca_totals = list()
eca_totals[[1]] = sum_eca( c( 2))
eca_totals[[2]] = sum_eca( c( 4))
eca_totals[[3]] = sum_eca( c(13,14))
eca_totals[[4]] = sum_eca( c( 5))
eca_totals[[5]] = sum_eca( c( 6))
eca_totals[[6]] = sum_eca( c( 9))
eca_totals[[7]] = sum_eca( c(2,4,13,14,5,6,9), T )

cols = c("blue","green","orange","purple","yellow","red","grey")
labs = c("Basswood","Sugar maple","White/Green ash", "Hemlock",
  "Yellow birch", "Red maple","Other")

per_sp_eca = c()
for (i in 1:7) { per_sp_eca[i] = sum(eca_totals[[i]]) }

order = sort(per_sp_eca, decreasing=T, index.return=T)$ix

z = c()
for (i in 1:7){
  z = rbind( z, get_scale() * eca_totals[[ order[i] ]] )
}

if (length(data$eca)>0){
  tot_eca = get_scale() * sum( data$eca )
} else {
  tot_eca = 0
}

barplot(z, names.arg=rollmean(sclass,2),
        xlab="DBH (cm)", ylab="ECA m^2/Ha",
        col=cols[order],
        legend.text= if(plt_key){labs[order] } else { NULL },
        args.legend=list(x='right'),
        ylim=c(0,1500),
```

```
          main=if(plt_key){paste("Simulation year",year)} else{NULL})

  mtext( sprintf("ECA/(Plot area): %.3f", tot_eca / (100*100) ),
         side=3, line=-2.5)
}

# ECA and Diameter distributions for a single replication:
ba_dist = function(year) {
  sum_ba = function(sp, others=F){
    if (others){
      ix = which( !data$species %in% sp )
    } else {
      ix = which(data$species %in% sp)
    }

    rc = rep(0, 30)

    if(length(ix)>0){
      size_class = 1+pmin(floor(data$dbh[ix]/4),29)
      for (i in 1:length(ix) ){
        rc[size_class[i]] = rc[size_class[i]] + data$ba[ix[i]]
      }
    }
    rc[1] = NA # Drop the partial class

    rc
  }

  data=sql("select species,dbh," %.%
    "(3.14/4)*(nexp*eexp+eexp*sexp+sexp*wexp+wexp*nexp) as eca " %.%
    "from live_trees where dbh > 3 and year=" %.% year )
  sclass=seq(0,120,4) - 1

  data$ba = pi*(data$dbh^2/200)

  ba_totals = list()
  ba_totals[[1]] = sum_ba( c( 2))
  ba_totals[[2]] = sum_ba( c( 4))
  ba_totals[[3]] = sum_ba( c(13,14))
  ba_totals[[4]] = sum_ba( c( 5))
  ba_totals[[5]] = sum_ba( c( 6))
  ba_totals[[6]] = sum_ba( c( 9))
  ba_totals[[7]] = sum_ba( c(2,4,13,14,5,6,9), T )

  per_sp_ba = c()
  for (i in 1:7) { per_sp_ba[i] = sum(ba_totals[[i]], na.rm=T) }

  order = sort(per_sp_ba, decreasing=T, index.return=T)$ix

  z = c()
  for (i in 1:7){
    z = rbind( z, get_scale() * ba_totals[[ order[i] ]] )
  }

  list(z=z, order=order, sclass=sclass)
}

ba_dist_one = function(year, plt_key=T){
  ba_data = ba_dist(year)
```

```
  cols = c("blue","green","orange","purple","yellow","red","grey")
  labs = c("Basswood","Sugar maple","White/Green ash", "Hemlock",
    "Yellow birch", "Red maple","Other")

  barplot(ba_data$z, names.arg=rollmean(ba_data$sclass,2),
          xlab="DBH (cm)", ylab="BA m^2/Ha",
          col=cols[ba_data$order],
          legend.text= if(plt_key){labs[ba_data$order] } else { NULL },
          args.legend=list(x='topright'),
          ylim=c(0,1500),
          main=if(plt_key){paste("Simulation year",year)} else{NULL})
  mtext( sprintf("Stage %i", stand_dev_yr( db_site_quality, year, 0.25, 3.75)),
        side=3, line=-1)
}

dbh_dist_one = function(year, log_plot=F, ymax=200, plt_legend=T, plt_labels=T,
  xaxt="s", yaxt="s") {
  get_sp = function(sp,others=F){
    clamp = function(x, lim) { if(length(x)>0){ pmin(x,lim) } else { numeric(0) } }
    if(others){
      ix = which( !data$species %in% sp )
    } else {
      ix = which( data$species %in% sp )
    }
    hist(clamp(data$dbh[ix],118),breaks=sclass,plot=F)
  }

  sclass=seq(0,120,4) - 1
  if (log_plot){
    data=sql("select dbh from sizes where dbh > 3 and year=" %.% year)
    all=hist(clamp(data$dbh,118),breaks=sclass,plot=F)
    z=get_scale() * all$counts
    plot(all$mids,pmax(1,z),log="y",xlab="DBH (cm)",ylab="T/Ha",ylim=c(1,ymax),
         type="l")
  } else {
    data=sql("select species,dbh from live_trees where dbh > 3 and year=" %.% year)

    counts = list()
    counts[[1]] = get_sp( c(2)  )
    counts[[2]] = get_sp( c(4)  )
    counts[[3]] = get_sp( c(13,14) )
    counts[[4]] = get_sp( c(5)  )
    counts[[5]] = get_sp( c(6)  )
    counts[[6]] = get_sp( c(9)  )
    counts[[7]] = get_sp( c(2,4,13,14,5,6,9), T)

    cols = c("blue","green","orange","purple","yellow","red","grey")
    labs = c("Basswood","Sugar maple","White/Green ash", "Hemlock",
      "Yellow birch", "Red maple","Other")

    ba_mid = pi*(rollmean(sclass,2)/200)^2

    ba_list = c()
    for (i in 1:7) { ba_list[i] = sum(counts[[i]]$counts * ba_mid) }

    order = sort(ba_list, decreasing=T, index.return=T)$ix
```

```
    ba = get_scale()*sum( pi*(data$dbh/200)^2 )

    z = c()
    for (i in 1:7){
      z = rbind( z, get_scale() * counts[[ order[i] ]]$counts )
    }

    if (plt_legend){
      l_text = labs[order]
    } else {
      l_text = NULL
    }

    barplot(z, names.arg=rollmean(sclass,2),
            xlab="", ylab="", ylim=c(0,ymax),
            col = cols[order],
            legend.text=l_text,
            args.legend=list(x='right'),
            xaxs="i", yaxs="i", xaxt=xaxt, yaxt=yaxt,xpd=F )
    if (plt_labels){
      title(paste("Simulation year",year))
      mtext("DBH (cm)", side=1, line=2 )
      mtext("Trees/ha", side=2, line=2 )
      mtext(sprintf("BA: %.1f", ba), side=3, line=-2.5)
    }
  }
}

age_dist_one = function(year){
  sp_age = function(i,sp){
    ix = which(d$species==sp & (mids[i]-2) < d$dbh & d$dbh <= (mids[i]+2) )
    mean( d$age[ix] )
  }

  d=sql("select species,dbh,age from live_trees where dbh > 4 and year=" %.% year)
  mids = seq(6,118,by=4)

  bw = rep(0, length(mids))
  sm = rep(0, length(mids))
  wa = rep(0, length(mids))
  hm = rep(0, length(mids))
  yb = rep(0, length(mids))
  rm = rep(0, length(mids))

  for (i in 1:length(mids) ){
    bw[i] = sp_age(i, 2)
    sm[i] = sp_age(i, 4)
    wa[i] = sp_age(i,13)
    hm[i] = sp_age(i, 5)
    yb[i] = sp_age(i, 6)
    rm[i] = sp_age(i, 9)
  }

  plot( mids, bw, type="l", lwd=2, col="blue", xlab="", ylab="", ylim=c(0,500) )
  lines( mids, sm, lwd=2, col="green" )
  lines( mids, wa, lwd=2, col="orange")
  lines( mids, hm, lwd=2, col="purple")
  lines( mids, yb, lwd=2, col="yellow")
```

```
  lines( mids, rm, lwd=2, col="red")
  legend("topright", lty=1, lwd=2,
         c("Basswood", "Sugar maple", "White ash",
           "Hemlock", "Yellow birch","Red maple"),
         col=c("blue","green","orange","purple","yellow","red") )
  mtext("Midpoint diameter class (cm)", side=1, line=2)
  mtext("Mean age (yr)", side=2, line=2)
}

stocking_dist_one = function(yr){
  d = sql(sprintf('select stocking_level from stocking where year=%i and type=900',
    yr))

  hist( pmin(499,d$stocking_level), seq(0,500,by=20), xlab='Stocking (%)', main='')
  mtext( sprintf('mean = %.1f', mean(d$stocking_level)) )
}

dgr_vs_dbh_plot_one = function(yr){
  d_now = get_trees(yr)
  d_nxt = get_trees(yr+10)

  d_both = merge(d_now, d_nxt, by="treeno")

  col = rep('black', length(d_both$species.x) )
  col[ d_both$species.x ==  2] = 'blue'
  col[ d_both$species.x ==  4] = 'green'
  col[ d_both$species.x == 13] = 'orange'
  col[ d_both$species.x ==  5] = 'purple'
  col[ d_both$species.x ==  6] = 'yellow'
  col[ d_both$species.x ==  9] = 'red'

  pch = rep(0, length(d_both$species.x))
  pch[d_both$species.x ==  2] = 1
  pch[d_both$species.x ==  4] = 2
  pch[d_both$species.x == 13] = 15
  pch[d_both$species.x ==  5] = 4
  pch[d_both$species.x ==  6] = 19
  pch[d_both$species.x ==  9] = 6

  plot( d_both$dbh.x,d_both$dbh.y - d_both$dbh.x,
        pch = pch, col = col,
        xlim=c(0,90), ylim=c(0,10),
        xlab='DBH (cm)', ylab='DGR (mm/yr)',
        main=sprintf('Simulation year %i',yr) )
}

ba_plot = function() {
  d = ba_data()
  top = min(2*median(d), 1.1*max(d))
  ix = which(d>= top)
  plot(years, pmin(d, top), xlab='Simulation year', ylab='BA (m2/ha)',
       ylim=c(0,top))
  points( years[ix], rep(top,length(ix)), col=2, pch=24)
}

rba_plot_one = function() {
  scale=get_scale()
```

```
  r=sql(paste("select year,",
    scale,"*sum(3.14159*dbh*dbh/40000) as ba,",
    scale,"*sum(3.14159*dbh*dbh*(species== 2)/40000) as bw_ba,",
    scale,"*sum(3.14159*dbh*dbh*(species== 4)/40000) as sm_ba,",
    scale,"*sum(3.14159*dbh*dbh*(species==13)/40000) as wa_ba,",
    scale,"*sum(3.14159*dbh*dbh*(species==24)/40000) as iw_ba,",
    scale,"*sum(3.14159*dbh*dbh*(species== 5)/40000) as hm_ba,",
    scale,"*sum(3.14159*dbh*dbh*(species== 6)/40000) as yb_ba,",
    scale,"*sum(3.14159*dbh*dbh*(species== 9)/40000) as rm_ba",
    "from live_trees where year%",meas_interval,"=0 group by year;"));

  plot (r$year, 100*r$bw_ba/r$ba, type='l', ylim=c(0,100), col="blue",
        xlab='Simulation Year', ylab='% BA')
  lines(r$year, 100*r$sm_ba/r$ba, col="green")
  lines(r$year, 100*r$wa_ba/r$ba, col="orange")
  lines(r$year, 100*r$iw_ba/r$ba, col="cyan")
  lines(r$year, 100*r$hm_ba/r$ba, col="purple")
  lines(r$year, 100*r$yb_ba/r$ba, col="yellow")
  lines(r$year, 100*r$rm_ba/r$ba, col="red")
  legend("topright",c( "Sugarmaple", "Hemlock", "Yellow Birch", "Red Maple",
    "Ash", "Basswood", "Ironwood"),
         col=c("green","purple","yellow","red","orange","blue","cyan"), lty=1)
  invisible(r)
}

#########################
# Growth function engines
#  These engines will pull growth data out of the database,
#  run it through a function to compute some size metric,
#  and then summarize by 5-year interval
#
# d_thresh - a diameter threshold, growth is computed above this
# metric   - a function of dbh, height, and species to compute
#            the size metric of interest (ba, cuft, bdft, etc).

ingrowth_engine = function( d_thresh, metric, my_filter=function(d){d} ){
  res_cache( eval(substitute(
    function(){ ingrowth_engine_base( d_thresh, metric, my_filter ) },
    list( d_thresh=d_thresh, metric=metric, my_filter=my_filter) )) )
}

ingrowth_engine_base = function( d_thresh, metric, my_filter=function(d){d} ){
  rc = matrix(0, ncol=length(years), nrow=8)
  colnames(rc) = years
  rownames(rc) = c(2,4,13,24,5,6,9,NA)

  cat('i')
  d_prv = get_trees(years[1])
  for( i in 2:length(years)){
    cat('.')
    d_now = get_trees(years[i])

    big_prv = d_prv[which(d_prv$dbh>=d_thresh),]
    big_now = d_now[which(d_now$dbh>=d_thresh),]
```

```
    tn_ig = setdiff( big_now$treeno, big_prv$treeno )
    d = my_filter( d_now[ d_now$treeno %in% tn_ig, ] )

    if (dim(d)[1] > 0 ){
      rc[8,i] = 0

      s2 = tapply( get_scale()*metric(d)/meas_interval, d$species, sum)

      for (j in 1:length(s2)){
        ix = which(rownames(rc)==names(s2)[j])
        if (length(ix)==0){ ix = 8 }
        rc[ix,i] = rc[ix,i] + s2[j]
      }
    }

    d_prv = d_now
  }

  cat('\n')
  rc
}

survivor_growth_engine = function( d_thresh, metric, my_filter=function(d){d} ){
  res_cache( eval(substitute(
    function(){ survivor_growth_engine_base( d_thresh, metric, my_filter ) },
    list( d_thresh=d_thresh, metric=metric, my_filter=my_filter) )) )
}

survivor_growth_engine_base = function( d_thresh, metric, my_filter=function(d){d}) {

  rc = matrix(0, ncol=length(years), nrow=8)
  colnames(rc) = years
  rownames(rc) = c(2,4,13,24,5,6,9,NA)

  cat('s')

  prev_live = get_trees(years[1])
  prev_live = prev_live[prev_live$dbh>=d_thresh,]

  for (i in 2:length(years)){
    cat('.')
    this_live = get_trees(years[i])
    this_live = this_live[this_live$dbh>=d_thresh,]

    r <- sql('select year,treeno,dbh,species,ht from cut_trees')
    this_cut <- r[r$dbh>=d_thresh & years[i-1]<r$year & r$year <= years[i],]

    this_trees = merge(this_live, this_cut, all=T)
    d = my_filter(merge(prev_live, this_trees, by="treeno"))

    if (dim(d)[1]>0){
      s1=metric(data.frame(dbh=d$dbh.x, ht=d$ht.x, species=d$species.x))
      s2=metric(data.frame(dbh=d$dbh.y, ht=d$ht.y, species=d$species.x))

      delta_s = tapply(get_scale()*(s2-s1)/meas_interval, d$species.x, sum)

      for (j in 1:length(delta_s)){
        ix = which(rownames(rc) == names(delta_s)[j])
        if (length(ix) == 0 ){ ix = 8 }
```

```
        rc[ix,i] = rc[ix,i] + delta_s[j]
      }
    }

    prev_live = this_live
  }
  cat('\n')
  rc
}

mortality_engine = function( d_thresh, metric, my_filter=function(d){d} ){
  res_cache( eval(substitute(
    function(){ mortality_engine_base( d_thresh, metric, my_filter ) },
    list( d_thresh=d_thresh, metric=metric, my_filter=my_filter) )) )
}

mortality_engine_base = function( d_thresh, metric, my_filter=function(d){d}){
  rc = matrix(0, ncol=length(years), nrow=8)
  colnames(rc) = years
  rownames(rc) = c(2,4,13,24,5,6,9,NA)

  cat('d')
  for( i in 2:length(years)){
    cat('.')
    d <- sql('select year,treeno,dbh from dead_trees')
    dead_tn = d[d$dbh>=d_thresh & years[i-1]<d$year & d$year<=years[i],]$treeno

    prev_live = get_trees(years[i-1])
    dead = my_filter( prev_live[ prev_live$treeno %in% dead_tn, ] )

    if (dim(dead)[1] > 0 ){
      s_dead = tapply( get_scale()*metric(dead)/meas_interval, dead$species, sum)

      for (j in 1:length(s_dead)){
        ix = which(rownames(rc) == names(s_dead)[j])
        if (length(ix)==0) { ix = 8 }
        rc[ix,i] = rc[ix,i] + s_dead[j]
      }
    }
  }
  cat('\n')
  rc
}


harvest_engine = function( d_thresh, metric ){
  res_cache( eval(substitute(
    function(){ harvest_engine_base( d_thresh, metric ) },
    list( d_thresh=d_thresh, metric=metric) )) )
}

harvest_engine_base = function( d_thresh, metric ){
  rc = matrix(0, ncol=length(years), nrow=8)
  colnames(rc) = years
  rownames(rc) = c(2,4,13,24,5,6,9,NA)

  d <- sql('select year,treeno,dbh,species,ht from cut_trees')
  for (i in 2:length(years)){
    d_i <- d[d$dbh>=d_thresh & years[i-1]<d$year & d$year<=years[i],]
    if (length(d_i$treeno)>0){
```

```
      d_prev = get_trees(years[i-1])
      d_prev = d_prev[ d_prev$treeno %in% d_i$treeno, ]
      if (dim(d_prev)[1] > 0 ){
        s_harv = tapply( get_scale()*metric(d_prev)/meas_interval, d_prev$species, sum)
        for (j in 1:length(s_harv)){
          ix = which(rownames(rc) == names(s_harv)[j])
          if (length(ix)==0) { ix = 8 }
          rc[ix,i] = rc[ix,i] + s_harv[j]
        }
      }
    }
  }
  rc
}

windstorm_engine = function( d_thresh, metric, my_filter=function(d){d} ){
  res_cache( eval(substitute(
    function(){ windstorm_engine_base( d_thresh, metric, my_filter) },
      list(d_thresh=d_thresh,metric=metric, my_filter=my_filter))))
}

windstorm_engine_base = function( d_thresh, metric, my_filter=function(d){d}){
  rc = matrix(0, ncol=length(years), nrow=8)
  colnames(rc) = years
  rownames(rc) = c(2,4,13,24,5,6,9,NA)

  d = sql('select year,treeno,dbh,species,ht from wind_trees');
  for (i in 2:length(years)){
    d_i = d[d$dbh >= d_thresh & years[i-1] < d$year & d$year <= years[i],]
    if (length(d_i$treeno)>0){
      d_prev = my_filter(get_trees(years[i-1]))
      d_prev = d_prev[ d_prev$treeno %in% d_i$treeno, ]
      if (dim(d_prev)[1] > 0 ){
        s_wind = tapply( get_scale()*metric(d_prev)/meas_interval, d_prev$species, sum)
        for (j in 1:length(s_wind)){
          ix = which(rownames(rc) == names(s_wind)[j])
          if (length(ix)==0) { ix = 8 }
          rc[ix,i] = rc[ix,i] + s_wind[j]
        }
      }
    }
  }
  rc
}

############################
# Misc regression estimators
estimate_eca = function(dbh_cm, species) {
  rc = rep(NA, length(dbh_cm))

  for (i in 1:length(dbh_cm)){
    if      (species[i]==5) { rc[i] = 0.145 * (dbh_cm[i])^1.34 }
    else if (species[i]==6) { rc[i] = 200.0 * exp(-65.8/dbh_cm[i]) }
    else                    { rc[i] = 0.377 * (dbh_cm[i])^1.22 }
  }

  rc
}

##################################
```

```
# Individual tree volume estimates

# Cubic-foot volumes from G&O:
estimate_volume_go = function( dbh_cm, ht_m ){
  if (length(dbh_cm)>0) {
    dbh_ft = (dbh_cm/2.54)/12
    ht_ft =  3.2808399*ht_m
    ba_ft = pi*(dbh_ft/2)^2
    V_ft=(20<=ht_ft)*(0.42 + 0.06*pmin(1,pmax(0,(30-ht_ft)/10)))*ba_ft*ht_ft
    rc = 0.0283168466*V_ft
  } else {
    rc = numeric(0)
  }
  rc
}

# From Hahn and Hansen (NJAF 8(2):47-57)
estimate_volume_hh = function( dbh_cm, ht_m ){
  S = 65 # SI, assuming Sugar Maple
  b1 = 118.80
  b2 = 0.2106
  b3 = -0.07184/1000
  b4 = 2.724

  dbh_in = dbh_cm/2.54

  V_ft = b1 * S^b2 * (1-exp(b3*dbh_in^b4))
  0.0283168466*V_ft
}

estimate_scribner_volume_go = function( dbh_cm, logs ){
  rc = rep(0, length(dbh_cm))
  dbh_in = dbh_cm / 2.54

  ## Table 1 from Tech Bull 1104, excluding the softwood numbers
  bdft = rbind(
    c(   NA,   NA,   NA,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   13,   NA,   NA,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   17,   30,   NA,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   22,   38,   51,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   28,   48,   66,   78,   NA,   NA,   NA,   NA,   NA),
    c(   34,   59,   81,   96,  112,   NA,   NA,   NA,   NA),
    c(   40,   70,   96,  116,  141,  160,   NA,   NA,   NA),
    c(   47,   81,  113,  137,  166,  188,  204,   NA,   NA),
    c(   54,   93,  129,  158,  191,  224,  248,  263,   NA),
    c(   63,  106,  148,  182,  218,  257,  285,  308,  340),
    c(   72,  122,  168,  207,  248,  292,  325,  355,  395),
    c(   81,  137,  190,  234,  280,  328,  368,  405,  455),
    c(   90,  156,  212,  262,  317,  366,  415,  450,  520),
    c(  100,  173,  238,  293,  351,  405,  460,  505,  585),
    c(  111,  194,  262,  328,  392,  450,  510,  560,  660),
    c(  123,  215,  290,  360,  435,  500,  560,  620,  730),
    c(  137,  236,  319,  400,  470,  550,  620,  690,  800),
    c(  149,  258,  348,  440,  520,  600,  680,  760,  880),
    c(  165,  281,  381,  480,  565,  650,  740,  820,  950),
```

```
    c(  179,  305,  415,  520,  620,  710,  800,  890, 1030),
    c(  195,  331,  450,  560,  670,  760,  860,  960, 1120),
    c(  210,  356,  485,  600,  720,  830,  930, 1030, 1200),
    c(  227,  383,  520,  650,  770,  890, 1000, 1110, 1290),
    c(  245,  410,  560,  700,  830,  950, 1080, 1200, 1380),
    c(  260,  440,  600,  740,  890, 1020, 1150, 1280, 1470),
    c(  279,  470,  640,  790,  950, 1080, 1230, 1370, 1560),
    c(  294,  500,  680,  840, 1010, 1160, 1300, 1460, 1670),
    c(  312,  530,  720,  900, 1080, 1230, 1390, 1560, 1790),
    c(  330,  565,  770,  960, 1140, 1310, 1480, 1650, 1900),
    c(  349,  600,  820, 1020, 1210, 1390, 1570, 1750, 2010),
    c(  365,  630,  860, 1070, 1270, 1470, 1660, 1840, 2120),
    c(  384,  660,  900, 1130, 1330, 1550, 1750, 1940, 2240),
    c(  405,  700,  950, 1180, 1400, 1630, 1850, 2050, 2350)
  )
  col_ix = c(1,2,3,4,5,6,7,8,8,9)

  for (i in 1:length(dbh_in)){
    if (dbh_in[i] >= 8 && logs[i] > 0){
      ri = min( 40, round(dbh_in[i])) - 7
      ci = col_ix [ min(9,round(2*logs[i])) ]

      rc[i] = bdft[ ri, ci ]
    }
  }
  rc
}

estimate_scribner_volume = function (dbh_cm, ht_m, species,
  dbh_cm_residual=c() ){
  ## Coefficients from Table 5 in USFS RP NC-222
  ## For converting from cubic foot volume to scribner board feet
  coefs = rbind (
    ##   SM    BW    WA    YB
    c( 1.7, 1.4, 1.3, 2.1 ), # 10
    c( 2.3, 2.1, 2.2, 2.5 ), # 10.5
    c( 2.8, 2.6, 2.7, 2.7 ), # 11
    c( 3.2, 2.9, 3.0, 2.9 ), # 11.5
    c( 3.5, 3.2, 3.3, 3.1 ), # 12
    c( 3.7, 3.5, 3.4, 3.2 ), # 12.5
    c( 3.9, 3.7, 3.5, 3.4 ), # 13
    c( 4.2, 3.9, 3.7, 3.6 ), # 14
    c( 4.4, 4.1, 3.9, 3.8 ), # 15
    c( 4.6, 4.3, 4.0, 3.9 ), # 16
    c( 4.7, 4.4, 4.1, 4.0 ), # 17
    c( 4.8, 4.5, 4.2, 4.1 ), # 18
    c( 4.8, 4.6, 4.3, 4.2 ), # 19
    c( 4.6, 4.7, 4.4, 4.3 )  # 10
  )

  ## Glue the cut trees + residual together to get a list of
  ## stand diameters
  dbh_stand = c(dbh_cm[which(dbh_cm>2.54*10)],
                dbh_cm_residual[which(dbh_cm_residual>2.54*10)])
```

```
  ## Compute the mean stand diameter:
  qmsd = sqrt(sum((2.54*dbh_stand)^2)/length(dbh_stand));

  ## Figure out which row in the table to use
       if (qmsd < 10   ) { stop("Stand too small"); }
  else if (qmsd < 10.5 ) { row_ix =  1 }
  else if (qmsd < 11   ) { row_ix =  2 }
  else if (qmsd < 11.5 ) { row_ix =  3 }
  else if (qmsd < 12   ) { row_ix =  4 }
  else if (qmsd < 12.5 ) { row_ix =  5 }
  else if (qmsd < 13   ) { row_ix =  6 }
  else if (qmsd < 14   ) { row_ix =  7 }
  else if (qmsd < 15   ) { row_ix =  8 }
  else if (qmsd < 16   ) { row_ix =  9 }
  else if (qmsd < 17   ) { row_ix = 10 }
  else if (qmsd < 18   ) { row_ix = 11 }
  else if (qmsd < 19   ) { row_ix = 12 }
  else if (qmsd < 20   ) { row_ix = 13 }
  else                   { row_ix = 14 }

  ## Get the cubic foot volumes
  V_m = sum( estimate_volume_go( dbh_cm, ht_m ) )
  V_ft = V_m / 0.0283168466

  V_sbft = 0
  for ( i in 1:length(V_ft) ){
    if ( dbh_cm[i] >= 2.54*10) {
      if ( species[i] ==  2 ) { col_ix = 2 } # Basswood
      else if ( species[i] == 13 ) { col_ix = 3 } # White Ash
      else if ( species[i] ==  6 ) { col_ix = 4 } # Yellow Birch
      else                         { col_ix = 1 } # Sugar maple, other

      V_sbft = V_sbft + V_ft * coefs[row_ix, col_ix];
    }
  }

  V_sbft
}

estimate_international_volume_go = function(dbh_cm, logs){
  rc = rep(0, length(dbh_cm))
  dbh_in = dbh_cm / 2.54

  ## Table 2 from Tech Bull 1104, excluding the softwood numbers
  bdft = rbind(
    c(   NA,   NA,   NA,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   18,   NA,   NA,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   21,   39,   NA,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   25,   48,   68,   NA,   NA,   NA,   NA,   NA,   NA),
    c(   30,   57,   80,  100,   NA,   NA,   NA,   NA,   NA),
    c(   36,   68,   96,  118,  134,   NA,   NA,   NA,   NA),
    c(   42,   79,  110,  140,  163,  184,   NA,   NA,   NA),
    c(   50,   92,  128,  160,  188,  214,  232,   NA,   NA),
    c(   59,  105,  147,  180,  213,  247,  274,  295,   NA),
    c(   66,  118,  166,  208,  245,  281,  314,  340,  378),
    c(   74,  135,  188,  235,  278,  320,  360,  400,  440),
```

```
    c(   83,  152,  212,  265,  314,  360,  405,  450,  500),
    c(   92,  170,  236,  295,  350,  400,  450,  500,  570),
    c(  102,  189,  262,  328,  390,  450,  505,  550,  635),
    c(  112,  209,  290,  362,  430,  495,  555,  610,  715),
    c(  122,  228,  316,  396,  470,  540,  610,  680,  800),
    c(  133,  252,  346,  430,  510,  595,  670,  740,  870),
    c(  145,  275,  376,  470,  555,  645,  730,  810,  950),
    c(  158,  300,  410,  510,  605,  700,  790,  880, 1020),
    c(  172,  325,  440,  550,  650,  760,  850,  950, 1100),
    c(  187,  348,  480,  595,  700,  810,  920, 1020, 1190),
    c(  203,  378,  515,  640,  760,  870,  990, 1100, 1280),
    c(  220,  410,  550,  685,  810,  930, 1060, 1180, 1360),
    c(  237,  440,  595,  740,  870, 1000, 1140, 1260, 1450),
    c(  254,  470,  635,  790,  930, 1070, 1210, 1350, 1550),
    c(  270,  500,  680,  840,  990, 1140, 1290, 1440, 1650),
    c(  291,  530,  725,  900, 1060, 1210, 1380, 1530, 1760),
    c(  311,  565,  770,  950, 1120, 1290, 1460, 1630, 1880),
    c(  333,  600,  820, 1010, 1190, 1370, 1550, 1725, 2000),
    c(  353,  635,  860, 1070, 1260, 1450, 1640, 1830, 2120),
    c(  374,  670,  910, 1120, 1330, 1530, 1730, 1930, 2240),
    c(  394,  705,  960, 1180, 1400, 1620, 1830, 2040, 2360),
    c(  415,  745, 1010, 1250, 1480, 1700, 1930, 2160, 2480)
  )
  col_ix = c(1,2,3,4,5,6,7,8,8,9)

  for( i in 1:length(dbh_in)){
    if ( dbh_in[i] >= 8 && logs[i] > 0 ){
      ri = min( 40, round(dbh_in[i])) - 7
      ci = col_ix [ min(9,round(2*logs[i])) ]
      rc[i] = bdft[ ri, ci ]
    }
  }
  rc
}

# From Hahn and Hansen (NJAF 8(2):47-57)
estimate_international_volume_hh = function(dbh_cm, ht_m){
  S = 65 # SI, assuming Sugar Maple
  b1 = 585.0
  b2 = 0.2590
  b3 = -0.03684/1000
  b4 = 2.925

  dbh_in = dbh_cm/2.54

  b1*S^b2*(1-exp(b3*dbh_in^b4))
}

###
# Functions to estimate merchantable height

# The Pubanz Rule:
estimate_logs_rule_2 = function ( dbh_cm ) {
  rc = rep(0, length(dbh_cm))

  for( i in 1:length(dbh_cm)){
```

```
    if      ( dbh_cm[i] < 23 ) { rc[i] = 0   }
    else if ( dbh_cm[i] < 40 ) { rc[i] = 1.5 }
    else if ( dbh_cm[i] < 50 ) { rc[i] = 2   }
    else {
      rc[i] = sample( c(2,2.5), 1, prob=rep(0.5,2))
    }
  }

  rc
}

# The Schmierer rule:
estimate_logs_rule_3 = function ( dbh_cm ) {
  flip = function(heads,tails) { sample( c(heads,tails), 1, prob=rep(0.5,2)) }

  rc = rep(0, length(dbh_cm))

  for( i in 1:length(dbh_cm)) {
    if      ( dbh_cm[i] < 25 ) { rc[i] = 0   }
    else if ( dbh_cm[i] < 30 ) { rc[i] = 0.5 }
    else if ( dbh_cm[i] < 35 ) { rc[i] = flip( 0.5, 1   ) }
    else if ( dbh_cm[i] < 40 ) { rc[i] = flip( 1.0, 1.5) }
    else if ( dbh_cm[i] < 45 ) { rc[i] = flip( 1.5, 2.0) }
    else                       { rc[i] = flip( 2.0, 2.5) }
  }

  rc
}

# Merchantable height estimates calibrated from the NH-25 experiment.

estimate_merch_ht_nh25 = function(dbh) {
  logistic = function(x) { 1/(1+exp(-x)) }
  m1 = function(d){  75.852551  - 6.2071040*d + 0.1551870*d^2 - 0.0012440*d^3 }
  m2 = function(d){  -3.014e+01 + 2.272e+00*d - 6.077e-02*d^2 + 4.914e-04*d^3 }
  m3 = function(d){  53.7442009 - 4.2361741*d + 0.1046736*d^2 - 0.0008441*d^3 }
  m4 = function(d){  9.818118   - 0.5562640*d + 0.0065630*d^2 }
  m5 = function(d){ -15.511793  + 0.6627420*d - 0.0069020*d^2 }
  m6 = function(d){ -33.55647   + 1.5691800*d - 0.0188500*d^2 }
  m7 = function(d){ -30.04685   + 1.2891100*d - 0.0151600*d^2 }

  rc = rep(NA, length(dbh))

  pr_raw = matrix(NA, nrow=length(dbh), ncol=7)
  pr_raw[,1] = 1.5549*logistic( m1(dbh) )
  pr_raw[,2] = 0.6886*logistic( m2(dbh) )
  pr_raw[,3] = 0.8835*logistic( m3(dbh) )
  pr_raw[,4] = 0.9615*logistic( m4(dbh) )
  pr_raw[,5] = 1.0487*logistic( m5(dbh) )
  pr_raw[,6] = 0.8403*logistic( m6(dbh) )
  pr_raw[,7] = 1.1864*logistic( m7(dbh) )

  for( i in 1:length(dbh) ){
    ## Force probability to zero for impossible dbh/size combinations
    if (dbh[i] <=  9*2.54) { pr_raw[i,2:7] = 0 }
    else if (dbh[i] <= 10*2.54) { pr_raw[i,3:7] = 0 }
    else if (dbh[i] <= 11*2.54) { pr_raw[i,4:7] = 0 }
    else if (dbh[i] <= 12*2.54) { pr_raw[i,5:7] = 0 }
```

```
    else if (dbh[i] <= 13*2.54) { pr_raw[i,6:7] = 0 }
    else if (dbh[i] <= 14*2.54) { pr_raw[i,  7] = 0 }

    rc[i] = sample( seq(0,3,by=0.5), 1, prob=pr_raw[i,]/sum(pr_raw[i,]))
  }
  rc
}

## Biomass estimation -- needs documetation on equation sources
biomass_kg = function(d){
  rc = rep(0, length(d$dbh))

  for (i in 1:length(d$dbh)){
    ## Standardize on biomass in grams:
    if (d$species[i] == 9){
      ## Red maple
      rc[i] = 178.9*d$dbh[i]^2.334  # Crow and Erdmann 1983
    } else if ( d$species[i] == 4){
      ## Sugar maple:
      rc[i] = 179.1*d$dbh[i]^2.3329  # Young 1980
    } else if (d$species[i] == 6) {
      ## Yellow birch:
      rc[i] = 158.8*d$dbh[i]^2.3376  # Young 1980
    } else if (d$species[i] == 13 || d$species[i] == 14){
      ## White/green ash:
      rc[i] = 153.5*d$dbh[i]^2.3213  # Ker 1980
    } else if (d$species[i] == 24 ){
      ## Ironwood
      rc[i] = 1000*(5.5247-3.352*d$dbh[i]+0.006551*(10*d$dbh[i])^2) # Monteith 1979
    } else if (d$species[i]==5){
      ## Hemlock:
      rc[i] = 1000*(6.1371-2.785*d$dbh[i]+0.004286*(10*d$dbh[i])^2) # Monteith 1979
    } else if (d$species[i]==2){
      ## Basswood:
      rc[i] = 87.2*d$dbh[i]^2.3539 # Perala and Alban 1994
    } else if (d$species[i]==43){
      ## White spruce:
      rc[i] = 107.7*d$dbh[i]^2.3308 # Ker 1984
    } else if (d$species[i]==46){
      ## White cedar:
      rc[i] = 230.5*d$dbh[i]^1.9269 # Young et al 1980
    } else if (d$species[i]==41){
      ## Balsam fir
      rc[i] = 174.6*d$dbh[i]^2.1555 # Ker 1984
    } else {
      ## Otherwise, use Monk et al 1970 for "General hardwoods"
      rc[i] = 10^(1.9757 + 2.5371*log10(d$dbh[i])) # Units checked
    }
  }
  rc/1000 # biomass in kg
}

tx_pr_table = function(year) {
  ## Compute the transition probabilities from
  ## year to year+20.
```

```
  tx = matrix(nrow=30, ncol=8);
  colnames(tx)=c("Initial", "NoChange", "+1", "+2", "+3", "+4","Died","Final");

  rn=c();
  for( sc in 1:30 ){
    rn=c(rn, sprintf("%i <= d < %i", 4*(sc-1), 4*sc))
  }
  rownames(tx)=rn;

  present=sql(sprintf('select treeno,dbh from sizes where year==%i',year));
  future =sql(sprintf('select treeno,dbh from sizes where year==%i',year+20));
  fdead  =sql(sprintf('select treeno from dead where year between %i and %i',
    year, year+20));

  colnames(tx)=c("Initial", "NoChange", "+1", "+2", "+3", "+4","Died","Final");

  for( sc in 1:30 ){
    lwr = 4*(sc-1);
    upr = 4*sc;

    cur_trees = present$treeno[which(lwr<=present$dbh & present$dbh<upr)];

    tx[sc,1]=length(cur_trees);
    n=max(1,tx[sc,1]);

    for( up in 0:4 ){
      ix=which( lwr+4*up <= future$dbh & future$dbh < upr+4*up );
      tx[sc,2+up] = length( intersect( cur_trees, future$treeno[ix]))/n;
    }

    tx[sc,7]=length(intersect(cur_trees, fdead$treeno))/n;
    tx[sc,8]=length( which( lwr<= future$dbh & future$dbh < upr ) );
  }
  print(tx,3);
}

volume_plot_go = function(sm_lwr=NA, sm_upr=NA) {
  scale=get_scale()/2.47105381;

  d=sql('select year,dbh,ht from cut_trees');

  if( length(d)>0 ){
    tree_vol=estimate_volume_go( d$dbh, d$ht );
    harvest_vol = tapply( tree_vol, d$year, sum );

    plot( unique(d$year), scale*harvest_vol,
         xlab='Simulation Year', ylab='Harvested Volume (ft^3/ac)');
    lines(c(min(d$year),max(d$year)),rep(sm_lwr,2),col=3);
    lines(c(min(d$year),max(d$year)),rep(sm_upr,2),col=3);
  }
}

###################
## standard_analysis.r
##   Routines to analyze a batch of simulations.

yearly_summary_engine = function( metric, use_scale=T ){
  res_cache( eval(substitute(
    function(){ yearly_summary_engine_base( metric, use_scale ) },
    list( metric=metric, use_scale=use_scale) )) )
```

```
}

yearly_summary_engine_base = function( metric, use_scale=T ){
  rc = rep(NA, length(years))

  if (use_scale) { scale_factor = get_scale() }
  else           { scale_factor = 1 }

  for (i in 1:length(years)){
    rc[i] = scale_factor * metric( get_trees(years[i]) )
  }

  rc
}

stock_data = function(){ res_cache( function(){ stock_data_base() } ) }

stock_data_base = function() {
  rc = rep(NA, length(years))

  for (i in 1:length(years)){
    d = sql(sprintf('select avg(stocking_level) as x from stocking ' %.%
      'where year = %i and type=900' , years[i] ))
    rc[i] = d$x
  }

  rc
}

ba_data = function() {
  yearly_summary_engine( function(d){sum(pi*(d$dbh/200)^2)} )}

peca_data = function() {
  yearly_summary_engine( function(d){sum(d$eca)/100} )}

ptca_data = function() {
  yearly_summary_engine( function(d){sum(d$tca)/100} )}

standing_volume = function() {
  yearly_summary_engine( function(d){ sum(
      ifelse(d$dbh>=4.6*2.54,estimate_volume_go(d$dbh,d$ht),0))} )}

standing_biomass = function() {
  yearly_summary_engine( function(d){ sum( ifelse(d$dbh>=4.6*2.54,biomass_kg(d),0))/1000
      } )}


## Need is_over to be defined elsewhere:

ba_over = function(){
  yearly_summary_engine( function(d){sum( ifelse(is_over(d), pi*(d$dbh/200)^2, 0)) } )}

volume_over = function(){
  yearly_summary_engine( function(d){ sum( ifelse(d$dbh>=4.6*2.54 & is_over(d),
      estimate_volume_go(d$dbh,d$ht),0))} )}

biomass_over = function() {
  yearly_summary_engine( function(d){ sum( ifelse(d$dbh>=4.6*2.54 & is_over(d),
      biomass_kg(d),0))/1000 } )}

cnt_summary_by_size = function(lwr,upr) {
```

```
  yearly_summary_engine(
    eval( substitute( function(d){sum(d$sp!=24 & lwr<=d$dbh & d$dbh<upr)},
                      list(lwr=lwr, upr=upr) )) )}

ba_summary_by_size = function(lwr,upr) {
  yearly_summary_engine(
    eval(substitute( function(d){ sum( (d$sp!=24 & lwr<=d$dbh &
        d$dbh<upr)*pi*(d$dbh/200)^2 )},
                      list(lwr=lwr, upr=upr) )) )}

pba_summary_by_size = function(lwr,upr) {
  yearly_summary_engine( eval( substitute( function(d){
    ba = pi*(d$dbh/200)^2
    100 * sum( (d$sp!=24 & lwr<=d$dbh & d$dbh<upr)*ba) / sum(ba) }, list(lwr=lwr,
        upr=upr))) ,F )}

eca_summary_by_size = function(lwr,upr) {
  yearly_summary_engine(
    eval(substitute( function(d){ sum( (d$sp!=24 & lwr<=d$dbh & d$dbh<upr)*d$eca )},
                     list(lwr=lwr, upr=upr) )) )}

peca_summary_by_size = function(lwr,upr) {
  yearly_summary_engine(
    eval(substitute( function(d){ 100*sum((d$sp!=24 & lwr<=d$dbh &
        d$dbh<upr)*d$eca)/sum(d$eca)},
                      list(lwr=lwr, upr=upr) )), F )}

est_eca_by_size = function(lwr,upr) {
  yearly_summary_engine(
    eval(substitute(
      function(d){sum( (d$sp!=24 & lwr<=d$dbh & d$dbh<upr) * estimate_eca(d$dbh,
          d$species) ) },
      list(lwr=lwr, upr=upr) )) )}

seed_trees     = function(){ cnt_summary_by_size( 2,   6) }
sapling_trees  = function(){ cnt_summary_by_size( 2,  11) }
pole_trees     = function(){ cnt_summary_by_size(11,  26) }
mature_trees   = function(){ cnt_summary_by_size(26,  46) }
large_trees    = function(){ cnt_summary_by_size(46,1000) }
xlarge_trees   = function(){ cnt_summary_by_size(66,1000) }
g50_trees      = function(){ cnt_summary_by_size(50,1000) }
all_trees      = function(){ cnt_summary_by_size( 2,1000) }

seed_trees_ba     = function(){ ba_summary_by_size( 2,     6) }
sapling_trees_ba  = function(){ ba_summary_by_size( 2,    11) }
pole_trees_ba     = function(){ ba_summary_by_size( 11,   26) }
mature_trees_ba   = function(){ ba_summary_by_size( 26,   46) }
large_trees_ba    = function(){ ba_summary_by_size( 46, 1000) }
xlarge_trees_ba   = function(){ ba_summary_by_size( 66, 1000) }

lm_ba_ratio       = function(){
  large_trees_ba() / pmax(mature_trees_ba(),0.01)  }

seed_trees_pba     = function(){ pba_summary_by_size( 2,     6) }
sapling_trees_pba  = function(){ pba_summary_by_size( 2,    11) }
```

```
pole_trees_pba     = function(){ pba_summary_by_size( 11,   26) }
mature_trees_pba   = function(){ pba_summary_by_size( 26,   46) }
large_trees_pba    = function(){ pba_summary_by_size( 46, 1000) }
xlarge_trees_pba   = function(){ pba_summary_by_size( 66, 1000) }

gap_sap_trees_pba = function(eca_thresh=0.25, area_thresh=3.25){
  res_cache( eval(substitute(
    function(){ gap_sap_trees_pba_base( eca_thresh, area_thresh ) },
    list( eca_thresh=eca_thresh, area_thresh=area_thresh ) )) )
}

gap_sap_trees_pba_base = function(eca_thresh=0.25, area_thresh=3.25){
  rc = rep(NA, length(years))

  for (i in 1:length(years)){
    d_trees = get_trees(years[i])
    d_gaps = sql(sprintf(
      'select year, treeno, gap_area from gaps where year%%%i=0', meas_interval))
    d_gaps = d_gaps[ d_gaps$year==years[i], ]
    d = merge( d_trees, d_gaps, by="treeno", all.x=T)

    # Cut out the very big trees.
    d=d[d$dbh<=80,]

    all_ba = get_scale() * pi * sum( (d$dbh/200)^2 )
    rc[i] = 100*compute_gap_sap_ba(d, eca_thresh, area_thresh) / all_ba
  }

  rc
}

sp_trees_pba       = function(){ pba_summary_by_size(  2,   26) }
ml_trees_pba       = function(){ pba_summary_by_size( 26, 1000) }

seed_trees_eca     = function(){ eca_summary_by_size( 2,     6) }
sapling_trees_eca  = function(){ eca_summary_by_size( 2,    11) }
pole_trees_eca     = function(){ eca_summary_by_size( 11,   26) }
mature_trees_eca   = function(){ eca_summary_by_size( 26,   46) }
large_trees_eca    = function(){ eca_summary_by_size( 46, 1000) }
xlarge_trees_eca   = function(){ eca_summary_by_size( 66, 1000) }

seed_trees_peca     = function(){ peca_summary_by_size( 2,     6) }
sapling_trees_peca  = function(){ peca_summary_by_size( 2,    11) }
pole_trees_peca     = function(){ peca_summary_by_size( 11,   26) }
mature_trees_peca   = function(){ peca_summary_by_size( 26,   46) }
large_trees_peca    = function(){ peca_summary_by_size( 46, 1000) }
xlarge_trees_peca   = function(){ peca_summary_by_size( 66, 1000) }

est_seed_trees_eca     = function(){ est_eca_by_size( 2,     6) }
est_sapling_trees_eca  = function(){ est_eca_by_size( 2,    11) }
est_pole_trees_eca     = function(){ est_eca_by_size( 11,   26) }
est_mature_trees_eca   = function(){ est_eca_by_size( 26,   46) }
est_large_trees_eca    = function(){ est_eca_by_size( 46, 1000) }
est_xlarge_trees_eca   = function(){ est_eca_by_size( 66, 1000) }

mort_volume = function(){
  rc = rep(NA,length(years));
```

```
  for( i in 2:length(years)) {
    d <- sql('select year,treeno,dbh,ht from dead_trees')
    d <- d[ years[i-1] < d$year & d$year <= years[i], ]
    rc[i] = sum( estimate_volume_go( d$dbh, d$ht ) )
  }
  get_scale()*rc
}

harvest_volume = function() {
  rc=rep(NA, length(years))
  for (i in 2:length(years) ){
    r <- sql('select year,treeno,dbh,ht from cut_taken_trees')
    r <- r[ years[i-1]< r$year & r$year <= years[i], ]
    rc[i] = sum(estimate_volume_go( r$dbh, r$ht))
  }
  get_scale()*rc
}

canopy_tree_mean_diam = function(){
  rc = rep(NA,length(years))

  for (i in 1:length(years)){
    d = get_trees(years[i])
    d = d[ d$dbh >= 11 & d$eca / d$tca >= 0.20, ]

    rc[i] = mean(d$dbh)
  }
  rc
}

canopy_tree_mean_th = function(){
  rc = rep(NA,length(years))

  for (i in 1:length(years)){
    d = get_trees(years[i])
    d = d[ d$dbh >= 11 & d$eca / d$tca >= 0.20, ]

    rc[i] = mean(d$ht)
  }
  rc
}

canopy_tree_mean_age = function(){
  rc = rep(NA, length(years))

  for (i in 1:length(years)){
    d = get_trees(years[i])
    d = d[ d$dbh >= 11 & d$eca / d$tca >= 0.20, ]

    rc[i] = mean(d$age)
  }
  rc
}

canopy_tree_mean_mcr = function(){
  rc = rep(NA, length(years))

  for (i in 1:length(years)){
    d = get_trees(years[i])
    d = d[ d$dbh >= 11 & d$eca / d$tca >= 0.20, ]
```

```
    rc[i] = mean(d$mcr)
  }

  rc
}

qm_stand_diameter = function(){
  rc = rep(NA, length(years))

  for (i in 1:length(years)){
    d = get_trees(years[i])
    rc[i] = sqrt( mean(d$dbh^2) )
  }
  rc
}

canopy_tree_growth = function(){
  rc = rep(NA,length(years))

  d_prv = get_trees(years[1])
  for( i in 2:length(years)){
    d_now = get_trees(years[i])

    d_all = my_merge(
      d_prv[ d_prv$dbh >= 11 & d_prv$eca/d_prv$tca >= 0.20, ],
      d_now[ d_now$dbh >= 11 & d_now$eca/d_now$tca >= 0.20, ],
      by="treeno");

    rc[i]= 10*mean( d_all$dbh.y - d_all$dbh.x )/2
    d_prv = d_now
  }
  rc/meas_interval
}

canopy_tree_crown_growth = function(){
  rc = rep(NA,length(years));

  for( i in 2:length(years)){
    d_prv = get_trees(years[i-1]);
    d_now = get_trees(years[i]);

    d_all = my_merge(
      d_prv[ d_prv$dbh >= 11 & d_prv$eca/d_prv$tca >= 0.20, ],
      d_now[ d_now$dbh >= 11 & d_now$eca/d_now$tca >= 0.20, ],
      by="treeno");

    rc[i]= 10*mean( d_all$mcr.y - d_all$mcr.x )/2;
  }
  100*rc/meas_interval # Convert to cm/yr
}

radial_growth_by_size = function(lwr,upr){
  rc = rep(NA,length(years));

  d_prv = get_trees(years[1])
  for( i in 2:length(years)){
    d_now = get_trees(years[i])

    d_all = my_merge(
      d_prv[ lwr <= d_prv$dbh & d_prv$dbh < upr , ],
      d_now[ lwr <= d_now$dbh & d_now$dbh < upr , ],
```

```
      by="treeno");

    rc[i]= 10*mean( d_all$dbh.y - d_all$dbh.x )/2
    d_prv = d_now
  }
  rc/meas_interval
}

seed_radial_growth= function() { radial_growth_by_size( 0,   6); }
sap_radial_growth = function() { radial_growth_by_size( 0,  11); }
pol_radial_growth = function() { radial_growth_by_size(11,  26); }
mat_radial_growth = function() { radial_growth_by_size(26,  46); }
lrg_radial_growth = function() { radial_growth_by_size(46,1000); }
xlg_radial_growth = function() { radial_growth_by_size(66,1000); }

mortality_by_size = function(lwr,upr) {
  rc <- rep(NA,length(years));

  d <- sql('select year,treeno,dbh,ht from dead_trees')
  for (i in 2:length(years)){
    di <- d[lwr <= d$dbh & d$dbh < upr &
            years[i-1] < d$year & d$year <= years[i], ]
    rc[i] <- sum( pi*(di$dbh/200)^2 );
  }
  get_scale()*rc/meas_interval
}

seed_mort = function() { mortality_by_size( 0,   6); }
sap_mort  = function() { mortality_by_size( 0,  11); }
pol_mort  = function() { mortality_by_size(11,  26); }
mat_mort  = function() { mortality_by_size(26,  46); }
lrg_mort  = function() { mortality_by_size(46,1000); }
xlg_mort  = function() { mortality_by_size(66,1000); }

gap_formation_rate = function(){
  data_items = 'treeno,year,3.14/4*(nexp*eexp+eexp*sexp+sexp*wexp+wexp*nexp) as eca'

  dead_eca = sql(
    'select ' %.% data_items %.% ' from dead_trees ' %.%
    'union select ' %.% data_items %.% ' from wind_trees ' %.%
    'union select ' %.% data_items %.% ' from cut_trees' )

  rc = rep(NA, length(years))

  for (i in 2:length(years)){
    ix = which( years[i-1] <= dead_eca$year & dead_eca$year < years[i] )
    rc[i] = get_scale() * sum( dead_eca[ix,]$eca )
  }

  rc
}

ingrowth_ba = function(dlim){
  ingrowth_engine(dlim, function(d){pi*(d$dbh/200)^2})}

survivor_growth_ba = function(dlim) {
  survivor_growth_engine(dlim, function(d){pi*(d$dbh/200)^2})}

mortality_ba = function(dlim) {
```

```
  mortality_engine( dlim, function(d){pi*(d$dbh/200)^2})}

gross_growth_ba = function(dlim){
  ingrowth_ba(dlim) + survivor_growth_ba(dlim) }

net_growth_ba   = function(dlim){
  gross_growth_ba(dlim) - mortality_ba(dlim) }

ingrowth        = function(){         ingrowth_ba( 4.6*2.54 ) }
survivor_growth = function(){ survivor_growth_ba( 4.6*2.54 ) }
mortality       = function(){        mortality_ba( 4.6*2.54 ) }
gross_growth    = function(){     gross_growth_ba( 4.6*2.54 ) }
net_growth      = function(){       net_growth_ba( 4.6*2.54 ) }

## Need to define cohort_over and cohort_under functions
## elsewhere which can be used here.

ig_over_cohort = function(){
  cohort_over( ingrowth_engine, 4.6*2.54, function(d){pi*(d$dbh/200)^2} )}
sg_over_cohort = function(){
  cohort_over( survivor_growth_engine, 4.6*2.54, function(d){pi*(d$dbh/200)^2} )}
mt_over_cohort = function(){
  cohort_over( mortality_engine, 4.6*2.54, function(d){pi*(d$dbh/200)^2} )}
ng_over_cohort = function(){
  ig_over_cohort() + sg_over_cohort() - mt_over_cohort() }

ig_under_cohort = function(){
  cohort_under( ingrowth_engine, 4.6*2.54, function(d){pi*(d$dbh/200)^2} )}
sg_under_cohort = function(){
  cohort_under( survivor_growth_engine, 4.6*2.54, function(d){pi*(d$dbh/200)^2} )}
mt_under_cohort = function(){
  cohort_under( mortality_engine, 4.6*2.54, function(d){pi*(d$dbh/200)^2} )}
ng_under_cohort = function(){
  ig_under_cohort() + sg_under_cohort() - mt_under_cohort() }

ingrowth_saw         = function(){         ingrowth_ba( 9.6*2.54 ) }
survivor_growth_saw  = function(){ survivor_growth_ba( 9.6*2.54 ) }
mortality_saw        = function(){        mortality_ba( 9.6*2.54 ) }
gross_growth_saw     = function(){     gross_growth_ba( 9.6*2.54 ) }
net_growth_saw       = function(){       net_growth_ba( 9.6*2.54 ) }

ingrowth_kg         = function(){         ingrowth_engine(4.6*2.54, biomass_kg) }
survivor_growth_kg  = function(){ survivor_growth_engine(4.6*2.54, biomass_kg) }
mortality_kg        = function(){        mortality_engine(4.6*2.54, biomass_kg) }
gross_growth_kg     = function(){  ingrowth_kg() + survivor_growth_kg() }
net_growth_kg       = function(){  gross_growth_kg() - mortality_kg() - wind_kg() }
wind_kg             = function(){      windstorm_engine(4.6*2.54, biomass_kg) }

ig_kg_over_cohort = function(){ cohort_over(         ingrowth_engine, 4.6*2.54,
    biomass_kg)}
sg_kg_over_cohort = function(){ cohort_over( survivor_growth_engine, 4.6*2.54,
    biomass_kg)}
mt_kg_over_cohort = function(){ cohort_over(        mortality_engine, 4.6*2.54,
    biomass_kg)}
```

```
wind_kg_over_cohort=function(){ cohort_over(        windstorm_engine, 4.6*2.54,
    biomass_kg)}
ng_kg_over_cohort = function(){
  ig_kg_over_cohort() + sg_kg_over_cohort() - mt_kg_over_cohort() - wind_kg_over_cohort()
      }

ig_kg_under_cohort = function(){ cohort_under(         ingrowth_engine, 4.6*2.54,
    biomass_kg)}
sg_kg_under_cohort = function(){ cohort_under( survivor_growth_engine, 4.6*2.54,
    biomass_kg)}
mt_kg_under_cohort = function(){ cohort_under(        mortality_engine, 4.6*2.54,
    biomass_kg)}
wind_kg_under_cohort=function(){ cohort_under(        windstorm_engine, 4.6*2.54,
    biomass_kg)}
ng_kg_under_cohort = function(){
  ig_kg_under_cohort() + sg_kg_under_cohort() - mt_kg_under_cohort()-
      wind_kg_under_cohort() }

ingrowth_all_kg         = function(){         ingrowth_engine(3, biomass_kg) }
survivor_growth_all_kg = function(){ survivor_growth_engine(3, biomass_kg) }
mortality_all_kg        = function(){        mortality_engine(3, biomass_kg) }
gross_growth_all_kg     = function(){  ingrowth_all_kg() + survivor_growth_all_kg() }
net_growth_all_kg       = function(){  gross_growth_all_kg() - mortality_all_kg()   }

#####
# Cubic volume growth functions

survivor_growth_m3 = function(){ survivor_growth_engine( 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)} )}
ingrowth_m3         = function(){         ingrowth_engine( 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)} )}
mortality_m3        = function(){        mortality_engine( 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)} )}
harvest_m3          = function(){          harvest_engine( 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)} )}
gross_growth_m3     = function(){ ingrowth_m3() + survivor_growth_m3() }
net_growth_m3       = function(){ gross_growth_m3() - mortality_m3()   }

ig_m3_over_cohort = function(){ cohort_over(         ingrowth_engine, 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)})}
sg_m3_over_cohort = function(){ cohort_over( survivor_growth_engine, 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)})}
mt_m3_over_cohort = function(){ cohort_over(        mortality_engine, 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)})}
ng_m3_over_cohort = function(){
  ig_m3_over_cohort() + sg_m3_over_cohort() - mt_m3_over_cohort() }

ig_m3_under_cohort = function(){ cohort_under(         ingrowth_engine, 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)})}
sg_m3_under_cohort = function(){ cohort_under( survivor_growth_engine, 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)})}
mt_m3_under_cohort = function(){ cohort_under(        mortality_engine, 4.6*2.54,
    function(d){estimate_volume_go(d$dbh,d$ht)})}
ng_m3_under_cohort = function(){
  ig_m3_under_cohort() + sg_m3_under_cohort() - mt_m3_under_cohort() }
```

```
############################
# bf Volume growth functions
#  These don't yet support plotting
#  By default, they use the nh25 rule to estimate
#    but that can be changed by passing in a different est function
survivor_growth_bf = function() {
  survivor_growth_engine( 9.6*2.54,
                          function(dbh,ht){
                            estimate_scribner_volume_go(dbh,
                                estimate_merch_ht_nh25(dbh))})}
ingrowth_bf = function(){
  ingrowth_engine( 9.6*2.54,
                   function(dbh,ht){
                     estimate_scribner_volume_go(dbh, estimate_merch_ht_nh25(dbh))})}
mortality_bf = function() {
  mortality_engine( 9.6*2.54,
                    function(dbh,ht){
                      estimate_scribner_volume_go(dbh, estimate_merch_ht_nh25(dbh))})}
harvest_bf = function() {
  harvest_engine( 9.6*2.54,
                  function(dbh,ht){
                    estimate_scribner_volume_go(dbh, estimate_merch_ht_nh25(dbh))})}

## CWD volume:
cwd_volume = function(year,cl,sp=NA){
  scale=get_scale()
  d=get_cwd_dbh(year,cl);
  if( is.na( sp ) )     { ix = which( d$dbh>=20) }
  else if ( sp=="HM" ) { ix = which( d$dbh>=20 & d$species==5) }
  else                  { ix = which( d$dbh>=20 & d$species!=5) }
  scale*sum(estimate_volume_go( d$dbh[ix], d$ht[ix]));
}

vol_log1 = function(year,sp=NA){ cwd_volume(year, 'log_1',sp) }
vol_log2 = function(year,sp=NA){ cwd_volume(year, 'log_2',sp) }
vol_log3 = function(year,sp=NA){ cwd_volume(year, 'log_3',sp) }
vol_log4 = function(year,sp=NA){ cwd_volume(year, 'log_4',sp) }
vol_log5 = function(year,sp=NA){ cwd_volume(year, 'log_5',sp) }

vol_snag1 = function(year){ cwd_volume(year, 'snag_1') }
vol_snag2 = function(year){ cwd_volume(year, 'snag_2') }
vol_snag3 = function(year){ cwd_volume(year, 'snag_3') }
vol_snag4 = function(year){ cwd_volume(year, 'snag_4') }
vol_snag5 = function(year){ cwd_volume(year, 'snag_5') }

cwd_vol_compute = function(logs, snags){
  rc = rep(0,length(years));
  for( i in 1:length(years) ){
    if( sum(logs)>0 ) {
      rc[i] = rc[i]                                          +
              logs[1]*vol_log1(years[i])                     +
              logs[2]*vol_log2(years[i],"SM")*(1-0.19)       +
              logs[3]*vol_log3(years[i],"SM")*(1-0.32)       +
```

```
              logs[4]*vol_log4(years[i],"SM")*(1-0.54) +
              logs[5]*vol_log5(years[i],"SM")*(1-0.73) +
              logs[2]*vol_log2(years[i],"HM")*(1-0.14) +
              logs[3]*vol_log3(years[i],"HM")*(1-0.42) +
              logs[4]*vol_log4(years[i],"HM")*(1-0.73) +
              logs[5]*vol_log5(years[i],"HM")*(1-0.73)
    }
    if( sum(snags)>0 ){
      rc[i] = rc[i]                          +
              snags[1]*vol_snag1(years[i]) +
              snags[2]*vol_snag2(years[i]) +
              snags[3]*vol_snag3(years[i]) +
              snags[4]*vol_snag4(years[i]) +
              snags[5]*vol_snag5(years[i])
    }
  }
  rc;
}

vol_all    = function() { cwd_vol_compute(rep(1,5),rep(1,5)) }
vol_logs   = function() { cwd_vol_compute(rep(1,5),rep(0,5)) }
vol_snags  = function() { cwd_vol_compute(rep(0,5),rep(1,5)) }
vol_log1s  = function() { cwd_vol_compute(c(1,0,0,0,0), rep(0,5)) }
vol_log2s  = function() { cwd_vol_compute(c(0,1,0,0,0), rep(0,5)) }
vol_log3s  = function() { cwd_vol_compute(c(0,0,1,0,0), rep(0,5)) }
vol_log4s  = function() { cwd_vol_compute(c(0,0,0,1,0), rep(0,5)) }
vol_log5s  = function() { cwd_vol_compute(c(0,0,0,0,1), rep(0,5)) }
vol_snag1s = function() { cwd_vol_compute(rep(0,5), c(1,0,0,0,0)) }
vol_snag2s = function() { cwd_vol_compute(rep(0,5), c(0,1,0,0,0)) }
vol_snag3s = function() { cwd_vol_compute(rep(0,5), c(0,0,1,0,0)) }
vol_snag4s = function() { cwd_vol_compute(rep(0,5), c(0,0,0,1,0)) }
vol_snag5s = function() { cwd_vol_compute(rep(0,5), c(0,0,0,0,1)) }

## CWD biomass:
cwd_biomass = function(year,cl,sp=NA){
  rc = 0
  d = get_cwd_dbh(year,cl);
  if( is.na( sp ) )     { ix = which( d$dbh>=20) }
  else if ( sp=="HM" ) { ix = which( d$dbh>=20 & d$species==5) }
  else                  { ix = which( d$dbh>=20 & d$species!=5) }
  if (length(ix)>0){
    rc = get_scale() * sum(biomass_kg( d[ix,])) / 1000
  }
  rc
}

mass_log1 = function(year,sp=NA){ cwd_biomass(year, 'log_1',sp) }
mass_log2 = function(year,sp=NA){ cwd_biomass(year, 'log_2',sp) }
mass_log3 = function(year,sp=NA){ cwd_biomass(year, 'log_3',sp) }
mass_log4 = function(year,sp=NA){ cwd_biomass(year, 'log_4',sp) }
mass_log5 = function(year,sp=NA){ cwd_biomass(year, 'log_5',sp) }

mass_snag1 = function(year){ cwd_biomass(year, 'snag_1') }
mass_snag2 = function(year){ cwd_biomass(year, 'snag_2') }
mass_snag3 = function(year){ cwd_biomass(year, 'snag_3') }
mass_snag4 = function(year){ cwd_biomass(year, 'snag_4') }
```

```
mass_snag5 = function(year){ cwd_biomass(year, 'snag_5') }

cwd_mass_compute = function(logs, snags){
  rc = rep(0,length(years));
  for( i in 1:length(years) ){
    if( sum(logs)>0 ) {
      rc[i] = rc[i]                                          +
              logs[1]*mass_log1(years[i])                     +
              logs[2]*mass_log2(years[i],"SM")*(1-0.19)  +
              logs[3]*mass_log3(years[i],"SM")*(1-0.32)  +
              logs[4]*mass_log4(years[i],"SM")*(1-0.54)  +
              logs[5]*mass_log5(years[i],"SM")*(1-0.73)  +
              logs[2]*mass_log2(years[i],"HM")*(1-0.14)  +
              logs[3]*mass_log3(years[i],"HM")*(1-0.42)  +
              logs[4]*mass_log4(years[i],"HM")*(1-0.73)  +
              logs[5]*mass_log5(years[i],"HM")*(1-0.73)
    }
    if( sum(snags)>0 ){
      rc[i] = rc[i]                             +
              snags[1]*mass_snag1(years[i]) +
              snags[2]*mass_snag2(years[i]) +
              snags[3]*mass_snag3(years[i]) +
              snags[4]*mass_snag4(years[i]) +
              snags[5]*mass_snag5(years[i])
    }
  }
  rc;
}

mass_all    = function() { cwd_mass_compute(rep(1,5),rep(1,5)) }
mass_logs   = function() { cwd_mass_compute(rep(1,5),rep(0,5)) }
mass_snags  = function() { cwd_mass_compute(rep(0,5),rep(1,5)) }
mass_log1s  = function() { cwd_mass_compute(c(1,0,0,0,0), rep(0,5)) }
mass_log2s  = function() { cwd_mass_compute(c(0,1,0,0,0), rep(0,5)) }
mass_log3s  = function() { cwd_mass_compute(c(0,0,1,0,0), rep(0,5)) }
mass_log4s  = function() { cwd_mass_compute(c(0,0,0,1,0), rep(0,5)) }
mass_log5s  = function() { cwd_mass_compute(c(0,0,0,0,1), rep(0,5)) }
mass_snag1s = function() { cwd_mass_compute(rep(0,5), c(1,0,0,0,0)) }
mass_snag2s = function() { cwd_mass_compute(rep(0,5), c(0,1,0,0,0)) }
mass_snag3s = function() { cwd_mass_compute(rep(0,5), c(0,0,1,0,0)) }
mass_snag4s = function() { cwd_mass_compute(rep(0,5), c(0,0,0,1,0)) }
mass_snag5s = function() { cwd_mass_compute(rep(0,5), c(0,0,0,0,1)) }

## CWD DBH distributions.

cwd_ddist = function(year, cl){
  cl_names = c( 'log_1',  'log_2',  'log_3',  'log_4',  'log_5',
                'snag_1', 'snag_2', 'snag_3', 'snag_4', 'snag_5'  );
  rc=NULL;
  for( i in 1:length(cl) ){
    if( cl[i] ){
      d = get_cwd_dbh(year, cl_names[i]);
      if( is.null(rc) ){ rc = d }
      else { rc = my_merge( rc, d, all.x=T, all.y=T) }
    }
  }
```

```
  sclass=seq(0,120,4);
  if(length(rc$dbh)>0){
    XX=get_scale() * hist(pmin(rc$dbh,118),breaks=sclass,plot=F)$counts;
  } else {
    XX=rep(0,length(sclass)-1);
  }
  XX
}

cwd_ddist_log1  = function(){ cwd_ddist(max_year, c(1,0,0,0,0, 0,0,0,0,0)) }
cwd_ddist_log2  = function(){ cwd_ddist(max_year, c(0,1,0,0,0, 0,0,0,0,0)) }
cwd_ddist_log3  = function(){ cwd_ddist(max_year, c(0,0,1,0,0, 0,0,0,0,0)) }
cwd_ddist_log4  = function(){ cwd_ddist(max_year, c(0,0,0,1,0, 0,0,0,0,0)) }
cwd_ddist_log5  = function(){ cwd_ddist(max_year, c(0,0,0,0,1, 0,0,0,0,0)) }
cwd_ddist_snag1 = function(){ cwd_ddist(max_year, c(0,0,0,0,0, 1,0,0,0,0)) }
cwd_ddist_snag2 = function(){ cwd_ddist(max_year, c(0,0,0,0,0, 0,1,0,0,0)) }
cwd_ddist_snag3 = function(){ cwd_ddist(max_year, c(0,0,0,0,0, 0,0,1,0,0)) }
cwd_ddist_snag4 = function(){ cwd_ddist(max_year, c(0,0,0,0,0, 0,0,0,1,0)) }
cwd_ddist_snag5 = function(){ cwd_ddist(max_year, c(0,0,0,0,0, 0,0,0,0,1)) }
cwd_ddist_logs  = function(){ cwd_ddist(max_year, c(1,1,1,1,1, 0,0,0,0,0)) }
cwd_ddist_snags = function(){ cwd_ddist(max_year, c(0,0,0,0,0, 1,1,1,1,1)) }
cwd_ddist_all   = function(){ cwd_ddist(max_year, c(1,1,1,1,1, 1,1,1,1,1)) }

rba_sp = function(sp){
 res_cache( eval(substitute(
    function(){ rba_sp_base( sp ) },
    list( sp=sp) )) )
}

rba_sp_base = function(sp){
  rc = rep(0,length(years))
  for (i in 1:length(years)) {
    dbh     = get_trees(years[i])$dbh
    species = get_trees(years[i])$species
    rc[i] = 100*sum(dbh*dbh*(species==sp))/sum(dbh*dbh)
  }
  rc
}

rba_plot = function(base, n, label) {
  rc = c()
  sp = c(4,5,6,13,9,2)
  cols=c('green','purple','yellow','orange','red','blue')
  ltys = 1:6

  plot_started=F
  for (i in 1:length(sp) ){
    d = envelope_engine(base, n, function(){ rba_sp(sp[i]) } )

    if (!plot_started){
      plot(c(min(years),max(years)), c(0,100), type='n',
           xlab="Simulation year", ylab='Relative basal area', main=label)
      rc = cbind( rc, years )
      plot_started=T
    }
```

```
    lines( years, d$Up, lty=ltys[i], lwd=1, col=cols[i])
    lines( years, d$Mu, lty=ltys[i], lwd=3, col=cols[i])
    lines( years, d$Dn, lty=ltys[i], lwd=1, col=cols[i])

    rc = cbind( rc, d )
  }

  legend("topright", c("Sugar maple", "Hemlock", "Yellow birch",
                       "Ash","Red maple","Basswood"),
         lty=1:6, lwd=3, col=cols)

  rc
}

dbh_dist = function(year=max_year) {
  clamp = function(x, lim) {
    if(length(x)>0){ rc=pmin(x,lim); }
    else { rc = numeric(0); }
    rc;
  }

  data=get_trees(year)$dbh

  all=hist(clamp(data,118),breaks=seq(0,120,4),plot=F)
  all$counts[1]=NA # Exclude the half-class

  get_scale()*all$counts
}

eca_dist = function(year=max_year) {
  rc <- rep(0, 30)
  data <- get_trees(year)

  size_class <- 1+pmin(floor(data$dbh/4),29)
  for (i in 1:length(data$dbh)){
    rc[size_class[i]] = rc[size_class[i]] + data$eca[i]
  }

  rc[1] = NA # Exclude the half-class at the bottom

  rc * get_scale()
}

stand_dev = function(eca_thresh=0.25, area_thresh=3.25){
  res_cache( eval(substitute(
    function(){ stand_dev_base( qual, eca_thresh, area_thresh ) },
    list( qual=db_site_quality, eca_thresh=eca_thresh, area_thresh=area_thresh) )) )
}

stand_dev_base = function(qual, eca_thresh, area_thresh){
  rc = rep(NA, length(years))
  for ( i in 1:length(years)){
    rc[i] = stand_dev_yr( qual, years[i], eca_thresh, area_thresh)
  }
  rc
}

compute_gap_sap_ba = function(d, eca_thresh, area_thresh){
  ## d is expected to contain at least:
  ##  species, dbh, eca, tca, gap_area
```

```
  ## Make sure ironwood is excluded:
  d = d[d$species != 24, ]

  ## Pull out just the saplings
  d = d[d$dbh <= 11, ]

  ## With ECA greater than the threshold
  d = d[ d$eca / pmax(d$tca, 0.01) >= eca_thresh , ]

  ## For trees which have no gap area computed, force it to zero:
  gap_area = ifelse( !is.na(d$gap_area), d$gap_area, 0)

  ## And with enough gap area:
  d = d[ d$eca >= area_thresh | gap_area >= area_thresh , ]

  get_scale() * sum( pi * (d$dbh/200)^2 )
}

stand_dev_yr = function(site_quality, yr, eca_thresh, area_thresh, diagnostic=F){
  ## Pull out the needed data:
  d_trees = get_trees(yr)

  d_gaps = sql(sprintf(
    'select year, treeno, gap_area from gaps where year%%%i=0', meas_interval))
  d_gaps = d_gaps[ d_gaps$year==yr, ]

  d = merge( d_trees, d_gaps, by="treeno", all.x=T)

  ## Define stand stages
  st_sap    = 1
  st_pol    = 2
  st_msm    = 3
  st_mat    = 4
  st_og_la  = 5
  st_og_et  = 6
  st_og_lt  = 7
  st_og_ss  = 8

  ## Set the min threshold for large based on site quality
  if (site_quality=="F") { lrg_min = 44 }
  else                   { lrg_min = 46 }

  ## Exclude ironwood
  d = d[d$species != 24, ]

  ## Compute BA for each tree on a per ha basis
  ba =  get_scale() * pi*(d$dbh/200)^2

  ## Split it out by size class
  sap_ba = sum( ba[                    d$dbh <= 11       ] )
  pol_ba = sum( ba[       11 < d$dbh & d$dbh <= 26       ] )
  mat_ba = sum( ba[       26 < d$dbh & d$dbh <= lrg_min ] )
  lrg_ba = sum( ba[ lrg_min < d$dbh                     ] )
  all_ba = sap_ba + pol_ba + mat_ba + lrg_ba


  et_thr = ifelse( sum(ba[d$sp == 5])/all_ba > 0.25, 0.10, 0.08 )

  gap_sap_ba = compute_gap_sap_ba(d, eca_thresh, area_thresh)
```

```
  ## Do the decision tree
  rc=NA
  if ( mat_ba + lrg_ba >= 20 ){
    if (lrg_ba/all_ba > 0.45 &
        lrg_ba/mat_ba > 1.1) {
      if (lrg_ba/mat_ba > 1.4             &
          (sap_ba+pol_ba)/all_ba >= 0.10 &
          (sap_ba+pol_ba)/all_ba <  0.20 &
          round(gap_sap_ba/all_ba,3) >= 0.006) { rc = st_og_ss }
      else {
        if (lrg_ba/mat_ba > 1.75 &
            (sap_ba+pol_ba)/all_ba<et_thr)  { rc = st_og_et }
        else {
          if (lrg_ba/mat_ba >= 1.7 &
              (sap_ba+pol_ba)/all_ba<0.20)  { rc = st_og_lt }
          else                              { rc = st_og_la }
        }
      }
    } else                                  { rc = st_mat   }
  } else {
    if (mat_ba + lrg_ba > 10) {
      if (pol_ba/all_ba > 0.30)             { rc = st_pol   }
      else                                  { rc = st_msm   }
    } else {
      if (pol_ba+mat_ba+lrg_ba >= 10 &
          pol_ba/all_ba         >  0.30)    { rc = st_pol   }
      else                                  { rc = st_sap   }
    }
  }

  ## Debugging output:
  if (diagnostic){
    rc = data.frame(
      stage=rc,
      sap_ba=sap_ba,
      pol_ba=pol_ba,
      mat_ba=mat_ba,
      lrg_ba=lrg_ba,
      all_ba=all_ba,
      gap_sap_ba=gap_sap_ba)
  }

  rc
}

stand_dev_old = function(){
  # Define stand stages
  st_sap   = 1
  st_pol   = 2
  st_msm   = 3
  st_mat   = 4
  st_og_la = 5
  st_og_et = 6
  st_og_lt = 7
```

```
  st_og_ss = 8

  # BA variables that we'll need:
  sap_ba <- sapling_trees_ba()
  pol_ba <- pole_trees_ba()
  mat_ba <- mature_trees_ba()
  lrg_ba <- large_trees_ba()
  xlg_ba <- xlarge_trees_ba()
  tot_ba <- sap_ba + pol_ba + mat_ba + lrg_ba + xlg_ba

  # ECA Variables that we'll need:
  sap_eca <- est_sapling_trees_eca()
  pol_eca <- est_pole_trees_eca()
  mat_eca <- est_mature_trees_eca()
  lrg_eca <- est_large_trees_eca()
  xlg_eca <- est_xlarge_trees_eca()
  tot_eca <- sap_eca + pol_eca + mat_eca + lrg_eca + xlg_eca

  lm_eca_ratio <- (lrg_eca + xlg_eca)/mat_eca

  # Set up the result variable:
  rc = rep(NA,length(years))

  # Foreach year:
  for (i in 1:length(years)){

    # This is the decision tree from
    # "Ecological Benchmarks for Stand Structural Stages"
    if (mat_ba[i]+lrg_ba[i]+xlg_ba[i] >= 20){
      if ((lrg_ba[i]+xlg_ba[i])/tot_ba[i] >= 0.45 & lm_eca_ratio[i] > 1.0 ){
        if ( (lrg_eca[i]+xlg_eca[i])/tot_eca[i] >= 0.58 ) { rc[i] = st_og_et }
        else {
          if (mat_eca[i]/tot_eca[i] <= 0.36 &
              sap_eca[i]/tot_eca[i] >= 0.03 )              { rc[i] = st_og_ss }
          else {
            if (lm_eca_ratio[i] > 1.5 )                    { rc[i] = st_og_lt }
            else                                           { rc[i] = st_og_la }
          }
        }
      } else                                               { rc[i] = st_mat   }
    } else { # Less than 20m2/ha in mat+lrg+xlg
      if( mat_ba[i]+lrg_ba[i]+xlg_ba[i] > 10){
        if( pol_ba[i]/tot_ba[i] > 0.30)                    { rc[i] = st_pol   }
        else                                               { rc[i] = st_msm   }
      } else {
        if (tot_ba[i]-sap_ba[i]>10 &
            pol_ba[i]/tot_ba[i]>0.30)                      { rc[i] = st_pol   }
        else                                               { rc[i] = st_sap   }
      }
    }
  }

  rc
}

##

compute_productivity = function(){
```

```
  sim_prod = function( dbase, tgt_yr) {
    cut_vol = 0;
    std_vol = 0;

    nrep = 3;
    for( sr in 1:nrep ){
      fname = sprintf('%s.r%02i.db', dbase, sr);

      db_open(fname)
      cut      = sql(sprintf('select treeno,dbh,ht ' %.%
        'from cut_taken_trees where year<=%i', tgt_yr ));
      standing = sql(sprintf('select treeno,dbh,ht ' %.%
        'from live_trees where year==%i', tgt_yr));
      cut_vol = cut_vol + sum(estimate_volume_go(      cut$dbh,      cut$ht ));
      std_vol = std_vol + sum(estimate_volume_go( standing$dbh, standing$ht ));
      db_close()
    }
    c( cut_vol / nrep, std_vol / nrep );
  }

  chvol = c(); chvol50 = c();
  fvol  = c();  fvol50 = c();
  sname = c();

  sims=list.files(pattern='.*.r01.db$')
  sims=sub('.r01.db','',sims)

  for( fname_base in sims ){
    x = sim_prod( fname_base, 300 );
    chvol = c(chvol, x[1])
    fvol  = c(fvol,  x[2])

    x = sim_prod( fname_base, 45);
    chvol50 = c(chvol50, x[1])
    fvol50  = c(fvol50,  x[2])

    sname = c(sname, fname_base)
  }

  print( data.frame(sname, fv45 =fvol50/9, hv45 =chvol50/(9*45 ),
                    fv300=fvol  /9, hv300=chvol  /(9*300) ))
}

mort_by_size_class = function(){
  mort_sim = function(dbase) {
    cat( sprintf('\n\n=== %s ===\n',dbase));
    if( length(list.files(pattern=sprintf('^%s_pct_mort.png',dbase)))==0 ) {
      tot_ba     = rep(0,30)
      dead_count = rep(0,30)
      live_count = rep(0,30)

      for( sr in 1:3 ){
        fname = sprintf('%s.r%02i.db', dbase, sr);
        db_open(fname)

        d_prv = get_trees(290)
        d_now = get_trees(300)
```

```
      tn_cut = sql(sprintf('select treeno from cut where ' %.%
        '%i <= year and year <= %i', 290, 300))

      dead_tn = setdiff( d_prv$treeno, union(d_now$treeno, tn_cut$treeno) )

      dbh = d_prv$dbh[ d_prv$treeno %in% dead_tn ]
      ba= get_scale() * pi/40000 * dbh^2

      ## Now to compute %ba by size class, and get dead counts
      for( i in 1:length(dbh) ){
        sc = 1+min(floor(dbh[i]/4),29)
        tot_ba[sc] = tot_ba[sc]+ba[i]
        dead_count[sc] = dead_count[sc] + 1
      }

      ## And live counts for %annual mort
      for( i in 1:length( d_prv$dbh ) ){
        sc = 1+min(floor(d_prv$dbh[i]/4),29)
        live_count[sc] = live_count[sc] + 1
      }
      db_close();
    }

    pct_mort = 1-(1-dead_count/live_count)^(1/10)
    pct_ba   = tot_ba / sum(tot_ba)

    png(sprintf('%s_pct_mort.png', dbase), width=600, height=600)
      barplot(100*pct_mort, names.arg = seq(4,120,by=4), ylim=c(0,20),
              xlab='DBH (cm)', ylab='Percent Annual Mortality' )
    dev.off()
    cat( sprintf('%%mort(%s): ',dbase) %.%
         paste(signif(100*pct_mort,3),collapse='\t') %.% '\n' )

    png(sprintf('%s_pct_mort_ba.png', dbase), width=600, height=600)
      barplot(100*pct_ba,   names.arg = seq(4,120,by=4), ylim=c(0,20),
              xlab='DBH (cm)',ylab='Percenta BA of dead trees (m^2/ha)' )
    dev.off()
    cat( sprintf('%%mortba(%s): ',dbase) %.%
         paste(signif(100*pct_ba,3),collapse='\t') %.% '\n' )
  }
}

sims=list.files(pattern='.*.r01.db$')
sims=sub('.r01.db','',sims)

for( sim in sims ){ mort_sim(sim) }
}

stand_dev_diagnostic = function(dbfile){
  stand_dev_yr_diagnostic = function(
    yr, eca_change, stdata,
    site_quality="G", eca_thresh=0.25, area_thresh=3.25){
    sink('/dev/null')
    stage_data = stand_dev_yr( site_quality, yr, eca_thresh, area_thresh, T)
    sink()

    stage = stage_data$stage
    all_ba = stage_data$all_ba
    sap_ba = stage_data$sap_ba
```

```
pol_ba = stage_data$pol_ba
mat_ba = stage_data$mat_ba
lrg_ba = stage_data$lrg_ba
gap_sap_ba = stage_data$gap_sap_ba

if      (stage==1) { cat('Sapling &') }
else if (stage==2) { cat('Pole &')    }
else if (stage==3) { cat('MSM &')     }
else if (stage==4) { cat('Mature &')  }
else if (stage==5) { cat('OG: LA &')  }
else if (stage==6) { cat('OG: ET &')  }
else if (stage==7) { cat('OG: LT &')  }
else if (stage==8) { cat('OG: SS &')  }
else { stop('Unknown stage') }

if (length(stdata$year)>0){
  for (i in 1:length(stdata$year)){
    cat( sprintf(' %i:%i,', stdata$year[i], stdata$severity[i] ))
  }
}
cat(' &')

cat(sprintf('%.1f &', 100*eca_change))
cat(sprintf('%.1f &', sap_ba ))
cat(sprintf('%.1f &', pol_ba ))
cat(sprintf('%.1f &', mat_ba ))
cat(sprintf('%.1f &', lrg_ba ))

if (mat_ba+lrg_ba>=20) {
  cat(sprintf('\\textbf{%.2f} &', mat_ba + lrg_ba))
} else {
  cat(sprintf('%.2f &', mat_ba + lrg_ba))
}

if (lrg_ba/all_ba > 0.45 ) {
  cat(sprintf('\\textbf{%.1f} &', 100*lrg_ba/all_ba))
} else {
  cat(sprintf('%.1f &', 100*lrg_ba/all_ba))
}

if (lrg_ba/max(0.01,mat_ba) > 1.1 ) {
  cat(sprintf('\\textbf{%.2f} &', lrg_ba/mat_ba))
} else {
  cat(sprintf('%.2f &', lrg_ba/mat_ba))
}

if (lrg_ba/max(0.01,mat_ba) > 1.4 ){
  cat(sprintf('\\textbf{%.2f} &', lrg_ba/mat_ba))
} else {
  cat(sprintf('%.2f &', lrg_ba/mat_ba))
}

if ( (sap_ba+pol_ba)/all_ba >= 0.10 ){
  cat(sprintf('\\textbf{%.1f} &', 100*(sap_ba+pol_ba)/all_ba))
} else {
  cat(sprintf('%.1f &', 100*(sap_ba+pol_ba)/all_ba))
}
```

```
  if ( gap_sap_ba/all_ba >= 0.006) {
    cat(sprintf('\\textbf{%.2f} &', 100*gap_sap_ba/all_ba))
  } else {
    cat(sprintf('%.1f &', 100*gap_sap_ba/all_ba))
  }

  if (lrg_ba/max(0.01,mat_ba) > 1.75 ){
    cat(sprintf('\\textbf{%.2f} &', lrg_ba/mat_ba))
  } else {
    cat(sprintf('%.2f &', lrg_ba/mat_ba))
  }


  if ((sap_ba+pol_ba)/all_ba<0.10) {
    cat(sprintf('\\textbf{%.1f} &', 100*(sap_ba+pol_ba)/all_ba))
  } else {
    cat(sprintf('%.1f &', 100*(sap_ba+pol_ba)/all_ba))
  }

  if (lrg_ba/max(0.01,mat_ba)>1.75) {
    cat(sprintf('\\textbf{%.2f} &', lrg_ba/mat_ba))
  } else {
    cat(sprintf('%.2f &', lrg_ba/mat_ba))
  }

  if (mat_ba+lrg_ba>10) {
    cat(sprintf('\\textbf{%.2f} &', mat_ba + lrg_ba))
  } else {
    cat(sprintf('%.2f &', mat_ba + lrg_ba))
  }

  if (pol_ba/all_ba > 0.30 ) {
    cat(sprintf('\\textbf{%.1f} &', 100*pol_ba/all_ba ))
  } else {
    cat(sprintf('%.1f &', 100*pol_ba/all_ba ))
  }

  if (pol_ba+mat_ba+lrg_ba >= 10 ){
    cat(sprintf('\\textbf{%.2f} &', pol_ba+mat_ba+lrg_ba ))
  } else {
    cat(sprintf('%.2f &', pol_ba+mat_ba+lrg_ba ))
  }

  if ( pol_ba/all_ba > 0.30 ) {
    cat(sprintf('\\textbf{%.1f} \\\\\n', 100*pol_ba/all_ba))
  } else {
    cat(sprintf('%.1f \\\\\n', 100*pol_ba/all_ba))
  }

  stage
}

print_head = function(){
  cat('Year & Stage & Storms & \\(\\Delta\\)ECA & S & P & M & L & \n')
  cat('M+L\\(\\ge\\)20 &\n')
  cat('\\%L\\(>\\)45 & L:M\\(>\\)1.1 &\n')
  cat('L:M\\(>\\)1.4 &\\%(S+P)\\(>\\)10 & \\%gS\\(>\\)0.6 &\n')
  cat('L:M\\(>\\)1.75 &\\%(S+P)\\(<\\)10 &\n')
  cat('L:M\\(>\\)1.75 &\n')
```

```
  cat('M+L\\(>\\)10 &\n')
  cat('\\%P\\(>\\)30 &\n')
  cat('P+M+L\\(\\ge\\)10 & \\%P\\(>\\)30 \\\\\n')
  cat('\\hline\n')
}

db_open(dbfile)

sink(gsub('\\.db$','_stdev_diagnostic.tex',dbfile))

sink('/dev/null')
get_trees(0)
get_scale()
st_data = sql('select * from storms')
sink()

cat('\\documentclass[10pt,landscape,english]{article}\n')
cat('\\usepackage[letterpaper]{geometry}\n')
cat('\\usepackage{longtable}\n')
cat('\\usepackage{fancyhdr}\n')
cat('\\geometry{verbose,tmargin=2cm,bmargin=2cm,lmargin=1cm,rmargin=1cm}\n')
cat('\\pagestyle{fancy}\n\n')
cat('\\begin{document}\n')
cat('\\lhead{', gsub('_','\\\\_',dbfile),'}\n')
cat('\\(\\Delta\\)ECA is the percent change in ECA from the previous ')
cat('to the current measurement.\\\\\n')
cat('Values for S,P,M,L are all in m2/ha.\\\\\n')
cat('Storms column shows Year:\\%ECA removal\\\\\n')

cat('\\tiny\n')
cat('\\begin{longtable}{')
cat('c|c|c|ccccc') # Year, ECA, S,P,M,L
cat('|c')       # M+L ge 20
cat('|c@{\\hspace{2mm}}c')
cat('|c@{\\hspace{2mm}}c@{\\hspace{2mm}}c|c@{\\hspace{2mm}}c|c|c|c|c@{\\hspace{2mm}}c}\n')
print_head()
cat('\\endfirsthead\n')
print_head()
cat('\\endhead\n')


d=get_trees(years[1])
prev_eca = sum(d$eca)

prev_eca = NA
for ( i in 2:length(years)){
  d = get_trees(years[i])
  this_eca = sum(d$eca)

  st_ix = which( years[i-1] < st_data$year & st_data$year <= years[i])

  cat(sprintf('%i &', years[i]))

  stand_dev_yr_diagnostic(
    years[i],  (this_eca-prev_eca)/prev_eca, st_data[st_ix,] )

  prev_eca = this_eca
```

```
  }

  cat('\\end{longtable}\n')


  cat('\\end{document}')

  db_close();

  sink();
}

avg_q_below_mode = function(year){
  d_dist  = dbh_dist(year)
  ba_dist = apply(ba_dist(year)$z, 2, sum)

  ix_mode = which.max( ba_dist )

  # Start at 2 to cut out the half class (it's NA anyhow)
  # And go up through the mode.
  d_dist = d_dist[2:ix_mode]

  qi = rep(0, length(d_dist)-1)
  for (i in 2:length(d_dist)){
    qi[i-1] = d_dist[i-1] / max(0.1, d_dist[i])
  }

  mean(qi)
}

pct_ba_around_mode = function(year){
  ba_dist = apply(ba_dist(year)$z, 2, sum)

  ix_mode = which.max( ba_dist )

  ix = max(1, ix_mode-2):min( length(ba_dist), ix_mode+2)

  sum( ba_dist[ix], na.rm=T ) / sum(ba_dist, na.rm=T)
}
```