Methods in Ecology and Evolution

Methods in Ecology and Evolution 2016, **7,** 136–146 doi: 10.1111/2041-210X.12525

DEMOGRAPHY BEYOND THE POPULATION

APPLICATION

`plant`: A package for modelling forest trait ecology and evolution

Daniel S. Falster[1*†], Richard G. FitzJohn[1,†], Åke Brännström[2,3], Ulf Dieckmann[3] and Mark Westoby[1]

[1]*Department of Biological Sciences, Macquarie University, Sydney, NSW 2109, Australia;* [2]*Department of Mathematics and Mathematical Statistics, Umeå University, 90187 Umeå, Sweden; and* [3]*Evolution and Ecology Program, International Institute for Applied Systems Analysis, Schlossplatz 1, A-2361 Laxenburg, Austria*

Summary

1. Population dynamics in forests are strongly size-structured: larger plants shade smaller plants while also expending proportionately more energy on building and maintaining woody stems. Although the importance of size structure for demography is widely recognized, many models either omit it entirely or include only coarse approximations.

2. Here, we introduce the `plant` package, an extensible framework for modelling size- and trait-structured demography, ecology and evolution in simulated forests. At its core, `plant` is an individual-based model where plant physiology and demography are mediated by traits. Individual plants from multiple species can be grown in isolation, in patches of competing plants or in metapopulations under a disturbance regime. These dynamics can be integrated into metapopulation-level estimates of invasion fitness and vegetation structure. Because fitness emerges as a function of traits, `plant` provides a novel arena for exploring eco-evolutionary dynamics.

3. `plant` is an open source R package and is available at github.com/traitecoevo/plant. Accessed from R, the core routines in `plant` are written in C++. The package provides for alternative physiologies and for capturing trade-offs among parameters. A detailed test suite is provided to ensure correct behaviour of the code.

4. `plant` provides a transparent platform for investigating how physiological rules and functional trade-offs interact with competition and disturbance regimes to influence vegetation demography, structure and diversity.

Key-words: demography, emergent, fitness, growth, metapopulation, mortality, physiology reproduction, size structure, trade-off

Introduction

Plant growth and demography are fundamentally size- and trait-structured, influencing dynamics over time-scales ranging from instantaneous physiological effects to long-term evolutionary outcomes (Harper 1977, Westoby *et al.* 2002, Griffith *et al.* 2016; Rees & Ellner 2016). As an individual plant increases its leaf area, it increases its potential to generate photosynthate. On the other hand, as individuals grow larger, they must allocate increasing fractions of their photosynthetic income to activities other than building new leaves, for example to maintaining support tissues (Givnish 1988; Enquist *et al.* 2007) or to reproduction (Thomas 2011). Consequently, rates of growth, mortality and reproduction change with individual size (Muller-Landau *et al.* 2006; Rüger *et al.* 2011; Thomas 2011). Ontogenetic patterns of growth also vary with traits, for example, leaf and wood construction costs influence growth rates (Falster *et al.* 2011; Visser *et al.* 2016), while seed size and height at maturation determine start and endpoints of ontogenetic trajectories (Westoby *et al.* 2002). Strong feedbacks emerge between individuals within a forest via competition for light and other resources, such that the growth rate of one individual depends on the size and traits of nearby individuals (Shugart & West 1980; Pacala *et al.* 1996). Such feedbacks make it difficult to link observable species traits to phenomena such as self-thinning, successional transitions, trait evolution and species coexistence, without modelling both individual growth and the competitive interactions between individuals.

Although the importance of size structure for trait evolution, vegetation dynamics and diversity has long been recognized (e.g. Harper 1977; Shugart & West 1980; Huston & Smith 1987), current research in this area is dominated by models and theory that either omit size structure entirely or only include coarse approximations. Theoretical investigations

*Correspondence author. E-mail: daniel.falster@mq.edu.au
†These authors contributed equally.

This article forms part of the British Ecological Society journals' Demography Beyond the Population special feature http://wileyonlinelibrary.com/BES_demography

exploring questions about niche-based species coexistence and ecological drift (neutrality) both rely primarily on unstructured population models (e.g. e.g., MacArthur & Levins 1967; Tilman 1985; Geritz *et al.* 1998; Hubbell 2001; Calcagno *et al.* 2006). It is common to assume competitive outcomes to be influenced by size-related traits such as adult size and seed size, but detailed size structure is rarely considered, at least in plant models (for animal examples, see De Roos, Taljapurkar & Caswell, 1997). Similarly, many of the models used to study global vegetation dynamics across the last 20 years have not included size-structured demography (for comparisons of some major models, see Sitch *et al.* 2008; De Kauwe *et al.* 2014). While the importance of explicitly including size-structured dynamics is increasingly recognized in both evolutionary ecology (e.g. Falster *et al.* 2015; Rees & Ellner 2016) and vegetation dynamics (e.g. Moorcroft, Hurtt & Pacala 2001; Purves & Pacala 2008; Smith *et al.* 2014; Weng *et al.* 2015; Sakschewski *et al.* 2015), our understanding of how these features impact the ecology and evolution of vegetation communities remains relatively limited.

In this note, we introduce the `plant` package for R (R-core Team, 2015); a framework for studying the effects of size structure and trait variation on the demography of individual plants, of patches of competing plants and of metapopulations structured by a disturbance regime. Our own purpose in developing `plant` has been mainly to investigate how species differing in traits may be able to coexist with one another (following Falster *et al.* 2011, 2015); as size-structured models provide unique opportunities for studying coexistence via differentiation in successional strategy (see also Huston & Smith 1987; Moorcroft, Hurtt & Pacala 2001; Uriate *et al.* 2016). At the same time, we expect `plant` to be useful for modelling other demographic and evolutionary phenomena.

Broadly, the `plant` package falls into a class we refer to as trait-, size- and patch-structured models (TSPMs, following Falster *et al.* 2011). TSPMs are direct descendants from the 'gap' models developed in the 1980s (e.g., Shugart & West 1980; Huston & Smith 1987; Kohyama 1993); other modern examples include ED (Moorcroft, Hurtt & Pacala 2001), LPJ- GUESS (Smith *et al.* 2014) and LPJmL-FIT (Sakschewski *et al.* 2015). Common to TSPMs is that they explicitly model size-structured competition among individual plants within a metapopulation of 'patches', where individuals can differ in their traits and thus demographic behaviour. TSPMs typically ignore the spatial structure of plants within patches and the spatial structure of patches relative to one another. Instead, they focus computational effort on resolving the dynamics of size-structured competition for light and successional turnover. Differences among TSPMs arise from the core physiological models used (describing how the demography of individual plants responds to resource availability), and also from the numerical techniques applied to scaling the physiological model up to estimate emergent behaviours of individuals, patches and vegetation.

Below, we describe the general approach of the `plant` package, then provide sections outlining potential applications at nested levels of ecological organization. More detailed technical documents are provided as supplementary information (see Appendices; updated versions of these technical documents are available within the `plant` package itself). These documents outline (S1) the system of equations being solved when modelling the demography of plants, patches and metapopulations; (S2) a description of the core physiological model used in `plant`; and (S3) a compendium of worked code examples showing how to interact with the `plant` package from R.

Overview of approach

The `plant` package implements an individual-based model, meaning that the dynamics of the population arise from rules specifying how individuals grow and interact. Being driven by traits, the model can be extended to potentially very many species. The core rules in `plant` are about the short-term physiological functioning of an individual plant and how this is influenced by its traits, size and light environment. While `plant` includes a core physiological model, this article uses it only as an example, to illustrate how one can take a given physiological model and simulate ecological and evolutionary outcomes. Thus, we do not aim to justify the particular physiological model used, to provide detailed comparisons to data or to promote this physiological model over those in other TSPMs. (These are all topics deserving substantially more attention than can be afforded here.) An important feature of `plant` considered as a software package is that it allows for users to substitute their own physiological models.

On top of the physiological model, `plant` implements methods for population dynamics and adaptive dynamics (Fig. 1), following methods described by Falster *et al.* (2011, 2015). Demographic phenomena can be studied at three levels: individual plants, stands of competing plants and entire metapopulations. The dynamics at higher levels of organization arise as emergent properties, driven by growth physiology, competition for light and disturbance (Fig. 1). `plant` offers the capacity to model emergent phenomena in patches with specified area, and also through a deterministic approximation where specifying a patch size is not required. Trait evolution and community assembly can then be modelled using the estimates of invasion fitness provided by `plant`.

Effects of size, trait and light environment on the demography of individual plants

The core of `plant` is a model for an individual species' physiological strategy as specified by its traits (Fig. 1a). For present purposes, a species is defined as a group of individuals with identical traits. The effects of trait variation are to modify parameters of the core physiological model, which in turn translate into different demographic outcomes.

The default physiological model used in `plant` largely reflects that presented in Falster *et al.* (2011, 2015), but with extensions allowing for growth in plant diameters also to be estimated. We refer to this default model as `FF16`, reflecting the initials of the first two authors and publication year of this article The `FF16` model estimates rate of biomass production

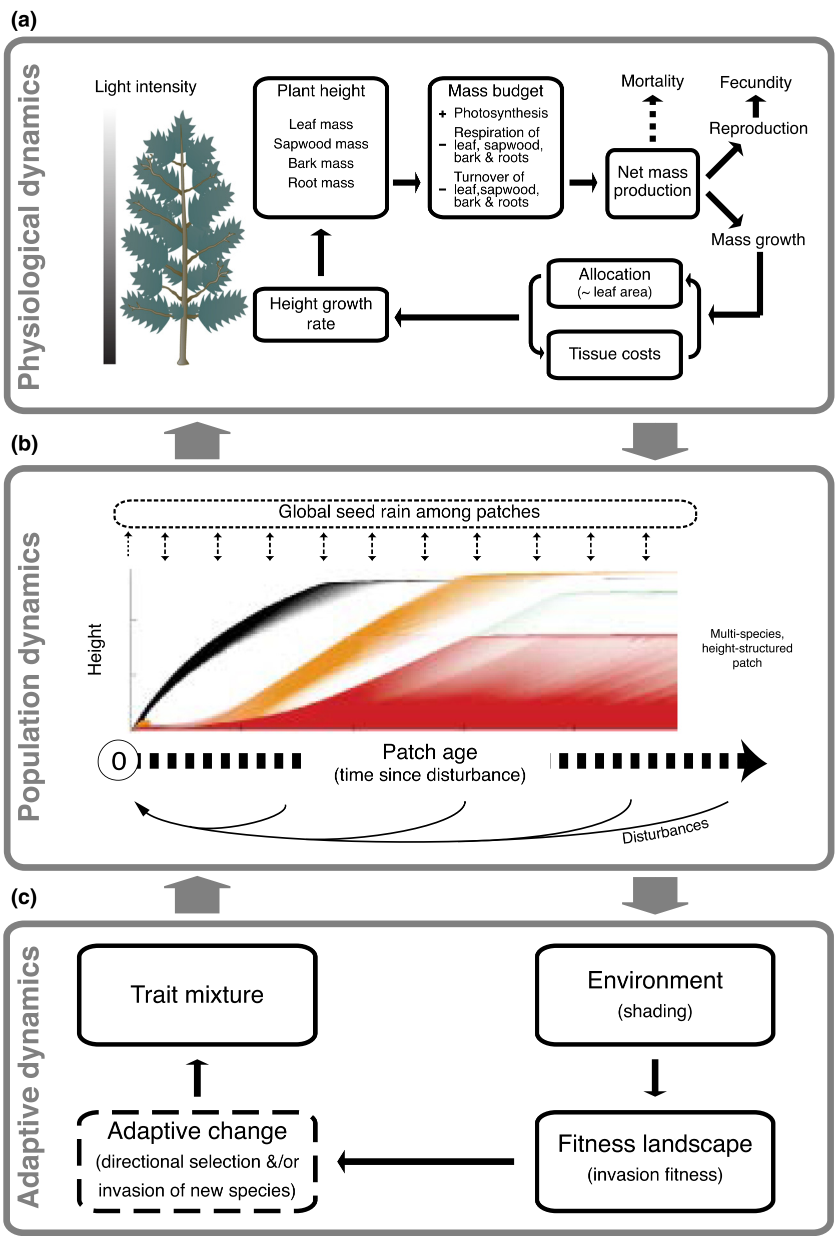


Fig. 1. Processes modelled within `plant` include physiological, population and adaptive dynamics. (a) Dynamics across all three levels are driven by the physiological sub-model determining an individual's physiological and demographic rates on the basis of its traits, size and light environment. (b) Competitive hierarchies are modelled by tracking the height distribution of individuals within a patch. The intensity of shading indicates the density of individuals at a given height for different species (distinguished by colours). A size- and patch-structured metapopulation consists of a distribution of patches linked by a global seed rain. Disturbances occasionally remove all vegetation within a patch, resetting its vegetation. (c) The traits of the resident species determine the light environment in all patches across the metapopulation, which in turn determines the invasion fitness. The dashed outline indicates that users must supply their own algorithms for modelling trait evolution and community assembly, using the estimate of invasion fitness provided by `plant`. Figure adapted from Falster *et al.* (2011, 2015).

for a plant, given its size and the current light environment. Gross photosynthetic income is calculated from the total leaf area and the light distribution across the plant's canopy. Costs of tissue respiration and turnover are subtracted. The remaining biomass production is allocated between growth and reproduction. The key outputs needed by higher levels of `plant` are height growth rate, mortality rate and rate of seed production (Fig. 1). Additional quantities are also computed, such as total assimilation, as well as respiration, turnover and allocation rates for different tissues; these intermediates can also be accessed (for further details on the `FF16` model, see Appendices S2 and S3).

While we have implemented a particular physiological model (`FF16`), the package is designed to allow arbitrary additional physiologies to be introduced. Users are able to modify the physiological model in three different ways. First, one can vary the parameters of the `FF16` model directly (see Appendix S2 for the list of parameters and Appendix S3 for code examples). Secondly, we allow for changes in key parameters to bring about changes in other parameters (a hyperparameterization), enabling the straightforward modelling of trade-offs. For example, the trait LMA (leaf mass per unit leaf area) is used by default to estimate the rate of leaf turnover (k_l), based on an observed scaling relationship spanning across diverse vegetation and plant functional types (Wright *et al.* 2004),

$$k_l = \beta_{kl1}(\phi/\phi_0)^{-\beta_{kl2}},$$

where k_l and ϕ (LMA) are both parameters of the core physiological model, and $\beta_{kl1}, \beta_{kl2}, \phi_0$ are hyper-parameters. Such linkages are defined within a user-supplied hyperparameterization function (see Appendix S2 for details, or Appendix S3 for code examples). Finally, rather than varying the parameters of the existing equations in the `FF16` model, the equations themselves can be fundamentally changed. This requires defining a new physiological model in a new C++ file and compiling that into the package. As an example, in the `FF16r` physiology, we have shown how to modify the function determining allocation to reproduction.

With a physiological model in place, `plant` can be used to estimate essential physiological rates for individual plants (Fig. 2). The function `grow_plant_to_size` takes a given strategy and light environment and grows the plant, producing a trajectory of plant size over time (Fig. 2a). This is achieved by integrating an ordinary differential equation with a size-dependent growth rate (see Appendices S1 and S3 for details). As the plant grows, allocation to different tissue types varies; specifically, the composition of the plant changes to include more stem and less leaf (Fig. 2b). This, in turn, affects maintenance and turnover costs and the growth rate of the plant. By itself, the shift in allocation can generate the widely observed hump-shaped dependence of absolute height growth on plant height (Fig. 2c; King 2011), as well as the decline in relative mass growth rate with plant size from birth onwards (Enquist *et al.* 2007).

By varying the light environment and measuring growth rate, the whole-plant light compensation point (WPLCP) can be computed (Fig. 2d). The WPLCP is the light level where the plant stops growing, increasingly regarded as the most useful measure of a species' shade tolerance (Givnish 1988; Baltzer & Thomas 2007; Lusk & Jorgensen 2013). As expected, the WPLCP increases with plant size (Fig. 2d), due to increased costs of building and maintaining stem and leaf tissues (Givnish 1988). Likewise, the WPLCP decreases with LMA (Fig. 2d), because high-LMA species have slower leaf turnover (Baltzer & Thomas 2007).

Plants competing in a patch

Within patches of competing plants, competition for light generates strong nonlinear feedbacks on growth, survival and reproduction. In the `FF16` physiological model, we consider only the effects of shading on rates of biomass production. Competition for other resources such as nitrogen or water could be considered by extending the physiological model. We assume that patches are vertically – but not horizontally – structured; in other words, we account for size differences, but not for the spatial layout within patches. Similar assumptions are made in other TSPMs (Shugart & West 1980; Huston & Smith 1987; Kohyama 1993; Moorcroft, Hurtt & Pacala 2001; Smith *et al.* 2014), and also in related approaches, such as integral projection models (e.g. Rees & Ellner 2016). While one would ideally also consider spatial interactions within patches, such models are very computationally demanding (Shugart & West 1980; Pacala *et al.* 1996). Simplifying spatial interactions within patches can be motivated from the observation that competitive thinning tends to break down spatial clusters (Strigul *et al.* 2008).

When modelling a patch of competing plants, the focus is on how the size distribution $N(H|x,a)$ changes with patch age a, with the latter defined as the time that has passed since the last

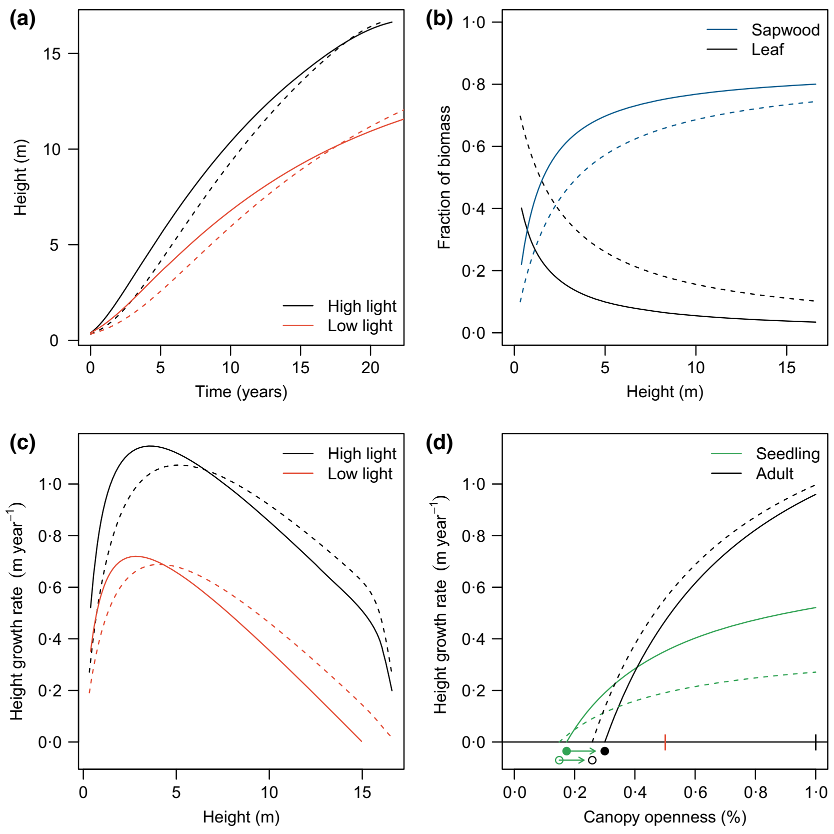


Fig. 2. Physiological dynamics for individual plants varying in size, trait and light environment. Solid lines refer to the low-LMA (leaf mass per unit leaf area) strategy (LMA = 0·0825), while dashed lines refer to the high-LMA strategy (LMA = 0·2625). (a) Growth trajectories are influenced by both light environment and traits. (b) Over time, the fraction of living tissue switches from leaf towards sapwood, with high-LMA species having relatively more mass in leaf than low-LMA species. (c) Size-dependent height growth rates, being the derivatives of the functions in (a), peak at a height of around 5m, but the location of this peak varies with both trait and light level. (d) Declines in height growth rate with light level vary with plant size and traits. Whole-plant light compensation points (indicated by circles) emerge at zero growth rate (intercepts on horizontal axis). The light level used in (a) and (c) is indicated by coloured ticks. See Appendix S3 for code reproducing this figure.

disturbance. This distribution describes the density N of individuals with height H for given traits x and patch age a. The density N is measured as individuals per unit height and per unit ground area. Modelling the dynamics of $N(H|x,a)$ requires that both the initial size distribution and the inflow of new recruits be specified. We have primarily been interested in the dynamics of a patch recovering from disturbance and have therefore started with an empty patch and a constant flow of seeds from a global seed rain (Fig. 1b). A (nontrivial) extension would result from allowing some plants to survive disturbance (following Kohyama 1993).

`plant` offers two methods for modelling the dynamics of $N(H|x,a)$ in a competing population. In the first stochastic mode, users specify an average seed rain and patch size, and `plant` then generates a vector of seed arrival events and simulates the stochastic development of the resulting finite-sized population.

For most applications, however, rather than modelling dynamics within a finite-sized stochastic patch, it will be preferable to use `plant`'s deterministic mode. This assumes that patches are sufficiently large that population dynamics within a patch approach their deterministic limit and can be approximated via a physiologically structured population model (Metz & Diekmann 1986; Kohyama 1993; De Roos, Taljapurkar & Caswell, 1997). These models are often formulated as partial differential equations where boundary conditions and coefficients may depend on the population state. Such structured population models are also thought to capture the average behaviour across a large number of small patches (Moorcroft, Hurtt & Pacala 2001). Importantly, the deterministic mode in `plant` is much faster than the stochastic mode and eliminates the demographic noise that inevitably arises in finite-sized populations (see Appendix S1 for details).

Our approach for numerically solving deterministic size-structured population dynamics is based on the characteristic method (Angulo & López-Marcos 2004). This method describes the development of the size distribution N by approximating it along a collection of individual growth trajectories spanning the size spectrum. Following a disturbance, a series of such trajectories are introduced into each patch. These trajectories then change according to the growth of the corresponding individuals, conditioned on their survival (Fig. 3a). Meanwhile, growth and mortality combine to alter the density of individuals along these trajectories (Fig. 3b). The characteristic method is similar to the Escalator Boxcar Train (EBT) method (De Roos, Taljapurkar & Caswell, 1997; Brännström, Carlsson & Simpson, 2013) used by Falster *et al.* (2011, 2015), but not identical. An advantage of the characteristic method is that it allows direct approximation of the size distribution. For the same reason, however, the characteristic method may be less suitable for resolving size distributions in models where those distributions naturally develop sharp peaks (which happens, e.g. when plants completely cease to grow at a given height).

We implement a novel technique for handling strongly size-asymmetric competitive feedbacks, such as those occurring under strong competition for light (see Appendix S1 for details). This technique works for both the characteristic method and the EBT method. It involves an adaptive refinement of the time points where new growth trajectories or cohorts are introduced into the population, an idea first applied in Falster *et al.* (2011) and described further in Falster *et al.* (2015). Normally, growth trajectories are introduced evenly, spaced out according to a fixed time interval. Under size-asymmetric competition, however, the growth trajectories of individuals born at nearby times can diverge substantially over time (Fig. 3a; Appendices S1 and S3). This can become a problem during patch development, because much of the population can be located in between widely spaced trajectories, causing low numerical accuracy. To maintain numerical accuracy, we use an iterative algorithm to adaptively refine the cohort spacing until adjacent trajectories remain adequately resolved throughout patch development (see Appendix S1 for details). This refinement gives rise to an uneven distribution of cohort introduction times across patch age (Fig. 3a), with a considerably tighter spacing of trajectories in younger patches.

Modelling the full size-structured dynamics within patches enables multiple demographic phenomena to be investigated. First, `plant` can simulate demographic phenomena typically observed in patches recovering from a disturbance, including self-thinning and successional replacement. Whereas other approaches for modelling self-thinning are limited to a single species (e.g. Barnes & Roderick 2004, Coomes & Allen 2007) TSPMs can accommodate multiple species with different traits (Figs 1b and 3). Secondly, the self-thinning and successional replacement emerges from the combined effects of physiology and competition for resources, rather than being prescribed by the model. As a consequence, it is possible to ask how alternative trait values affect fitness. Thirdly, TSPMs allow the combined effects of physiology and succession on a patch's emergent properties to be investigated (Moorcroft, Hurtt & Pacala 2001; Falster *et al.* 2011). In particular, leaf area cover and rates of biomass production tend to vary nonlinearly with patch age (Smith & Long 2001; Binkley *et al.* 2002; Ogawa *et al.* 2010); in `plant`, such outcomes naturally arise as emergent properties, driven by underlying physiology and demography (Fig. 3c).

Trait-, size- and patch-structured vegetation

All vegetation in `plant` is subject to a disturbance regime, implying that patches are cleared at a given rate (e.g. Treurnicht *et al.* 2016). Such clearance can be interpreted, for example as resulting from a fire, cyclone, clear-cutting or landslide. The vegetation thus comprises a series of patches differing in the time that has passed since their last disturbance, linked via seed dispersal (Fig. 1b). Such a system is often referred to as a structured metapopulation (Gyllenberg & Metz 2001).

In `plant`, following an approach established in previous TSPMs (Kohyama 1993; Moorcroft, Hurtt & Pacala 2001; Falster *et al.* 2011), we assume an infinite number of patches, all experiencing the same disturbance regime and sharing a common seed dispersal pool, or global seed rain (as in the

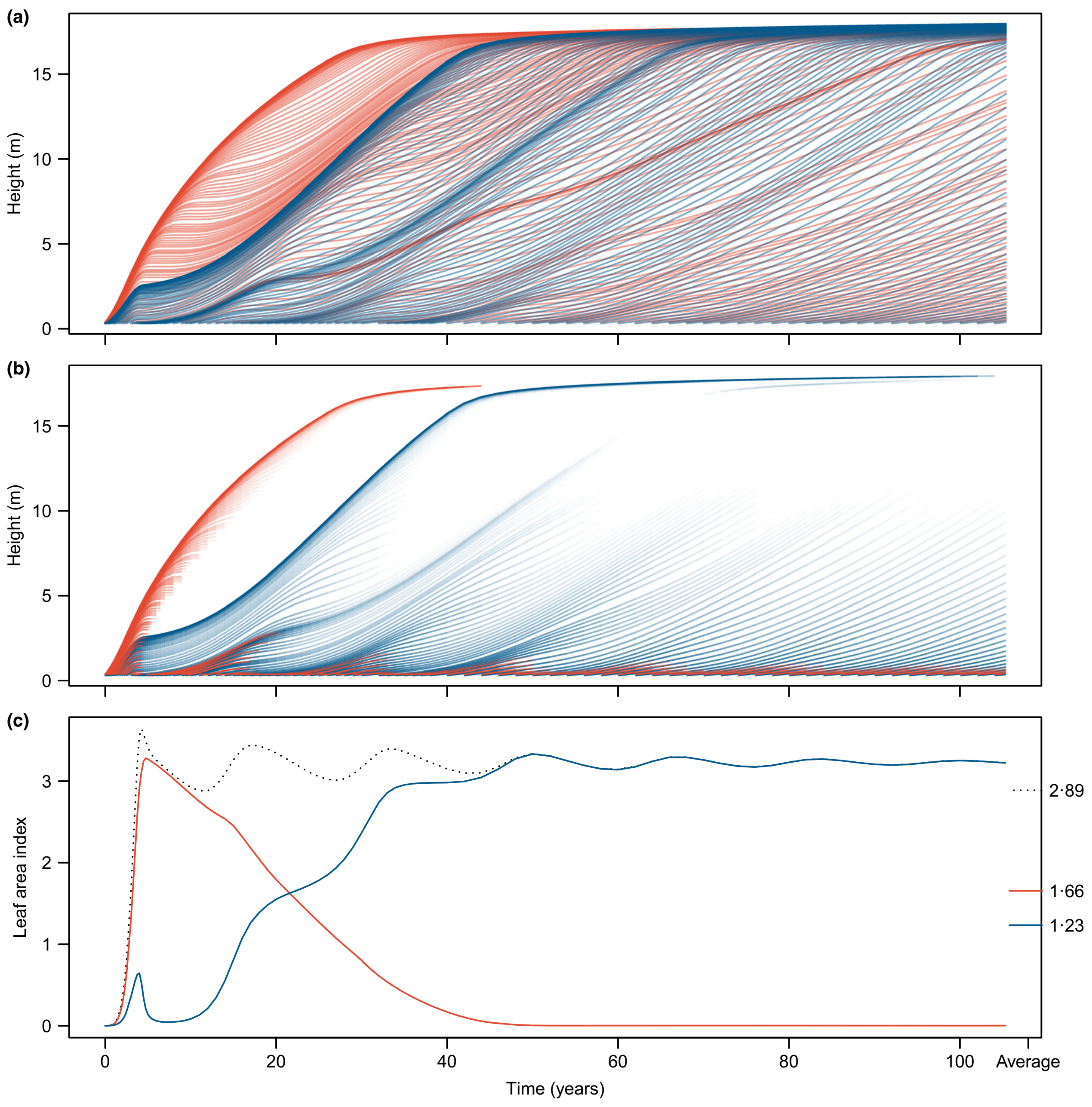


Fig. 3. Population dynamics for two species competing within a patch. Red and blue lines refer to the low- and high-LMA species from Fig. 2. (a) Growth trajectories for individuals germinating across a range of patch ages for each species, using a schedule of introduction times generated by `plant`'s adaptive algorithm. The initial growth rate advantage of the low-LMA species (Fig. 2c) means it quickly overtops the high-LMA species, suppressing its growth. Self-thinning then creates space for the high-LMA species to establish below the canopy of the faster growing species. (b) Same trajectories as in (a), but shaded according to the density N of individuals at given size and patch age (light = low density, dark = high density). (c) Leaf area index at ground level for each species. The dotted line is the total leaf area index (summed across both species), while the lines on the right-hand side are averages integrated over the whole metapopulation. See Appendix S3 for code reproducing this figure.

island model; Fig. 1b). As mentioned before, we also assume that disturbances remove all vegetation. With this approach, we can scale up from deterministically solving the dynamics of a single patch to solving the dynamics of an entire metapopulation. Under these assumptions, the frequency distribution $P(a)$ of patches aged a changes deterministically according to a second partial differential equation (see Appendix S1 for details). The scaling from patches to the metapopulation is then achieved by weighting the temporal dynamics of a single patch by $P(a)$, that is by the relative abundance of patches aged a in the metapopulation (Fig. 3c).

The main numerical challenge in stepping from a single patch to a metapopulation is to identify the equilibrium seed rain of the metapopulation. An input seed rain is required to simulate the demography of the metapopulation, and this in turn produces an output seed rain. Equilibrium occurs when, for each species, the input seed rain equals the output seed rain. Identifying this equilibrium requires finding the root of a multidimensional function, subject to the constraint that a unique positive root exists and the root is dynamically stable (the trivial equilibrium of zero seed rain always exists, but is often not stable; Appendix S3). Solving for multidimensional roots can be computationally challenging, with no single technique guaranteed to find a solution. In `plant`, we sequentially apply rounds of iteration and nonlinear root finding, while also using some heuristics to try to ensure that the root returned is an attracting point (see Appendices S1, S3 for details). Real seed rains are of course noisy, but the aforementioned approach allows for mean tendencies to be estimated (which is required for computing fitness; see below).

An attractive feature of the metapopulation concept implemented in TSPMs is that it integrates equilibrium and non-equilibrium approaches to modelling ecological dynamics (Kohyama 1993; Moorcroft, Hurtt & Pacala 2001; Falster *et al.* 2011). An equilibrium being approached at the level of the metapopulation implies that seed rain and size structure across the metapopulation are approximately stable. Yet, even under this condition, the structure of vegetation within individual patches remains constantly in flux: patches continue to age, and their residents continue to grow, until the next disturbance occurs.

Such dynamic equilibria arising in TSPMs potentially resolve several challenges faced by unstructured models. First, by scaling up from patches to a metapopulation, we close the demographic feedback loop and create a self-regulating system. `plant` yields as outputs the equilibrium seed rains and size distributions for each considered species, based on the combined effects of the model's physiological rules, species traits, competitive interactions and disturbance regime.

Secondly, the metapopulation concept allows for the effects of disturbance regimes on vegetation structure and dynamics to be properly incorporated. In `plant`, characteristics for the entire vegetation (metapopulation) – such as total leaf area, biomass and productivity – are obtained by averaging patch-level outcomes over the frequency distribution of patch ages (e.g. Fig. 3c). Similarly, the distributions of abundance and growth rate observed in large forest plots (e.g., Muller-Landau *et al.* 2006; Coomes & Allen 2007) arise as emergent properties. Moreover, the predictions from `plant` are for a frequency distribution of patch states across the landscape (Fig. 4, see Appendix S3 for details). As such, size- and patch-structured models provide unique opportunities for linking with satellite data and forest survey data (Moorcroft, Hurtt & Pacala 2001; Purves & Pacala 2008).

Finally, with the extension to the metapopulation level, we are modelling demography across the entire plant life cycle and are thus in a position to estimate invasion fitness – the rate at which a rare individual with particular traits can establish (positive invasion fitness) or cannot establish (negative invasion fitness) in a community of established residents at their equilibrium densities (Metz, Nisbet & Geritz 1992). While often calculated as the long-term per capita growth rate, a more convenient indicator of fitness in structured metapopulations is given by the logarithm of the basic reproduction ratio, R (Gyllenberg & Metz 2001; Metz & Gyllenberg 2001). The basic reproduction ratio is simply the average number of seeds produced from one seed of a new type over its lifetime, averaged across the metapopulation. So a new type can invade when $R > 1$. The calculation of R available in `plant` follows equations given by Falster *et al.* (2015) (see Appendix S1 for details).

With this measure of fitness, we can model trait-based adaptive dynamics, and community assembly (Metz, Nisbet & Geritz 1992; Geritz *et al.* 1998; Chesson 2000; Dieckmann & Ferrire 2004; Brännström, Johansson & von Festenberg, 2013). An example for forests was introduced by Falster *et al.* (2015). This is best thought of as occurring on a fitness landscape, a surface with fitness plotted against one or more traits (Fig. 5). The reproductive success of individuals depends both on their own traits and on the densities of other occupants in the current resident community. Fitness landscapes are positive in trait regions that allow invasion, and zero for resident species that are at equilibrium. The slope of the fitness landscape with respect to traits determines the direction of selection, for example the negative slope of the dashed line around the resident in Fig. 5a) indicates selection for smaller trait values (see Appendix S3 for details). Fitness landscapes are continually reshaped as species are introduced and their trait values evolve, reflecting the nonlinear demographic feedbacks that occur under frequency-dependent evolution. Stable coexistence is possible when species with different traits can mutually invade. Following the fitness gradient may lead to an evolutionary branching point; at such a point, there is no directional selection (solid line in Fig. 5a), but both smaller and larger trait values can invade. It is often possible to find a species configuration where fitness is negative everywhere (solid line in Fig. 5b) except for the zero points where species are present. Such communities are immune to invasions by species with any other trait values and represent predictions for the trait mixtures favoured by natural selection.

Identifying conditions promoting trait diversity is a valuable role for `plant`. The simple example shown in the present paper can be extended to multiple dimensions and can be used to investigate how trade-offs facilitate coexistence (Falster *et al.* 2015). Co-occurring plant species differ in a range of physiological traits, yet the conditions allowing for stable coexistence remain largely unknown (Adler *et al.* 2013). A variety of algorithms from the area of adaptive dynamics theory can be applied for modelling community assembly using the estimate of invasion fitness provided by `plant` (e.g. Geritz *et al.* 1998; Dieckmann & Ferrire 2004; Brännström, Johansson & von Festenberg, 2013; Falster *et al.* 2015, Fig. 1).

Implementation details

`plant` is written in C++ and R, using the packages `Rcpp` (Eddelbuettel & François 2011; Eddelbuettel 2013) and `RcppR6` (FitzJohn 2015) to bridge between the two languages. Due to the computationally intensive nature of the model, the core physiological and demographic components are written

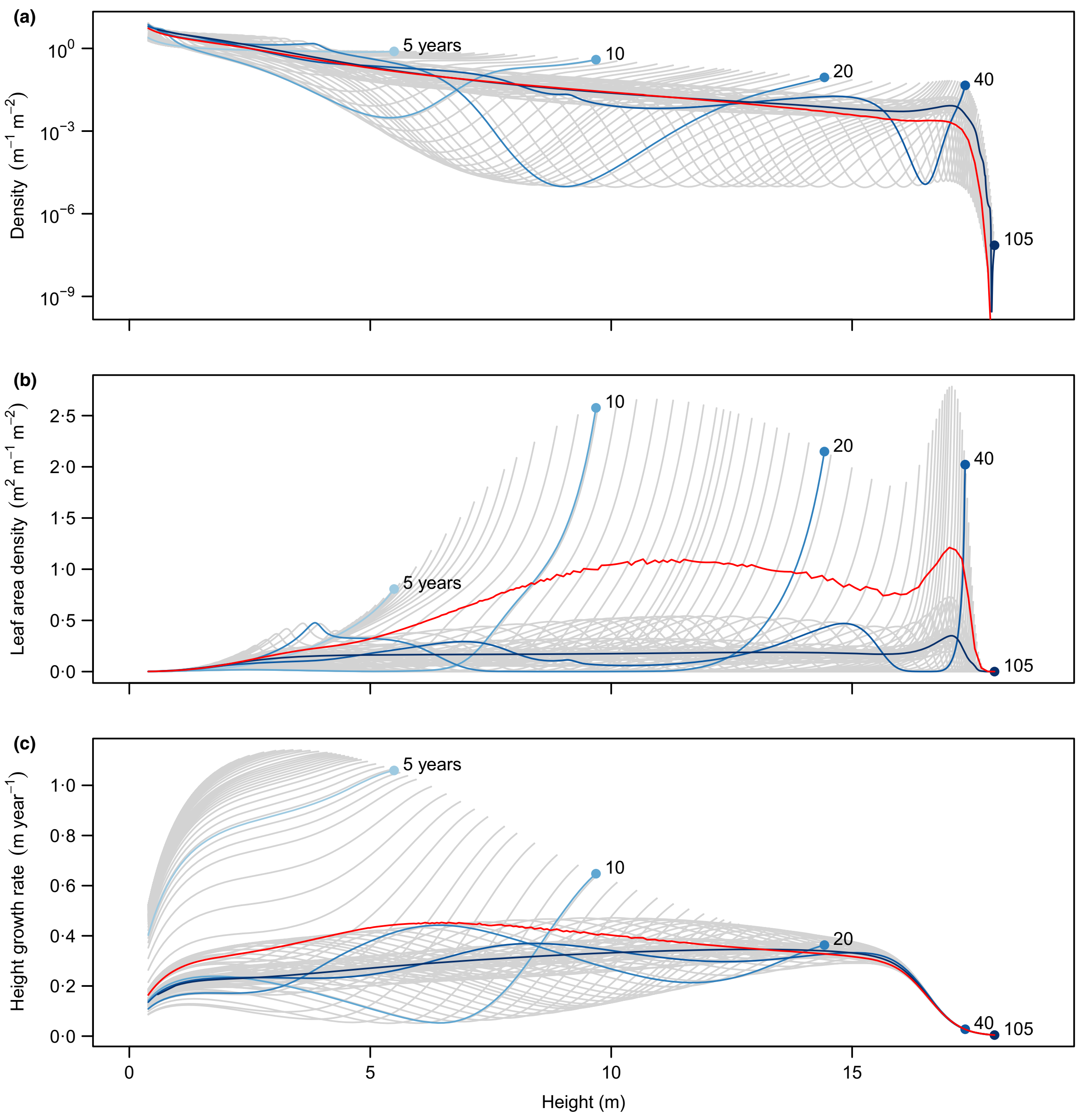


Fig. 4. Emergent size distributions and demography across a metapopulation. Grey lines show patterns within individual patches differing in patch age, that is in time since disturbance, with blue lines highlighting five selected patch ages. Red lines show the average relationship across the metapopulation, obtained by weighting the grey lines by patch frequency and individual abundance. (a) Density of individuals per unit height per unit ground area; (b) density of leaf area per unit height per unit ground area; and (c) height growth rate. See Appendix S3 for code reproducing this figure.

in C++ to maximize speed. We used templated types that allow the demographic and evolutionary components to be driven by alternative physiological models while retaining the same interface at higher levels. Despite mostly being written in C++, all components of the model, down to individual plants, can be extracted and interacted with dynamically in R.

`plant` makes use of much existing software, including the R computational environment (R Core Team, 2015), the R packages `Rcpp` (Eddelbuettel & François 2011; Eddelbuettel 2013), `R6` (Chang 2014), and the Boost Library for C++ (Schäling 2014) via the R package `BH` (Eddelbuettel, Emerson & Kane, 2015). Source code is hosted at github.com/traitecoevo/plant. Installation instructions are available at this web page, and binary releases will also be available there. (`plant` does not currently compile on Windows, but this will change as soon as R core upgrades the Windows C++ compiler.)

Comparison with existing software

Increased interest in capturing the dynamics of size-structured populations (e.g. Griffith *et al.* 2016) has prompted development of several tools, each with its own distinct

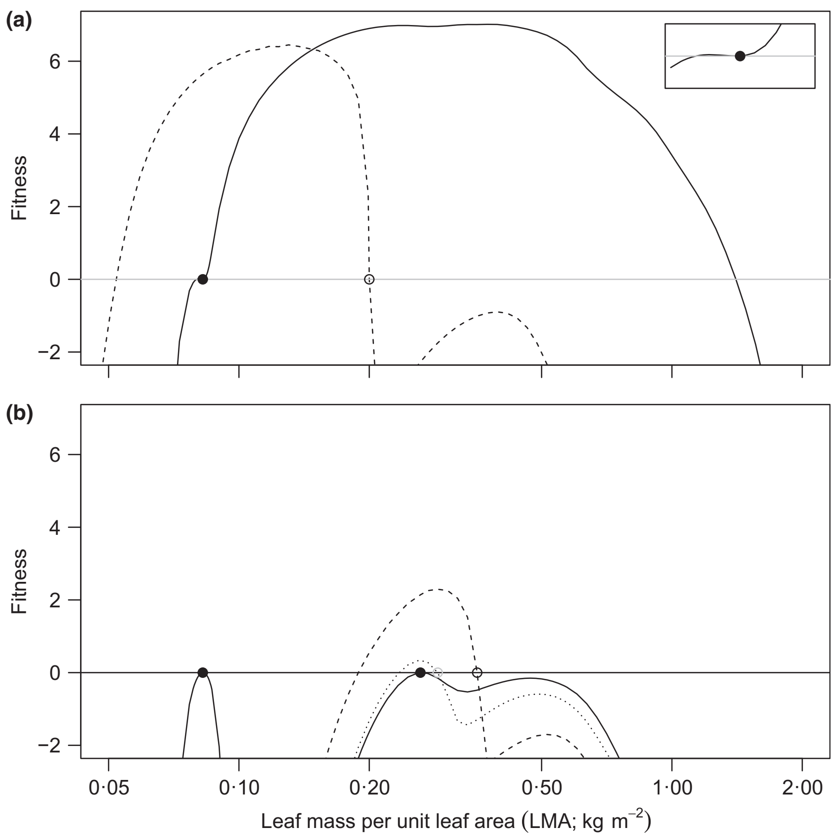


Fig. 5. Example of a fitness landscape for the LMA (leaf mass per unit leaf area) trait, showing potential for the stable coexistence of multiple plant types (adapted from Falster *et al.* 2015). Invasion fitness is the logarithm of the basic reproduction ratio – at zero fitness, species are at equilibrium, at positive fitness, they increase in density. (a) Fitness landscape generated by a single species (indicated by a circle). For an LMA of 0·2 (dashed line, open circle), there is directional selection for smaller LMA. For an LMA of 0·0825 (solid line, filled circle), there is a branching point (see also inset). (b) Fitness landscape generated by two species, holding the first at an LMA of 0·0825. Introducing a species at the maximum of the solid line in (a) reduces the region of positive fitness considerably (dashed line, open circle). Dotted lines show subsequent invasions and replacements of the second species. The solid line shows the fitness landscape of the resultant evolutionarily stable two-species community, with both species coexisting on separate peaks. At this point, no further invasion is possible. See Appendix S3 for code reproducing this figure.

strengths. The EBT software (De Roos, Taljapurkar & Caswell, 1997) provides a flexible framework that has been used to study a range of organisms. Compared to the EBT, `plant` is customized for plants and to that end offers refinements needed to handle modelling of the strongly size-asymmetric feedbacks that occur in plant communities, as well as the extension to a structured metapopulation. Other TSPMs also exist for modelling of vegetation, like ED (ver 1 and 2; Moorcroft, Hurtt & Pacala 2001; Medvigy *et al.* 2009), LPJ-GUESS (Smith *et al.* 2014) and LPJmL-FIT (Sakschewski *et al.* 2015). These models were designed for simulating biogeochemical cycles and vegetation structure across broad geographic areas, and their architectures are optimized for such use. By contrast, `plant` was designed so that the physiological rules driving the dynamics of the system could be easily manipulated and emergent demographic properties (including those of individual plants, patches, metapopulations of patches and fitness) studied in detail. Finally, like `plant`, the LM3-PPA model (Weng *et al.* 2015) also aims to scale from individual-level processes to emergent properties of vegetation, including estimating invasion fitness. Yet, the approaches taken for solving the dynamics are very different. The LM3-PPA relies on complex analytical solutions to the size-structured interactions, whose derivation required simplifying elements of the physiology and ecology in that model. By contrast, in `plant` we have retained the full range of physiological and ecological feedbacks and instead relied on using robust numerical methods to solve the system dynamics (see Appendix S1 for details).

Closing comments

The `plant` package is designed to make it easier for researchers and forest practitioners alike to investigate a variety of nonlinear demographic, ecological and evolutionary phenomena. Rather than provide a model with a few entry points, we have developed an extensible framework that we hope can be used to answer a number of questions at a variety of scales. The specific applications highlighted here arise from our interest in the assembly of trait mixtures in communities. A variety of other uses are possible, addressing other aspects of community assembly and function. For example, we expect the package will prove useful for (i) investigating how morphological and physiological traits influence growth rate and shade tolerance, (ii) studying emergent phenomena such as multispecies self-thinning, successional transitions and age-related changes in vegetation productivity and (iii) bridging between empirical data, large-scale simulators of global biogeochemical cycles and simple abstract theoretical models.

Acknowledgements

We thank J Camac, A De Roos, R Duursma, J Johansson and C Prentice for helpful discussions and three anonymous reviewers for comments on an earlier draft. Falster was supported by an Australian Research Council (ARC) discovery grant (DP110102086). FitzJohn was supported by the Science and Industry Endowment Fund (RP04-174). Brännström was supported by the Swedish Research Council Formas (2012–2008). Dieckmann was supported by the European Commission, the European Science Foundation, the Austrian Science Fund, the Austrian Ministry of Science and Research, and the Vienna Science and Technology Fund. Westoby was supported by a fellowship from the ARC. The authors have no conflict of interests to declare.

Data accessibility

Source code for reproducing the entire contents of this article is available at https://github.com/traitecoevo/plant_paper; see also Appendix S3 for worked examples reproducing the figures. This article does not use any data.

Author contribution

RGF and DSF wrote and developed the software, including numerical algorithms. ÅB and UD helped develop algorithms included in prototype software. DSF, RGF and MW wrote the article, with input from ÅB and UD. All authors contributed to material presented in Supporting information on demography and physiology. RGF and DSF developed and edited the worked examples showing how to interact with plant from R.

References

Adler, P.B., Fajardo, A., Kleinhesselink, A.R. & Kraft, N.J.B. (2013) Trait-based tests of coexistence mechanisms. *Ecology Letters*, **16**, 1294–1306.

Angulo, O. & López-Marcos, J.C. (2004) Numerical integration of fully nonlinear size-structured population models. *Applied Numerical Mathematics*, **50**, 291–327.

Baltzer, J.L. & Thomas, S.C. (2007) Determinants of whole-plant light requirements in Bornean rain forest tree saplings. *Journal of Ecology*, **95**, 1208–1221.

Barnes, B. & Roderick, M.L. (2004) An ecological framework linking scales across space and time based on self-thinning. *Theoretical Population Biology*, **66**, 113–128.

Binkley, D., Stape, J.L., Ryan, M.G., Barnard, H.R. & Fownes, J. (2002) Age-related decline in forest ecosystem growth: an individual-tree, stand-structure hypothesis. *Ecosystems*, **5**, 58–67.

Brännström, Å., Carlsson, L. & Simpson, D. (2013) On the convergence of the escalator boxcar train. *SIAM Journal on Numerical Analysis*, **51**, 3213–3231.

Brännström, Å., Johansson, J. & von Festenberg, N. (2013) The hitchhikers guide to adaptive dynamics. *Games*, **4**, 304–328.

Calcagno, V., Mouquet, N., Jarne, P. & David, P. (2006) Coexistence in a metacommunity: the competition-colonization trade-off is not dead. *Ecology Letters*, **9**, 897–907.

Chang, W. (2014) *R6: Classes with Reference Semantics*. URL: http://CRAN.R-project.org/package = R6.

Chesson, P. (2000) Mechanisms of maintenance of species diversity. *Annual Review of Ecology and Systematics*, **31**, 343–366.

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

De Kauwe, M.G., Medlyn, B.E., Zaehle, S., Walker, A.P., Dietze, M.C., Wang, Y.P. *et al.* (2014) Where does the carbon go? A model-data intercomparison of vegetation carbon allocation and turnover processes at two temperate forest free-air CO_2 enrichment sites. *New Phytologist*, **203**, 883–899.

De Roos, A.M., Taljapurkar, S. & Caswell, H. (1997) *A Gentle Introduction to Physiologically Structured Population Models. Structured Population Models in Marine, Terrestrial and Fresh-Water Systems*, pp. 119–204. Chapman & Hall, New York.

Dieckmann, U. & Ferrire, R. (2004) Adaptive dynamics and evolving biodiversity. *Evolutionary Conservation Biology* (eds R. Ferrire, U. Dieckmann & D. Couvet, eds), pp. 188–224. Cambridge University Press, Cambridge, UK.

Eddelbuettel, D. (2013) *Seamless R and C++ Integration with Rcpp*. Springer, New York. ISBN 978-1-4614-6867-7.

Eddelbuettel, D., Emerson, J.W. & Kane, M.J. (2015) *BH: Boost C++ Header Files*. R package version 1.55.0-3.

Eddelbuettel, D. & François, R. (2011) Rcpp: Seamless R and C++ integration. *Journal of Statistical Software*, **40**, 1–18.

Enquist, B.J., Kerkhoff, A.J., Stark, S.C., Swenson, N.G., McCarthy, M.C. & Price, C.A. (2007) A general integrative model for scaling plant growth, carbon flux, and functional trait spectra. *Nature*, **449**, 218–222.

Falster, D.S., Brännström, Å., Dieckmann, U. & Westoby, M. (2011) Influence of four major plant traits on average height, leaf-area cover, net primary productivity, and biomass density in single-species forests: a theoretical investigation. *Journal of Ecology*, **99**, 148–164.

Falster, D.S., Brännström, Å., Westoby, M. & Dieckmann, U. (2015) Multi-trait eco-evolutionary dynamics explain niche diversity and evolved neutrality in forests. *bioRxiv*, p. 014605.

FitzJohn, R.G. (2015) *RcppR6: Code-generation Wrapping C++ Classes as R6 Classes*. URL: https://github.com/richfitz/RCPPR6.

Geritz, S.A.H., Kisdi, É., Meszéna, G. & Metz, J.A.J. (1998) Evolutionarily singular strategies and the adaptive growth and branching of the evolutionary tree. *Evolutionary Ecology*, **12**, 35–57.

Givnish, T.J. (1988) Adaptation to sun and shade: a whole-plant perspective. *Australian Journal of Plant Physiology*, **15**, 63–92.

Griffith, A.B., Salguero-Gomez, R., Merow, C. & McMahon, S.M. (2016) Demography beyond the population. *Journal of Ecology*, **104**, 271–280.

Gyllenberg, M. & Metz, J.A.J. (2001) On fitness in structured metapopulations. *Journal of Mathematical Biology*, **43**, 545–560.

Harper, J.L. (1977) *Population Biology of Plants*. Academic Press, London.

Hubbell, S.P. (2001) *The Unified Neutral Theory of Biodiversity and Biogeography. Monographs in Population Biology*. Princeton University Press, Princeton and Oxford.

Huston, M. & Smith, T. (1987) Plant succession: life history and competition. *American Naturalist*, **130**, 168–198.

King, D.A. (2011) Size-related changes in tree proportions and their potential influence on the course of height growth. *Size- and Age-related Changes in Tree Structure and Function* (eds F.C.C. Meinzer, B. Lachenbruch, T.E.E. Dawson, F.C. Meinzer & Ü. Niinemets) volume 4, pp. 165–191. Springer, the Netherlands.

Kohyama, T. (1993) Size-structured tree populations in gap-dynamic forest: the forest architecture hypothesis for the stable coexistence of species. *Journal of Ecology*, **81**, 131–143.

Lusk, C.H. & Jorgensen, M.A. (2013) The whole-plant compensation point as a measure of juvenile tree light requirements. *Functional Ecology*, **27**, 1286–1294. Cited by 0001.

MacArthur, R. & Levins, R. (1967) The limiting similarity, convergence, and divergence of coexisting species. *American Naturalist*, **101**, 377–385.

Medvigy, D., Wofsy, S.C., Munger, J.W., Hollinger, D.Y. & Moorcroft, P.R. (2009) Mechanistic scaling of ecosystem function and dynamics in space and time Ecosystem Demography model version 2. *Journal of Geophysical Research*, **114**, G01002.

Metz, J.A.J. & Diekmann, O. (1986) *The Dynamics of Physiologically Structured Populations, Volume 68 of Lecture Notes in Biomathematics*. Springer, Berlin.

Metz, J.A.J. & Gyllenberg, M. (2001) How should we define fitness in structured metapopulation models? Including an application to the calculation of evolutionarily stable dispersal strategies. *Proceedings of the Royal Society of London – Series B: Biological Sciences*, **268**, 499–508.

Metz, J.A.J., Nisbet, R.M. & Geritz, S.A.H. (1992) How should we define 'fitness' for general ecological scenarios? *Trends in Ecology and Evolution*, **7**, 198–202.

Moorcroft, P.R., Hurtt, G.C. & Pacala, S.W. (2001) A method for scaling vegetation dynamics: the Ecosystem Demography model (ED). *Ecological Monographs*, **71**, 557–586.

Muller-Landau, H.C., Condit, R.S., Chave, J., Thomas, S.C., Bohlman, S.A., Bunyavejchewin, S. *et al.* (2006) Testing metabolic ecology theory for allometric scaling of tree size, growth and mortality in tropical forests. *Ecology Letters*, **9**, 575–588.

Ogawa, K., Adu-Bredu, S., Yokota, T. & Hagihara, A. (2010) Leaf biomass changes with stand development in hinoki cypress (*Chamaecyparis obtusa* [Sieb. et Zucc.] Endl.). *Plant Ecology*, **211**, 79–88.

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

Purves, D. & Pacala, S. (2008) Predictive models of forest dynamics. *Science*, **320**, 1452–1453.

R-Core and Team (2015) *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria.

Rees, M. & Ellner, S. (2016) Evolving integral projection models: evolutionary demography meets eco-evolutionary dynamics. *Methods in Ecology and Evolution*, **7**, 157–170.

Rüger, N., Berger, U., Hubbell, S.P., Vieilledent, G. & Condit, R. (2011) Growth strategies of tropical tree species: disentangling light and size effects. *PLoS One*, **6**, e25330.

Sakschewski, B., von Bloh, W., Boit, A., Rammig, A., Kattge, J., Poorter, L., Peñuelas, J. & Thonicke, K. (2015) Leaf and stem economics spectra drive diversity of functional plant traits in a dynamic global vegetation model. *Global Change Biology*, **21**, 2711–2725.

Schäling, B. (2014) *The Boost C++ Libraries*, 2nd edn. XML Press, Laguna Hills, California, USA.

Shugart, H.H. & West, D.C. (1980) Forest succession models. *BioScience*, **30**, 308–313.

Sitch, S., Huntingford, C., Gedney, N., Levy, P.E., Lomas, M., Piao, S.L. *et al.* (2008) Evaluation of the terrestrial carbon cycle, future plant geography and climate-carbon cycle feedbacks using five Dynamic Global Vegetation Models (DGVMs). *Global Change Biology*, **14**, 2015–2039.

Smith, B., Wårlind, D., Arneth, A., Hickler, T., Leadley, P., Siltberg, J. & Zaehle, S. (2014) Implications of incorporating N cycling and N limitations on primary production in an individual-based dynamic vegetation model. *Biogeosciences*, **11**, 2027–2054.

Smith, F.W. & Long, J.N. (2001) Age-related decline in forest growth: an emergent property. *Forest Ecology and Management*, **144**, 175–181.

Strigul, N., Pristinski, D., Purves, D., Dushoff, J. & Pacala, S. (2008) Scaling from trees to forests: tractable macroscopic equations for forest dynamics. *Ecological Monographs*, **78**, 523–545.

Thomas, S.C. (2011) Age-related changes in tree growth and functional biology: the role of reproduction. *Size- and Age-Related Changes in Tree Structure and Function* (eds F.C. Meinzer, B. Lachenbruch & T.E. Dawson), volume 4, pp. 33–64. Springer Netherlands, Dordrecht.

Tilman, D. (1985) The resource-ratio hypothesis of plant succession. *American Naturalist*, **125**, 827–852.

Treurnicht, M., Pagel, J., Esler, K. *et al.* (2016) Environmental drivers of demographic variation across the global geographical range of 26 plant species. *Journal of Ecology*, **104**, 331–342.

Uriate, M., Lasky, J., Boukili, V. & Chazdon, R. (2016) A trait-mediated, neighborhood approach to quantify climate impacts on successional dynamics of tropical rainforests. *Functional Ecology*, **104**, 331–342

Visser, M.D., Bruijning, M., Wright, S.J., Muller-Landau, H., Jongejans, E., Comita, L.S. & de Kroon, H. (2016) Functional traits as predictors of vital rates across the life-cycle of tropical trees. *Functional Ecology*, **30**, 168–180.

Weng, E.S., Malyshev, S., Lichstein, J.W., Farrior, C.E., Dybzinski, R., Zhang, T., Shevliakova, E. & Pacala, S.W. (2015) Scaling from individual trees to forests in an Earth system modeling framework using a mathematically tractable model of height-structured competition. *Biogeosciences*, **12**, 2655–2694.

Westoby, M., Falster, D.S., Moles, A.T., Vesk, P. & Wright, I.J. (2002) Plant ecological strategies: some leading dimensions of variation between species. *Annual Review of Ecology and Systematics*, **33**, 125–159.

Wright, I.J., Reich, P.B., Westoby, M., Ackerly, D., Baruch, Z., Bongers, F. *et al.* (2004) The world-wide leaf economics spectrum. *Nature*, **428**, 821–827.

Received 1 September 2015; accepted 25 October 2015
Handling Editor: Sean McMahon

Supporting Information

Additional Supporting Information may be found in the online version of this article.

Appendix S1. Modelling the demography of plants, patches and metapopulations.

Appendix S2. Description of the `FF16` physiological model.

Appendix S3. Worked examples showing how to interact with `plant` from R.

SUPPORTING INFORMATION FOR:
"PLANT: A PACKAGE FOR MODELLING FOREST TRAIT ECOLOGY & EVOLUTION"

Modelling the demography of plants, patches, and metacommunities

Daniel S. Falster†, Richard G. FitzJohn, Åke Brännström, Ulf Dieckmann, and Mark Westoby

†Department of Biological Sciences, Macquarie University, Sydney, Australia

daniel.falster@mq.edu.au, rich.fitzjohn@gmail.com

CONTENTS

1 INTRODUCTION

This document outlines methods used to model demography in the PLANT package. We first outline the system of equations being solved. These equations were primarily developed elsewhere, in particular in Falster *et al.* (2011) and Falster *et al.* (2015), following general principles laid out in De Roos, Taljapurkar & Caswell (1997), Kohyama (1993), and Moorcroft, Hurtt & Pacala (2001). They are presented here so that users can understand the full system of equations being solved within PLANT. We then describe the numerical techniques used to solve the equations. Table 1 provides a list of names and definitions used throughout this document.

2 METACOMMUNITY DYNAMICS

The following material is written assuming there exists a physiological sub-model that takes as inputs a plant's traits x, height H, and light environment E_a, and returns rates of growth, mortality, and fecundity. The physiological model must describe how much leaf area the plant contributes to shading other plants in the patch. Specifically, the physiological model is responsible for calculating the following variables from Table 1: $A_l(H)$, $Q(z,H)$, $H_0(x)$, $g(x,H,E_a)$, $f(x,H,E_a)$, $d(x,H,E_a)$, and $S_G(x,H_0,E_{a0})$. All are other variables in Table 1 are calculated in the demographic model using the equations provided below.

2.1 *Individual plants*

We first consider the dynamics of an individual plant. Throughout, we refer to a plant as having traits x and size H, with the latter given by its height. The plant grows in a light environment E, a function describing the distribution of light with respect to height within a patch. Ultimately, E depends on the composition of plants in a patch, and thus on the patch age a. To indicate this dependence, we write E_a. Further, let $a_0$be the age in which the plant germinated, and $H_)(x)$ its initial height. The functions $g(x,H,E_a)$, $f(x,H,E_a)$, and $d(x,H,E_a)$ denote the growth, death, and fecundity rates of the plant. Then,

$$H(x,a_0,a) = H_0(x) + \int_{a_0}^{a} g(x, H(x,a_0,a'), E_{a'})\,\mathrm{d}a' \tag{1}$$

is the trajectory of plant height,

$$S_\mathrm{I}(x,a_0,a) = S_\mathrm{G}(x,H_0,E_\mathrm{a0})\,\exp\left(-\int_{a_0}^{a} d(x,H(x,a_0,a'),E_{a'})\,\mathrm{d}a'\right) \tag{2}$$

is the probability of plant survival within the patch, where $S_\mathrm{G}(x,H_0,E_\mathrm{a0})$ is the probability a seed germinates successfully, and

$$\tilde{R}(x,a_0,a) = \int_{a_0}^{a} f(x,H(x,a_0,a'),E_{a'})\,S_\mathrm{I}(x,a_0,a')\,\mathrm{d}a' \tag{3}$$

is the plant's cumulative seed output, from its birth at patch age $a = a_0$ until patch age a.

The notational complexity required in Eqs. 1-3 potentially obscures an important point: these equations are simply the general, nonlinear solutions to integrating growth, mortality, and fecundity over time.

2.2 *Patches of competing plants*

We now consider a patch of competing plants. At any age a, the patch is described by the size-density distribution $N(H|x,a)$ of plants with traits x and height H. In a finite-sized patch, N is described by a collection of points, each indicating the height of one individual, whereas in a very (infinitely) large patch, N is a continuous function. In either case, the demographic behaviour of the plants within the patch is given by Eqs. 1-3. Integrating the dynamics over time is complicated by two other factors: (i) plants interact, thereby altering E_a with age, and (ii) new individuals may establish, expanding the system of equations.

In the current version of PLANT, plants interact by shading one another. Following standards biophysical principles, we let the canopy openness $E_a(z)$ at height z in a patch of age a decline exponentially with the total amount of leaf area above z,

$$E_a(z) = \exp\left(-k_{\rm I} \sum_{i=1}^{K} \int_0^\infty A_{\rm l}(H)\, Q(z,H)\, N(H|x_i,a)\, \mathrm{d}H\right), \tag{4}$$

where $A_{\rm l}(H)$ is the total leaf area of a plant with height H, $Q(z,H)$ is the fraction of this leaf area situated above height z for plants of height H, $k_{\rm I}$ is the light extinction coefficient, and K is the number of species in the metacommunity.

Assuming patches are sufficiently large, the dynamics of N can be modelled deterministically via the following partial differential equation (PDE) (Kohyama, 1993; De Roos, Taljapurkar & Caswell, 1997; Moorcroft, Hurtt & Pacala, 2001),

$$\frac{\partial}{\partial a} N(H|x,a) = -d(x,H,E_a)\, N(H|x,a) - \frac{\partial}{\partial H}\left[g(x,H,E_a)\, N(H|x,a)\right]. \tag{5}$$

See section 4.2 for the derivation of this PDE.

Eq. 5 has two boundary conditions. The first links the flux of individuals across the lower bound $(H_0(x))$ of the size-density distribution to the rate Y_x at which seeds arrive in the patch,

$$N(H_0|x,a_0) = \begin{cases} \frac{Y_x\, S_{\rm G}(x,H_0,E_{\rm a0})}{g(x,H_0,E_{\rm a0})} & \text{if } g(x,H_0,E_{\rm a0}) > 0 \\ 0 & \text{otherwise.} \end{cases} \tag{6}$$

The function $S_{\rm G}(x,H_0,E_{\rm a0})$ denotes survival through germination and must be chosen such that $S_{\rm G}(x,H_0,E_{\rm a0})/g(x,H_0,E_{\rm a0}) \to 0$ as $g(x,H_0,E_{\rm a0}) \to 0$, to ensure a smooth decline in initial density as conditions deteriorate (Falster *et al.*, 2011).

The second boundary condition of Eq. 5 specifies the initial size-density distribution in patches right after a disturbance, i.e., for $a = 0$. Throughout, we consider only situations starting with an empty patch,

$$N(H|x,0) = 0, \tag{7}$$

although non–zero distributions could alternatively be specified (e.g., Kohyama, 1993; Moorcroft, Hurtt & Pacala, 2001).

2.3 *Age structure of patches*

We now consider the distribution of patch age a in the metacommunity. With a denoting the time since last disturbance, $P(a)$ denotes the frequency-density of patch age a and $d_{\rm P}(a)$ is the age-dependent probability that a patch of age a is transformed into a patch of age 0 by a disturbance event. Here we focus on situations where the age structure has reached an equilibrium. See section 4.1 for the derivation and non-equilibrium situations. The dynamics of P are given by (McKendrick, 1926; Foerster, 1959)

$$\frac{\mathrm{d}}{\mathrm{d}a} P(a) = -d_{\mathrm{P}}(a)\, P(a), \tag{8}$$

with the boundary condition

$$P(0) = \int_0^\infty d_{\mathrm{P}}(a)\, P(a)\, \mathrm{d}a. \tag{9}$$

The probability that a patch remains undisturbed from patch age a_0 to patch age a is then given by

$$S_{\mathrm{P}}(a_0, a) = \exp\left(- \int_{a_0}^{a} d_{\mathrm{P}}(a')\, \mathrm{d}a' \right). \tag{10}$$

These equations lead to an equilibrium distribution of patch ages,

$$P(a) = P(0) S_{\mathrm{P}}(0, a), \tag{11}$$

where

$$P(0) = \frac{1}{\int_0^\infty S_{\mathrm{P}}(0, a) \mathrm{d}a} \tag{12}$$

is the average disturbance frequency of a patch and, at the same time, the frequency-density of patches of age 0.

The default approach in PLANT is to assume that $d_{\mathrm{P}}(a)$ increases linearly with patch age, which leads to a Weibull distribution for P, specified by a single parameter $\bar{a}$ measuring the mean interval between disturbances (see section 4.1 for details).

2.4 *Trait-, size-, and patch-structured metacommunities*

We consider a large area of habitat where: (i) disturbances (such as fires, storms, landslides, floods, or disease outbreaks) strike patches of the habitat on a stochastic basis, killing all individuals within the affected patches; (ii) individuals compete for resources within patches, with a spatial scale of competitive interactions that renders negligible such interactions between individuals in adjacent patches; and (iii) there is high connectivity via dispersal between all patches in the habitat, allowing empty patches to be quickly re-colonised. Such a system can be modelled as a metapopulation (for a single species) or metacommunity (for multiple species). The dynamics of such a metacommunity are described by the PDEs in Eqs. 5 and 8.

The seed rain of each species in the metacommunity is given by the rate at which seeds are produced across all patches,

$$Y_x = \int_0^\infty \int_0^\infty P(a)\, S_{\mathrm{D}}\, f(x, H, E_a)\, N(H|x, a)\, \mathrm{d}H\, \mathrm{d}a, \tag{13}$$

where S_{D} is the survival probability of seeds during dispersal.

A convenient feature of Eqs. 5 - 7 is that the dynamics of a single patch scale up to give the dynamics of the entire metacommunity. Note that the rate Y_x at which offspring of individuals with traits x arrive from the disperser pool is constant when the metacommunity is at equilibrium. Combined with the assumption that all patches have the same initial (empty) size-density distribution after a disturbance, the assumption of constant seed rains ensures that all patches show the same temporal behaviour, the only difference between them being their ages.

To model the temporal dynamics of an archetypal patch, we need only a value for Y_x. The numerical challenge is therefore to find the right value for Y_x, by solving Eqs. 6 and 13 simultaneously, for all species with traits x in the metacommunity.

2.5 *Emergent properties of metacommunities*

Summary statistics of a metacommunity are obtained by integrating over the size-density distribution, weighting by the frequency-density $P(a)$. In particular, the average density of individuals per unit ground area across the metacommunity is

$$\hat{N}(x) = \int_0^\infty \int_0^\infty P(a)\, N(H|x,a)\, \mathrm{d}a\, \mathrm{d}H, \tag{14}$$

and the average size-density of plants with height H is

$$\bar{N}(x,H) = \int_0^\infty P(a)\, N(H|x,a)\, \mathrm{d}a. \tag{15}$$

Averages for other individual-level quantities can also be calculated. Denoting by $w(x,H,E_a)$ a quantity of interest, either a demographic rate (growth, mortality) or a state (plant height, leaf area, light environment), the average of w for plants with height H and traits x is

$$\bar{w}(x,H) = \frac{1}{\bar{N}(x,H)} \int_0^\infty P(a)\, N(H|x,a)\, w(x,H,E_a)\, \mathrm{d}a. \tag{16}$$

The average of w across all individuals of the species is

$$\hat{w}(x) = \frac{1}{\hat{N}(x)} \int_0^\infty \int_0^\infty P(a)\, N(H|x,a)\, w(x,H,E_a)\, \mathrm{d}a\, \mathrm{d}H. \tag{17}$$

When calculating the average mortality rate, one must decide whether mortality due to patch disturbance is included. Non-disturbance mortality is obtained by setting $w(x,H,E_a) = d(x,H,E_a)$, while the total mortality due to growth processes and disturbance is obtained by setting $w(x,H,E_a) = d(x,H,E_a) + d_{\mathrm{P}}(a)S_{\mathrm{P}}(0,a)$.

Similarly, we can integrate over the size-density distribution to extract aggregate features of the vegetation within a patch,

$$W(a) = \sum_{i=1}^{N} \int_0^\infty N(H|x_i,a)\, w(x_i,H,E_a)\, \mathrm{d}H, \tag{18}$$

and across the entire metacommunity,

$$\hat{W} = \int_0^\infty P(a)\, W(a)\, \mathrm{d}a. \tag{19}$$

2.6 *Invasion fitness*

We now consider how to estimate the fitness of a rare mutant individual with traits x' growing in the light environment of a resident community with traits x. We focus on phenotype-dependent components of fitness – describing the aggregate consequences of a given set of traits for growth, fecundity, and mortality – taking into account the non-linear effects of competition, but ignoring the underlying genetic basis for trait inheritance and expression. We also adhere to standard conventions in such analyses by assuming that the mutant phenotype is sufficiently rare to have a negligible effect on the light environment where it is growing (Geritz *et al.*, 1998). The approach for calculating fitness implemented here follows the approach described by Falster *et al.* (2015).

In general, invasion fitness is defined as the long-term per capita growth rate of a rare mutant phenotype in the environment determined by a resident phenotype (Metz, Nisbet &

Geritz, 1992). Calculating per capita growth rates, however, is particularly challenging in a structured metacommunity model (Gyllenberg & Metz, 2001; Metz & Gyllenberg, 2001). As an alternative measure of invasion fitness in metacommunities, we can use the basic reproduction ratio measuring the expected number of new dispersers arising from a single dispersal event. For metacommunities at demographic equilibrium, evolutionary inferences made using basic reproduction ratios are equivalent to those made using per capita growth rates (Gyllenberg & Metz, 2001; Metz & Gyllenberg, 2001).

We denote by $R\left(x^{\prime}, x\right)$ the basic reproduction ratio of individuals with traits $x^{\prime}$ growing in the competitive light environment of the resident traits x. Recalling that patches of age a have frequency-density $P(a)$ in the landscape, it follows that any seed with traits $x^{\prime}$ has a probability of $P(a)$ of landing in a patch of age a. The basic reproduction ratio for individuals with traits $x^{\prime}$ is then

$$R\left(x^{\prime}, x\right)=\int_{0}^{\infty} P(a)\ \tilde{R}\left(x^{\prime}, a, \infty\right) \mathrm{d} a, \tag{20}$$

where $\tilde{R}\left(x^{\prime}, a_{0}, a\right)$ is the expected number of dispersing offspring produced by a single dispersing seed with traits $x^{\prime}$ arriving in a patch of age a_{0} up until age a (Gyllenberg & Metz, 2001; Metz & Gyllenberg, 2001). In turn, $\tilde{R}\left(x^{\prime}, a, \infty\right)$ is calculated by integrating an individual's fecundity over the expected lifetime of a patch, taking into account competitive shading from residents with traits x, the individual's probability of surviving, and its traits,

$$\tilde{R}(x^{\prime}, a_{0}, \infty)=\int_{a_{0}}^{\infty} S_{\mathrm{D}}\, f(x^{\prime}, H(x^{\prime}, a_{0}, a), E_{a})\, S_{\mathrm{I}}(x^{\prime}, a_{0}, a)\, S_{\mathrm{P}}(a_{0}, a)\, \mathrm{d} a. \tag{21}$$

3 NUMERICAL METHODS FOR SOLVING METACOMMUNITY DYNAMICS

The equations below describe how we solve the trait-, size-, and patch-structured population dynamics specified in the previous section. Our approach to solving for the size-density distribution is based on the characteristic method (Angulo & López-Marcos, 2004; Angulo, López-Marcos & López-Marcos, 2014, 2016). The characteristic method is related to the Escalator Boxcar Train technique (De Roos, 1988; De Roos, Taljapurkar & Caswell, 1997; De Roos, Diekmann & Metz, 1992), but not identical to it.

When simulating an individual plant, or the development of a patch, we need to solve for the size, survival, and seed output of individual plants. When solving for the size-density distribution in a large patch, we also need to estimate the average abundance of individuals. Each of these problems is formulated as an initial-value ODE problem (IVP), which can be solved using an ODE stepper.

3.1 *Approach*

SIZE — The size of an individual is obtained via Eq. 1, which is solved via the IVP

$$\frac{dy}{dt}=g(x, y, t),$$
$$y(0)=H_{0}(x).$$

SURVIVAL — The probability of an individual surviving from patch age a_{0} to patch age a is obtained via Eq. 2, which is solved via the IVP

$$\frac{dy}{dt}=d(x, H_{i}(t), E_{t}),$$

$$y(0) = -\ln\left(S_G(x, H_0, E_{a0})\right).$$

Survival is then

$$S_I(x, a_0, a) = \exp\left(-y(a)\right).$$

Seed production — The lifetime seed production of individuals is obtained via Eq. 21, which is solved via the IVP

$$\frac{dy}{dt} = S_D\, f(x, H_i(t), E_t)\, S_I(x, a_0, t)\, S_P(a_0, t),$$

$$y(0) = 0,$$

where S_I is calculated as described above and S_P is calculated as in Eq. 10.

Size-density of individuals — By integrating along the characteristics of Eq. 5, the size-density of individuals born with height H_0 and traits x at patch age a_0 is given by (De Roos, Taljapurkar & Caswell, 1997; Angulo & López-Marcos, 2004)

$$N(H|x, a) = N(H_0|x, a_0) \exp\left(-\int_{a_0}^{a}\left[\frac{\partial g(x, H(x, a_0, a'), E_{a'})}{\partial H} + d(x, H(x, a_0, a'), E_{a'})\right] \mathrm{d}a'\right). \tag{22}$$

Eq. 22 states that the size-density N at a specific patch age a is the product of the size-density at patch age a_0 adjusted for changes through growth and mortality. Size-density decreases through time because of mortality, as in a typical survival equation, but also changes because of growth. If growth is slowing with size, (i.e., $\partial g/\partial H < 0$), size-density will increase since the characteristics compress. Conversely, size-density will increases if $\partial g/\partial H > 0$.

Denoting by $[H_0, H_+)$ the range of heights attainable by any individual, our algorithm for solving metacommunity dynamics proceeds by sub-dividing this interval into a series of cohorts with heights $H_0 < H_1 < \ldots < H_k$ at the initial points of the characteristic curves. These cohorts are then transported along the characteristics of Eq. 5. The placement of cohorts is controlled indirectly, via the schedule of patch ages at which new cohorts are introduced into the metacommunity. We then track the demography of each such cohort.

The integral in Eq. 22 is solved via the IVP

$$\frac{dy}{dt} = \frac{\partial g(x, H_i(t), E_t)}{\partial H} + d(x, H_i(t), E_t),$$

$$y(0) = -\ln\left(N(H_0|x, a_0)\, S_G(x, H_0, E_{a0})\right),$$

from which we obtain the size-density

$$N(H_0|x, a_0) = \exp(-y(a)).$$

3.2 *Controls on approximation error*

We now outline how to control the error of the approximate solutions to the system of equations described above. In our algorithm, numerical solutions are required to address a variety of problems:

- To estimate the amount of light at a given height in a patch requires numerically integrating over the size-density distribution within that patch.
- To calculate the assimilation of a plant requires numerically integrating photosynthesis over this light profile.

- To simulate patch dynamics requires numerically identifying a vector of patch ages at which new cohorts are introduced, and then numerically stepping the equations for each cohort forward in time to estimate their size, survival, and fecundity at different subsequent patch ages.
- To solve for the initial height of a plant given its seed mass, and for the equilibrium seed rains across the metacommunity, requires numerical root finding.

As with all numerical techniques, solutions to each of these problems are accurate only up to a specified level. These levels are controlled via parameters in the PLANT code. Below, we provide a brief overview of the different numerical techniques being applied and outline how error tolerance can be increased or decreased. We refer to various control parameters that can be found within the `control` object. For a worked example illustrating how to modify these control parameters, see the section `parameters` of Appendix S3.

INITIAL PLANT HEIGHTS — When a seed germinates, it produces a seedling of given height. The height of these seedlings is assumed to vary with the seed mass. Because there is no analytical solution relating seedling height to seed mass – at least when using the default `FF16` physiological model – we must solve for this height numerically. The calculation is performed by the function `height_seed` within the physiological model, using the Boost library's one-dimensional `bisect` routine (Schäling, 2014; Eddelbuettel, Emerson & Kane, 2015). The accuracy of the solution is controlled by the parameter `plant_seed_tol`.

STEPPING OF ODES — All of the IVPS outlined above (in section 3.1) must be stepped through time. For this, PLANT uses the embedded Runge-Kutta Cash-Karp 4-5 algorithm (Cash & Karp, 1990), with code ported directly from the GNU Scientific Library (Galassi, 2009). The accuracy of the solver is controlled by two control parameters for relative and absolute accuracy, `ode_tol_rel` and `ode_tol_abs`.

APPROXIMATION OF SIZE-DENSITY DISTRIBUTION VIA THE CHARACTERISTIC METHOD — Errors in the approximation of the size-density distribution arise from two sources: (i) coarse stepping of cohorts through time and (ii) poor spacing of cohorts across the attainable size range.

As described above, the stepping of the ODE solver is controlled by two control parameters for relative and absolute accuracy, `ode_tol_rel` and `ode_tol_abs`.

A second factor controlling the accuracy with which cohorts are stepped through time is the accuracy of the derivative calculation according to Eq. 22, calculated via standard finite differencing (Abramowitz & Stegun, 2012). When the parameter `cohort_gradient_richardson` is `TRUE` a Richardson extrapolation (Stoer & Bulirsch, 2002) is used to refine the estimate, up to depth `cohort_gradient_richardson`. The overall accuracy of the derivative is controlled by `cohort_gradient_eps`.

The primary factor controlling the spacing of cohorts is the schedule of cohort introduction times. Because the system of equations to be integrated is deterministic, the schedule of cohort introduction times determines the spacing of cohorts throughout the entire development of a patch. Poor cohort spacing introduces error because various emergent properties – such as total leaf area, biomass, or seed output – are estimated by integrating over the size-density distribution. The accuracy of these integrations declines directly with the inappropriate spacing of cohorts. Thus, our algorithm aims to build an appropriately refined schedule, which allows the required integrations to be performed with the desired accuracy at every time point. At the same time, for reasons of computational efficiency, we want as few cohorts as possible.

A suitable schedule is found using the function `build_schedule`. This function takes an initial vector of introduction times and considers for each cohort whether removing that cohort causes the error introduced when integrating two specified functions over the size-density distribution to jump over the allowable error threshold `schedule_eps`. This calculation is repeated for every time step in the development of the patch. A new cohort is introduced immediately prior to any cohort failing these tests. The dynamics of the patch are then simulated again and the process is repeated, until all integrations at all time points have an error below the tolerable limit `schedule_eps`. Decreasing `schedule_eps` demands higher accuracy from the solver, and thus increases the number of cohorts being introduced. Note that we are assessing whether removing an existing cohort causes the error to jump above the threshold limit, and using this to decide whether an extra cohort – in addition to the one used in the test – should be introduced. Thus, the actual error is likely to be lower than, but at most equal to, `schedule_eps`. The general idea of adaptively refining the vector of introduction times was first applied by Falster *et al.* (2011), and was described further by Falster *et al.* (2015).

To determine the error associated with a given cohort, we integrate two different functions over the size-density distribution, within the function `run_scm_error`. We then assess how much removing the focal cohort increases the error in these two integrations. The first integration, performed by the function `area_leaf_error`, determines how much the removal of the focal cohort increases the error in the estimate of the total leaf area in the patch. The second integration, performed by the function `seed_rain_error`, determines how much the removal of the focal cohort increases the error in the estimate of the total seed production from the patch. The relative error in each integration is then calculated using the `local_error_integration` function.

For a worked example illustrating the `build_schedule` function, see the section `cohort_spacing` of Appendix S3.

CALCULATION OF LIGHT ENVIRONMENT AND INFLUENCE ON ASSIMILATION — To progress with solving the system of ODEs requires that we calculate the amount of shading on each of the cohorts, from all other plants in the patch.

Calculating the canopy openness $E_a(z)$ at a given height z in a patch of age a requires that we integrate over the size-density distribution (Eq. 4). This integration is performed using the trapezium rule, within the function `area_leaf_above` in `species.h`. The main factor controlling the accuracy of the integration is the spacing of cohorts. The cohort introduction times determining the spacing of cohorts are adaptively refined as described above. This implies that also the trapezium integration within the `area_leaf_above` function is adaptively refined via the `build_schedule` function.

The cost of calculating $E_a(z)$ linearly increases with the number of cohorts in the metacommunity. Since the same calculation must be repeated for every cohort, the overall computational cost of a step increases as $O(k^2)$, where k is the total number of cohorts across all species. This disproportionate increase in computational cost with the number of cohorts is highly undesirable.

We reduce the computational cost from $O(k^2)$ to $O(k)$ by approximating $E_a(z)$ with a spline. Eq. 4 describes a function monotonically increasing with size. This function is easily approximated using a piecewise continuous spline fitted to a limited number of points. Once fitted, the spline can be used to estimate any additional evaluations of competitive effect. Since spline evaluations are computationally cheaper than integrating over the size-density distribution, this approach reduces the overall cost of stepping the resident population. A new spline is constructed for each time step.

The accuracy of the spline interpolation depends on the number of points used in its construction and on their placement along the size axis. We select the number and locations of points via an adaptive algorithm. Starting with an initial set of 33 points, we assess how much each point contributes to the accuracy of the spline fit at the location of each cohort, first via exact calculation, and second by linearly interpolating from adjacent cohorts. The absolute difference in these values is compared to the control parameter `environment_light_tol`. If the error is greater than this tolerance, the interval is bisected and the is process repeated (see `adaptive_interpolator.h` for details).

Integration over light environment — Plants have leaf area distributed over a range of heights. Estimating a plant's assimilation at each time step thus requires integrating leaf-level rates over the plant. The integration is performed using Gaussian quadrature, using the QUADPACK routines (Piessens *et al.*, 1983) adapted from the GNU Scientific Library (Galassi, 2009); see `qag.h` for details. If the control parameter `plant_assimilation_adaptive` is `TRUE`, the integration is performed using adaptive refinement with an accuracy controlled by the parameter `plant_assimilation_tol`.

Solving for seed rains — For a single species, solving for Y_x is a straightforward one-dimensional root-finding problem, which can be solved with accuracy `equilibrium_eps` via a simple bisection algorithm (see `equlibrium.R` for details).

Solving for seed rains in metacommunities with multiple species is significantly harder, because there is no generally applicable method for multi-dimensional root finding. In PLANT, we have therefore implemented several different approaches (see `equlibrium.R` for details).

4 APPENDICES

4.1 *Derivation of* PDE *describing age-structured dynamics*

We consider patches of habitat that are subject to an intermittent disturbance, with the age of a patch measuring the time since the last disturbance. Denoting by $P(a,t)$ the frequency-density of patch age a at time t and by $d_P(a)$ the age-dependent probability that a patch of age a is transformed into a patch of age 0 through disturbance, the dynamics of $P(a,t)$ are given by

$$\frac{\partial}{\partial t}P(a,t) = -\frac{\partial}{\partial a}P(a,t) - d_P(a,t)P(a,t),$$

with boundary condition

$$P(0,t) = \int_0^\infty d_P(a,t)P(a,t)\,\mathrm{d}a.$$

The frequency-density of patches of age $a < a'$ is given by $\int_0^{a'} P(a,t)\,\mathrm{d}a$, with $\int_0^\infty P(a,t)\,\mathrm{d}a = 1$. If $\frac{\partial}{\partial t}d_P(a,t) = 0$, then $P(a)$ will approach an equilibrium solution given by

$$P(a) = P(0)\,S_P(0,a),$$

where

$$S_P(0,a) = \exp\left(-\int_0^a d_P(a')\,\mathrm{d}a'\right)$$

is the probability that a patch remains undisturbed for duration a, and

$$P(0) = \frac{1}{\int_0^\infty S_{\mathrm{P}}(0,a)\,\mathrm{d}a}$$

is the frequency-density of patches of age 0. The rate of disturbance for patches of age a is given by $\frac{\partial(1-S_{\mathrm{P}}(0,a))}{\partial a} = -\frac{\partial S_{\mathrm{P}}(0,a)}{\partial a}$, while the expected lifetime of patches is $-\int_0^\infty a \frac{\partial}{\partial a} S_{\mathrm{P}}(0,a)\,\mathrm{d}a = \int_0^\infty S_{\mathrm{P}}(0,a)\,\mathrm{d}a = \frac{1}{P(0)}$ (first step made using integration by parts).

An equilibrium distribution of patch ages may be achieved under a variety of conditions, for example, if $d_{\mathrm{P}}(a,t)$ depends on patch age a but not on time t. The rate of disturbance may also depend on features of the vegetation in the patch, rather than on patch age directly, in which case an equilibrium distribution of patch ages can still arise, provided the vegetation is also assumed to be at equilibrium.

Exponential distribution — If the rate d_{P} of patch disturbance is constant with respect to patch age, then the rates at which patches of age a are disturbed follow an exponential distribution, $-\partial S_{\mathrm{P}}(0,a)/\partial a = d_{\mathrm{P}} \exp(-d_{\mathrm{P}} a)$. The distribution of patch ages is then given by

$$S_{\mathrm{P}}(0,a) = \exp\left(-d_{\mathrm{P}} a\right),\, P(0) = d_{\mathrm{P}}.$$

Weibull distribution — If the rate of patch disturbance changes with patch age according to $d_{\mathrm{P}}(a) = \lambda \psi a^{\psi-1}$, then the rates at which patches of age a are disturbed follow a Weibull distribution, $-\partial S_{\mathrm{P}}(0,a)/\partial a = \lambda \psi a^{\psi-1} e^{-\lambda a^{\psi}}$. $\psi > 1$ implies that the probability of disturbance increases with patch age, while $\psi < 1$ implies that it decreases with patch age. For $\psi = 1$, we obtain the exponential distribution, a special case of the Weibull distribution. The Weibull distribution results in the following distribution of patch ages,

$$S_{\mathrm{P}}(0,a) = \exp(-\lambda a^{\psi}),\, P(0) = \frac{\psi \lambda^{\frac{1}{\psi}}}{\Gamma\left(\frac{1}{\psi}\right)},$$

where $\Gamma(x) = \int_0^\infty e^{-t} t^{x-1}\, dt$ is the gamma function. We can also specify this distribution by the mean disturbance interval $\bar{a} = \frac{1}{P(0)}$. From this, we can calculate the relevant value for $\lambda = \left(\frac{\Gamma\left(\frac{1}{\psi}\right)}{\psi \bar{a}}\right)^{\psi}$.

The default in PLANT is to assume $\psi = 2$, such that d_{P} increases as a linear function of patch age. The distribution of patch ages is then specified by a single parameter, $\bar{a}$.

4.2 *Derivation of* PDE *describing size-structured dynamics*

To model the metacommunity, we use a PDE describing the dynamics for a thin slice ΔH. This PDE can be derived as follows, following De Roos, Taljapurkar & Caswell (1997). Note that in this sub-section the dependencies on the traits x are deliberately not made explicit. Assuming that all rates are constant within the interval ΔH, the total number of individuals within the interval spanned by $[H - 0.5\Delta H, H + 0.5\Delta H)$ is $N(H,a)\Delta H$. The flux of individuals in and out of this size interval can be expressed as

$$\begin{array}{l} g(H-0.5\Delta H,a)\, N(H-0.5\Delta H,a) - g(H+0.5\Delta H,a)\, N(H+0.5\Delta H,a) \\ -d(H,a)\, N(H,a)\Delta H \end{array}. \qquad (23)$$

The first two terms describe the flux in and out of the considered size interval through growth, while the last term describes losses through mortality. The change in the density of individuals within this size interval during a time step Δa is thus

$$\begin{aligned} N(H, a+\Delta a)\Delta H - N(H,a)\Delta H = \; & g(H-0.5\Delta H, a)\, N(H-0.5\Delta H, a)\Delta a \\ & - g(H+0.5\Delta H, a)\, N(H+0.5\Delta H, a)\Delta a \\ & - d(H,a)\, N(H,a)\Delta H \Delta a. \end{aligned} \tag{24}$$

By rearranging, we obtain

$$\begin{aligned} \frac{N(H,a+\Delta a)-N(H,a)}{\Delta a} = \; & -d(H,a)\, N(H,a) \\ & - \frac{g(H+0.5\Delta H,a)\, N(H+0.5\Delta H,a) - g(H-0.5\Delta H,a)\, N(H-0.5\Delta H,a)}{\Delta H}. \end{aligned} \tag{25}$$

The left-hand side above corresponds to the derivative of N as $\Delta a \to 0$. For thin slices, $\Delta H \to 0$, this yields

$$\frac{\partial}{\partial a} N(H,a) = -d(H,a)\, N(H,a) - \frac{\partial}{\partial H}(g(H,a)\, N(H,a)). \tag{26}$$

To complete the model, this PDE must be supplemented with boundary conditions that specify the density at the lower end of heights for all a, as well as the initial distribution $N(H,0)$. The former is derived by integrating the PDE with respect to H over the interval (H_0, H_∞), yielding

$$\frac{\partial}{\partial a} \int_{H_0}^{H_\infty} N(H,a)\, \mathrm{d}H = g(H_0,a)\, N(H_0,a) - g(H_\infty,a)\, N(H_\infty,a) - \int_{H_0}^{H_\infty} d(H,a)\, N(H,a)\, \mathrm{d}H. \tag{27}$$

The left-hand size above is evidently the rate of change of the total density of individuals in the population, while the right-hand side is the population's total death rate. Further, we can assume that $N(H_\infty, a) = 0$. Thus, to balance total births and deaths, $g(H_0,a)\, N(H_0,a)$ must equal the population's total birth rate B, yielding the boundary condition

$$g(H_0,a)\, N(H_0,a) = B. \tag{28}$$

4.3 *Converting density from one size unit to another*

The quantification of size-density depends on the chosen size unit (Eq. 5). Thus, what if we want to express size-density with respect to another size unit? A relation between the two corresponding values of size-density can be derived by noting that the total density of individuals within a given size range must be equal. We consider a situation in which size-density is expressed in units of size M, but we want it in units of size H. First, we require a one-to-one function that translates the first size unit into the second, $H = \hat{H}(M)$. Then the following must hold

$$\int_{M_1}^{M_2} N(M|x,a)\, \mathrm{d}M = \int_{\hat{H}(M_1)}^{\hat{H}(M_2)} N'(H|x,a)\, \mathrm{d}H. \tag{29}$$

For very small size intervals, this equation is equivalent to

$$(M_2 - M_1)\; N(M_1|x,a) = \left(\hat{H}(M_2) - \hat{H}(M_1)\right)\; N'(\hat{H}(M_1)|x,a). \tag{30}$$

Rearranging gives

$$N'(\hat{H}(M_1)|x,a) = N(M_1|x,a)\, \frac{M_2 - M_1}{\hat{H}(M_2) - \hat{H}(M_1)}. \tag{31}$$

Noting that the second term on the right-hand side is simply the definition of $\frac{\mathrm{d}M}{\mathrm{d}H}$ evaluated at M_1, we have

$$N'(H|x,a) = N(M|x,a)\,\frac{\mathrm{d}M}{\mathrm{d}H}. \tag{32}$$

TABLE 1: Variable names and definitions in the demographic model of the PLANT package.

Symbol	Unit	Description
Patch state variables		
K		Number of species in the metacommunity
a, a'	yr	Patch age (time since disturbance)
a_0	yr	Patch age when plant germinates
E_a		Profile of canopy openness within a patch of age a
$E_a(z)$		Canopy openness at height z within a patch of age a
Plant state variables		
x		Vector of traits for a species
H	m	Height of a plant
$H_0(x)$	m	Height of a seedling with traits x after germination
z	m	Height in canopy
$A_l(H)$	m^2	Leaf area of a plant with height H
$Q(z, H)$		Fraction of leaf area above height z for a plant with height H
Abundance measures		
$P(a)$	yr^{-1}	Frequency-density of patches of age a
Y_x	m^{-2} yr^{-1}	Global seed rain for a species with traits x
$N(H\|x, a)$	m^{-1} m^{-2}	Density of plants at height H per unit ground area for given traits x and patch age a
Demographic rates		
$g(x, H, E_a)$	m yr^{-1}	Height growth rate of a plant with traits x and height H in the light environment E_a in a patch of age a
$f(x, H, E_a)$	yr^{-1}	Seed production rate of a plant with traits x and height H in the light environment E_a in a patch of age a
$d(x, H, E_a)$	yr^{-1}	Instantaneous mortality rate of a plant with traits x and height H in the light environment E_a in a patch of age a
$d_P(a)$	yr^{-1}	Instantaneous disturbance rate of a patch of age a
Demographic outcomes		
$H(x, a_0, a)$	m	Height of a plant with traits x in a patch of age a having germinated at patch age a_0
S_D		Probability a seed survives dispersal
$S_G(x, H_0, E_{a0})$		Probability the seed of a plant with traits x germinates successfully at height H_0 in the light environment E_{a0} in a patch of age a_0
$S_I(x, a_0, a)$		Probability a plant survives from patch age a_0 to patch age a
$S_P(a_0, a)$		Probability a patch remains undisturbed from patch age a_0 to patch age a
$\tilde{R}(x, a_0, a)$		Cumulative seed output for plants with traits x from patch age a_0 to patch age a
$R(x', x)$		Basic reproduction ratio for a mutant plant with traits x' growing in a metapopulation of resident plants with traits x
Miscellaneous constants		
k_I		Light extinction coefficient
$\bar{a}$	yr	Mean interval between patch disturbances

REFERENCES

Abramowitz, M. & Stegun, I.A. (2012) *Handbook of mathematical functions: with formulas, graphs, and mathematical tables*. Courier Corporation.

Angulo, O. & López-Marcos, J.C. (2004) Numerical integration of fully nonlinear size-structured population models. *Applied Numerical Mathematics*, **50**, 291–327.

Angulo, O., López-Marcos, J.C. & López-Marcos, M.A. (2014) Analysis of an efficient integrator for a size-structured population model with a dynamical resource. *Computers & Mathematics with Applications*, **68**, 941–961.

Angulo, O., López-Marcos, J.C. & López-Marcos, M.A. (2016) Study on the efficiency in the numerical integration of size-structured population models: error and computational cost. *Journal of Computational and Applied Mathematics*, **291**, 391–401.

Cash, J.R. & Karp, A.H. (1990) A variable order Runge-Kutta method for initial value problems with rapidly varying right-hand sides. *ACM Transactions on Mathematical Software*, **16**, 201–222.

De Roos, A.M. (1988) Numerical methods for structured population models: the Escalator Boxcar Train. *Numerical Methods for Partial Differential Equations*, **4**, 173–195.

De Roos, A.M., Diekmann, O. & Metz, J.A.J. (1992) Studying the dynamics of structured population models: a versatile technique and its application to daphnia. *American Naturalist*, **139**, 123.

De Roos, A.M., Taljapurkar, S. & Caswell, H. (1997) A gentle introduction to physiologically structured population models. *Structured population models in marine, terrestrial and freshwater systems*, pp. 119–204. Chapman & Hall, New York.

Eddelbuettel, D., Emerson, J.W. & Kane, M.J. (2015) *BH: Boost C++ Header Files*. R package version 1.55.0-3.

Falster, D.S., Brännström, Å., Dieckmann, U. & Westoby, M. (2011) Influence of four major plant traits on average height, leaf-area cover, net primary productivity, and biomass density in single-species forests: a theoretical investigation. *Journal of Ecology*, **99**, 148–164.

Falster, D.S., Brännström, Å., Westoby, M. & Dieckmann, U. (2015) Multi-trait eco-evolutionary dynamics explain niche diversity and evolved neutrality in forests. *bioRxiv*, p. 014605.

Foerster, H.V. (1959) Some remarks on changing populations. F. Stohlman, ed., *The kinetics of cellular proliferation*, pp. 382–407. Grune et Stratton, New York.

Galassi, M., ed. (2009) *GNU scientific library: reference manual*. A GNU manual. Network Theory, s.l., 3. ed., for gsl version 1.12 edition.

Geritz, S.A.H., Kisdi, É., Meszéna, G. & Metz, J.A.J. (1998) Evolutionarily singular strategies and the adaptive growth and branching of the evolutionary tree. *Evolutionary Ecology*, **12**, 35–57.

Gyllenberg, M. & Metz, J.A.J. (2001) On fitness in structured metapopulations. *Journal of Mathematical Biology*, **43**, 545–560.

Kohyama, T. (1993) Size-structured tree populations in gap-dynamic forest: the forest architecture hypothesis for the stable coexistence of species. *Journal of Ecology*, **81**, 131–143.

McKendrick, A.G. (1926) Applications of mathematics to medical problems. *Proceedings of the Edinburgh Mathematical Society*, **44**, 1–34.

Metz, J.A.J. & Gyllenberg, M. (2001) How should we define fitness in structured metapopulation models? Including an application to the calculation of evolutionarily stable dispersal strategies. *Proceedings of the Royal Society of London - Series B: Biological Sciences*, **268**, 499–508.

Metz, J.A.J., Nisbet, R.M. & Geritz, S.A.H. (1992) How should we define 'fitness' for general ecological scenarios? *Trends in Ecology and Evolution*, **7**, 198–202.

Moorcroft, P.R., Hurtt, G.C. & Pacala, S.W. (2001) A method for scaling vegetation dynamics: the Ecosystem Demography model (ED). *Ecological Monographs*, **71**, 557–586.

Piessens, R., Doncker-Kapenga, E.D., Überhuber, C. & Kahaner, D. (1983) *QUADPACK, a subroutine package for automatic integration*. Springer, Berlin Heidelberg.

Schäling, B. (2014) *The Boost C++ libraries*. XML Press, 2nd edition.

Stoer, J. & Bulirsch, R. (2002) *Introduction to numerical analysis*. Springer Science & Business Media.

supporting information for:
"plant: a package for modelling forest trait ecology & evolution"

Description of the FF16 physiological model

Daniel S. Falster†, Richard G. FitzJohn, Åke Brännström, Ulf Dieckmann, and Mark Westoby

†Department of Biological Sciences, Macquarie University, Sydney, Australia

daniel.falster@mq.edu.au, rich.fitzjohn@gmail.com

contents

1 introduction

This document outlines the core physiological model used in the plant package. This model has primarily been developed elsewhere, in particular in Falster *et al.* (2011). The model's equations are presented here not as original findings, but rather so that users can understand the full system of equations being solved within plant.

The purpose of a physiological model in PLANT is to take a plant's current size, light environment, and physiological parameters as inputs, and return its growth, mortality, and fecundity rates. In the FF16 physiological model, these vital rates are all derived from the rate at which living biomass is produced by the plant, which in turn is calculated based on well-understood physiology (Fig. 1). Various physiological parameters influence demographic outcomes. Varying these parameters allows accounting for species differences, potentially via traits (see last section). Tables 1, 3, 4 summarize the units and definitions of all variables, parameters, and hyper-parameters used in the material below.

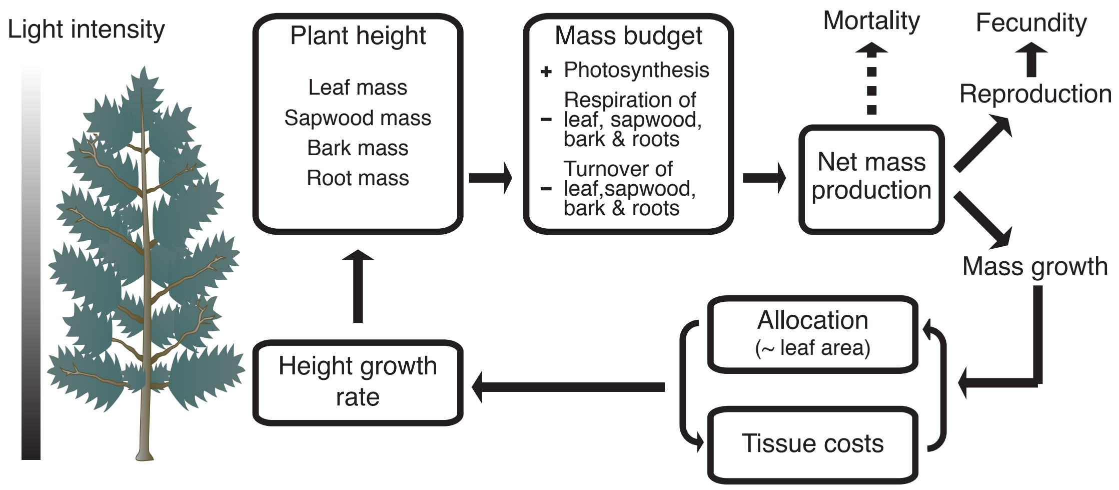


FIGURE 1: Physiological model in PLANT, giving demographic rates on the basis of its traits, size, and light environment, as functions of net mass production. The dashed arrow towards mortality indicates that, although the mortality rate is assumed to depend on mass production, no mass is actually allocated there. Figure adapted from Falster *et al.* (2011) and Falster *et al.* (2015).

2 GROWTH

2.1 *Leaf photosynthesis*

We denote by $p(x, E)$ the gross rate of leaf photosynthesis per unit leaf area within the canopy of a plant with traits x at light level $E(z)$, where z is height within the canopy. We assume a relationship of the form

$$p(x, E(z)) = \frac{\alpha_{\mathrm{p1}}}{E(z) + \alpha_{\mathrm{p2}}}, \tag{1}$$

for the average of p across the year. The parameters α_{p1} and α_{p2} are derived from a detailed leaf-level model and measure, respectively, maximum annual photosynthesis and the light levels at 50% of this maximum. The average rate of leaf photosynthesis across the plant is then

$$\bar{p}(x, H, E_a) = \int_0^H p(x, E_a(z))\, q(z, H)\, \mathrm{d}z, \tag{2}$$

where $q(z, H)$ is the vertical distribution of leaf area with respect to height z (Eq. 13).

2.2 *Mass production*

The amount of biomass available for growth, $\mathrm{d}B/\mathrm{d}t$, is given by the difference between income (total photosynthetic rate) and losses (respiration and turnover) within the plant (Mäkelä, 1997; Thornley & Cannell, 2000; Falster *et al.*, 2011),

$$\underbrace{\frac{\mathrm{d}B}{\mathrm{d}t}}_{\text{net biomass production}} = \underbrace{\alpha_{\mathrm{bio}}}_{\text{mass per C}} \underbrace{\alpha_{\mathrm{y}}}_{\text{yield}} \Big(\underbrace{A_{\mathrm{l}}\, \bar{p}}_{\text{photosynthesis}} - \underbrace{\sum_{i=\mathrm{l,b,s,r}} M_i\, r_i}_{\text{respiration}} \Big) - \underbrace{\sum_{i=\mathrm{l,b,s,r}} M_i\, k_i}_{\text{turnover}} . \tag{3}$$

Here, M, r, and k refer to the mass, maintenance respiration rate, and turnover rate of different tissues, denoted by subscripts l = leaves, b = bark, s = sapwood, and r = roots. $\bar{p}$ is the assimilation rate of $\mathrm{CO_2}$ per unit leaf area, α_{y} is yield (i.e., the fraction of assimilated carbon fixed in biomass, with the remaining fraction being lost as growth respiration; this comes in addition to the costs of maintenance respiration), and α_{bio} is the amount of biomass per unit carbon fixed. Gross photosynthetic production is proportional to leaf area, $A_{\mathrm{l}} = M_{\mathrm{l}}/\phi$, where ϕ is leaf mass per area. The total mass of living tissues is $M_{\mathrm{a}} = M_{\mathrm{l}} + M_{\mathrm{b}} + M_{\mathrm{s}} + M_{\mathrm{r}}$.

2.3 *Height growth*

The key measure of growth required by the demographic model is the rate of height growth, denoted $g(x, H, E_a)$. To model height growth requires that we translate mass production into height increment, accounting for the costs of building new tissues, allocation to reproduction, and architectural layout. Using the chain rule, height growth can be decomposed into a product of physiologically relevant terms (Falster *et al.*, 2011),

$$g(x, H, E_a) = \frac{\mathrm{d}H}{\mathrm{d}t} = \frac{\mathrm{d}H}{\mathrm{d}A_{\mathrm{l}}} \times \frac{\mathrm{d}A_{\mathrm{l}}}{\mathrm{d}M_{\mathrm{a}}} \times \frac{\mathrm{d}M_{\mathrm{a}}}{\mathrm{d}B} \times \frac{\mathrm{d}B}{\mathrm{d}t}. \tag{4}$$

The first factor, $\mathrm{d}H/\mathrm{d}A_{\mathrm{l}}$, is the growth in plant height per unit growth in total leaf area – accounting for the architectural strategy of the plant. Some species tend to leaf out more than grow tall, while other species emphasise vertical extension.

The second factor, $\mathrm{d}A_{\mathrm{l}}/\mathrm{d}M_{\mathrm{a}}$, accounts for the marginal cost of deploying an additional unit of leaf area, including construction of the leaf itself and various support structures. As such, $\mathrm{d}A_{\mathrm{l}}/\mathrm{d}M_{\mathrm{a}}$ can itself be expressed as a sum of construction costs per unit leaf area,

$$\frac{\mathrm{d}A_{\mathrm{l}}}{\mathrm{d}M_{\mathrm{a}}} = \left(\frac{\mathrm{d}M_{\mathrm{l}}}{\mathrm{d}A_{\mathrm{l}}} + \frac{\mathrm{d}M_{\mathrm{s}}}{\mathrm{d}A_{\mathrm{l}}} + \frac{\mathrm{d}M_{\mathrm{b}}}{\mathrm{d}A_{\mathrm{l}}} + \frac{\mathrm{d}M_{\mathrm{r}}}{\mathrm{d}A_{\mathrm{l}}} \right)^{-1}. \tag{5}$$

The third factor, $\mathrm{d}M_{\mathrm{a}}/\mathrm{d}B$, is the fraction of net biomass production (Eq. 3) that is allocated to growth rather than reproduction or storage. In the `FF16` physiological model, we let this growth fraction decrease with height according to the function

$$\frac{\mathrm{d}M_{\mathrm{a}}}{\mathrm{d}B}(H) = 1 - \frac{\alpha_{\mathrm{f1}}}{1 + \exp\left(\alpha_{\mathrm{f2}}\left(1 - H/H_{\mathrm{mat}}\right)\right)}, \tag{6}$$

where α_{f1} is the maximum possible allocation $(0 - 1)$ and α_{f2} determines the sharpness of the transition (Falster *et al.*, 2011).

2.4 *Diameter growth*

Analogously, the growth in basal area A_{st} can be expressed as the sum of growth in sapwood, bark, and heartwood areas (A_{s}, A_{b}, and A_{h}, respectively),

$$\frac{\mathrm{d}A_{\mathrm{st}}}{\mathrm{d}t} = \frac{\mathrm{d}A_{\mathrm{b}}}{\mathrm{d}t} + \frac{\mathrm{d}A_{\mathrm{s}}}{\mathrm{d}t} + \frac{\mathrm{d}A_{\mathrm{h}}}{\mathrm{d}t}.$$

Applying the chain rule, we derive an equation for basal area growth that contains many of the same elements as Eq. 4,

$$\frac{\mathrm{d}A_{\mathrm{st}}}{\mathrm{d}t} = \left(\frac{\mathrm{d}A_{\mathrm{s}}}{\mathrm{d}A_{\mathrm{l}}} + \frac{\mathrm{d}A_{\mathrm{b}}}{\mathrm{d}A_{\mathrm{l}}}\right) \times \frac{\mathrm{d}A_{\mathrm{l}}}{\mathrm{d}M_{\mathrm{a}}} \times \frac{\mathrm{d}M_{\mathrm{a}}}{\mathrm{d}B} \times \frac{\mathrm{d}B}{\mathrm{d}t} + \frac{\mathrm{d}A_{\mathrm{h}}}{\mathrm{d}t}. \tag{7}$$

Diameter growth is then given by the geometric relationship between stem diameter D and A_{st},

$$\frac{\mathrm{d}D}{\mathrm{d}t} = (\pi A_{\mathrm{st}})^{-0.5} \frac{\mathrm{d}A_{\mathrm{st}}}{\mathrm{d}t}. \tag{8}$$

3 FUNCTIONAL-BALANCE MODEL FOR ALLOCATION

Here we describe an allometric model linking to a plant's height its various other size dimensions required by most ecologically realistic vegetation models (i.e., the masses of leaves, sapwood, bark, and fine roots). This approach allows us to track only the plant's height, while still accounting for the mass needs to build leaves, roots, and stems. The growth rates of various tissues can then also be derived (Table 2).

3.1 *Leaf area*

Based on empirically observed allometry (Falster *et al.*, 2011), we assume an allometric power-law scaling relationship between the accumulated leaf area of a plant and its height,

$$H = \alpha_{\mathrm{l1}} \left(A_{\mathrm{l}}\,\mathrm{m}^{-2}\right)^{\alpha_{\mathrm{l2}}}. \tag{9}$$

This relationship is normalised around a leaf area of 1m^2.

3.2 *Vertical distribution of leaf area*

We follow the model of Yokozawa & Hara (1995) describing the vertical distribution of leaf area within the crowns of individual plants. This model can account for a variety of canopy profiles through a single parameter η. Setting $\eta = 1$ results in a conical canopy, as seen in many conifers, while higher values, e.g., $\eta = 12$, give a top-heavy canopy profile similar to those seen among angiosperms. We denote by $A_{\mathrm{s},z}$ the sapwood area at height z within the plant, by $q(z, H)$ the vertical distribution of leaf area of leaf area with respect to height z, and by $Q(z, H)$ the cumulative fraction of a plant's leaves above height z. As defined previously, A_{s} is the sapwood area at the base of the plant. Following Yokozawa & Hara (1995), we assume a relationship between $A_{\mathrm{s},z}$ and height such that

$$\frac{A_{\mathrm{s},z}}{A_{\mathrm{s}}} = \left(1 - \left(\frac{z}{H}\right)^{\eta}\right)^{2}. \tag{10}$$

We also assume that each unit of leaf area is supported by a fixed area θ of sapwood (in agreement with the pipe model; Shinozaki *et al.*, 1964), so that the total canopy area of a plant relates to its basal sapwood area A_s,

$$A_s = \theta A_l. \tag{11}$$

The pipe model is assumed to hold within individual plants, as well as across plants of different size. It follows that

$$Q(z, H) = \left(1 - \left(\frac{z}{H}\right)^\eta\right)^2. \tag{12}$$

Differentiating with respect to z then yields a solution for the probability density of leaf area as a function of height,

$$q(z, H) = 2\frac{\eta}{H}\left(1 - \left(\frac{z}{H}\right)^\eta\right)\left(\frac{z}{H}\right)^{\eta-1}. \tag{13}$$

3.3 *Sapwood mass*

Integrating $A_{s,z}$ yields the total mass of sapwood in a plant,

$$M_s = \rho \int_0^H A_{s,z}\,dz = \rho A_s H \eta_c, \tag{14}$$

where $\eta_c = 1 - \frac{2}{1+\eta} + \frac{1}{1+2\eta}$ (Yokozawa & Hara, 1995). Substituting from Eq. 11 into Eq. 14 then gives an expression for sapwood mass as a function of leaf area and height,

$$M_s = \rho\,\eta_c\,\theta\,A_l\,H. \tag{15}$$

3.4 *Bark mass*

Bark and phloem tissue are modelled using an analogue of the pipe model, leading to a similar equation as that for sapwood mass (Eq. 15). The cross-section area of bark per unit leaf area is assumed to be a constant fraction α_{b1} of sapwood area per unit leaf area such that

$$M_b = \alpha_{b1} M_s. \tag{16}$$

3.5 *Root mass*

Also consistent with the pipe model, we assume a fixed ratio of root mass per unit leaf area,

$$M_r = \alpha_{r1}\,A_l. \tag{17}$$

Even though nitrogen and water uptake are not modelled explicitly, imposing a fixed ratio of root mass to leaf area ensures that approximate costs of root production are included in calculations of carbon budget.

4 SEED PRODUCTION

The rate of seed production, $f(x, H, E_a)$, is a direct function of the mass allocated to reproduction,

$$f(x, H, E_a) = \frac{(1 - \frac{dM_a}{dB}) \times \frac{dB}{dt}}{\omega + \alpha_{f3}}, \tag{18}$$

where ω is the mass of the seed and α_{f3} is the cost per seed of accessories, such as fruits, flowers, and dispersal structures. The function $\frac{dM_a}{dB}$ is the fraction of $\frac{dB}{dt}$ that is allocated to growth (from Eq. 6, while $1 - \frac{dM_a}{dB}$ gives the fraction allocated to reproduction).

5 MORTALITY

Instantaneous rates of plant mortality are given by the sum of a growth-independent and a growth-dependent rate (Falster *et al.*, 2011; Moorcroft, Hurtt & Pacala, 2001),

$$d(x, H, E_a) = d_{\mathrm{I}}(x, H) + d_{\mathrm{G}}(x, H, E_a). \tag{19}$$

The growth-independent rate is taken to be constant, independent of plant performance, but potentially varying with species traits. The growth-dependent rate is assumed to decline exponentially with the rate of mass production per unit leaf area,

$$d_{\mathrm{G}}(x, H, E_a) = \alpha_{\mathrm{dG1}} \exp(-\alpha_{\mathrm{dG2}} X), \tag{20}$$

where $X = \mathrm{d}B/\mathrm{d}t/A_{\mathrm{l}}$. This relationship allows for plants to increase in mortality as their growth rate approaches zero, while allowing for species to differ in the parameters α_{dG1} and α_{dG2}.

We also require a function $S_{\mathrm{G}}(x', H_0, E_{\mathrm{a0}})$ for plant survival through germination. For the demographic model to behave smoothly, $S_{\mathrm{G}}(x', H_0, E_{\mathrm{a0}})/g(x, H_0, E_{\mathrm{a0}})$ should approach zero as $g(x, H_0, E_{\mathrm{a0}})$ approaches zero. Following Falster *et al.* (2011), we use the function

$$S_{\mathrm{G}}(x', H_0, E_{\mathrm{a0}}) = \frac{1}{1 + X^2}, \tag{21}$$

where $X = \alpha_{\mathrm{d0}} \frac{A_{\mathrm{l}}}{\mathrm{d}B/\mathrm{d}t}$ and α_{d0} is a constant. Eq. 21 is consistent with Eq. 20, as both cause survival to decline with mass production.

6 HYPER-PARAMETERISATION OF PHYSIOLOGICAL MODEL VIA TRAITS

The FF16 physiological model includes default values for all needed parameters (Table 3). Species are known to vary considerably in many of these parameters, such as ϕ, ρ, ν, and ω; so by varying parameters one can account for species differences. When altering a parameter in the model, however, one must also consider whether there are trade-offs linking parameters.

PLANT allows for the hyper-parameterisation of the FF16 physiological model via plant functional traits: this enables simultaneous variation in multiple parameters in accordance with an assumed trade-off. In the FF16 physiological model, we implement the relationships described below. For more details, see `make_FF16_hyperpar.R`.

6.1 *Leaf mass per unit area*

The trait leaf mass per unit area, denoted by ϕ, directly influences growth by changing $\mathrm{d}A_{\mathrm{l}}/\mathrm{d}M_{\mathrm{a}}$. In addition, we link ϕ to the rate of leaf turnover, based on a widely observed scaling relationship from Wright *et al.* (2004),

$$k_{\mathrm{l}} = \beta_{\mathrm{kl1}} \left(\frac{\phi}{\phi_0}\right)^{-\beta_{\mathrm{kl2}}}.$$

This relationship is normalised around ϕ_0, the global mean of ϕ. This allows us to vary β_{kl1} and β_{kl2} without displacing the relationship from the observed mean.

We also vary the mass-based leaf respiration rate so that it stays constant per unit leaf area and varies with ϕ and nitrogen per unit leaf area ν, as empirically observed by Wright *et al.*

(2004),

$$r_\mathrm{l} = \frac{\beta_\mathrm{lf4}\,\nu}{\phi}.$$

6.2 *Wood density*

The trait wood density, denoted by ρ, directly influences growth by changing $\mathrm{d}A_\mathrm{l}/\mathrm{d}M_\mathrm{a}$. In addition, we allow for ρ to influence the rate of growth-independent mortality,

$$d_\mathrm{I} = \beta_\mathrm{dI1}\left(\frac{\rho}{\rho_0}\right)^{-\beta_\mathrm{dI2}},$$

and also the rate of sapwood turnover,

$$k_\mathrm{s} = \beta_\mathrm{ks1}\left(\frac{\rho}{\rho_0}\right)^{-\beta_\mathrm{ks2}},$$

As for ϕ, these relationships are normalized around ρ_0, the global mean of ρ. By default, β_kI2 and β_ks2 are set to zero, so these linkages only become present when these parameters are set to something other than their default values.

The rate of sapwood respiration per unit volume is assumed to be constant, so sapwood respiration per unit mass varies as

$$r_\mathrm{s} = \frac{\beta_\mathrm{rs1}}{\rho},$$

where β_rs1 is a default rate per volume of sapwood. Similarly, the rate of bark respiration per unit mass varies as

$$r_\mathrm{b} = \frac{\beta_\mathrm{rb1}}{\rho},$$

with $\beta_\mathrm{rb1} = 2\beta_\mathrm{rs1}$.

6.3 *Seed mass*

Effects of the trait seed mass, denoted by ω, are naturally embedded in the equation determining fecundity (Eq. 18) and the initial height of seedlings. In addition, we let the accessory cost per seed be a multiple of seed size,

$$\alpha_\mathrm{f3} = \beta_\mathrm{f1}\omega,$$

as empirically observed (Henery & Westoby, 2001).

6.4 *Nitrogen per unit leaf area*

Photosynthesis per unit leaf area and respiration rates per unit leaf mass (or area) are assumed to vary with leaf nitrogen per unit area, ν. The calculation of respiration rates is already described above. To calculate the average annual photosynthesis for a leaf, we integrate the instantaneous rate per unit leaf area over the annual solar trajectory, using a rectangular-hyperbolic photosynthesis light response curve,

$$p(\nu, E) = \frac{1}{365\mathrm{d}}\int_0^{365\mathrm{d}} \frac{Y(t) + A_\mathrm{max} - \sqrt{(Y(t) + A_\mathrm{max})^2 - 4\beta_\mathrm{lf2}Y(t)A_\mathrm{max}}}{2\beta_\mathrm{lf2}}\mathrm{d}t,$$

where

- A_{max} is the maximum photosynthetic capacity of the leaf,
- β_{lf2} is the curvature of the light response curve,
- $Y(t) = \beta_{\text{lf3}} I(t)$ is the initial yield of the light response curve, with β_{lf3} being the quantum yield parameter,
- $I(t) = k_{\text{I}}\, I_0(t)\, E$ is the intensity of light on the leaf surface, and
- $I_0(t)$ is light incident on a surface perpendicular to the sun's rays directly above the canopy at time t.

The profile of $I_0(t)$ is given by a solar model adapted from Ter Steege (1997).

We allow for the maximum photosynthetic capacity of the leaf to vary with leaf nitrogen per unit area, as

$$A_{\text{max}} = \beta_{\text{lf1}} \left(\frac{\nu}{\nu_0}\right)^{\beta_{\text{lf5}}}, \tag{22}$$

where β_{lf1} and β_{lf5} are constants. The relationships is normalized around ν_0, the global mean of leaf nitrogen per unit area.

Values of $p(\nu, E)$ are calculated across a range of values of E, and then an expression of the form in Eq. 1 is fitted to extract the parameters α_{p1} and α_{p2}, such that these become functions of ν.

7 TABLES

TABLE 1: Key variables of the FF16 physiological model. For mass (M), respiration (r), and turnover (k) variables, subscripts refer to any of the following tissues: l = leaves, b = bark, s = sapwood, r = roots, a = all living tissue. For area A variables, subscripts refer to any of the following: l = leaves, st = total stem cross-section, s = sapwood cross-section, b = bark cross-section, h = heartwood cross-section.

Symbol	Unit	Description
Plant construction		
x		Vector of traits for a species
H_0	m	Height of a seedling after germination
H	m	Height of a plant
B	kg	Biomass originating from parent plant
M_i	kg	Mass of tissue type i retained on plant
A_i	m^2	Surface area or area of cross-section of tissue type i
$q(z, H)$	m^{-1}	Vertical distribution of leaf area across heights z for a plant with height H
$Q(z, H)$		Fraction of leaf area above height z for a plant with height H
Mass production		
$p, \bar{p}$	mol yr^{-1} m^{-2}	Photosynthetic rate per unit area
r_i	mol yr^{-1} kg^{-1}	Respiration rate per unit mass of tissue type i
k_i	yr^{-1}	Turnover rate for tissue type i
Environment		
a	yr	Patch age
E_a		Profile of canopy openness within a patch of age a
$E_a(z)$		Canopy openness at height z within a patch of age a
Demographic outcomes		
$g(x, H, E_a)$	m yr^{-1}	Height growth rate of a plant with traits x and height H in the light environment E_a in a patch of age a
$f(x, H, E_a)$	yr^{-1}	Seed production rate of a plant with traits x and height H in the light environment E_a in a patch of age a
$d(x, H, E_a)$	yr^{-1}	Instantaneous mortality rate of a plant with traits x and height H in the light environment E_a in a patch of age a
$S_G(x, H_0, E_{a0})$		Probability that a seed germinates successfully

TABLE 2: Equations of the allometric growth model implemented within the `FF16` physiological model, based on functional-balance assumptions (see text for details). The key assumptions allocation model are listed in (a), under "Function". From these assumptions, allocation functions are derived for tissue areas and tissue masses (b). Also the growth rate of each tissue type can be expressed as a function of the growth rate of leaf area. For mass (M), respiration (r), and turnover (k) variables, subscripts refer to any of the following tissues: l = leaves, b = bark, s = sapwood, r = roots, a = all living tissue. For area A variables, subscripts refer to any of the following: l = leaves, st = total stem cross-section, s = sapwood cross-section, b = bark cross-section, h = heartwood cross-section.

Variable	Function	Allocation	Growth rate
(a) Assumed relationships with leaf area			
Height	$H = \alpha_{l1}\ \left(A_l\,\mathrm{m}^{-2}\right)^{\alpha_{l2}}$	$\frac{\mathrm{d}H}{\mathrm{d}A_l} = \alpha_{l2}\alpha_{l1}\ \left(A_l\,\mathrm{m}^{-2}\right)^{\alpha_{l2}-1}$	$\frac{\mathrm{d}H}{\mathrm{d}t} = \frac{\mathrm{d}H}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$
Sapwoood area	$A_s = \theta\, A_l$	$\frac{\mathrm{d}A_s}{\mathrm{d}A_l} = \theta$	$\frac{\mathrm{d}A_s}{\mathrm{d}t} = \frac{\mathrm{d}A_s}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$
Bark area	$A_b = \alpha_{b1}\,\theta\, A_l$	$\frac{\mathrm{d}A_b}{\mathrm{d}A_l} = \alpha_{b1}\,\theta$	$\frac{\mathrm{d}A_b}{\mathrm{d}t} = \frac{\mathrm{d}A_b}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$
(b) Derived equations for mass of tissue			
Leaf mass	$M_l = \phi\, A_l$	$\frac{\mathrm{d}M_l}{\mathrm{d}A_l} = \phi$	$\frac{\mathrm{d}M_l}{\mathrm{d}t} = \frac{\mathrm{d}M_l}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$
Sapwood mass	$M_s = \rho\,\theta\,\eta_c\, A_l\, H$	$\frac{\mathrm{d}M_s}{\mathrm{d}A_l} = \rho\,\theta\,\eta_c\left(H + A_l\,\frac{\mathrm{d}H}{\mathrm{d}A_l}\right)$	$\frac{\mathrm{d}M_s}{\mathrm{d}t} = \frac{\mathrm{d}M_s}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$
Bark mass	$M_b = \alpha_{b1}\,\rho\,\theta\,\eta_c\, A_l\, H$	$\frac{\mathrm{d}M_b}{\mathrm{d}A_l} = \alpha_{b1}\,\rho\,\theta\,\eta_c\left(H + A_l\,\frac{\mathrm{d}H}{\mathrm{d}A_l}\right)$	$\frac{\mathrm{d}M_b}{\mathrm{d}t} = \frac{\mathrm{d}M_b}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$
Root mass	$M_r = \alpha_{r1}\, A_l$	$\frac{\mathrm{d}M_r}{\mathrm{d}A_l} = \alpha_{r1}$	$\frac{\mathrm{d}M_r}{\mathrm{d}t} = \frac{\mathrm{d}M_r}{\mathrm{d}A_l}\ \frac{\mathrm{d}A_l}{\mathrm{d}t}$

TABLE 3: Core parameters of the FF16 physiological model.

Description	Symbol	Unit	Code	Value
Plant construction				
Crown-shape parameter	η		eta	12
Leaf mass per area	ϕ	$\mathrm{kg\,m^{-2}}$	lma	0.1978791
Wood density	ρ	$\mathrm{kg\,m^{-3}}$	rho	608
Sapwood area per unit leaf area	θ		theta	0.0002141786
Height of plant with leaf area of 1m^2	α_{l1}	m	a_l1	5.44
Exponent of relationship between height and leaf area	α_{l2}		a_l2	0.306
Root mass per unit leaf area	α_{r1}	$\mathrm{kg\,m^{-2}}$	a_r1	0.07
Ratio of bark area to sapwood area	α_{b1}		a_b1	0.17
Production				
Leaf photosynthesis per area	α_{p1}	$\mathrm{mol\,yr^{-1}\,m^{-2}}$	a_p1	151.1778
Saturation of leaf photosynthesis per area	α_{p2}		a_p2	0.2047162
Yield = fraction of carbon fixed converted into mass	α_{y}		a_y	0.7
Biomass per mol carbon	α_{bio}	$\mathrm{kg\,mol^{-1}}$	a_bio	0.0245
Leaf respiration per mass	r_{l}	$\mathrm{mol\,yr^{-1}\,kg^{-1}}$	r_l	198.4545
Fine-root respiration per mass	r_{r}	$\mathrm{mol\,yr^{-1}\,kg^{-1}}$	r_r	217
Sapwood respiration per mass	r_{s}	$\mathrm{mol\,yr^{-1}\,kg^{-1}}$	r_s	6.598684
Bark respiration per mass	r_{b}	$\mathrm{mol\,yr^{-1}\,kg^{-1}}$	r_b	13.19737
Turnover rate for leaves	k_{l}	$\mathrm{yr^{-1}}$	k_l	0.4565855
Turnover rate for sapwood	k_{s}	$\mathrm{yr^{-1}}$	k_s	0.2
Turnover rate for bark	k_{b}	$\mathrm{yr^{-1}}$	k_b	0.2
Turnover rate for fine roots	k_{r}	$\mathrm{yr^{-1}}$	k_r	1
Fecundity				
Seed mass	ω	kg	omega	0.000038
Height at maturation	H_{mat}	m	hmat	16.59587
Maximum allocation to reproduction	α_{f1}		a_f1	1
Parameter determining rate of change in $r(x, m_{\mathrm{l}})$ around H_{mat}	α_{f2}		a_f2	50
Accessory cost per seed	α_{f3}	kg	a_f3	0.000114
Mortality				
Survival probability during dispersal	S_{D}		S_D	0.25
Parameter influencing survival through germination	α_{d0}	$\mathrm{kg\,yr^{-1}\,m^{-2}}$	a_d0	0.1
Intrinsic or growth-independent mortality	d_{I}	$\mathrm{yr^{-1}}$	d_I	0.01
Baseline rate for growth-dependent mortality	α_{dG1}	$\mathrm{yr^{-1}}$	a_dG1	5.5
Risk coefficient for dry-mass production per unit leaf area in growth-dependent mortality	α_{dG2}	$\mathrm{yr\,m^{2}\,kg^{-1}}$	a_dG2	20

TABLE 4: Parameters for hyper-parameterisation of the FF16 physiological model.

Description	Symbol	Unit	Code	Value
Leaf turnover				
Global average leaf mass per area	ϕ_0	$\mathrm{kg\,m^{-2}}$	`lma_0`	0.1978791
Rate of leaf turnover at average leaf mass per unit leaf area, ϕ_0	β_{kl1}	$\mathrm{yr^{-1}}$	`B_kl1`	0.4565855
Scaling exponent for ϕ in leaf turnover	β_{kl2}		`B_kl2`	1.71
Sapwood turnover				
Global average wood density	ρ_0	$\mathrm{kg\,m^{-3}}$	`rho_0`	608
Rate of sapwood turnover at average wood density, ρ_0	β_{ks1}	$\mathrm{yr^{-1}}$	`B_ks1`	0.2
Scaling exponent for ρ in sapwood turnover	β_{ks2}		`B_ks2`	0
Growth-independent mortality				
Rate of instantaneous mortality at average wood density, ρ_0	β_{dI1}	$\mathrm{yr^{-1}}$	`B_dI1`	0.01
Scaling exponent for wood density in intrinsic mortality	β_{dI2}		`B_dI2`	0
Photosynthesis				
Leaf nitrogen per unit leaf area	ν	$\mathrm{kg\,m^{-2}}$	`narea`	0.00187
Global average nitrogen per unit leaf area	ν_0	$\mathrm{kg\,m^{-2}}$	`narea_0`	0.00187
Potential CO_2 photosynthesis at average leaf nitrogen, ν_0	β_{lf1}	$\mathrm{mol\,d^{-1}\,m^{-2}}$	`B_lf1`	0.8273474
Curvature of light response curve	β_{lf2}		`B_lf2`	0.5
Quantum yield of leaf photosynthesis (CO_2 per unit photosynthetically active radiation)	β_{lf3}		`B_lf3`	0.04
Scaling exponent for leaf nitrogen in maximum leaf photosynthesis	β_{lf5}		`B_lf5`	1
Respiration				
CO_2 respiration per unit leaf nitrogen	β_{lf4}	$\mathrm{mol\,yr^{-1}\,kg^{-1}}$	`B_lf4`	21000
CO_2 respiration per unit sapwood volume	β_{rs1}	$\mathrm{mol\,yr^{-1}\,m^{-3}}$	`B_rs1`	4012
CO_2respiration per unit bark volume	β_{rb1}	$\mathrm{mol\,yr^{-1}\,m^{-3}}$	`B_rb1`	8024
Reproduction				
Cost of seed accessories per unit seed mass	β_{f1}		`B_f1`	3

REFERENCES

Falster, D.S., Brännström, Å., Dieckmann, U. & Westoby, M. (2011) Influence of four major plant traits on average height, leaf-area cover, net primary productivity, and biomass density in single-species forests: a theoretical investigation. *Journal of Ecology*, **99**, 148–164.

Falster, D.S., Brännström, Å., Westoby, M. & Dieckmann, U. (2015) Multi-trait eco-evolutionary dynamics explain niche diversity and evolved neutrality in forests. *bioRxiv*, p. 014605.

Henery, M. & Westoby, M. (2001) Seed mass and seed nutrient content as predictors of seed output variation between species. *Oikos*, **92**, 479–490.

Mäkelä, A. (1997) A carbon balance model of growth and self-pruning in trees based on structural relationships. *Forest Science*, **43**, 7–24.

Moorcroft, P.R., Hurtt, G.C. & Pacala, S.W. (2001) A method for scaling vegetation dynamics: the Ecosystem Demography model (ED). *Ecological Monographs*, **71**, 557–586.

Shinozaki, K., Yoda, K., Hozumi, K. & Kira, T. (1964) A quantitative analysis of plant form - the pipe model theory. I. Basic analyses. *Japanese Journal of Ecology*, **14**, 97–105.

Ter Steege, H. (1997) *Winphot 5: a programme to analyze vegetation indices, light and light quality from hemispherical photographs*. Number 97-3 in Tropenbos Guyana Reports. Tropenbos Guyana Programme, Georgetown, Guyana.

Thornley, J.H.M. & Cannell, M.G.R. (2000) Modelling the components of plant respiration: representation and realism. *Annals of Botany*, **85**, 55–67.

Wright, I.J., Reich, P.B., Westoby, M., Ackerly, D., Baruch, Z., Bongers, F., Cavender-Bares, J., Chapin, F., Cornelissen, J., Diemer, M., Flexas, J., Garnier, E., Groom, P., Gulias, J., Hikosaka, K., Lamont, B., Lee, T., Lee, W., Lusk, C., Midgley, J., Navas, M.L., Niinemets, Ü., Oleksyn, J., Osada, N., Poorter, H., Poot, P., Prior, L., Pyankov, V., Roumet, C., Thomas, S., Tjoelker, M., Veneklaas, E. & Villar, R. (2004) The world-wide leaf economics spectrum. *Nature*, **428**, 821–827.

Yokozawa, M. & Hara, T. (1995) Foliage profile, size structure and stem diameter plant height relationship in crowded plant-populations. *Annals of Botany*, **76**, 271–285.

supporting information for:
"plant: a package for modelling forest trait ecology & evolution"

Worked examples showing how to interact with plant from R

Daniel S. Falster†, Richard G. FitzJohn, Åke Brännström, Ulf Dieckmann, and Mark Westoby

†Department of Biological Sciences, Macquarie University, Sydney, Australia

daniel.falster@mq.edu.au, rich.fitzjohn@gmail.com

CONTENTS

1 INTRODUCTION

This document provides a number of worked examples that replicate the figures in the paper. The material in this section has been developed and edited by the first two authors. The code is not identical to the version used to generate the paper because through this document we try for a bit more exposition at the cost of more global variables, etc.

1.1 *Design*

The `plant` package is designed around a set of nested components.

- At the highest level there is a metapopulation, represented by the `SCM` object. (`SCM` denotes “Solver Characteristic Method”.)
- The `SCM` object contains a single `Patch` object, but we use a technique described in the “demography” document to demonstrate how we can treat this patch as an infinite number of patches.
- The `Patch` object contains one or more `Species` objects.
- Each `Species` contains a number of of `Cohort` objects, each of which contains a single `Plant` object representing the first plant born in that cohort.
- Each `Plant` object has a `Strategy` which contains all the physiological rules around growing a plant. This `Strategy` is shared across all members of a `Species`.

The document here works through these components in the reverse order, starting with `Plant` (and really `Strategy`) objects, and finishing by showing how fitness emerges from the `SCM`. The physiological model is described in detail in a separate “physiology” document, while the details about demographic calculations are described in the “demography” document.

2 PLANT-LEVEL PROPERTIES

This vignette illustrates the sort of analysis used to generate the plant figure (Fig 2) in the manuscript, i.e. modelling the dynamics of individual plants. It also shows some of the features of working with plant components of the model.

```
library(plant)
```

Plants are constructed with the `FF16_Plant` function. That function takes as its only argument a "strategy" object; the default is `FF16_Strategy`, but alternative strategies can be provided (see below). The "strategy" object contains all the physiological underpinning the dynamics of individual plants and entire metapopulations.

```
pl <- FF16_Plant()
```

Plants are an R6 class, and have a number of elements and fields that can be accessed:

```
pl
```

```
## <Plant<FF16>>
##   Inherits from: <Plant>
##   Public:
##     .ptr: externalptr
##     area_leaf_above: function (h)
##     clone: function (deep = FALSE)
##     compute_vars_phys: function (environment)
##     fecundity: active binding
##     germination_probability: function (environment)
##     height: active binding
##     initialize: function (ptr)
##     internals: active binding
##     mortality: active binding
##     mortality_probability: active binding
##     ode_names: active binding
##     ode_rates: active binding
##     ode_size: active binding
##     ode_state: active binding
##     reset_mortality: function ()
##     strategy: active binding
```

Things labelled 'active binding' are "fields" and can be read from and (sometimes) set:

```
pl$height
```

```
## [1] 0.3441948
```

```
pl$height <- 10
pl$height
```

```
## [1] 10
```

```
pl$fecundity
```

```
## [1] 0
```

```
pl$mortality
```

```
## [1] 0
```

Height, fecundity and mortality are the three key variables propagated by the internal system of differential equations:

```
pl$ode_state
```

```
## [1] 10  0  0  0  0
```

To compute rates of change for these variables we need a light environment. The function `fixed_environment` creates an environment that has the same canopy openness (here 100%) at all heights. The plant *does not affect* this light environment.

```
env <- fixed_environment(1.0)
```

The `compute_vars_phys` method computes net mass production for the plant, and from that demographic rates:

```
pl$ode_rates
```

```
## [1] NA NA NA NA NA
```

```
pl$compute_vars_phys(env)
pl$ode_rates
```

```
## [1] 9.231625e-01 1.002618e-02 6.905594e-05 3.132290e-04 1.687620e+00
```

Some internals from the calculations are available in the `internals` field:

```
names(pl$internals)
```

```
##  [1] "area_leaf"         "height"            "height_dt"
##  [4] "mortality"         "mortality_dt"      "fecundity"
##  [7] "fecundity_dt"      "area_heartwood"    "area_heartwood_dt"
## [10] "mass_heartwood"    "mass_heartwood_dt"
```

Height growth rate

```
pl$internals$height_dt
```

```
## [1] 0.9231625
```

Leaf area (m^2)

```
pl$internals$area_leaf
```

```
## [1] 7.312332
```

There's not actually that much that can be done with Plant objects; they're designed to be small and light to work well with the larger simulation code that does not particularly care about most of the internal calculations.

Because of this, we have a "PlantPlus" object that exposes more of a strategy, and in addition stem diameter growth:

```
pp <- FF16_PlantPlus(pl$strategy)
pp$compute_vars_phys(env)
names(pp$internals)
```

```
##  [1] "mass_leaf"                          "area_leaf"
##  [3] "height"                             "area_sapwood"
##  [5] "mass_sapwood"                       "area_bark"
##  [7] "mass_bark"                          "area_heartwood"
##  [9] "mass_heartwood"                     "area_stem"
## [11] "mass_root"                          "mass_live"
## [13] "mass_total"                         "mass_above_ground"
## [15] "diameter_stem"                      "assimilation"
## [17] "respiration"                        "turnover"
## [19] "net_mass_production_dt"             "fraction_allocation_reproduction"
## [21] "fraction_allocation_growth"         "fecundity_dt"
## [23] "area_leaf_dt"                       "darea_leaf_dmass_live"
## [25] "height_dt"                          "area_heartwood_dt"
## [27] "mass_heartwood_dt"                  "mortality_dt"
## [29] "mortality"                          "fecundity"
## [31] "dheight_darea_leaf"                 "dmass_sapwood_darea_leaf"
## [33] "dmass_bark_darea_leaf"              "dmass_root_darea_leaf"
## [35] "area_sapwood_dt"                    "area_bark_dt"
## [37] "area_stem_dt"                       "ddiameter_stem_darea_stem"
## [39] "diameter_stem_dt"                   "mass_root_dt"
## [41] "mass_live_dt"                       "mass_total_dt"
## [43] "mass_above_ground_dt"
```

Some of the internals require `compute_vars_internals` to be run (the zapsmall function rounds numbers close to zero to zero):

```
pp$compute_vars_growth()
zapsmall(unlist(pp$internals))
```

```
##                         mass_leaf                         area_leaf
##                             0.000                             0.000
##                            height                      area_sapwood
##                             0.344                             0.000
##                      mass_sapwood                         area_bark
##                             0.000                             0.000
##                         mass_bark                    area_heartwood
##                             0.000                             0.000
##                    mass_heartwood                         area_stem
##                             0.000                             0.000
##                         mass_root                         mass_live
##                             0.000                             0.000
##                        mass_total                mass_above_ground
##                             0.000                            0.000
##                     diameter_stem                     assimilation
##                             0.000                            0.015
##                       respiration                         turnover
##                             0.007                            0.000
##            net_mass_production_dt fraction_allocation_reproduction
##                             0.000                            0.000
##        fraction_allocation_growth                      fecundity_dt
##                             1.000                            0.000
##                      area_leaf_dt            darea_leaf_dmass_live
##                             0.000                            3.043
##                         height_dt                area_heartwood_dt
##                             0.334                            0.000
##                 mass_heartwood_dt                     mortality_dt
##                             0.000                            0.010
##                         mortality                        fecundity
##                             0.000                            0.000
##                dheight_darea_leaf         dmass_sapwood_darea_leaf
##                           871.275                            0.052
##             dmass_bark_darea_leaf            dmass_root_darea_leaf
##                             0.009                            0.070
##                   area_sapwood_dt                     area_bark_dt
##                             0.000                            0.000
##                      area_stem_dt        ddiameter_stem_darea_stem
##                             0.000                         3241.596
##                  diameter_stem_dt                     mass_root_dt
##                             0.000                            0.000
##                      mass_live_dt                    mass_total_dt
##                             0.000                            0.000
##              mass_above_ground_dt
##                             0.000
```

This PlantPlus object also includes heartwood area and mass as two more variables for the ODE system (this might move into Plant soon – see this issue). Tracking of these variables is needed to estimate stem diameter growth

```
pp$ode_names
```

```
## [1] "height"          "mortality"       "fecundity"       "area_heartwood"
## [5] "mass_heartwood"
```

Plants are a type of *reference object*. They are different to almost every other R object you regularly interact with in that they do not make copies when you rename them. So changes to one will be reflected in another.

```
pl2 <- pl
pl2$height <- 1
pl2$height
```

```
## [1] 1
```

```
pl$height # also 1!
```

```
## [1] 1
```

2.1 *Growing plants*

Rather than setting plant physical sizes to given values, it will often be required to *grow* them to a size. This is required to compute seed output (integrated over the plant's lifetime) stem diameter, survival, etc; basically everything except for height.

It's possible to directly integrate the equations exposed by the plant, using the `ode_state` field, `compute_vars_phys` method and `ode_rates` field. For example, we can use the R package `deSolve`:

```
derivs <- function(t, y, plant, env) {
  plant$ode_state <- y
  plant$compute_vars_phys(env)
  list(plant$ode_rates)
}
pl <- FF16_Plant()
tt <- seq(0, 50, length.out=101)
y0 <- setNames(pl$ode_state, pl$ode_names)
yy <- deSolve::lsoda(y0, tt, derivs, pl, env=env)
plot(height ~ time, yy, type="l")
```

Alternatively, it might desirable to grow a plant to a particular size. We could interpolate from the previous results easily enough. E.g., to find a plant with height of 15 m:

```
h <- 15.0
```

that happened approximately here:

```
t <- spline(yy[, "height"], yy[, "time"], xout=h)$y
```

Interpolate to find the state:

```
y <- apply(yy[, -1], 2, function(y) spline(yy[, "time"], y, xout=t)$y)

pl2 <- FF16_Plant()
pl2$ode_state <- y
pl2$compute_vars_phys(env)
```

Plant is expected height:

```
pl2$height
```

```
## [1] 15
```

And at this height, here is the total seed production:

```
pl2$fecundity
```

```
## [1] 289.9376
```

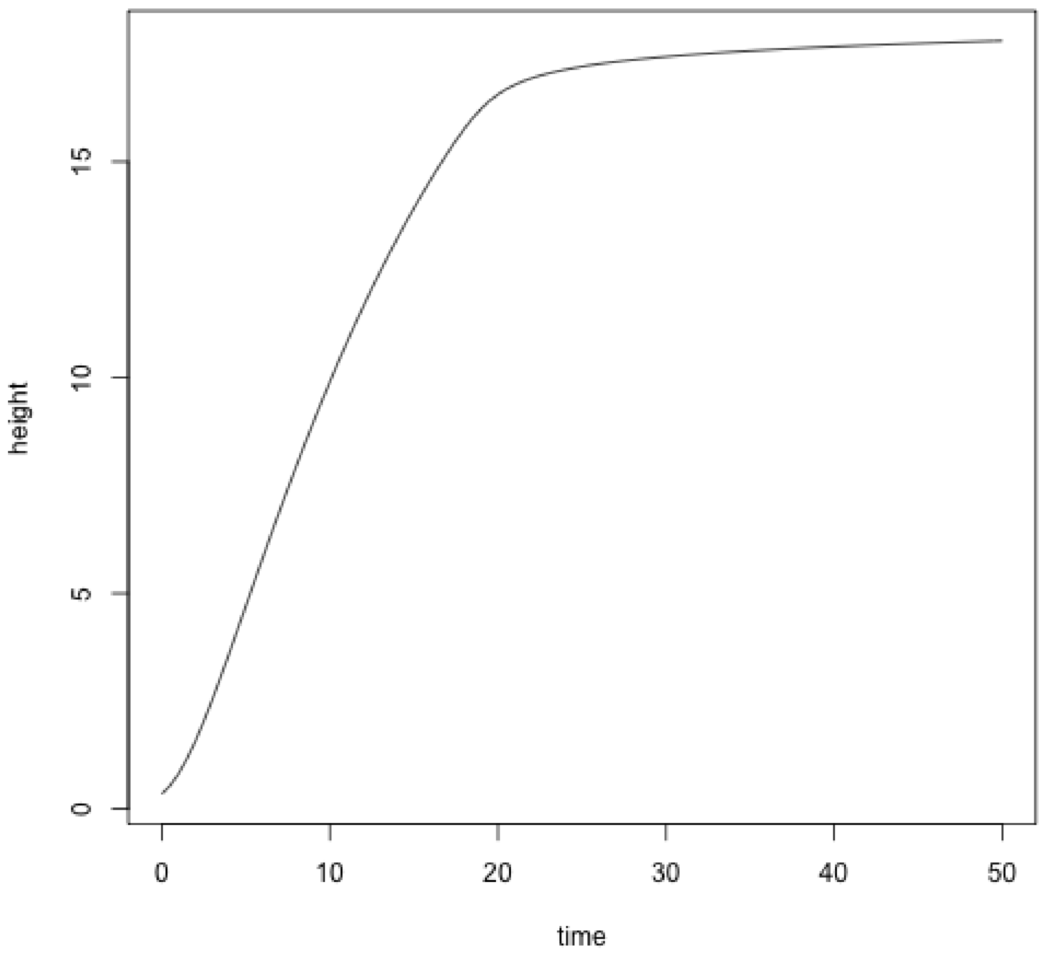


FIGURE 1

To make this type of operation easier, we provide the function `grow_plant_to_time`

```
res <- grow_plant_to_time(FF16_PlantPlus(FF16_Strategy()), tt, env)
```

Here is the result, plotted against the result obtained from using `deSolve` above:

```
plot(height ~ tt, res$state, type="l", las=1,
     xlab="Time (years)", ylab="Height (m)")
points(height ~ time, yy, col="grey", cex=.5)
```

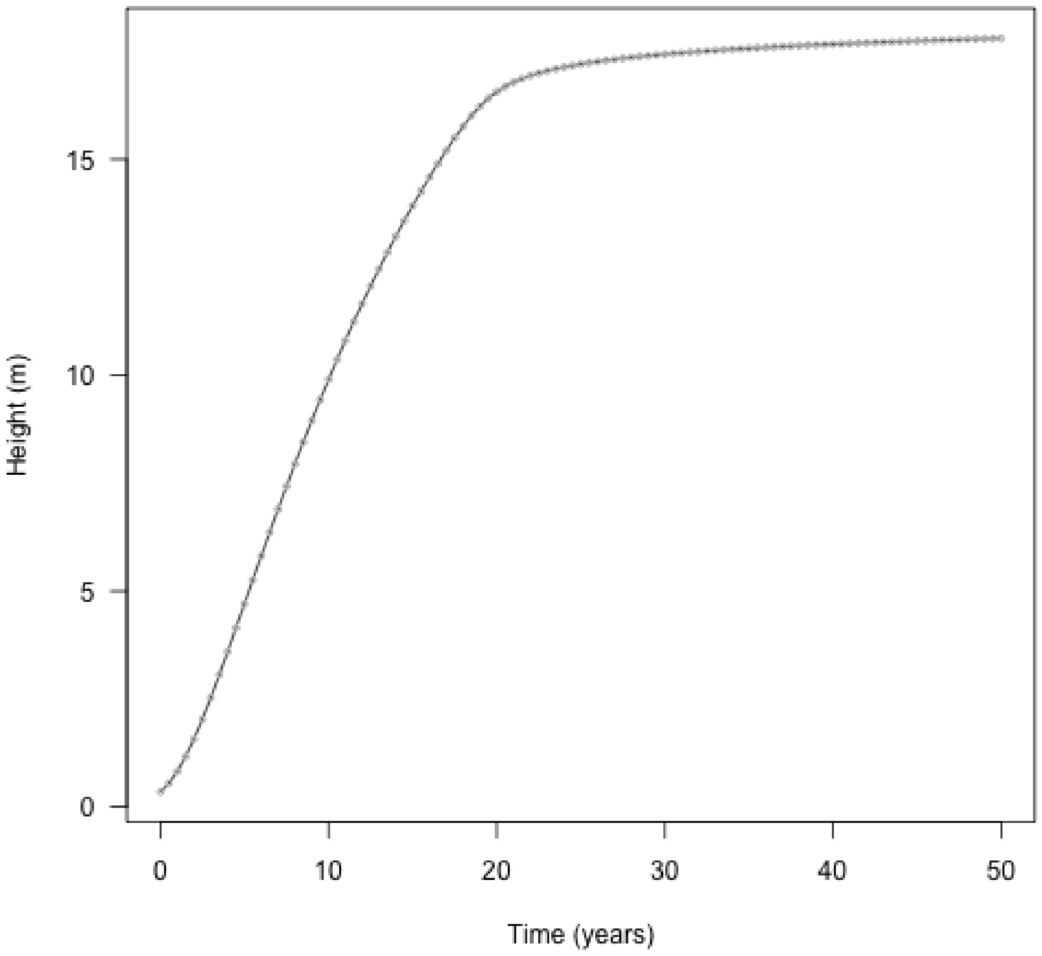


Figure 2

Completing the set, `plant` also provides a function for growing plants to a particular size; `grow_plant_to_size`. This takes *any* size measurement in the plant and can grow the plant to that size. So, for height:

```
pl <- FF16_PlantPlus(FF16_Strategy())
heights <- seq(pl$height, pl$strategy$hmat, length.out=20)
res <- grow_plant_to_size(pl, heights, "height", env)
```

This returns a vector of times; this is when the heights were achieved

```
res$time
```

```
##  [1]  0.000000  1.545681  2.523393  3.365033  4.155087  4.926508  5.696593
##  [8]  6.476387  7.274127  8.096757  8.950785  9.842785 10.779771 11.769706
## [15] 12.821689 13.946774 15.158668 16.476692 17.957500 20.137320
```

A matrix of state:

```
head(res$state)
```

```
##         height  mortality    fecundity area_heartwood mass_heartwood
## [1,] 0.3441948 0.00000000 0.000000e+00   0.000000e+00   0.000000e+00
## [2,] 1.1995465 0.01545682 7.754892e-20   1.280271e-07   6.544158e-05
## [3,] 2.0548958 0.02523396 5.654011e-18   9.826616e-07   8.617440e-04
## [4,] 2.9102485 0.03365038 2.252986e-16   3.837305e-06   4.790943e-03
## [5,] 3.7656006 0.04155098 6.750246e-15   1.081677e-05   1.755419e-02
## [6,] 4.6209512 0.04926531 1.710259e-13   2.510437e-05   5.019027e-02
```

And a list of plants

```
res$plant[[10]]
```

```
## <PlantPlus<FF16>>
##   Inherits from: <PlantPlus>
##   Public:
##     .ptr: externalptr
##     area_heartwood: active binding
##     area_leaf: active binding
##     area_leaf_above: function (h)
##     clone: function (deep = FALSE)
##     compute_vars_growth: function ()
##     compute_vars_phys: function (environment)
##     fecundity: active binding
##     germination_probability: function (environment)
##     height: active binding
##     initialize: function (ptr)
##     internals: active binding
##     mass_heartwood: active binding
##     mortality: active binding
##     ode_names: active binding
##     ode_rates: active binding
##     ode_size: active binding
##     ode_state: active binding
##     strategy: active binding
##     to_plant: function ()
```

```
res$plant[[10]]$height
```

```
## [1] 8.042356
```

```
heights[[10]]
```

```
## [1] 8.042356
```

Also included is `trajectory`; the points that the ODE stepper used with the system state at those times.

There is a convenience function `run_plant_to_heights` that achieves the same thing. Alternatively, and variable within `plant$internals` can be used, so long as it increases monotonically with respect to time.

```
pl <- FF16_PlantPlus(FF16_Strategy())
mass <- seq_log(pl$internals$mass_above_ground + 1e-8, 1000, length.out=21)
res_mass <- grow_plant_to_size(pl, mass, "mass_above_ground", env,
                               time_max=100, warn=FALSE)
```

(this seems on the low side - see this issue).

```
plot(res_mass$time, mass, type="o", pch=19, las=1, xlab="Time (years)")
```

With all these bits in place, let's look at growth trajectories of two species that differ in their LMA values. This what is presented In Fig. 2a of the paper.

```
p <- scm_base_parameters("FF16")
```

Low LMA ("fast growth") species

```
s1 <- strategy(trait_matrix(0.0825,  "lma"), p)
```

High LMA ("low growth") species

```
s2 <- strategy(trait_matrix(0.2625, "lma"), p)
```

Note that we're using an alternative way of specifying strategies here, to trigger our "hyper-parametrisation" approach. This may be simplified in future, but currently the "hyper-parametrisation" function resides on the `p` object.

Then, generate a sequence of heights to collect information at

```
pl1 <- FF16_PlantPlus(s1)
pl2 <- FF16_PlantPlus(s2)
```

(they are different for the two plants because they have different starting heights, the lower LMA of s1 allows it to achieve a greater initial height for given seed mass)

```
heights1 <- seq(pl1$height, s1$hmat, length.out=100L)
heights2 <- seq(pl2$height, s2$hmat, length.out=100L)

dat1 <- grow_plant_to_height(pl1, heights1, env,
                             time_max=100, warn=FALSE, filter=TRUE)
dat2 <- grow_plant_to_height(pl2, heights2, env,
```

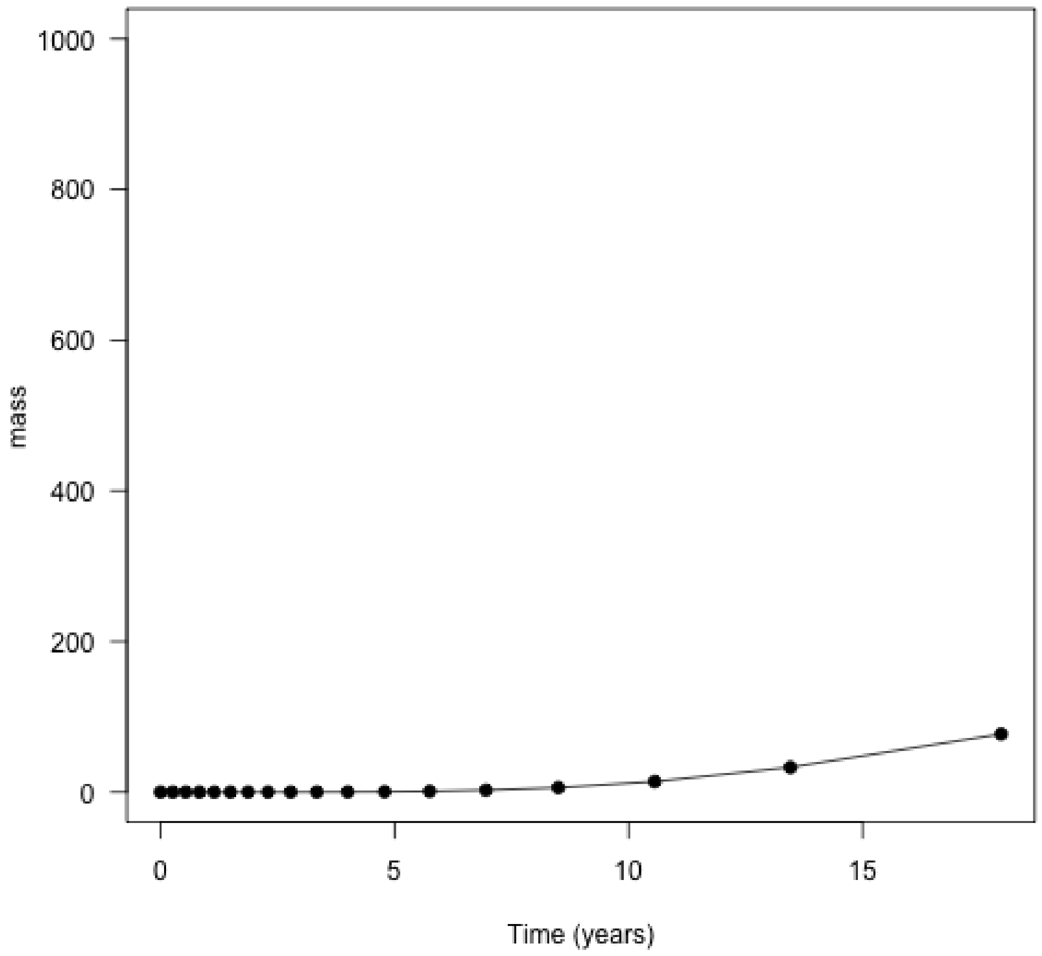


Figure 3

```
                              time_max=100, warn=FALSE, filter=TRUE)

plot(dat1$trajectory[, "time"], dat1$trajectory[, "height"],
     type="l", lty=1,
     las=1, xlab="Time (years)", ylab="Height (m)")
lines(dat2$trajectory[, "time"], dat2$trajectory[, "height"], lty=2)
legend("bottomright", c("Low LMA", "High LMA"), lty=1:2, bty="n")
```

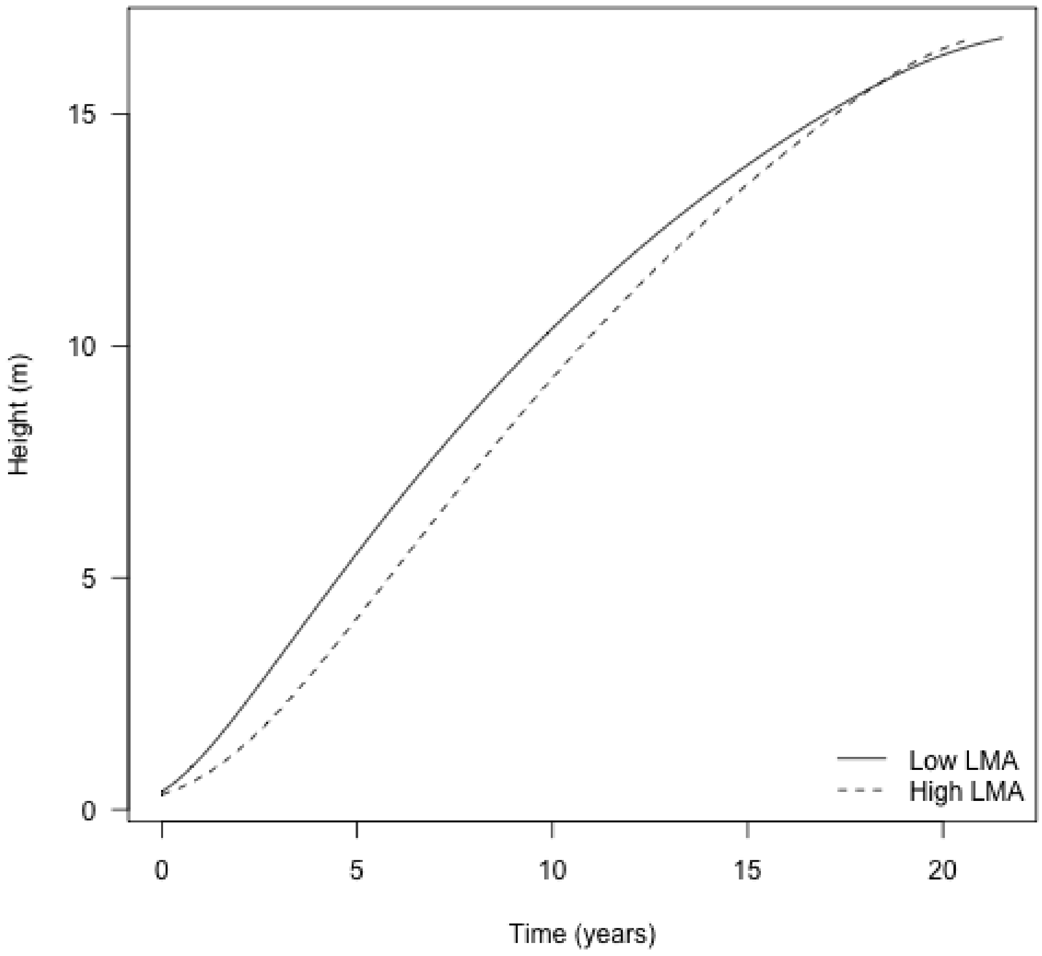


Figure 4

Similarly, growing the plants under lower light:

```
env_low <- fixed_environment(0.5)
dat1_low <- grow_plant_to_height(pl1, heights1, env_low,
                                  time_max=100, warn=FALSE, filter=TRUE)
dat2_low <- grow_plant_to_height(pl2, heights2, env_low,
                                  time_max=100, warn=FALSE, filter=TRUE)

cols <- c("black", "#e34a33")
plot(dat1$trajectory[, "time"], dat1$trajectory[, "height"],
     type="l", lty=1,
     las=1, xlab="Time (years)", ylab="Height (m)")
lines(dat2$trajectory[, "time"], dat2$trajectory[, "height"], lty=2)
lines(dat1_low$trajectory[, "time"], dat1_low$trajectory[, "height"],
      lty=1, col=cols[[2]])
lines(dat2_low$trajectory[, "time"], dat2_low$trajectory[, "height"],
      lty=2, col=cols[[2]])
legend("bottomright",
       c("High light", "Low light"), lty=1, col=cols,
       bty="n")
```

The *height growth rate* is the derivative of height with respect to time - the slope of the plot above. It is really the core quantity in the model; the actual heights are computed by solving the set of ODEs that includes height growth rate.

Growth rate with respect to height shows a hump-shaped pattern that is affected by both traits and by light environment. To extract this information from the trajectories takes a little more work though.

Here is a plant from part way through one run

```
pl <- dat1$plant[[50]]
```

Here is the set of ODE state variables:

```
setNames(pl$ode_state, pl$ode_names)

##          height      mortality      fecundity area_heartwood mass_heartwood
##    8.412120e+00   7.797210e-02   9.998839e-08   3.403727e-04   1.262008e+00
```

And the set of *rate* variables

```
setNames(pl$ode_rates, pl$ode_names)

##          height      mortality      fecundity area_heartwood mass_heartwood
##    9.530474e-01   1.003019e-02   3.243850e-07   1.780143e-04   8.068135e-01
```

. . . however, the rates might not be correct. They are whatever was left by the ODE stepper when it was advancing the plant, so it's best to update them:

```
pl$compute_vars_phys(dat1$env)
setNames(pl$ode_rates, pl$ode_names)

##          height      mortality      fecundity area_heartwood mass_heartwood
##    9.530474e-01   1.003019e-02   3.243850e-07   1.780143e-04   8.068135e-01
```

(in this case they are the same because the light environment is unchanging, but that not be the case generally)

Alternatively, we can access the height growth rate via the internals, which is the same as accessing directly from the ODE rates but more explicit:

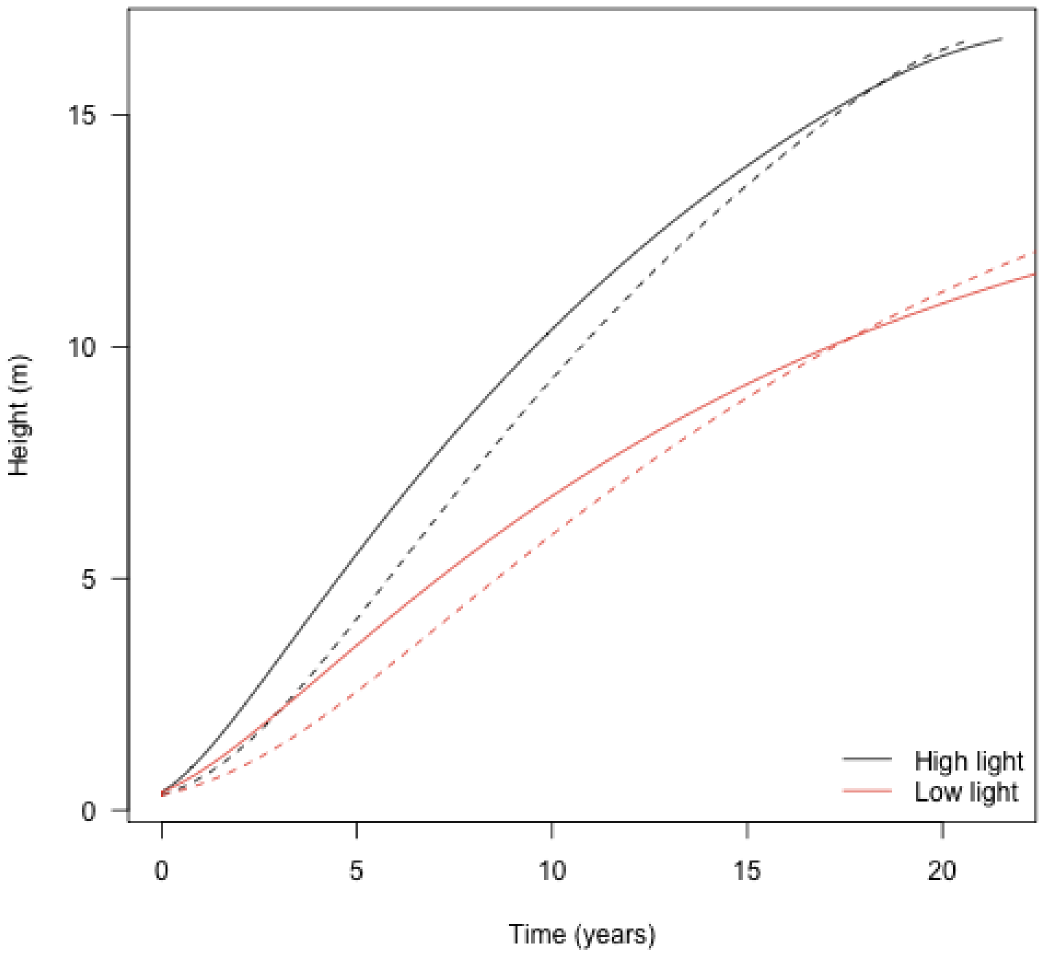


FIGURE 5

```
pl$internals$height_dt
```

```
## [1] 0.9530474
```

```
pl$ode_rates[[1]]
```

```
## [1] 0.9530474
```

Collecting height growth for all plants:

```
dhdt1 <- sapply(dat1$plant, function(x) x$internals$height_dt)
dhdt2 <- sapply(dat2$plant, function(x) x$internals$height_dt)
dhdt1_low <- sapply(dat1_low$plant, function(x) x$internals$height_dt)
dhdt2_low <- sapply(dat2_low$plant, function(x) x$internals$height_dt)

plot(dat1$time, dhdt1, type="l", lty=1,
     las=1, xlab="Time (years)", ylab="Height growth rate (m / yr)")
lines(dat2$time, dhdt2, lty=2)
lines(dat1_low$time, dhdt1_low, lty=1, col=cols[[2]])
lines(dat2_low$time, dhdt2_low, lty=2, col=cols[[2]])
legend("topright",
       c("High light", "Low light"), lty=1, col=cols,
       bty="n")
```

Alternatively, change in height plotted against height itself:

```
ylim <- c(0, max(dhdt1))
plot(dat1$state[, "height"], dhdt1, type="l", lty=1,
     las=1, xlab="Height (m)", ylab="Height growth rate (m / yr)", ylim=ylim)
lines(dat2$state[, "height"], dhdt2, lty=2)
lines(dat1_low$state[, "height"], dhdt1_low, lty=1, col=cols[[2]])
lines(dat2_low$state[, "height"], dhdt2_low, lty=2, col=cols[[2]])
legend("topright",
       c("High light", "Low light"), lty=1, col=cols,
       bty="n")
```

Over a plant's life, allocation to different structures varies. This is captured by a set of variables stored within the internals: e.g., `mass_leaf` `mass_sapwood`.

```
pl$internals$mass_leaf
```

```
## [1] 0.3428489
```

```
pl$internals$mass_sapwood
```

```
## [1] 4.034067
```

(these numbers seem a bit off: one of the motivations here is to develop and use better models of plant allometry. The parameterisation used at present are derived from adults and perform poorly with small plants. However, based on height / area relationships [Falster 2011, supporting information], for an 8m tall plant total leaf areas of 5-10 m are plausible and with an LMA of 0.08 that implies a total *dry* weight of 400 - 800 g).

Total live dry mass fraction to leaf and stem can be computed as:

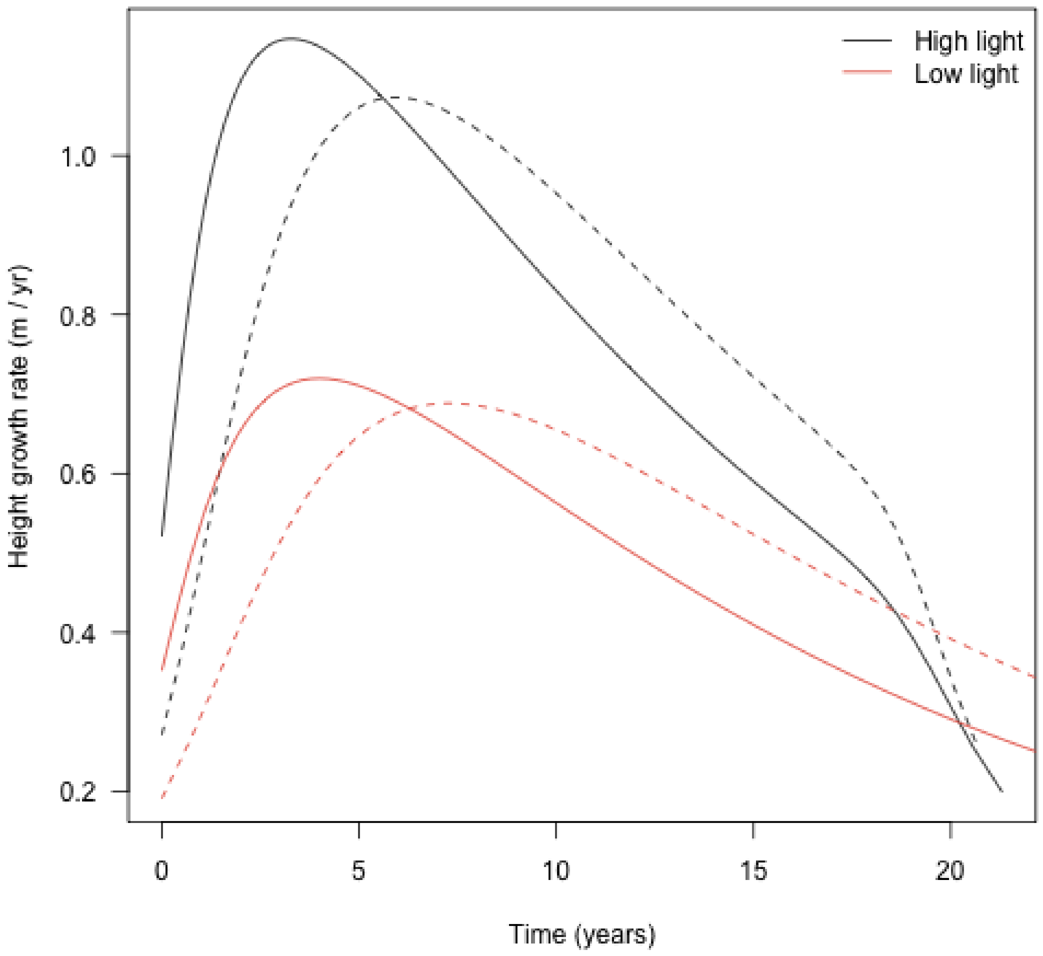

High light
Low light
Height growth rate (m / yr)
1.0
0.8
0.6
0.4
0.2
0
5
10
15
20
Time (years)

Figure 6

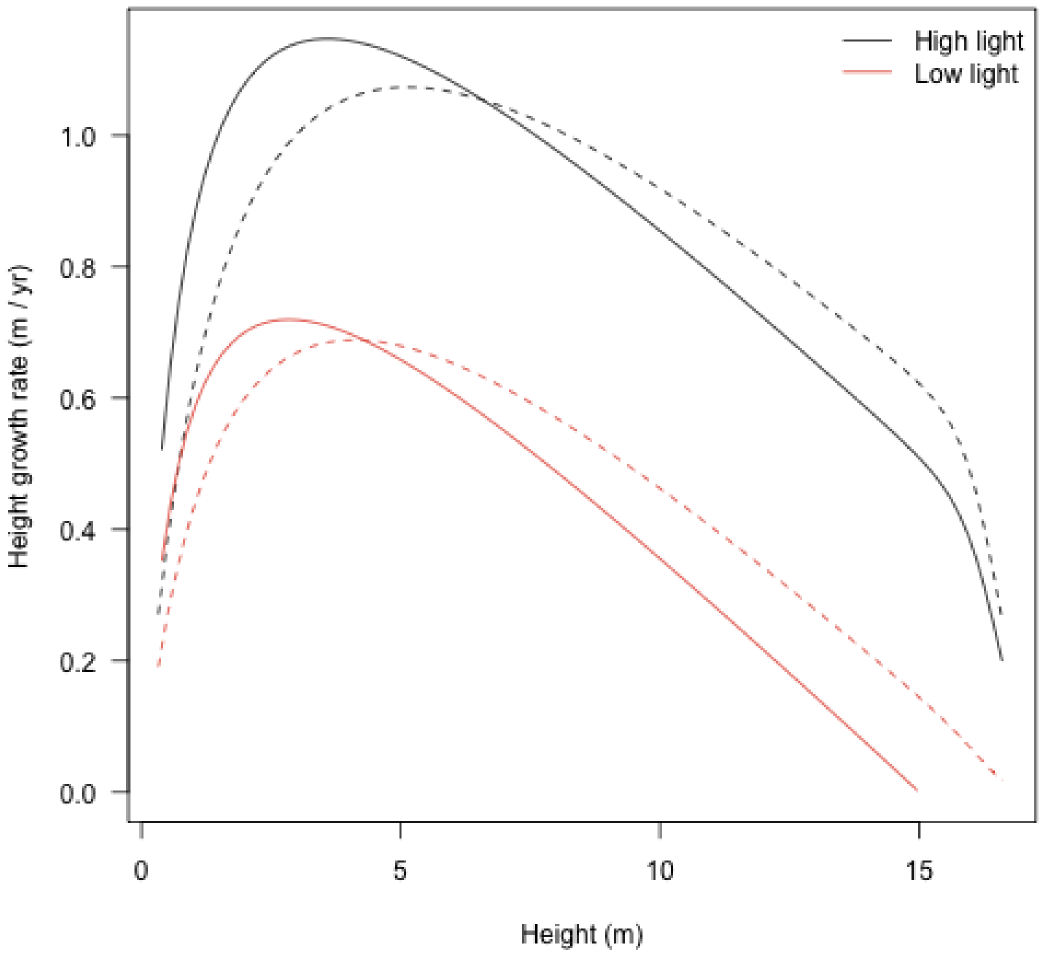

High light
Low light
1.0
0.8
0.6
0.4
0.2
0.0
Height growth rate (m / yr)
0
5
10
15
Height (m)

Figure 7

```
f <- function(p) {
  p_ints <- p$internals
  c(leaf=p_ints$mass_leaf / p_ints$mass_live,
    sapwood=p_ints$mass_sapwood / p_ints$mass_live)
}

cols_part <- c("black", "#045a8d")
matplot(dat1$state[, "height"], t(sapply(dat1$plant, f)),
        type="l", col=cols_part, lty=1, ylim=c(0, 1),
        xlab="Height (m)", ylab="Fractional allocation", las=1)
matlines(dat2$state[, "height"], t(sapply(dat2$plant, f)),
         col=cols_part, lty=2)
legend("topright", c("Sapwood", "Leaf"), lty=1, col=rev(cols_part), bty="n")
```

Relative allocation to leaf mass drops steeply as a plant grows and is replaced by allocation to sapwood mass.

The growth rates vary with both size and light environment (see above).

```
pl <- FF16_PlantPlus()
pl$height <- 10
pl$compute_vars_phys(fixed_environment(1.0))
pl$internals$height_dt # in full light
```

```
## [1] 0.9231625
```

```
pl$compute_vars_phys(fixed_environment(0.5))
pl$internals$height_dt # in 1/2 light
```

```
## [1] 0.4522332
```

At some point the plant cannot maintain positive carbon balance and therefore cannot grow; for example at 25% canopy openness:

```
pl$compute_vars_phys(fixed_environment(0.25))
pl$internals$height_dt
```

```
## [1] 0
```

The light level at which carbon gain becomes zero is the “whole plant light compensation point”.

```
lcp_whole_plant(pl)
```

```
## [1] 0.2991714
```

Consider a vector of canopy opennesses:

```
openness <- seq(0, 1, length.out=51)
```

```
lcp <- lcp_whole_plant(pl)
```

Height growth rate increases in a saturating fashion with increasing canopy openness from the light compensation point.

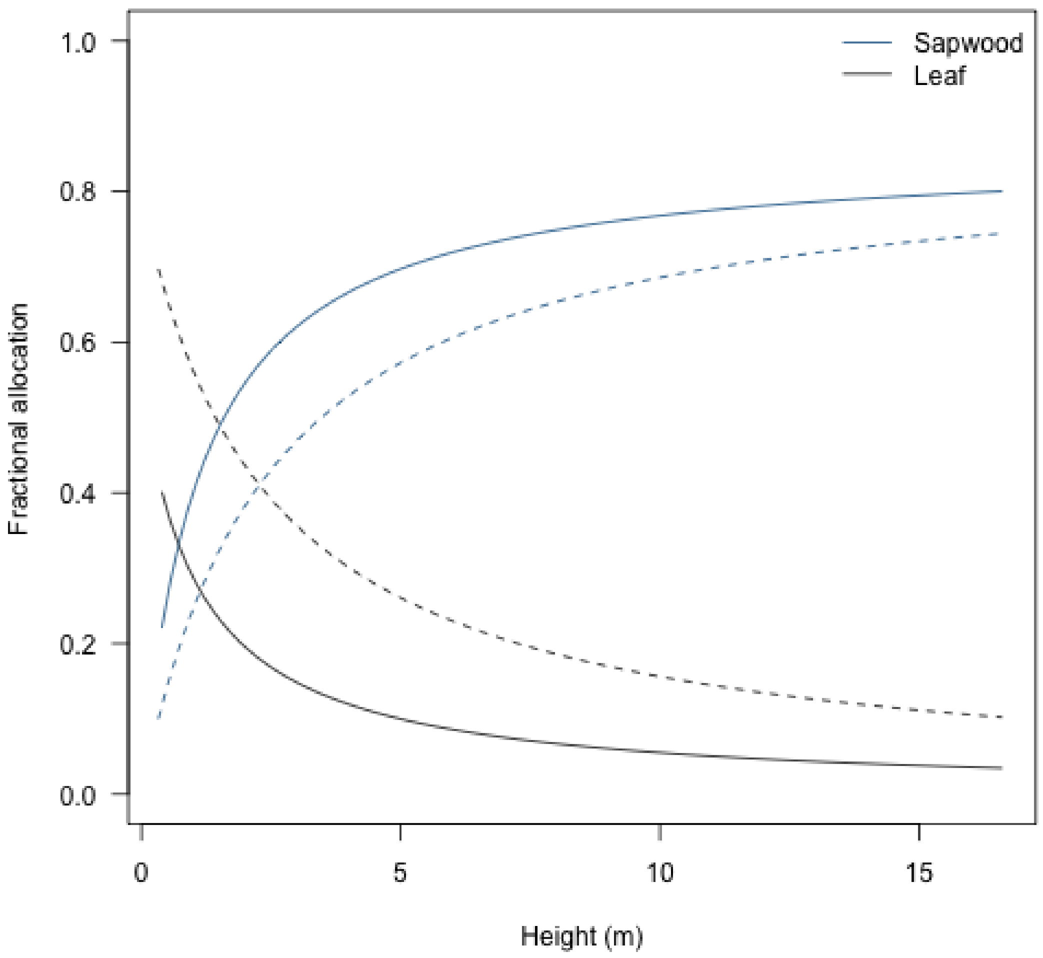

Sapwood
Leaf
1.0
0.8
0.6
0.4
0.2
0.0
Fractional allocation
0
5
10
15
Height (m)

Figure 8

```
f <- function(x, pl) {
  env <- fixed_environment(x)
  pl$compute_vars_phys(env)
  pl$internals$height_dt
}
x <- c(lcp, openness[openness > lcp])
plot(x, sapply(x, f, pl), type="l", xlim=c(0, 1),
     las=1, xlab="Canopy openness", ylab="Height growth rate (m / yr)")
points(lcp, 0.0, pch=19)
```

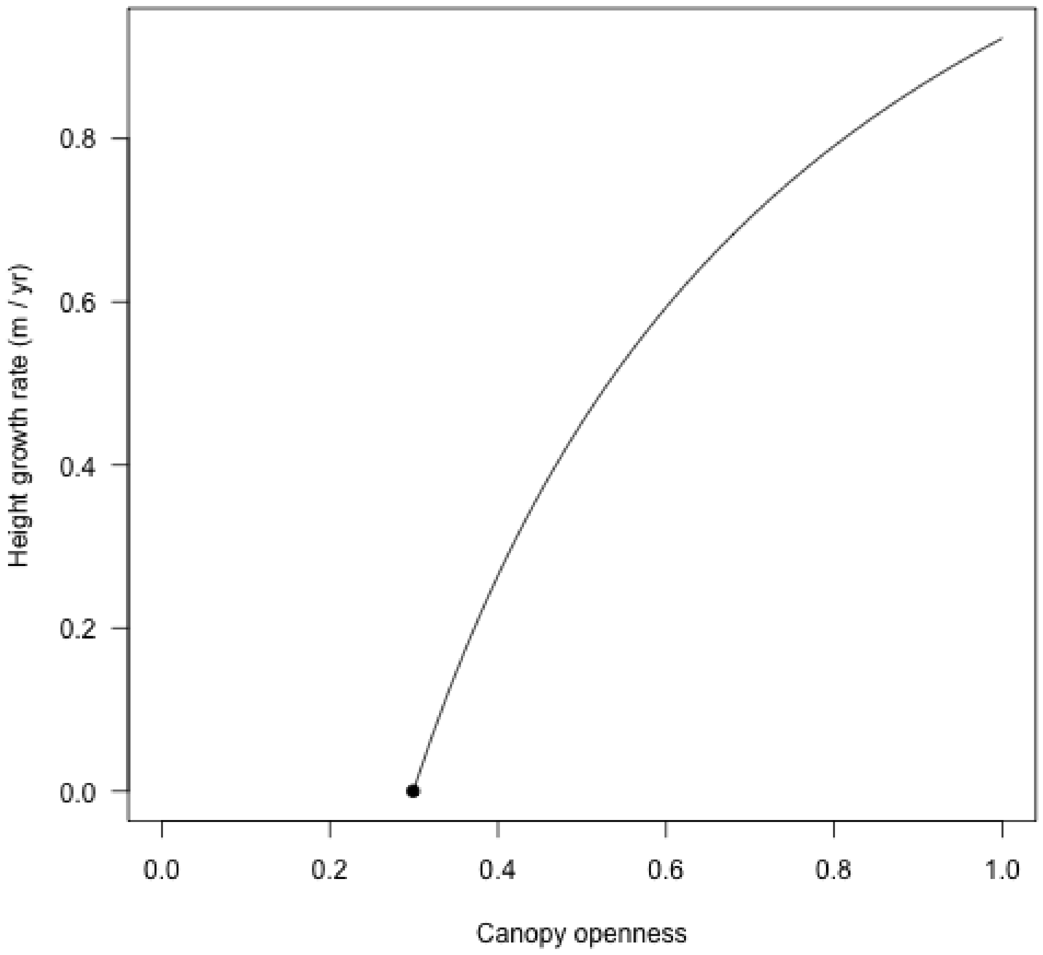


Figure 9

```
g <- function(x, pl) {
  lcp <- lcp_whole_plant(pl)
  x <- c(lcp, openness[openness > lcp])
  cbind(x=x, y=sapply(x, f, pl))
}
```

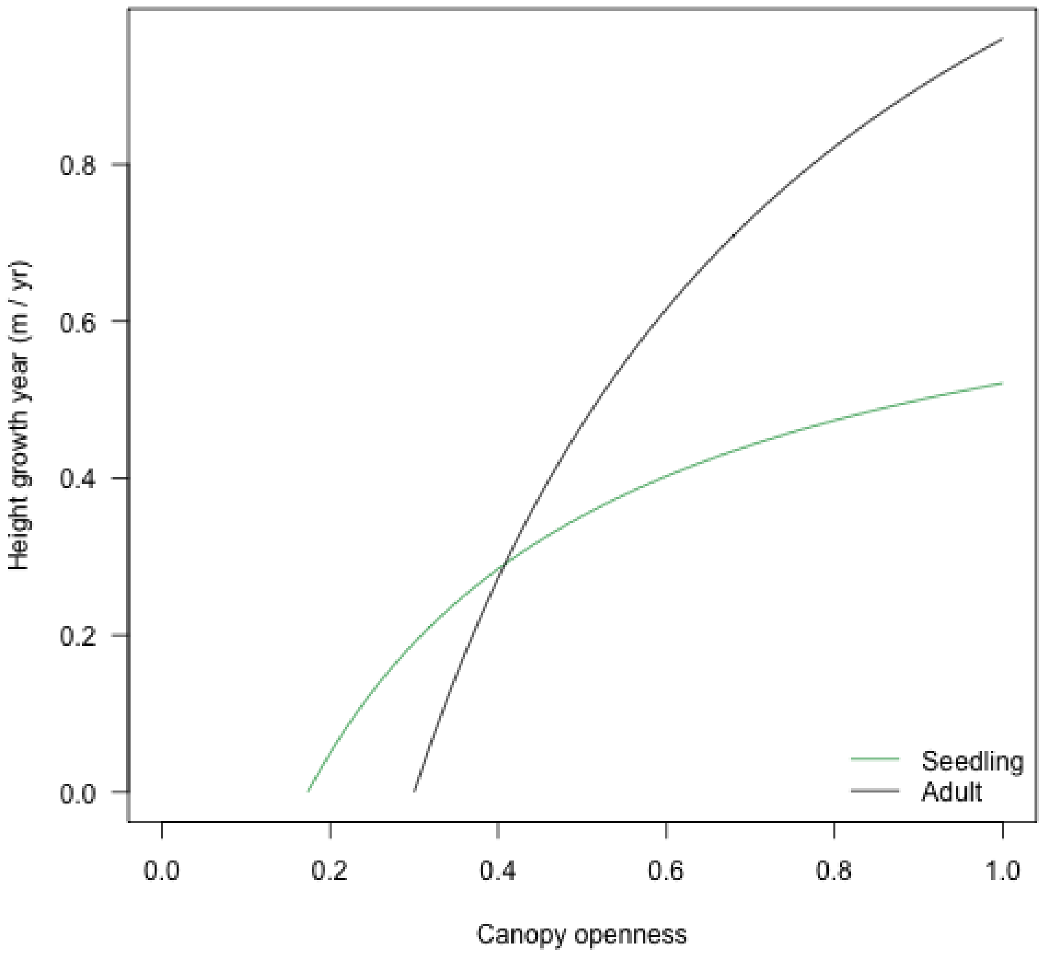
Now, consider this for a seedling and for a plant at half its maximum size size:

```
pl_seed <- FF16_PlantPlus(s1)
y_seed <- g(openness, pl_seed)
pl_adult <- FF16_PlantPlus(s1)
pl_adult$height <- pl_adult$strategy$hmat / 2
y_adult <- g(openness, pl_adult)
cols_height <- c("#31a354", "black")

ymax <- max(y_seed[, 2], y_adult[, 2])
plot(y_seed, type="l", col=cols_height[[1]],
     xlim=c(0, 1), ylim=c(0, ymax), las=1,
     xlab="Canopy openness", ylab="Height growth year (m / yr)")
lines(y_adult, col=cols_height[[2]])
legend("bottomright", c("Seedling", "Adult"), lty=1, col=cols_height, bty="n")
```

The light compensation point and curve varies with traits, too:

```
pl2_seed <- FF16_PlantPlus(s2)
pl2_adult <- FF16_PlantPlus(s2)
pl2_adult$height <- pl2_adult$strategy$hmat / 2
y2_seed <- g(openness, pl2_seed)
y2_adult <- g(openness, pl2_adult)

ymax <- max(ymax, y2_seed[, 2], y2_adult[, 2])
plot(y_seed, type="l", col=cols_height[[1]],
     xlim=c(0, 1), ylim=c(0, ymax), las=1,
     xlab="Canopy openness", ylab="Height growth year (m / yr)")
lines(y_adult, col=cols_height[[2]])
lines(y2_seed, col=cols_height[[1]], lty=2)
lines(y2_adult, col=cols_height[[2]], lty=2)
legend("bottomright", c("Seedling", "Adult"), lty=1, col=cols_height, bty="n")
```

Note that the traits with the lowest growth rate in most lights while a seedling (dotted lines) has the highest growth rate in all lights as an adult.



Height growth year (m / yr)
0.0
0.2
0.4
0.6
0.8
0.0
0.2
0.4
0.6
0.8
1.0
Canopy openness
Seedling
Adult

Figure 10

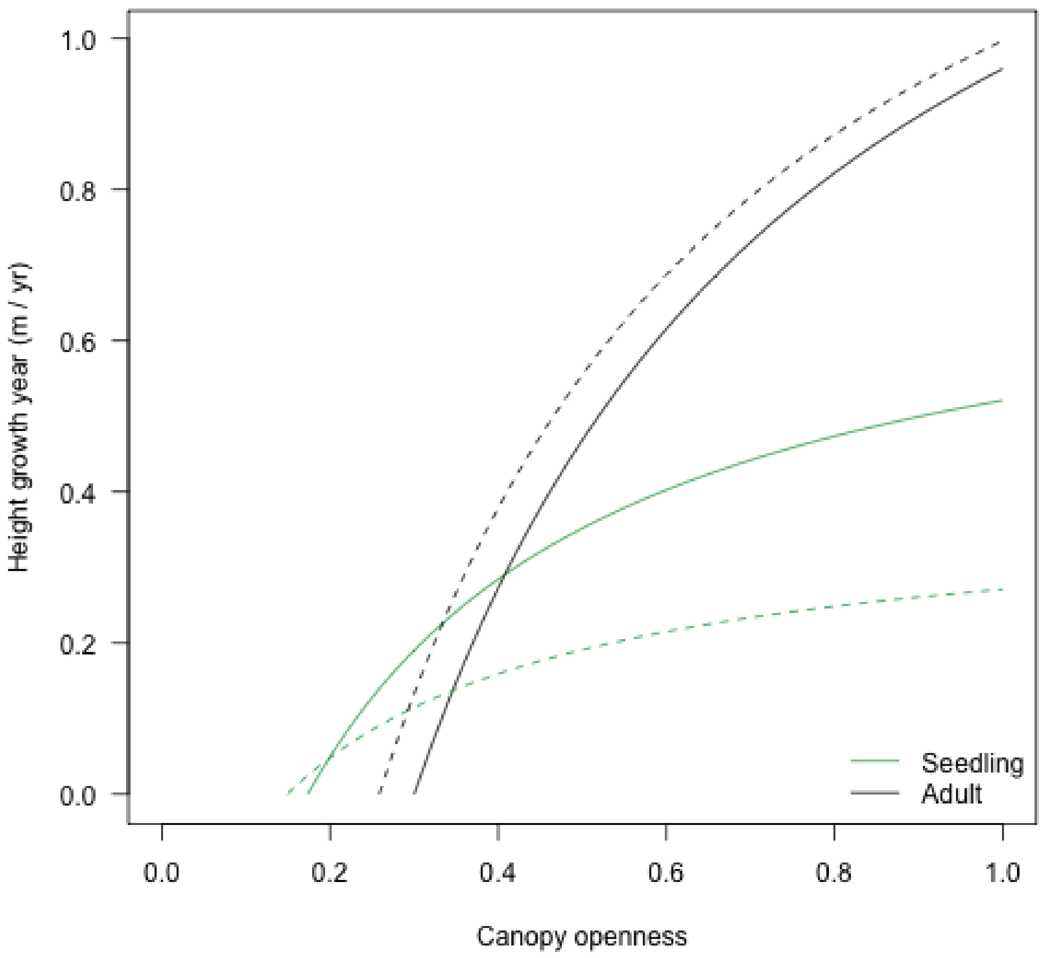

Height growth year (m / yr)
1.0
0.8
0.6
0.4
0.2
0.0
Seedling
Adult
0.0
0.2
0.4
0.6
0.8
1.0
Canopy openness

Figure 11

3 COHORT SPACING ALGORITHM

This vignette shows some details of cohort splitting. It's probably not very interesting to most people, only those interested in knowing how the SCM technique works in detail. It also uses a lot of non-exported, non-documented functions from plant so you'll see a lot of `plant:::` prefixes.

```
library(plant)
library(parallel)

p0 <- scm_base_parameters("FF16")
p <- expand_parameters(trait_matrix(0.0825, "lma"), p0, FALSE)
p$seed_rain <- 20 # close to equilibrium
```

The default cohort introduction times are designed to concentrate cohort introductions onto earlier times, based on empirical patterns of cohort refining:

```
t1 <- p$cohort_schedule_times[[1]]
plot(t1, cex=.2, pch=19, ylab="Time (years)", las=1)
```

The actual differences are stepped in order to increase the chance that cohorts from different species will be introduced at the same time and reduce the total amount of work being done.

```
plot(diff(t1), log="y", pch=19, cex=.2,
     ylab="Time difference (years)", las=1)
```

We can create more refined schedules by interleaving points between these points:

```
interleave <- function(x) {
  n <- length(x)
  xp <- c(x[-n] + diff(x) / 2, NA)
  c(rbind(x, xp))[- 2 * n]
}

t2 <- interleave(t1)
t3 <- interleave(t2)
```

Consider running the SCM and computing seed rain at the end; this is one of the key outputs from the model so a reasonable one to look for differences in.

```
run_with_times <- function(p, t) {
  p$cohort_schedule_times[[1]] <- t
  run_scm(p)$seed_rains
}

sr_1 <- run_with_times(p, t1)
sr_2 <- run_with_times(p, t2)
sr_3 <- run_with_times(p, t3)
```

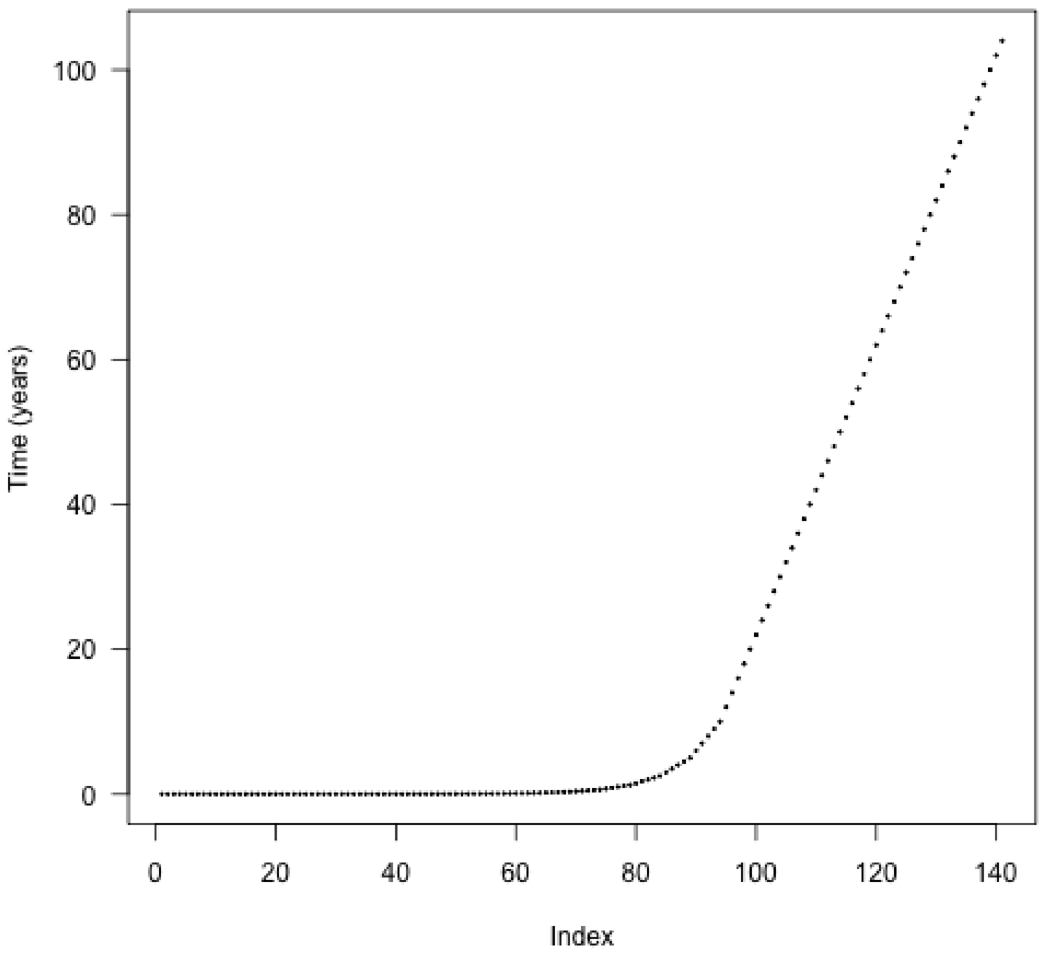


Figure 12

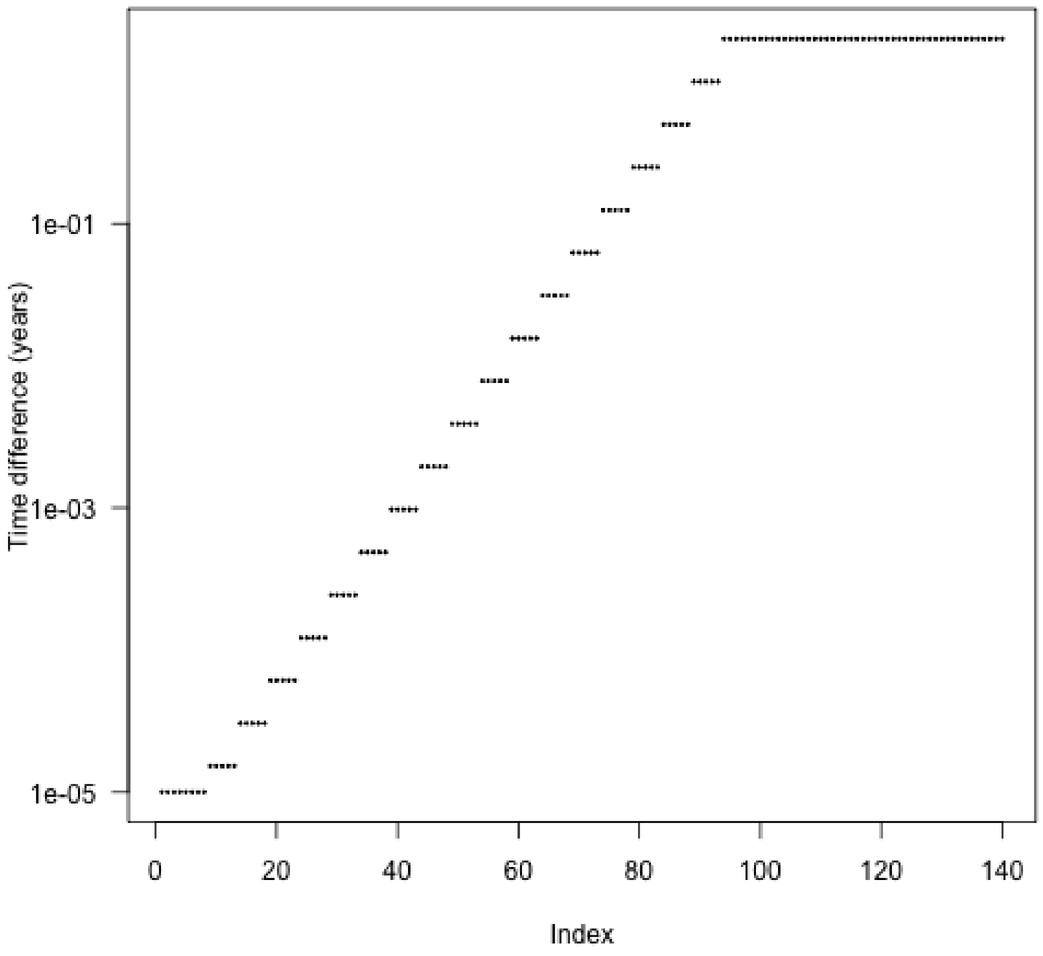

1e-01
1e-03
1e-05
Time difference (years)
0
20
40
60
80
100
120
140
Index

Figure 13

Seed rain increases as cohorts are introduced more finely, though at a potentially saturating rate. We're doing lots more work at the more refined end!

```
sr <- c(sr_1, sr_2, sr_3)
sr

## [1] 16.88934 16.87365 17.10484
```

The differences in seed rain are not actually that striking (perhaps 1%) but in some runs can be more, and the *variation* creates instabilities.

```
plot(c(length(t1), length(t2), length(t3)), sr,
     las=1, xlab="Number of cohort introductions", ylab="Seed rain")
```

Where is the fitness difference coming from?

Consider adding a *single* additional cohort at one of the points along the first vector of times `t1` and computing fitness:

```
insert_time <- function(i, x) {
  j <- seq_len(i)
  c(x[j], (x[i] + x[i+1])/2, x[-j])
}
run_with_insert <- function(i, p, t) {
  run_with_times(p, insert_time(i, t))
}
```

The internal function `plant:::run_scm_error` runs the SCM and computes errors as the integration proceeds; this helps shed some light.

```
i <- seq_len(length(t1) - 1)
res <- unlist(mclapply(i, run_with_insert, p, t1))
tm <- (t1[-1L] + t1[-length(t1)]) / 2
```

The biggest deviations in output seed rain come about half way through the schedule:

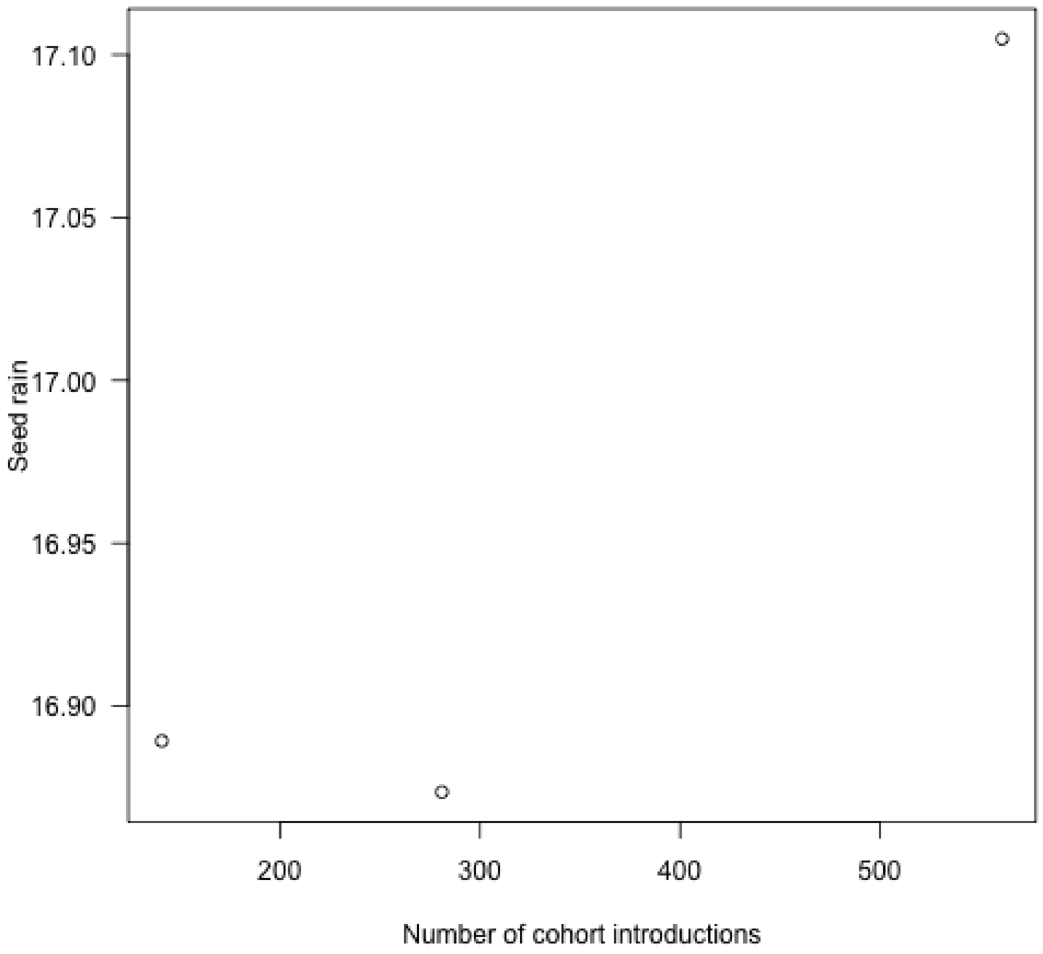
```
plot(i, res - sr_1, xlab="Index", ylab="Seed rain", las=1)
abline(h=0, col="grey")
```

Though because of the compression of early times it's still fairly early:

```
plot(tm, res - sr_1, xlab="Time (years)", ylab="Seed rain", las=1)
abline(h=0, col="grey")
```

```
ebt <- plant:::FF16_SCM(p)
ebt$run()
```

Now look at the contribution of different cohorts to see rain (x axis log scaled for clarity). In this case almost all the contribution comes from early cohorts (this is essentially a single-age stand of pioneers). Overlaid on this are the five cohorts with the largest change in total fitness (biggest difference in red). The difference is not coming from seed rain contributions from those cohorts, which is basically zero, though it is higher than the surrounding cohorts.



17.10
17.05
17.00
16.95
16.90
Seed rain
200
300
400
500
Number of cohort introductions

Figure 14

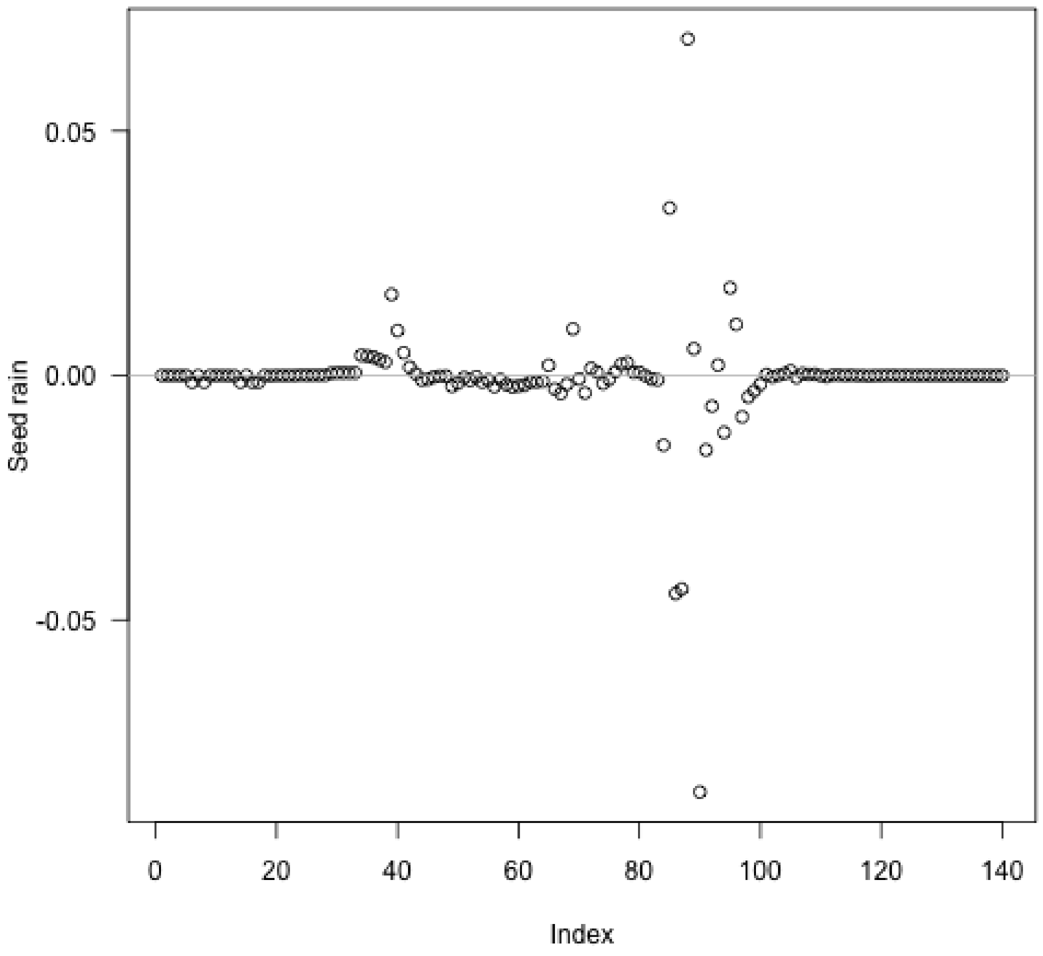


Figure 15

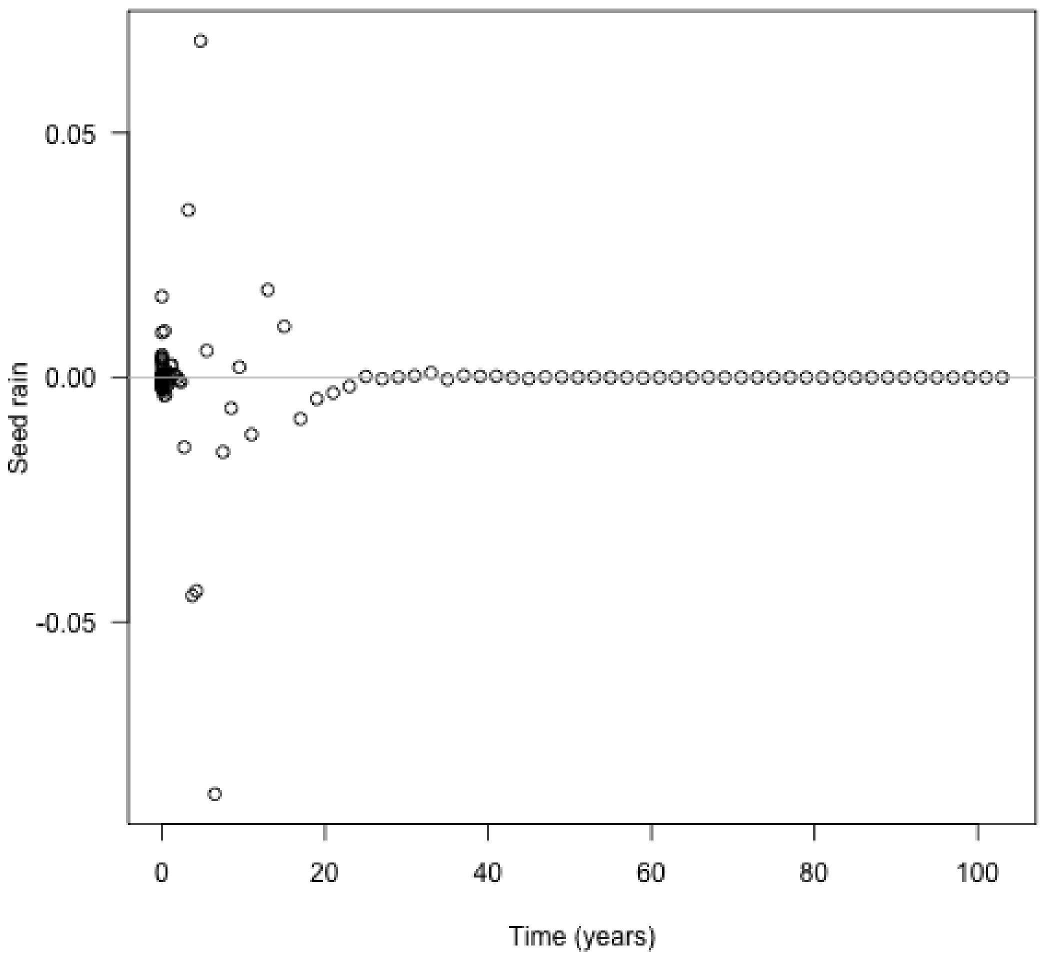


Figure 16

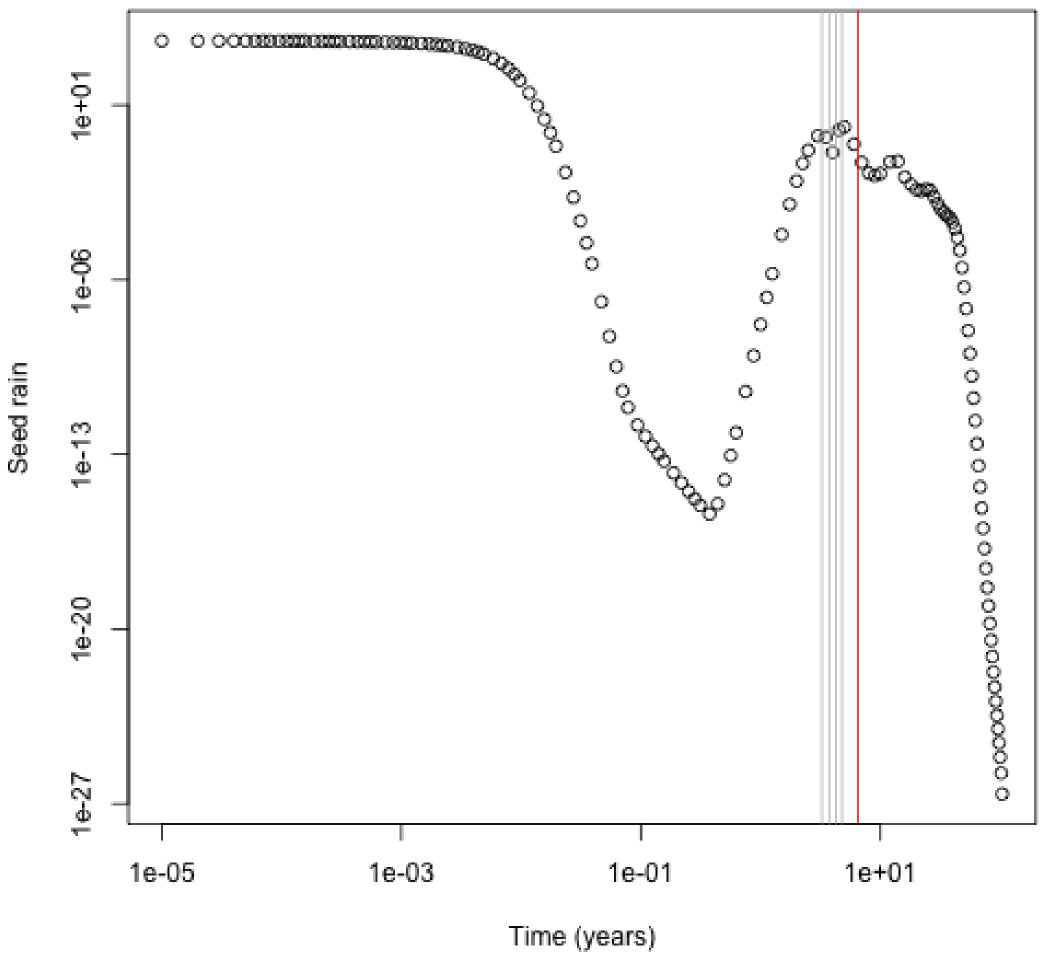

Seed rain
1e+01
1e-06
1e-13
1e-20
1e-27
1e-05
1e-03
1e-01
1e+01
Time (years)

Figure 17

```
plot(t1[-1], ebt$seed_rain_cohort(1)[-1], log="xy",
     xlab="Time (years)", ylab="Seed rain")
j <- order(abs(res - sr_1), decreasing=TRUE)[1:5]
abline(v=tm[j], col=c("red", rep("grey", 4)))
```

Next up, need to work out what the fitness contribution of each cohort is.

```
dat1 <- run_scm_collect(p)
p2 <- p
p2$cohort_schedule_times[[1]] <- insert_time(j[[1]], t1)
dat2 <- run_scm_collect(p2)
```

Then consider the light environment over time. This reconstructs the spline for the light environment for both runs, and computes canopy openness in both and computes the difference in light environments. The resulting image plot is blue in regions where the refined light environment is lighter (higher canopy openness) and red in regions where the the light environment is darker in the refined environment.

```
f <- function(e, h) {
  i <- plant:::Interpolator()
  i$init(e[, 1], e[, 2])
  y <- rep(1, length(h))
  j <- h < i$max
  y[j] <- i$eval(h[j])
  y
}
hmax <- max(dat1$light_env[[length(dat1$light_env)]][, "height"])
h <- seq(0, hmax, length.out=201)
y1 <- sapply(dat1$light_env, f, h)
y2 <- sapply(dat2$light_env, f, h)[, -(j[[1]] + 1L)]
dy <- t(y2 - y1)
dy[abs(dy) < 1e-10] <- NA
```

ColorBrewer's RdBu

```
cols <- c("#B2182B", "#D6604D", "#F4A582", "#FDDBC7",
          "#D1E5F0", "#92C5DE", "#4393C3", "#2166AC")
pal <- colorRampPalette(cols)(20)
image(dat1$time, h, dy, xlab="Time (years)", ylab="Height (m)", las=1,
      col=pal)
```

Because the differences are mostly manifest in the leaf area, we monitor error in both the leaf area and in fitness for all cohorts: (black line indicates the cohort identified as problematic above)

```
dat <- plant:::run_scm_error(p)
image(dat1$time, dat1$time, dat$err$lai[[1]],
      xlab="Patch age (years)", ylab="Introduction time (years)", las=1)
abline(h=t1[j[[1]]])
```

We then run through rounds of refining cohorts until estimated error has decreased down to an appropriate threshold:

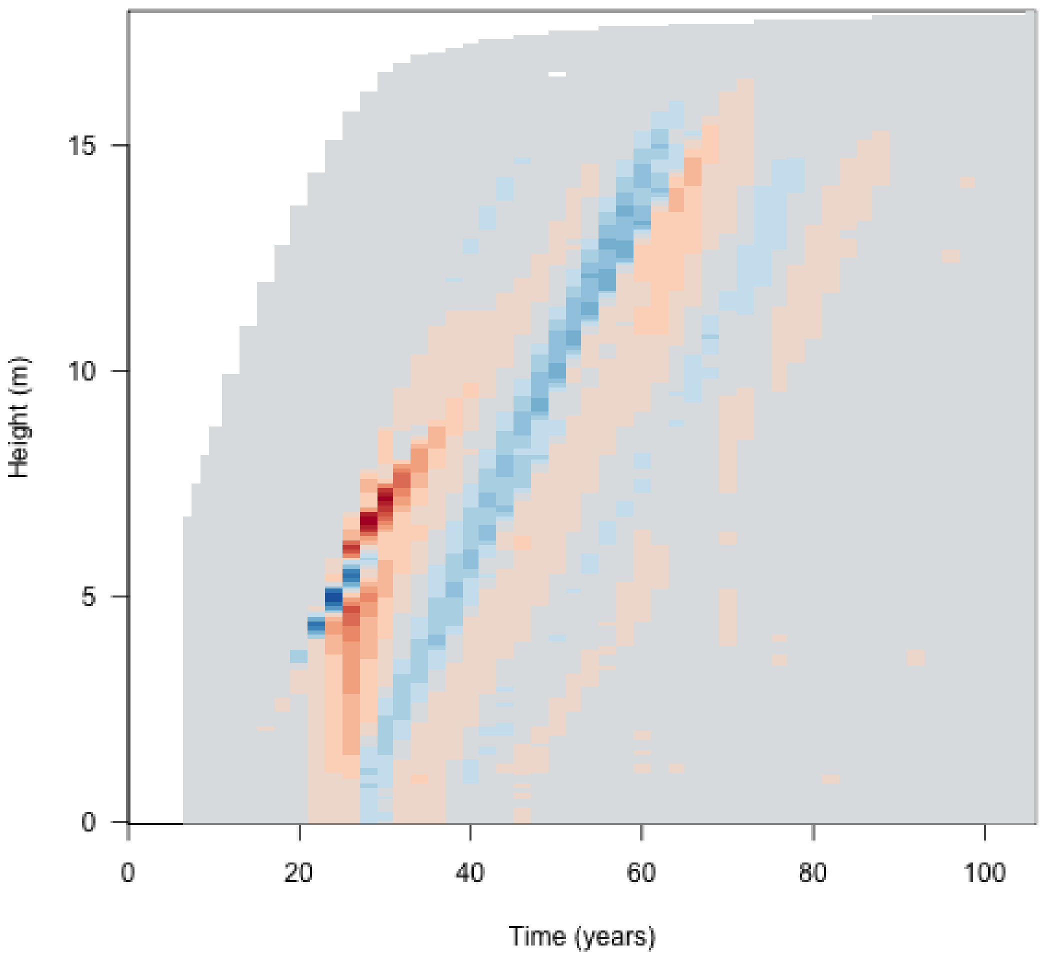


FIGURE 18

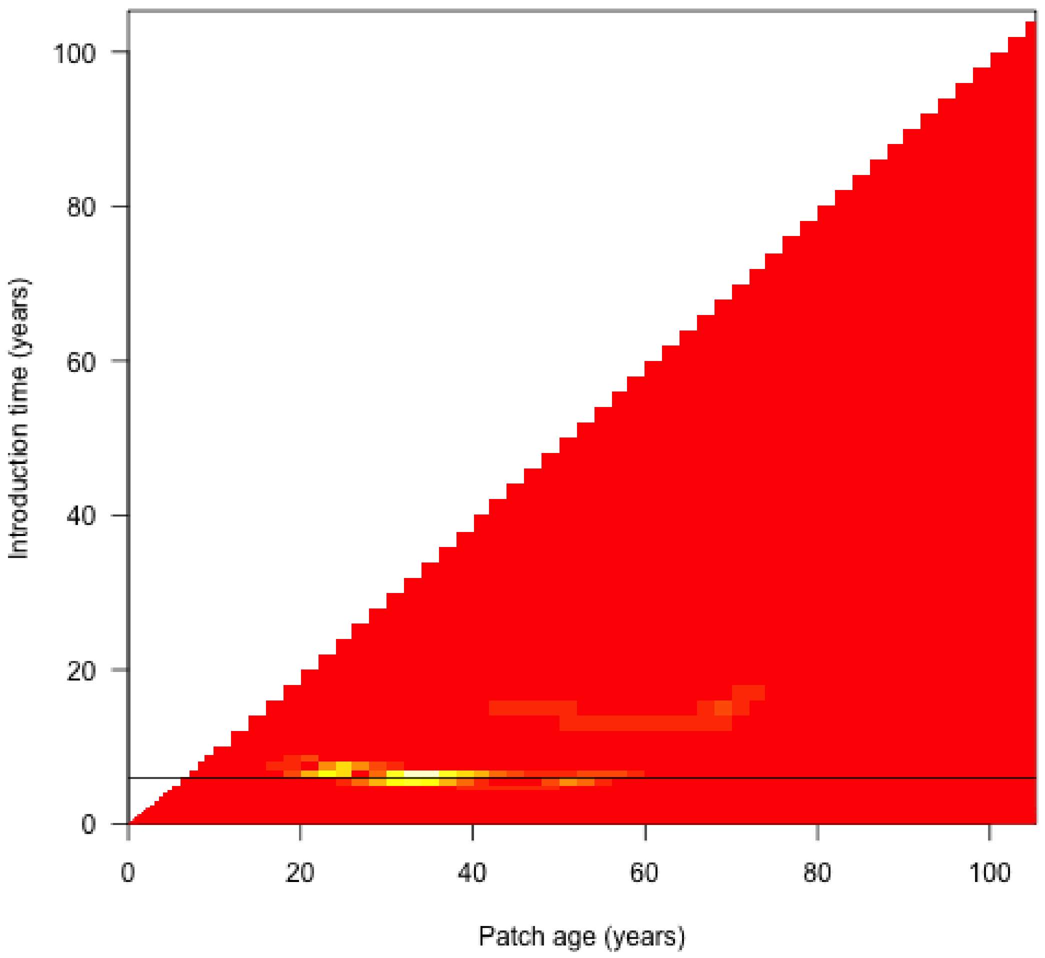

Introduction time (years)
Patch age (years)
0
20
40
60
80
100
0
20
40
60
80
100

Figure 19

```
p_refined <- build_schedule(p)
sr_refined <- attr(p_refined, "seed_rain_out")
```

Each round, the algorithm looks at the error in the leaf area calculations and in the fitness calculations and refines the worst cohorts, repeating as necessary.

```
t_refined <- p_refined$cohort_schedule_times[[1]]
i <- seq_len(length(t_refined) - 1)
res <- unlist(mclapply(i, run_with_insert, p_refined, t_refined))

tm <- (t1[-1L] + t1[-length(t1)]) / 2
```

The problem cohorts are still in about the same place, but are much less pronounced (and less so if considering relative error)

```
plot(i, res - sr_refined, xlab="Index", ylab="Seed rain", las=1)
abline(h=0, col="grey")


plot(i, res / sr_refined - 1, xlab="Index", ylab="Seed rain", las=1)
```

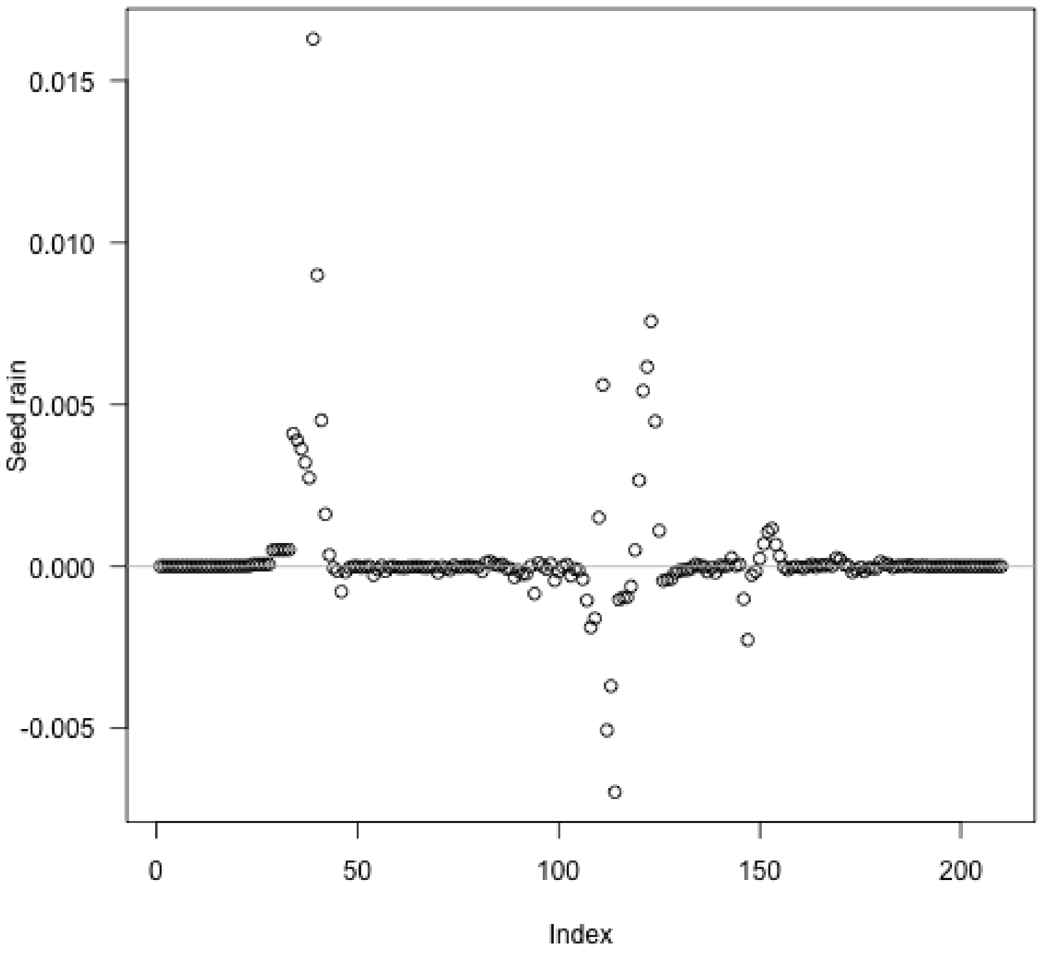


Figure 20

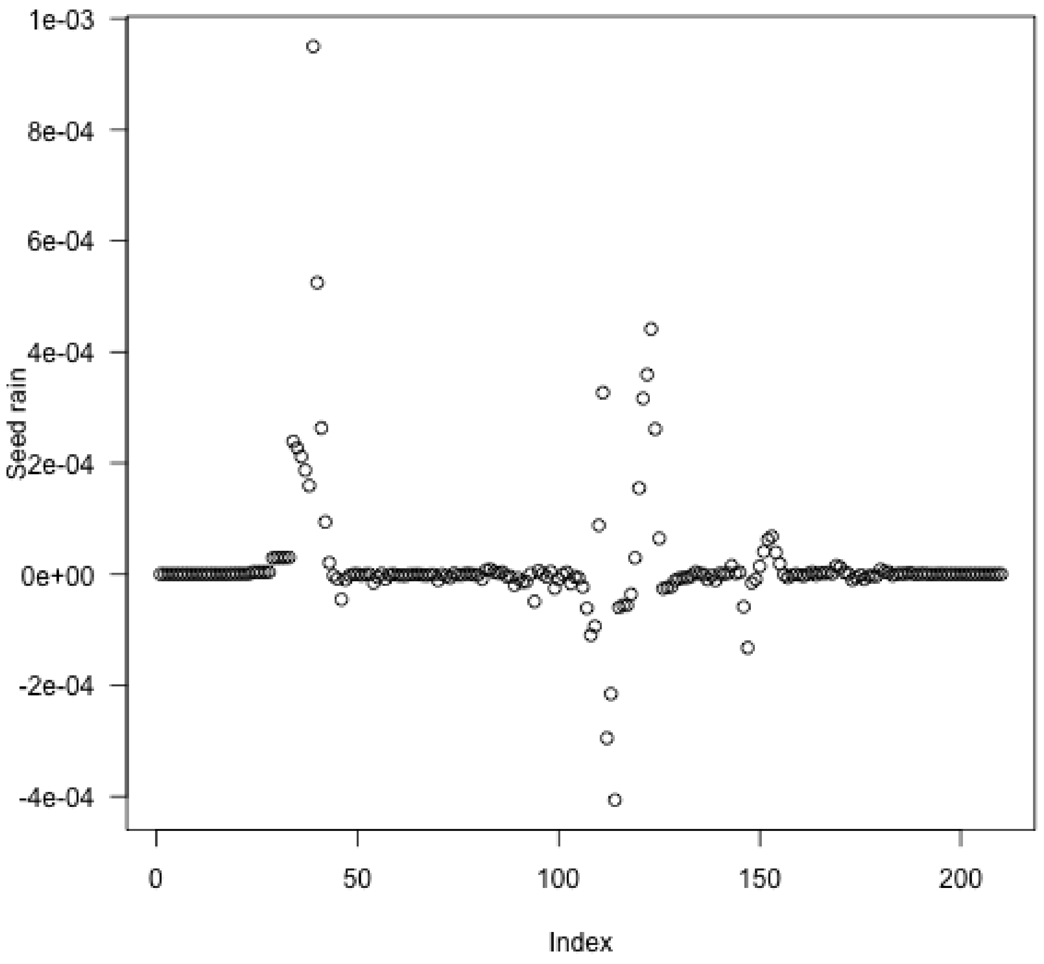

1e-03
8e-04
6e-04
4e-04
2e-04
0e+00
-2e-04
-4e-04
Seed rain
0
50
100
150
200
Index

Figure 21

4 FINDING DEMOGRAPHIC EQUILIBRIUM

```
library(plant)
library(parallel)

run <- function(seed_rain_in, p) {
  p$seed_rain <- seed_rain_in
  run_scm(p)$seed_rains
}

run_new_schedule <- function(seed_rain_in, p) {
  p$seed_rain <- seed_rain_in
  res <- build_schedule(p)
  attr(res, "seed_rain_out")
}

cobweb <- function(m, ...) {
  lines(rep(m[,1], each=2), c(t(m)), ...)
}
```

Try to establish what the equilibrium seed rain is.
Set model parameters

```
p0 <- scm_base_parameters("FF16")
p <- expand_parameters(trait_matrix(0.0825, "lma"), p0, FALSE)
```

Some output seed rains, given an input seed rain:

```
run(1.0, p)
```

```
## [1] 19.82443
```

```
run(10.0, p)
```

```
## [1] 17.63104
```

```
run(50.0, p)
```

```
## [1] 16.58796
```

When below the equilibrium, `run` returns a seed rain that is greater than the input, when above it, `run` returns a seed rain that is less than the input.

4.1 *1: Approach to equilibrium:*

```
p_eq <- equilibrium_seed_rain(p)
```

Equilibrium seed rain is right around where the last two values from run were producing

```
p_eq$seed_rain
```

```
## [1] 17.31512
```

From a distance, these both hone in nicely on the equilibrium, and rapidly, too.

```
approach <- attr(p_eq, "progress")
r <- range(approach)
plot(approach, type="n", las=1, xlim=r, ylim=r)
abline(0, 1, lty=2, col="grey")
cobweb(approach, pch=19, cex=.5, type="o")
```

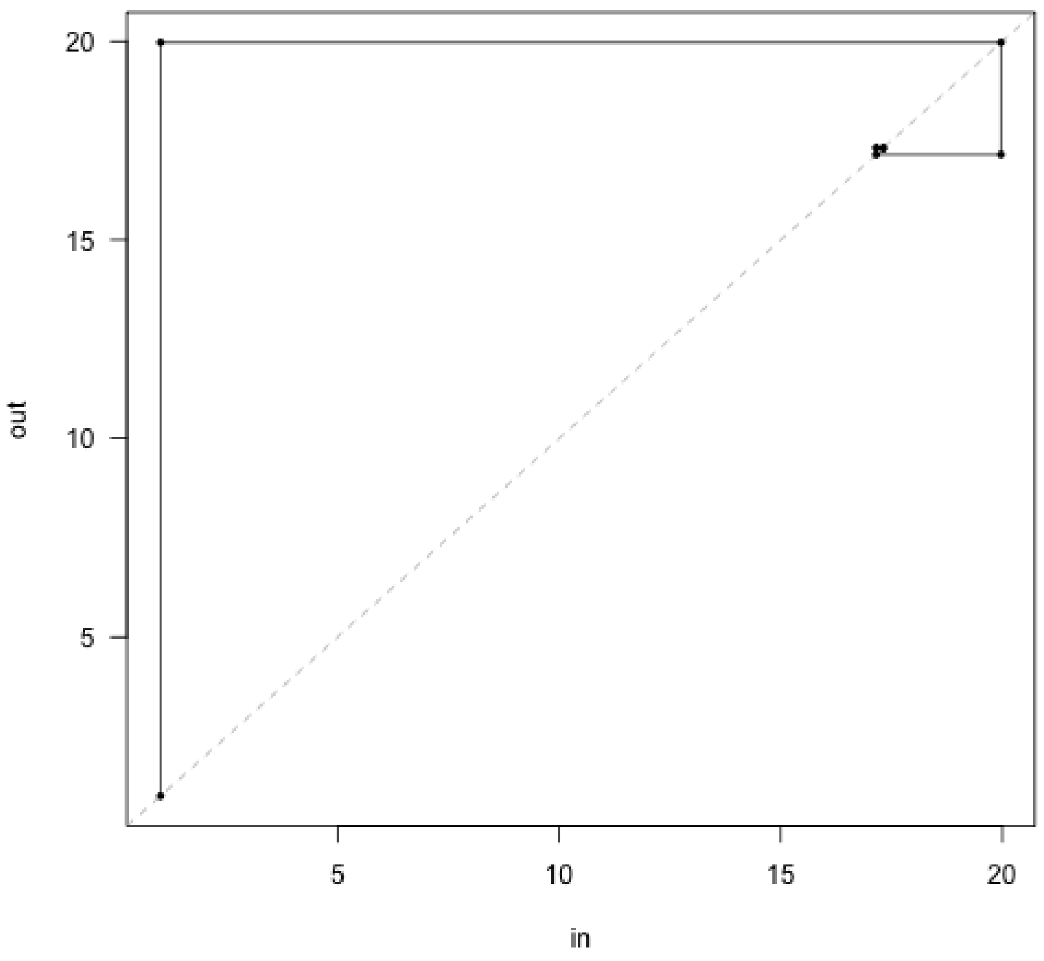


FIGURE 22

4.2 *2: Near the equilibrium point:*

Then, in the vicinity of the root we should look at what the curve actually looks like, without adaptive refinement.

```
dr <- 2 # range of input to vary (plus and minus this many seeds)
seed_rain_in <- seq(p_eq$seed_rain - dr,
                    p_eq$seed_rain + dr, length.out=31)
seed_rain_out <- unlist(mclapply(seed_rain_in, run, p_eq))
```

Here is input seeds vs. output seeds:

```
plot(seed_rain_in, seed_rain_out, xlab="Incoming seed rain",
     ylab="Outgoing seed rain", las=1, type="l", asp=5, col="red")
abline(0, 1, lty=2, col="grey")
cobweb(approach)
```

4.3 *4. Global function shape*

```
seed_rain_in_global <- seq(1, max(approach), length.out=51)
```

This takes quite a while to compute.

```
seed_rain_out_global <-
  unlist(mclapply(seed_rain_in_global, run_new_schedule, p))
```

This is pretty patchy, which is due to incompletely refining the cohort schedule, I believe. Tighten `schedule_eps` to make the curve smoother, at the cost of potentially a lot more effort.

```
plot(seed_rain_in_global, seed_rain_out_global,
     las=1, type="l",
     xlab="Incoming seed rain", ylab="Outgoing seed rain")
abline(0, 1, lty=2, col="grey")
cobweb(approach, lty=3)
```

4.4 *5. Multiple species at once:*

```
lma <- c(0.0825, 0.15)
p2 <- expand_parameters(trait_matrix(lma, "lma"), p0, FALSE)

p2_eq <- equilibrium_seed_rain(p2)
approach <- attr(p2_eq, "progress")
```

From a distance, these both hone in nicely on the equilibrium, and rapidly, too.

```
r <- range(unlist(approach))
plot(approach[[1]], type="n", las=1, xlim=r, ylim=r, xlab="in", ylab="out")
abline(0, 1, lty=2, col="grey")
cols <- c("black", "red")
for (i in 1:2) {
  cobweb(approach[, i + c(0, 2)], pch=19, cex=.5, type="o", col=cols[[i]])
}
abline(v=p2_eq$seed_rain, col=1:2, lty=3)
```

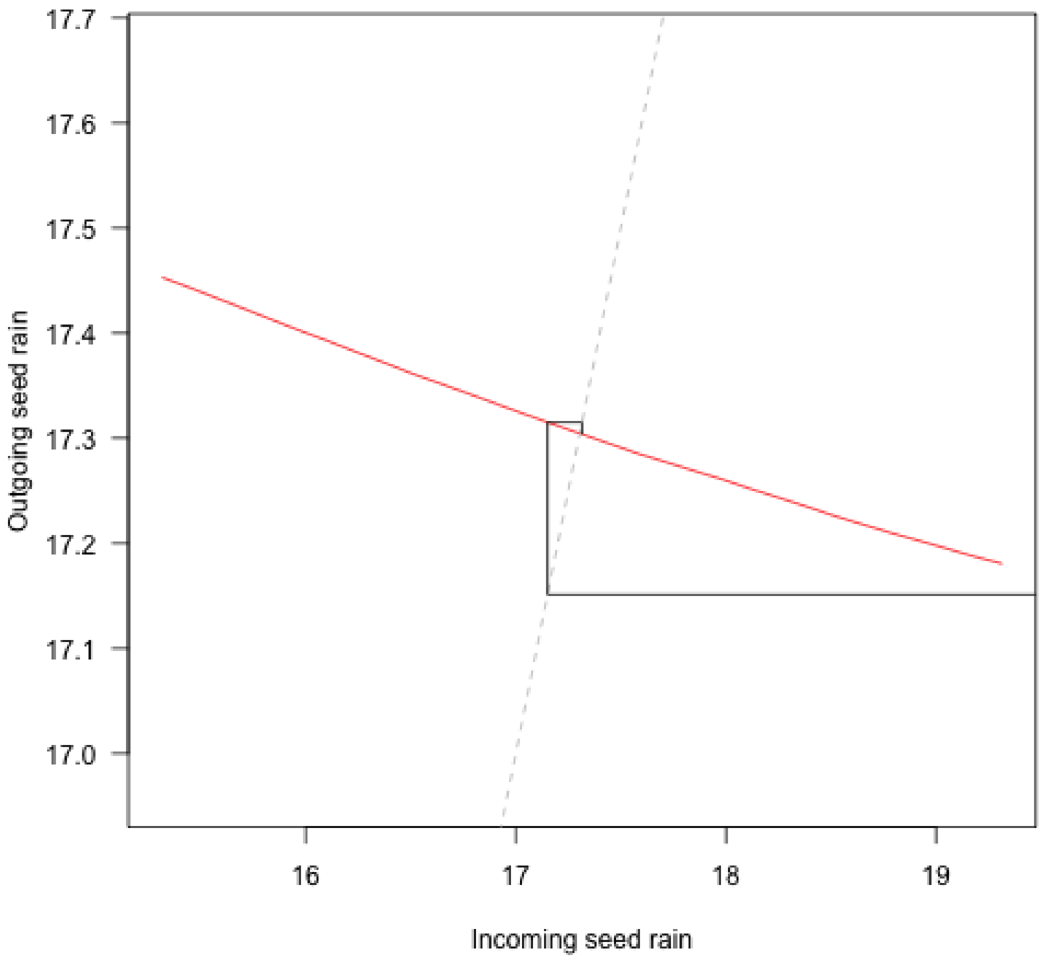
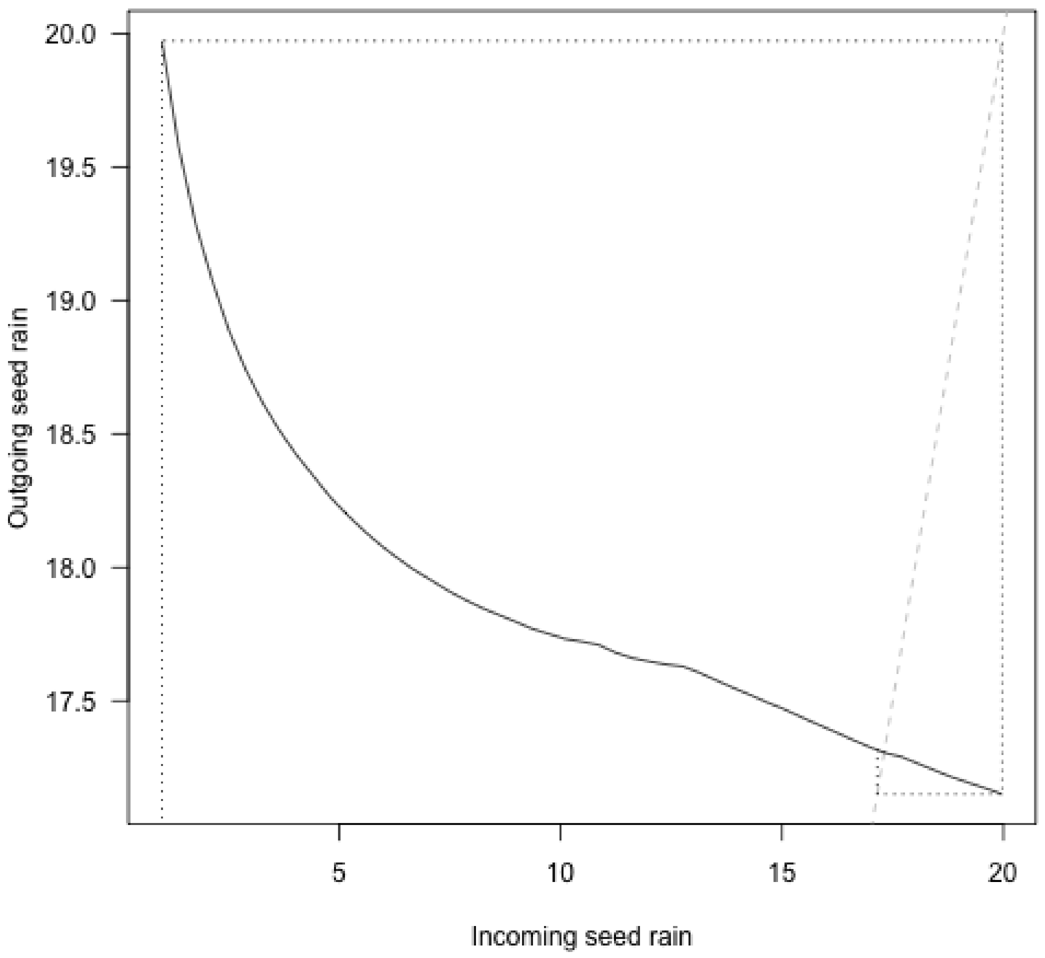

17.7
17.6
17.5
17.4
17.3
17.2
17.1
17.0
Outgoing seed rain
16
17
18
19
Incoming seed rain

Figure 23



20.0
19.5
19.0
18.5
18.0
17.5
Outgoing seed rain
5
10
15
20
Incoming seed rain

Figure 24

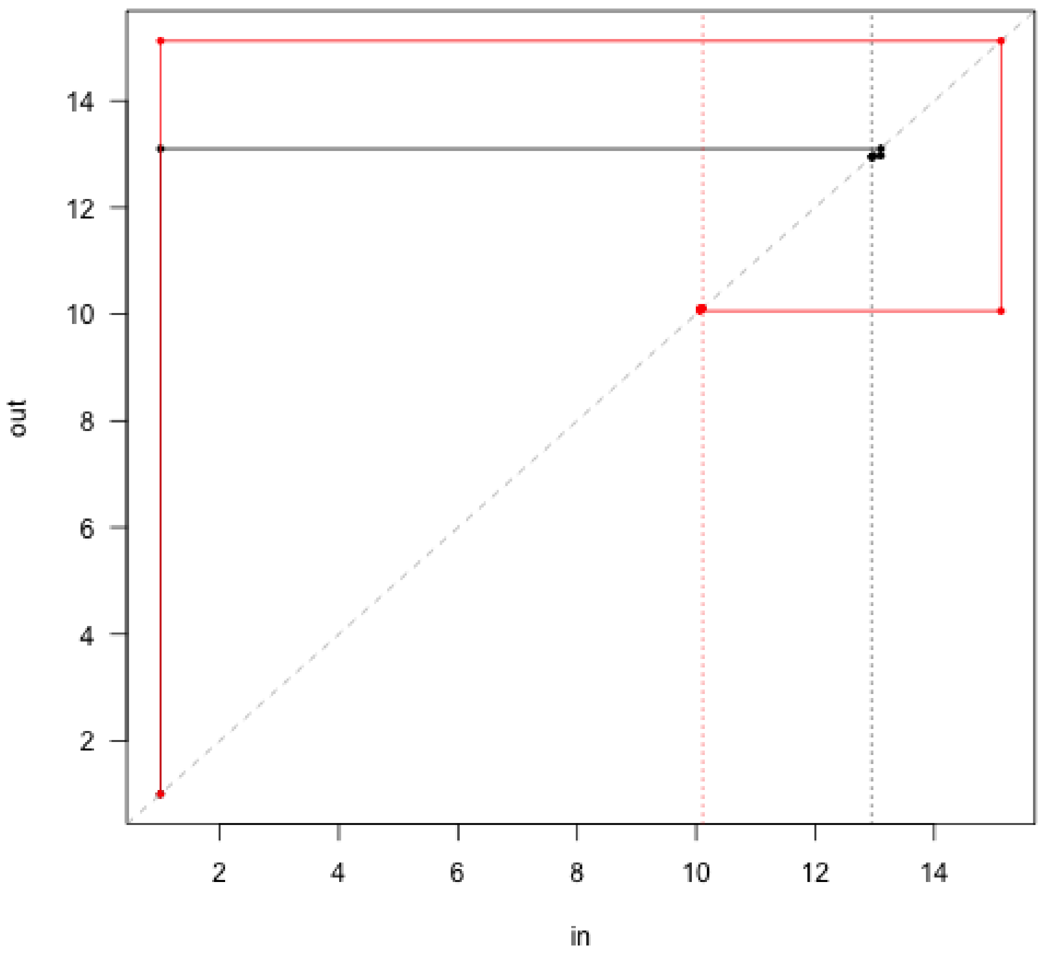


Figure 25

Note that the first guess position of the red species is higher than the black species, but in the end the output seed rain is lower. This is the difficulty in computing multi species equilibria - the different solutions affect each other. In general multi-dimensional root finding is difficult; even knowing that there are roots is not straightforward, much less proving that we've converged on the "correct" root (for example [0,0] is a root in this case but that root is not stable). plant uses some heuristics to try to ensure that the root returned is an attracting point but sequentially applying rounds of iteration and non-linear root finding algorithms, as well as rescaling seed rains to repel from known unstable roots.

To illustrate this a little further, though still in the fairly trivial 2d case, first identify the other two equilibria.

```
pp <- mclapply(lma, function(x)
  equilibrium_seed_rain(expand_parameters(trait_matrix(x, "lma"), p0, FALSE)))
```

Here's the seed rains of each species when alone:

```
seed_rain1 <- sapply(pp, function(x) x$seed_rain)
```

So that means that we have *four* equilibria: 1: The trivial equilibrium:

```
eq00 <- c(0, 0)
run(eq00, p2_eq) - eq00
```

```
## [1] 0 0
```

2: Species 1 alone

```
eq10 <- c(seed_rain1[[1]], 0)
run_new_schedule(eq10, p2) - eq10
```

```
## [1] -0.009399603  0.000000000
```

3: Species 2 alone

```
eq01 <- c(0, seed_rain1[[2]])
run_new_schedule(eq01, p2) - eq01
```

```
## [1] 0.000000000 0.007976816
```

4: Species 1 and 2 together:

```
eq11 <- p2_eq$seed_rain
run(eq11, p2_eq) - eq11
```

```
## [1] 0.0011239196 0.0003123585
```

(note that the approximations here mean that these equilibria are not terribly well polished - there are a set of nested approximations that make this difficult. Possibly the biggest culprit is the cohort refinement step).

```
len <- 21
dx <- max(seed_rain1) / (len - 1)
n1 <- seq(0.001, by=dx, to=seed_rain1[[1]] + dx)
n2 <- seq(0.001, by=dx, to=seed_rain1[[2]] + dx)
nn_in <- as.matrix(expand.grid(n1, n2))
tmp <- mclapply(unname(split(nn_in, seq_len(nrow(nn_in)))),
                run_new_schedule, p2_eq)
nn_out <- do.call("rbind", tmp)

len <- log(rowSums(sqrt((nn_out - nn_in)^2)))
rlen <- len / max(len) * dx * 0.8

theta <- atan2(nn_out[, 2] - nn_in[, 2], nn_out[, 1] - nn_in[, 1])

x1 <- nn_in[, 1] + rlen * cos(theta)
y1 <- nn_in[, 2] + rlen * sin(theta)
```

NOTE: I'm not sure why the point really close to the equilibrium here looks like it's moving so quickly, and possibly in the wrong direction. Cohort instability perhaps?

```
plot(nn_in, xlab="Species 1", ylab="Species 2", pch=19, cex=.25,
     col="grey", asp=1)
arrows(nn_in[, 1], nn_in[, 2], x1, y1, length=0.02)
lines(rbind(approach[1, 1:2], approach[, 3:4]), type="o", col="red",
      pch=19, cex=.5)
points(p2_eq$seed_rain[[1]], p2_eq$seed_rain[[2]], pch=19)
points(rbind(eq00, eq10, eq01))
```

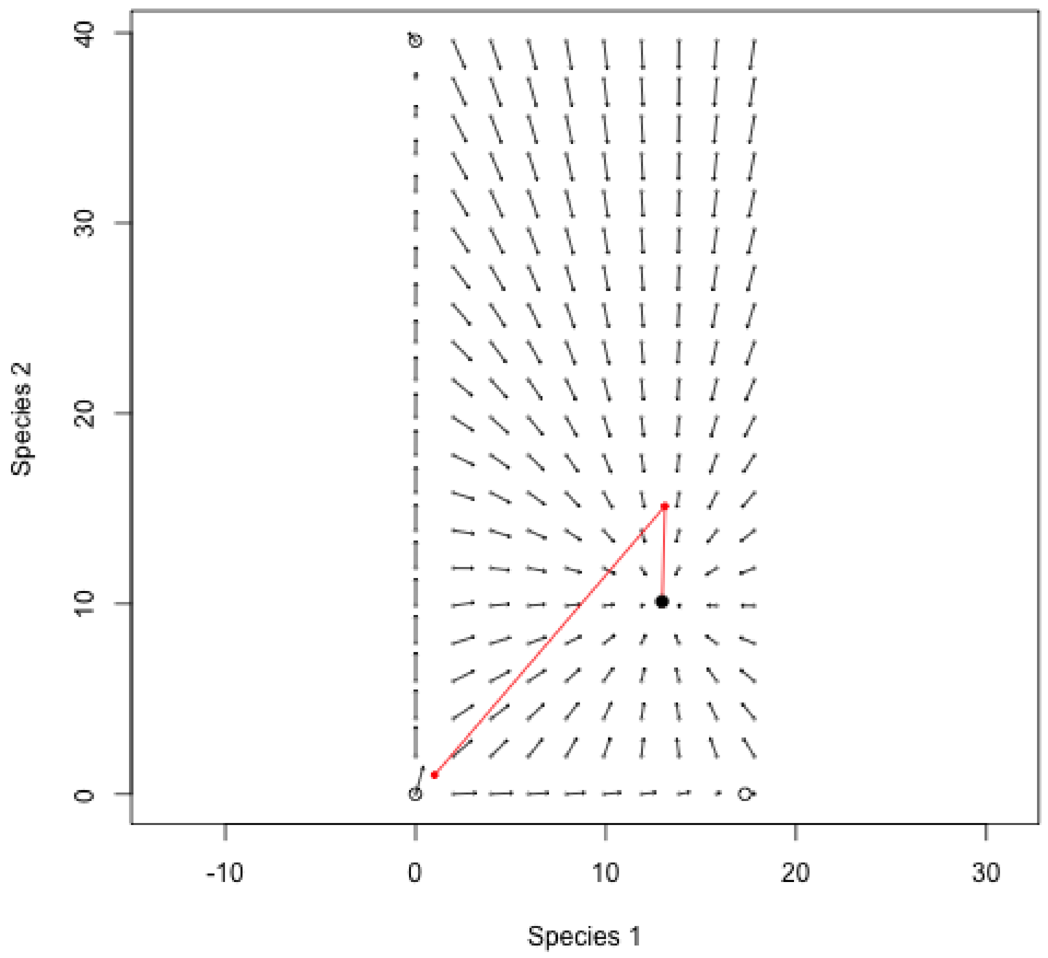


Figure 26

5 PATCH-LEVEL DYNAMICS

The aim here is to use plant to investigate dynamics of plants within a single patch.

```
library(plant)
```

```
p0 <- scm_base_parameters("FF16")
p0$control$equilibrium_nsteps <- 30
p0$control$equilibrium_solver_name <- "hybrid"
p0$disturbance_mean_interval <- 30.0
```

First, with a single species:

```
p1 <- expand_parameters(trait_matrix(0.0825, "lma"), p0, FALSE)
```

Run everything out to equilibrium:

```
p1_eq <- equilibrium_seed_rain(p1)
```

This collects information about the state of the system at every ODE step:

```
data1 <- run_scm_collect(p1_eq)
```

Entries are: * time: time of the step * species: a list with the state of each species (in this case there is only 1). The contents is a 3d array of variable / time step / cohort * light_env: a list with the points of the light environment spline (height / canopy openness) * seed_rain: output seed rain * p: input parameters

```
names(data1)
```

```
## [1] "time"          "species"       "light_env"     "seed_rain"
## [5] "patch_density" "p"
```

```
t <- data1$time
h <- data1$species[[1]]["height", , ]
```

h here is two of the three dimensions of the state array; each row is a *time* and each column is a *cohort*.

```
dim(h)
```

```
## [1] 214 213
```

So the ith row is the height of all extant cohorts in the patch (some are NA if they have not been introduced yet)

```
h[150, ]
```

```
##   [1] 10.2485992 10.2482823 10.2479654 10.2476483 10.2473311 10.2470138
##   [7] 10.2466964 10.2463789 10.2460613 10.2455764 10.2450913 10.2446059
##  [13] 10.2441202 10.2436342 10.2426616 10.2416879 10.2407131 10.2397372
##  [19] 10.2387603 10.2368034 10.2348421 10.2328766 10.2309067 10.2289326
##  [25] 10.2249714 10.2209928 10.2169967 10.2129830 10.2089516 10.2008353
##  [31] 10.1926469 10.1843857 10.1760509 10.1676416 10.1505967 10.1332448
##  [37] 10.1155797 10.0975952 10.0792849 10.0416630 10.0026648  9.9622402
##  [43]  9.9203426  9.8769319  9.8319750  9.7854454  9.6875796  9.5831993
##  [49]  9.4723566  9.4145938  9.3553453  9.2327455  9.1052512  8.9738718
##  [55]  8.8398917  8.5692273  8.4351001  8.3032636  8.1746287  8.0498992
##  [61]  7.8137910  7.5964892  7.3977024  7.2192089  7.0605443  6.9167563
##  [67]  6.7854018  6.6646825  6.5531909  6.4498147  6.2636984  6.1004274
##  [73]  5.9552741  5.8884229  5.8249981  5.6000828  5.5017418  5.4108713
##  [79]  5.3265873  5.2481566  5.1063563  4.9809935  4.8688142  4.7675043
##  [85]  4.6753646  4.5909377  4.4411792  4.3741009  4.3113747  4.1968742
##  [91]  4.0949065  4.0029292  3.9194395  3.7720448  3.6447932  3.5323634
##  [97]  3.4312027  3.3391155  3.1766769  3.0338193  2.9044425  2.7851320
## [103]  2.5713541  2.3754702  2.1919526  2.0198052  1.8596988  1.7156674
## [109]  1.5933398  1.4982955  1.4345562  1.4026839  1.3963969  1.3929998
## [115]  1.3863269  1.3819468  1.3769108  1.3713405  1.3653471  1.3589821
## [121]  1.3556643  1.3522573  1.3451835  1.3377756  1.3300350  1.3219608
## [127]  1.3135696  1.3048491  1.2957666  1.2863015  1.2764493  1.2662086
## [133]  1.2555801  1.2445605  1.2331284  1.2212655  1.2089375  1.1961458
## [139]  1.1692843  1.1408658  1.1108472  1.0790846  1.0111509  0.9381607
## [145]  0.8610027  0.7810595  0.6190055  0.5394430  0.4631125  0.3920458
## [151]         NA         NA         NA         NA         NA         NA
## [157]         NA         NA         NA         NA         NA         NA
## [163]         NA         NA         NA         NA         NA         NA
## [169]         NA         NA         NA         NA         NA         NA
## [175]         NA         NA         NA         NA         NA         NA
## [181]         NA         NA         NA         NA         NA         NA
## [187]         NA         NA         NA         NA         NA         NA
## [193]         NA         NA         NA         NA         NA         NA
## [199]         NA         NA         NA         NA         NA         NA
## [205]         NA         NA         NA         NA         NA         NA
## [211]         NA         NA         NA
```

And the jth column is the heights of a particular cohort, NA if that time is before it was introduced:

```
h[, 60]
```

```
##   [1]         NA         NA         NA         NA         NA         NA
##   [7]         NA         NA         NA         NA         NA         NA
##  [13]         NA         NA         NA         NA         NA         NA
##  [19]         NA         NA         NA         NA         NA         NA
##  [25]         NA         NA         NA         NA         NA         NA
##  [31]         NA         NA         NA         NA         NA         NA
##  [37]         NA         NA         NA         NA         NA         NA
##  [43]         NA         NA         NA         NA         NA         NA
##  [49]         NA         NA         NA         NA         NA         NA
##  [55]         NA         NA         NA         NA         NA  0.3920458
##  [61]  0.3940845  0.3961303  0.3981832  0.4002432  0.4023102  0.4043843
##  [67]  0.4064655  0.4085537  0.4106491  0.4127515  0.4169774  0.4212316
##  [73]  0.4255141  0.4276659  0.4298247  0.4385306  0.4429258  0.4473490
##  [79]  0.4518004  0.4562798  0.4653227  0.4744774  0.4837435  0.4931208
##  [85]  0.5026088  0.5122071  0.5317328  0.5416593  0.5516942  0.5720869
##  [91]  0.5929062  0.6141470  0.6358040  0.6803435  0.7264755  0.7741473
##  [97]  0.8233033  0.8738852  0.9790879  1.0892687  1.2039391  1.3226190
## [103]  1.5701667  1.8284238  2.0942049  2.3645807  2.6368113  2.9083260
## [109]  3.1767522  3.4400192  3.6964981  3.9451054  4.0662844  4.1853367
## [115]  4.4170927  4.5298461  4.6405712  4.7493074  4.8561088  4.9610261
## [121]  5.0127900  5.0640990  5.1653609  5.2648436  5.3625902  5.4586346
## [127]  5.5529997  5.6457097  5.7367767  5.8262163  5.9140329  6.0002406
## [133]  6.0848397  6.1678365  6.2492409  6.3290463  6.4072349  6.4837859
## [139]  6.6319101  6.7733356  6.9079253  7.0354559  7.2683888  7.4702753
## [145]  7.6394324  7.7749838  7.9506277  7.9990849  8.0298924  8.0498992
## [151]  8.0651195  8.0804571  8.0995138  8.1246074  8.1571601  8.1980377
## [157]  8.2477239  8.3064331  8.3742018  8.4509581  8.5365665  8.6308407
## [163]  8.7335586  8.9633063  9.2235945  9.5120416  9.8261846 10.1635309
## [169] 10.5215634 10.8977160 11.2894251 11.6940290 12.1086542 12.5301917
## [175] 13.3806883 14.2194470 15.0235590 15.7609024 16.3438806 16.5495043
## [181] 16.7055677 16.8254066 16.9198837 17.0602076 17.1614822 17.2401404
## [187] 17.3046010 17.3595142 17.4075158 17.4501098 17.4881937 17.5223569
## [193] 17.5530716 17.5808373 17.6062259 17.6297792 17.6518790 17.6727348
## [199] 17.6924404 17.7110416 17.7285831 17.7451385 17.7608150 17.7757472
## [205] 17.7900447 17.8037763 17.8169733 17.8296591 17.8418518 17.8535793
## [211] 17.8648844 17.8758038 17.8863635 17.8931463
```

With a single species there is always one more time step than cohort introduction, with the last two steps having the same number of cohorts. With multiple species there can be more time steps than cohort introductions, as we'll record data every cohort introduction for *either* species, which will be refined to different schedules.

Individual plants increase in height with respect to time, but because they are competing the growth rate depends on the amount of shading above them, in addition to size dependent growth rate (see vignette:plant)

```r
matplot(t, h, lty=1, col=make_transparent("black", 0.25), type="l",
        las=1, xlab="Time (years)", ylab="Height (m)")
```

The light environment is stored over each time step:

```r
xlim <- c(0, 1.1)
ylim <- range(data1$light_env[[length(data1$light_env)]][, "height"])
plot(NA, xlim=xlim, ylim=ylim, las=1,
     xlab="Canopy openness", ylab="Height (m)")
for (i in data1$light_env) {
  lines(i[, "canopy_openness"], i[, "height"], col="grey")
}

blues <- c("#DEEBF7", "#C6DBEF", "#9ECAE1", "#6BAED6",
```

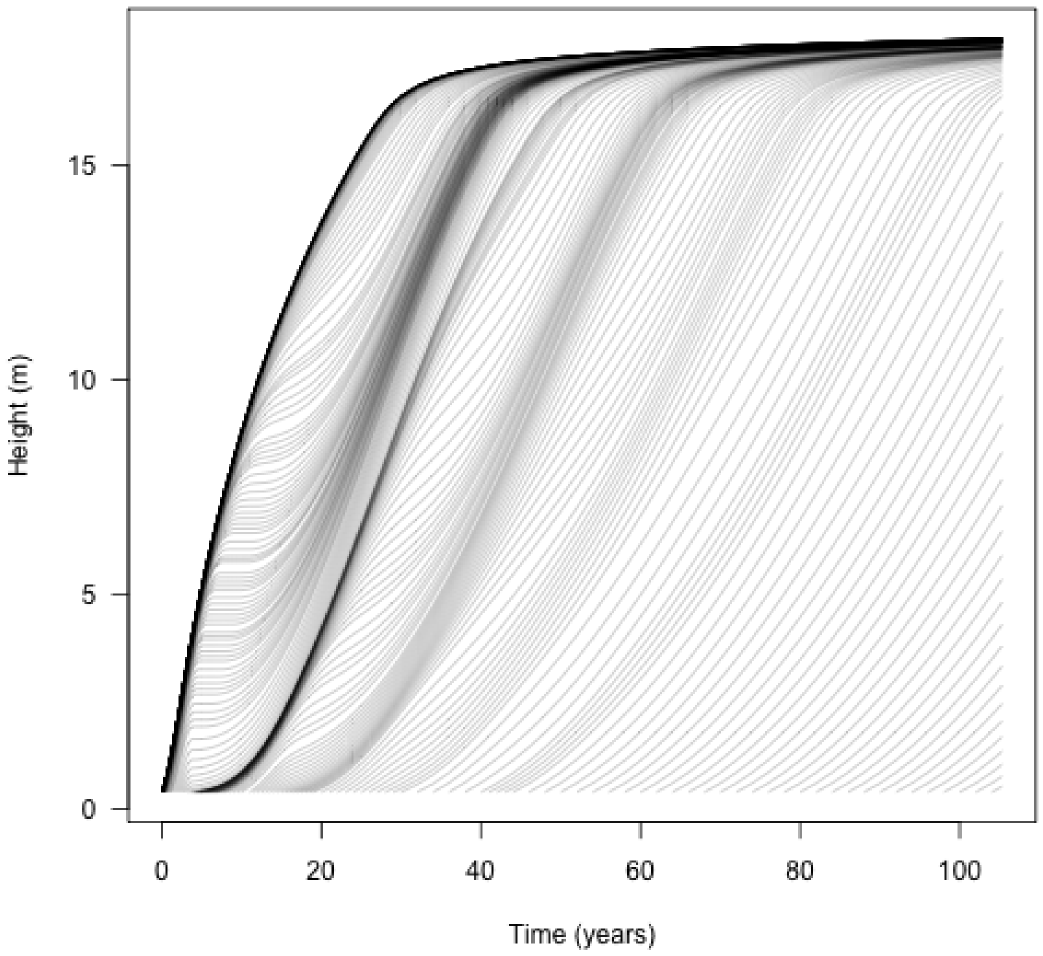


Figure 27

```
           "#4292C6", "#2171B5", "#08519C", "#08306B")
times <- c(5, 10, 20, 40, data1$time[[length(data1$time)]])
cols <- colorRampPalette(blues[-(1:2)])(length(times))
for (i in seq_along(times)) {
  x <- data1$light_env[[which.min(abs(times[[i]] - data1$time))]]
  lines(x[, "canopy_openness"], x[, "height"], col=cols[[i]])
  y <- x[nrow(x), "height"]
  points(1, y, pch=19, col=cols[[i]])
  text(1 + strwidth("x"), y, paste(round(times[[i]]), "years"),
       adj=c(0, 0))
}
```

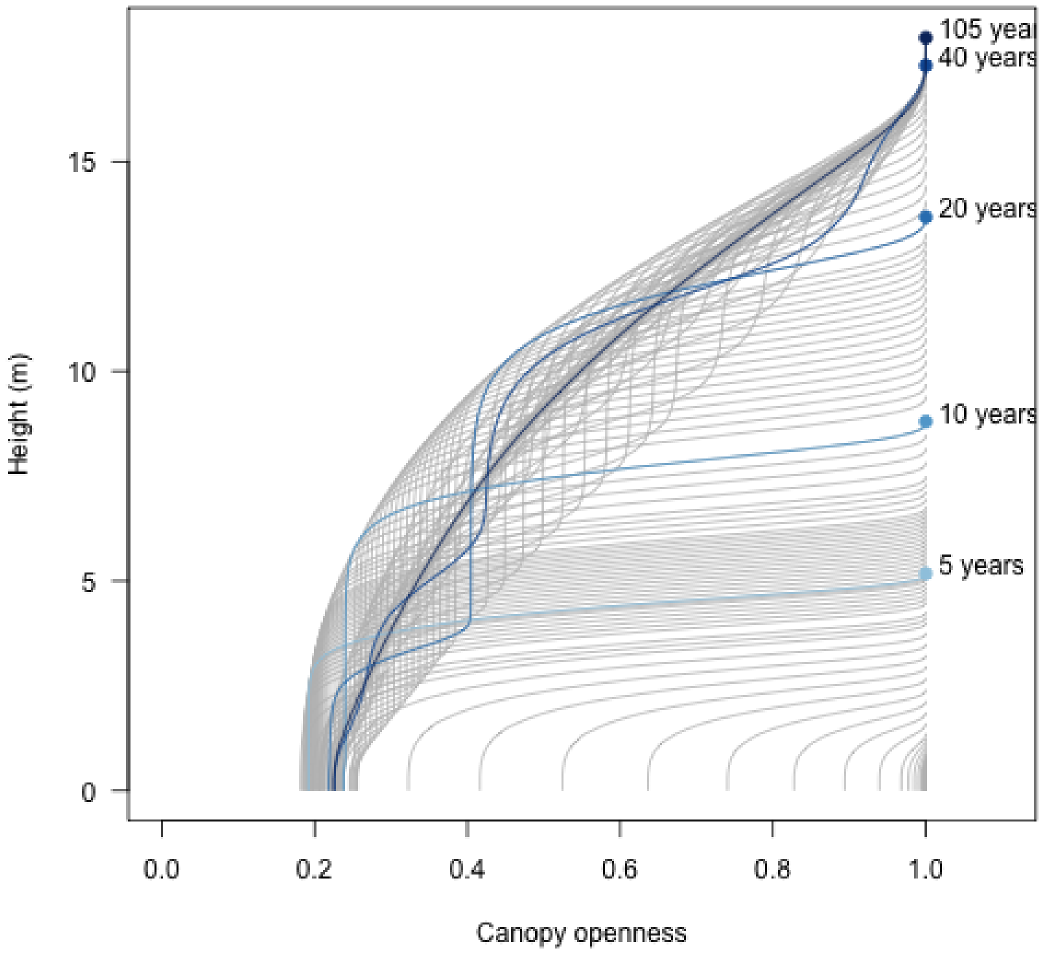


Figure 28

The amount of light at the ground level is perhaps the most relevant metric:

```
y <- sapply(data1$light_env, function(x) x[1, "canopy_openness"])
plot(data1$time, y, type="l", las=1,
     ylim=c(0, 1), xlab="Time (years)", ylab="Canopy openness")
```

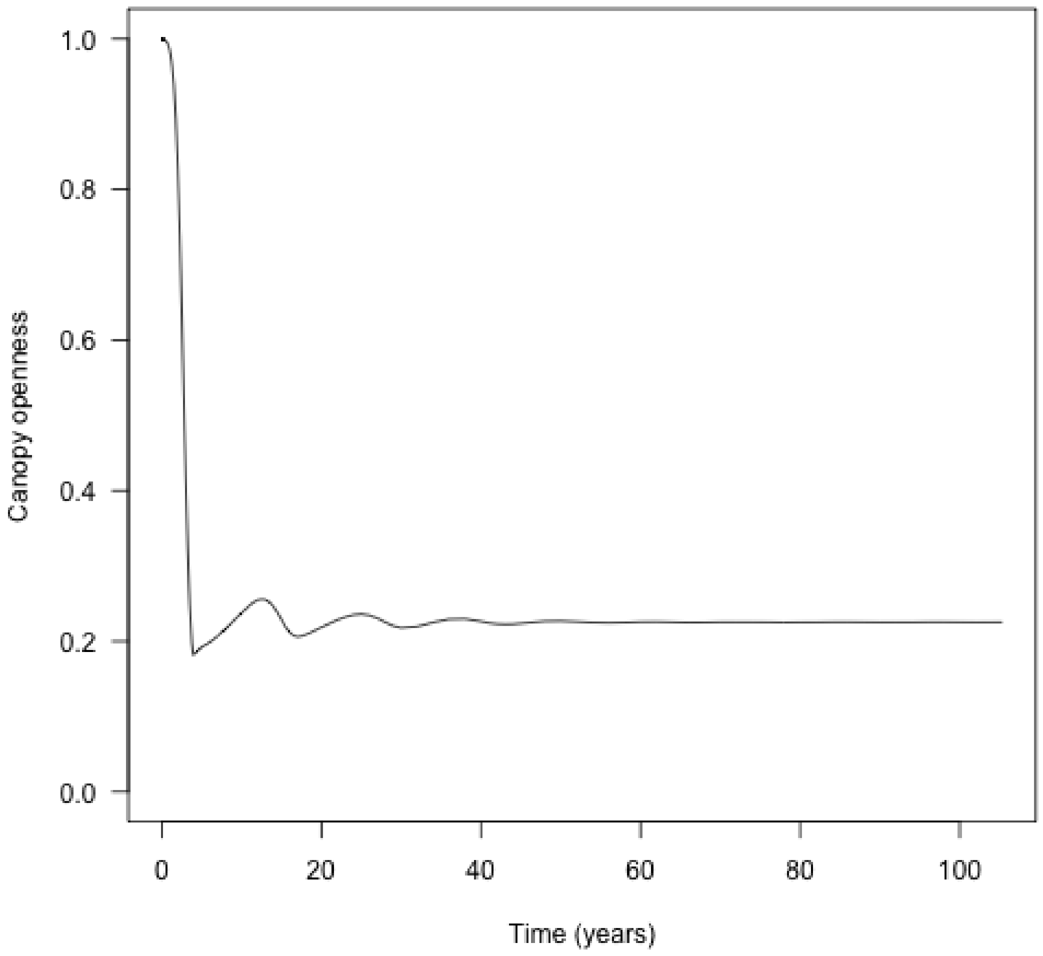


FIGURE 29

The waves here are due to rounds of recruitment and self thinning. Mortality is not instantaneous, so species recruit to a density that generates a canopy that they cannot survive under.

Leaf area index is the driver that controls the canopy openness (via the light extinction coefficient `p1$k_I`, following exponential extinction). This is not returned by `run_scm_collect` so instead we need to rebuild patches using `scm_patch`.

```
patches1 <- lapply(seq_along(data1$time), scm_patch, data1)
```

Each element of the resulting list is a Patch object, the same as was observed when running the model.

```
lai1 <- sapply(patches1, function(x) x$area_leaf_above(0.0))
plot(data1$time, lai1, type="l", las=1, xlab="Time (years)",
     ylab="Leaf area index")
```

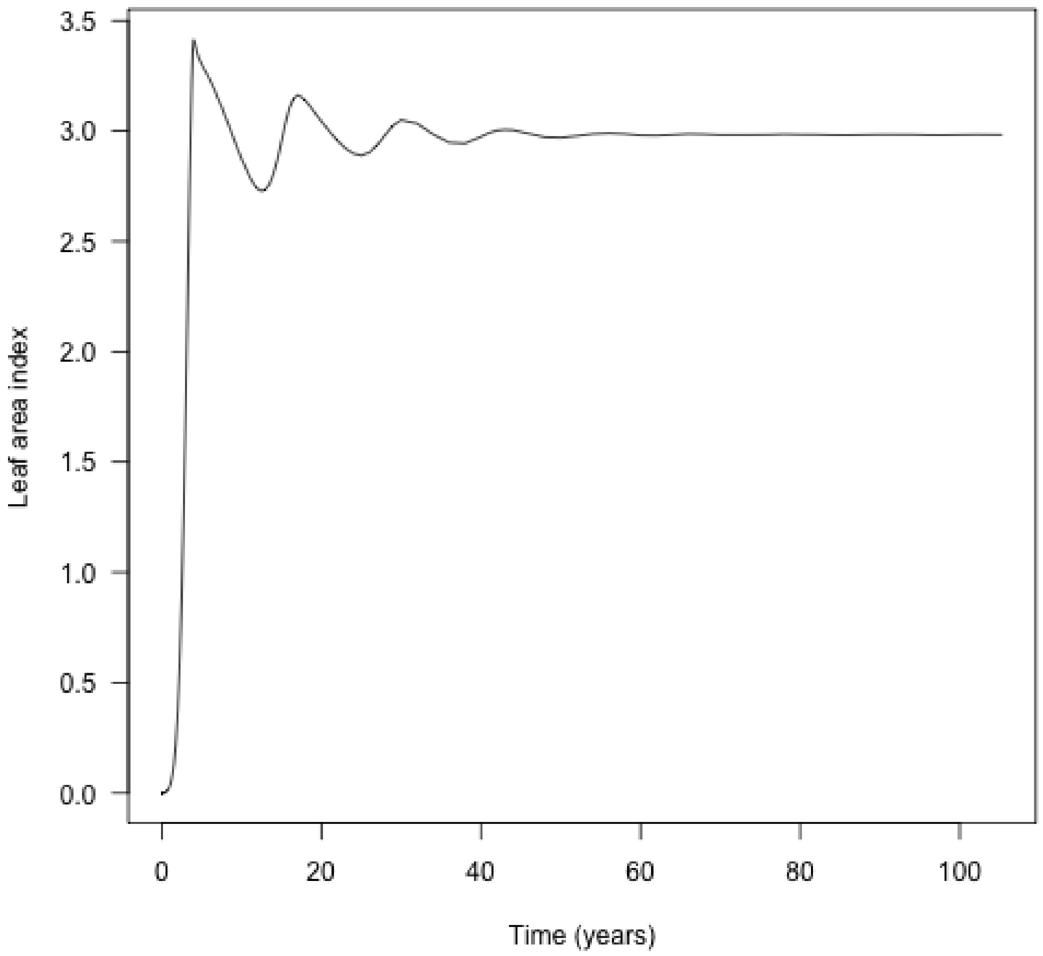


FIGURE 30

If multiple species are grown at once, they compete with one another. This adds a second species – with a higher lma value than the first species – to the population and runs them until they reach equilibrium seed rain at the metapopulation level.

```
p2 <- expand_parameters(trait_matrix(0.2625, "lma"), p1, FALSE)
p2_eq <- equilibrium_seed_rain(p2)
```

Then collect the patch-level dynamics:

```
data2 <- run_scm_collect(p2_eq)

t2 <- data2$time
h1 <- data2$species[[1]]["height", , ]
h2 <- data2$species[[2]]["height", , ]

cols <- c("#e34a33", "#045a8d")
matplot(t2, h1, lty=1, col=make_transparent(cols[[1]], .25), type="l",
        las=1, xlab="Time (years)", ylab="Height (m)")
matlines(t2, h2, lty=1, col=make_transparent(cols[[2]], .25))
```

Alternatively we can compare the growth of species 1 by itself or with another species:

```
matplot(t, h, lty=1, col=make_transparent("black", .25), type="l",
        las=1, xlab="Time (years)", ylab="Height (m)")
matlines(t2, h1, lty=1, col=make_transparent(cols[[1]], .25))
```

This shows that the additional species does not affect the growth of the *initial* wave of cohorts (because the second species is growing more slowly and is shorter than the first species), but subsequent waves are slowed or eliminated.

The dynamics are easier to see when coded by cohort density (some of the lines here represent cohorts at close to zero density).

Relativise the log densities onto (-4, max)

```
d1 <- data2$species[[1]]["log_density", , ]
d2 <- data2$species[[2]]["log_density", , ]
rel <- function(x, xmin) {
  x[x < xmin] <- xmin
  xmax <- max(x, na.rm=TRUE)
  (x - xmin) / (xmax - xmin)
}
rd1 <- rel(d1, -4)
rd2 <- rel(d2, -4)
```

R doesn't seem to offer a way to plot lines that vary in colour, so this is quite roundabout using `segments`, shaded by the density at the first part of the line segment:

```
n <- length(t2)
x <- matrix(rep(t2, ncol(h1)), nrow(h1))
col1 <- matrix(make_transparent(cols[[1]], rd1), nrow(d1))
col2 <- matrix(make_transparent(cols[[2]], rd2), nrow(d2))
plot(NA, xlim=range(t2), ylim=range(h1, na.rm=TRUE),
     las=1, xlab="Time (years)", ylab="Cohort height (m)")
segments(x[-1, ], h2[-1, ], x[-n, ], h2[-n, ], col=col2[-n, ], lend="butt")
segments(x[-1, ], h1[-1, ], x[-n, ], h1[-n, ], col=col1[-n, ], lend="butt")
```

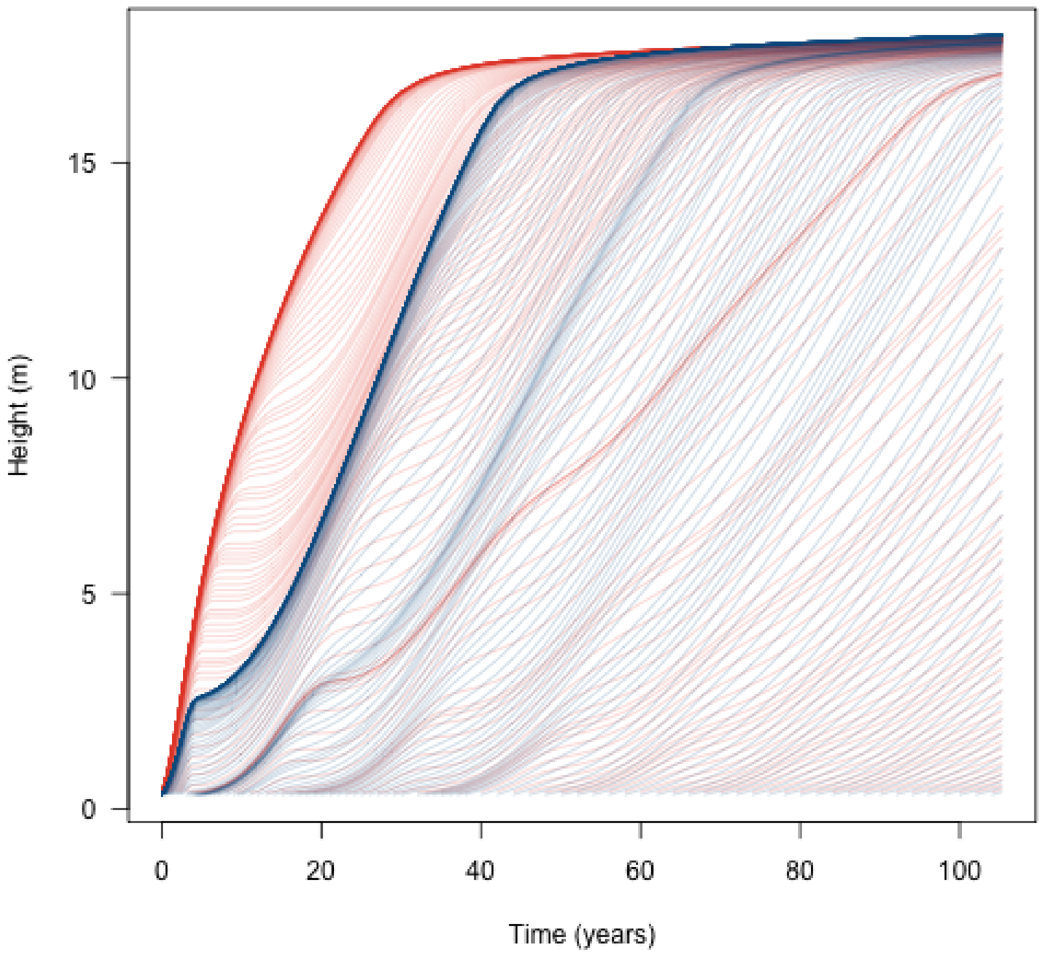


Figure 31

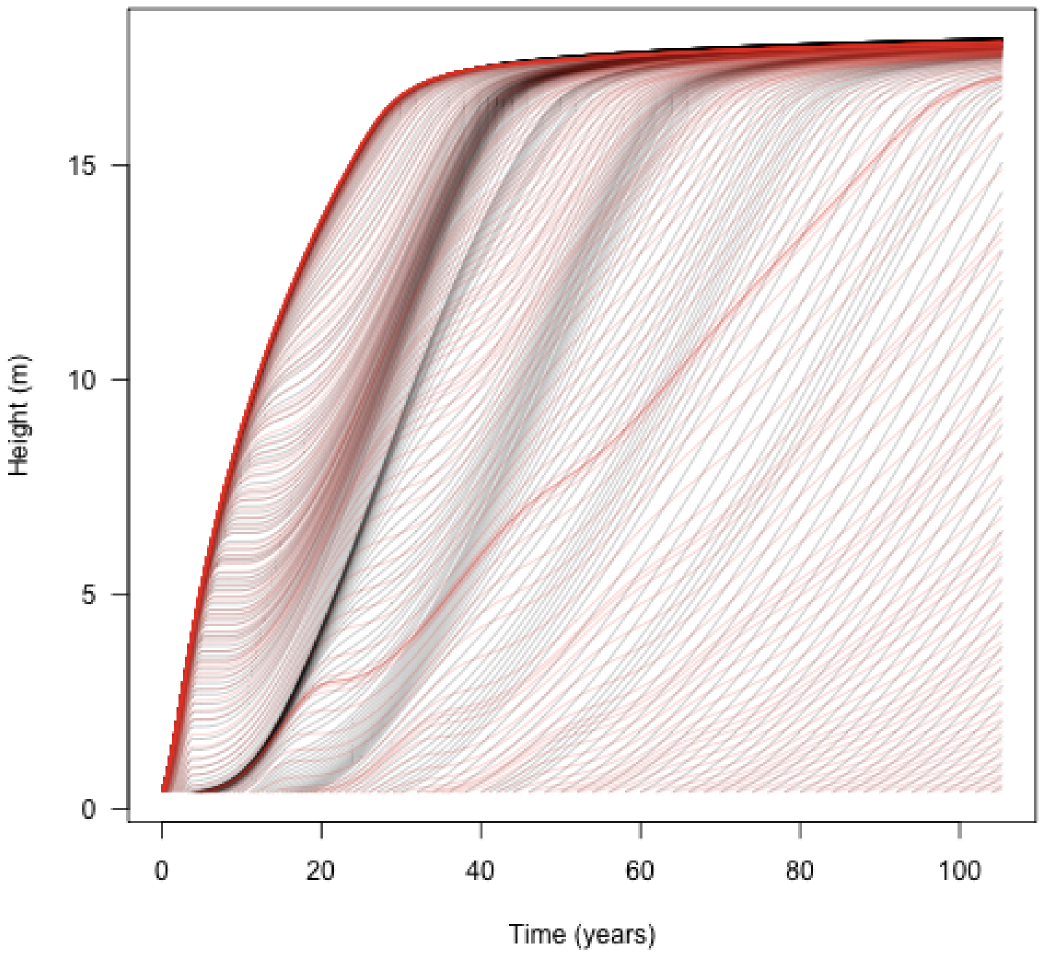


Figure 32

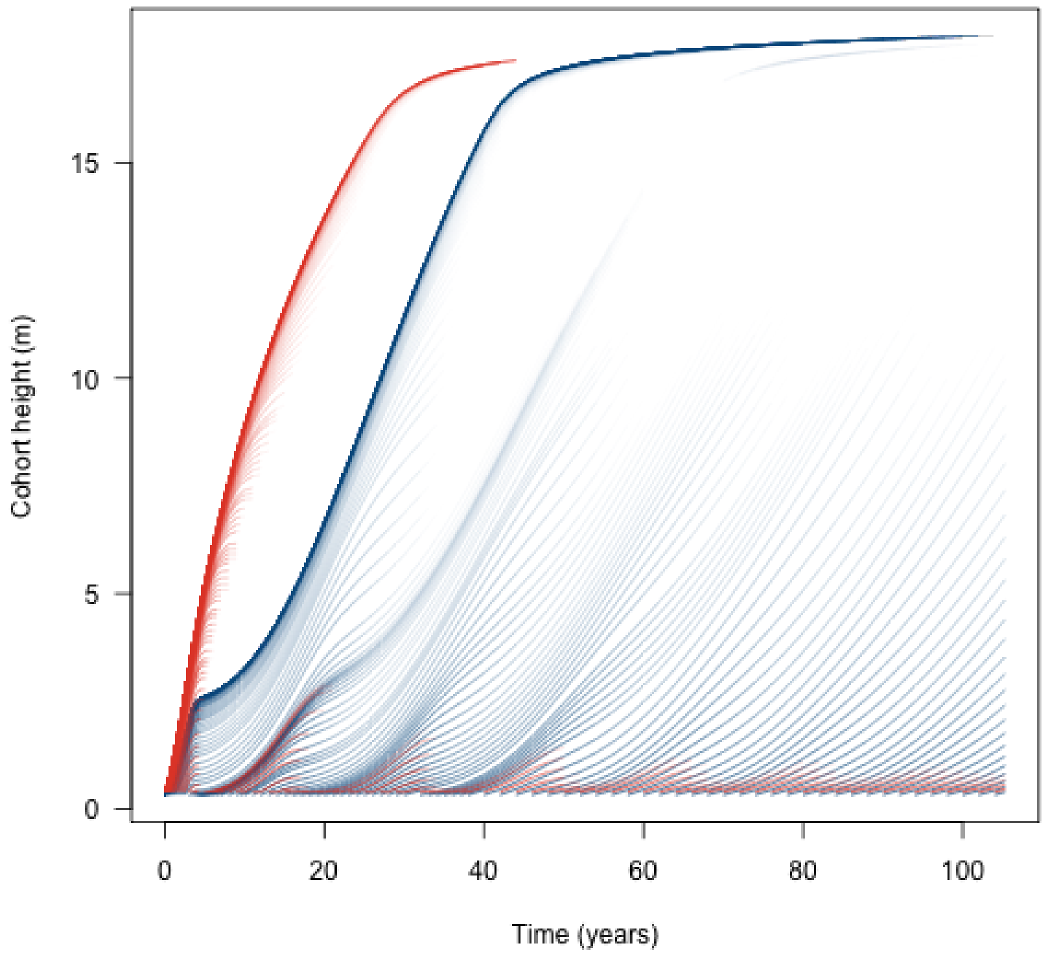


Figure 33

Then, the total leaf area:

Each element of the resulting list is a Patch object, the same as was observed when running the model.

```
patches2 <- lapply(seq_along(data2$time), scm_patch, data2)
lai2 <- sapply(patches2, function(x) x$area_leaf_above(0.0))

lai2_1 <- sapply(patches2, function(x) x$species[[1]]$area_leaf_above(0.0))
lai2_2 <- sapply(patches2, function(x) x$species[[2]]$area_leaf_above(0.0))

plot(t2, lai2, type="l", las=1, lty=2,
     xlab="Time (years)", ylab="Leaf area index")
lines(t2, lai2_1, col=cols[[1]])
lines(t2, lai2_2, col=cols[[2]])
```

To find the average value over the metapopulation we weight by patch abundance:

```
metapopulation <- function(x){
  plant:::trapezium(t2, x*data2$patch_density)
}

lai2_av <- metapopulation(lai2)
lai2_1_av <- metapopulation(lai2_1)
lai2_2_av <- metapopulation(lai2_2)

plot(t2, lai2, type="l", las=1, lty=2,
     xlab="Time (years)", ylab="Leaf area index")
lines(t2, lai2_1, col=cols[[1]])
lines(t2, lai2_2, col=cols[[2]])

axis(4, at=lai2_av,   tck=0.1, lty = 2, labels=NA)
axis(4, at=lai2_1_av, tck=0.1, col.ticks=cols[[1]], labels=NA)
axis(4, at=lai2_2_av, tck=0.1, col.ticks=cols[[1]], labels=NA)
axis(1, at=108, labels="Av")
```

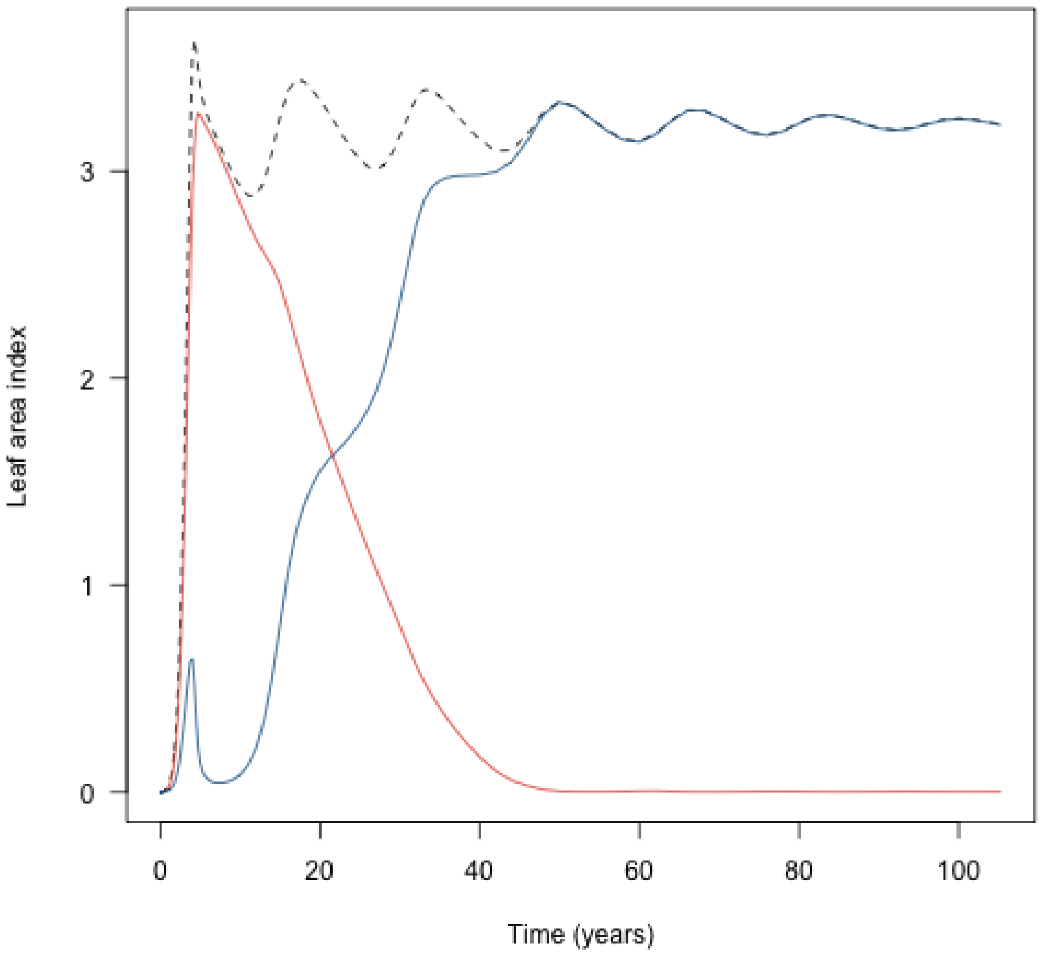


Figure 34

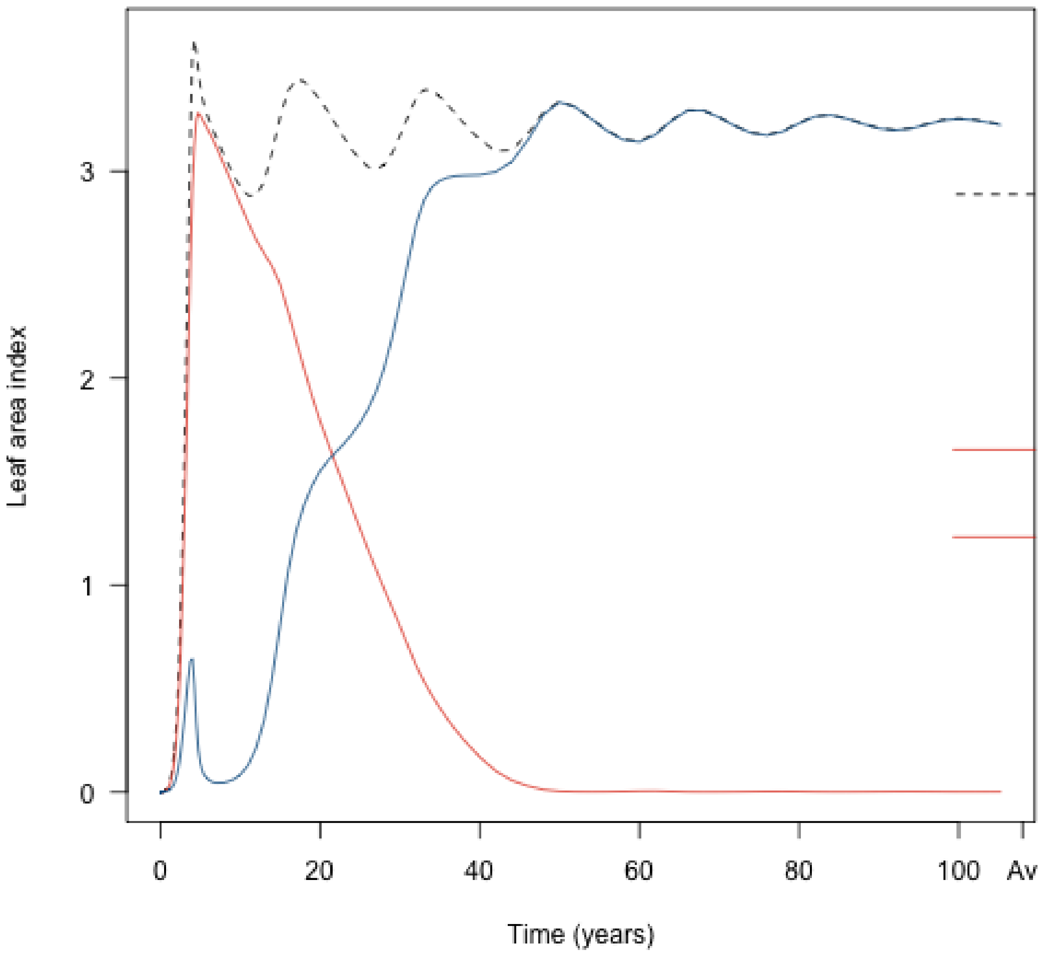


Figure 35

6 PATCH-LEVEL EMERGENT PROPERTIES

The aim here is to use the plant package to investigate dynamics within a patch of competing plants, focussing on emergent patch-level properties, rather than properties of plants within the patch.

```
library(plant)

p0 <- scm_base_parameters("FF16")
p0$control$equilibrium_nsteps <- 30
p0$control$equilibrium_solver_name <- "hybrid"
p0$disturbance_mean_interval <- 30.0
```

We'll work with a single species at equilibrium

```
p1 <- expand_parameters(trait_matrix(0.0825, "lma"), p0, FALSE)
p1_eq <- equilibrium_seed_rain(p1)
data1 <- run_scm_collect(p1_eq)
```

There was a bit of hassle in vignette:patch in plotting all trajectories along with a couple of lines corresponding to focal years. We're going to do the same thing here:

```
closest <- function(t, time) {
  which.min(abs(time - t))
}
last <- function(x) {
  x[[length(x)]]
}

times <- c(5, 10, 20, 40, last(data1$time))
i <- vapply(times, closest, integer(1), data1$time)
blues <- c("#DEEBF7", "#C6DBEF", "#9ECAE1", "#6BAED6",
           "#4292C6", "#2171B5", "#08519C", "#08306B")
cols <- colorRampPalette(blues[-(1:2)])(length(i))

height      <- t(data1$species[[1]]["height", , ])
log_density <- t(data1$species[[1]]["log_density", , ])
```

As with height in vignette:patch, the density is a matrix of time and cohort identity. However, to show the profiles for a given time slice using `matplot` it is convenient to transpose this matrix (`matplot` works column-by-column so we want each column to to be the state of the system at a given time)

```
density <- exp(log_density)
matplot(height, density, type="l", lty=1,
        col=make_transparent("black", 0.15),
        xlab="Height (m)", ylab="Density (1 / m / m2)", las=1,
        log="y")
```

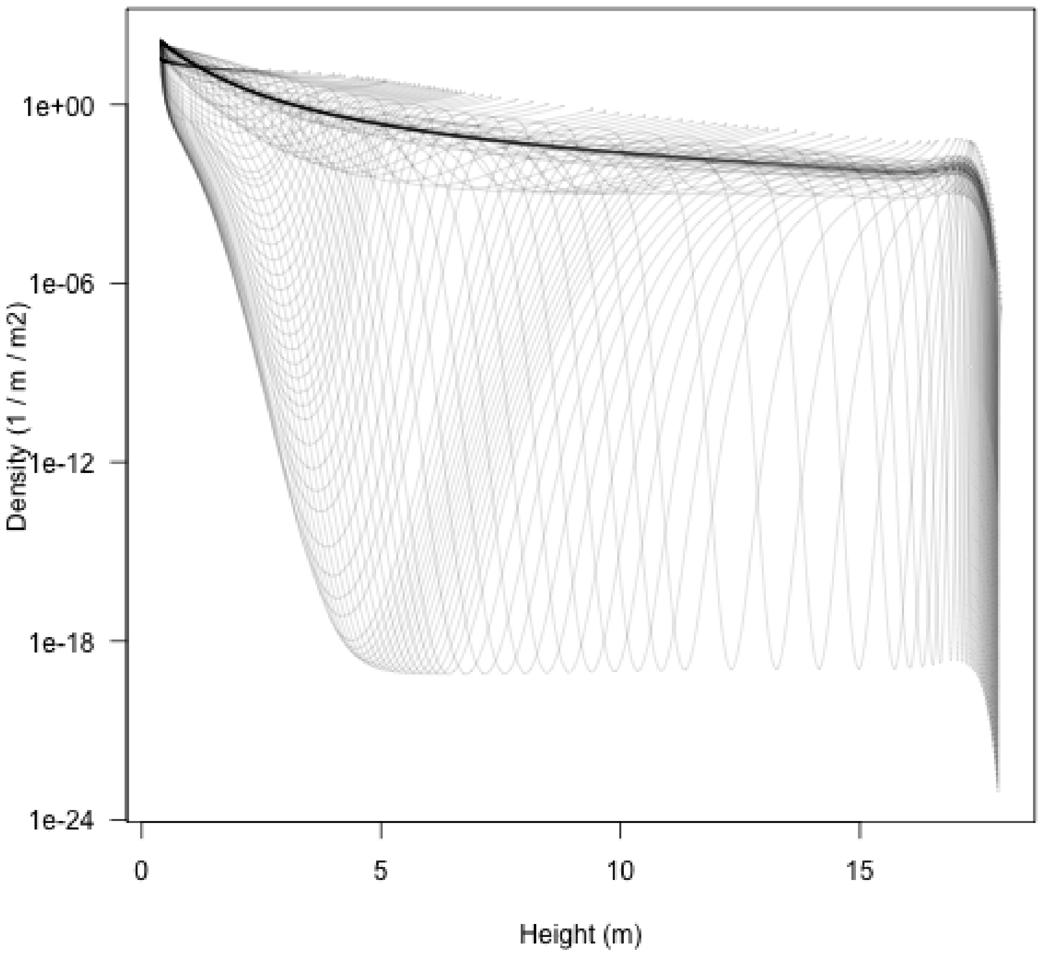


Figure 36

Note that the densities here can be *extremely* low, and yet individuals within the cohorts continue to grow in size; this is the "atto-fox problem", though here we drop down as low as 9 yocto plants / m / metre squared (a yocto-x being 1 millionth of an atto-x). *Anyway*. . . .

The trajectories are easier to understand if a few are highlighted.

```
xlim <- c(0, max(height, na.rm=TRUE) * 1.05)
matplot(height, density, type="l", lty=1,
        col=make_transparent("black", 0.15),
        xlab="Height (m)", ylab="Density (1 / m / m2)", las=1,
        log="y", xlim=xlim)
matlines(height[, i], density[, i], col=cols, lty=1, type="l")
points(height[1, i], density[1, i], pch=19, col=cols)
text(height[1, i] + strwidth("x"), density[1, i],
     paste0(round(times), c(" years", rep("", length(times) - 1))),
     adj=c(0, 0))
```

Early on there is a low density "dip" caused by self thinning (5 years). As the stand develops that dip broadens and deepens and moves forward in height (these very low density cohorts are still travelling the characteristic equations of the SCM). Later on (20 years) additional an second wave of recruitment gives a second pulse of high density (around 4 m tall), which can be seen travelling along in the 40 year profile to about 13 m. When the patch is very old the stand approaches a stable age distribution, though a very narrow window of "missing" heights exists just below the top of the canopy.

It's also possible see where the leaf area in the patch is coming from; a profile of leaf area with respect to height. Again, this requires reconstructing the patches, and using an unexported function from `plant` to put this into a matrix:

```
patch <- lapply(seq_along(data1$time), scm_patch, data1)
leaf_area <- lapply(patch, function(x) x$species[[1]]$area_leafs)
leaf_area <- do.call("cbind", plant:::pad_matrix(leaf_area))

matplot(height, leaf_area, type="l", lty=1, col="lightgrey",
        xlim=xlim, xlab="Height (m)",
        ylab="Leaf area density (m2 / m2 / m)", las=1)
matlines(height[, i], leaf_area[, i], col=cols, lty=1, type="l")
points(height[1, i], leaf_area[1, i], pch=19, col=cols)
text(height[1, i] + strwidth("x"), leaf_area[1, i],
     paste0(round(times), c(" years", rep("", length(times) - 1))),
     adj=c(0, 0))


ode_size <- patch[[1]]$species[[1]]$seed$ode_size
growth_rate <- lapply(patch, function(x)
                     matrix(x$species[[1]]$ode_rates, ode_size)[1, ])
growth_rate <- do.call("cbind", plant:::pad_matrix(growth_rate))
```

Finally, we can see where height growth rate is concentrated in the population. This differs from the profile in vignette:plant because it is reduced depending on the light environment, but that environment is the one constructed by the plant itself.

```
matplot(height, growth_rate, type="l", lty=1, col="lightgrey",
        xlim=xlim, xlab="Height (m)",
```

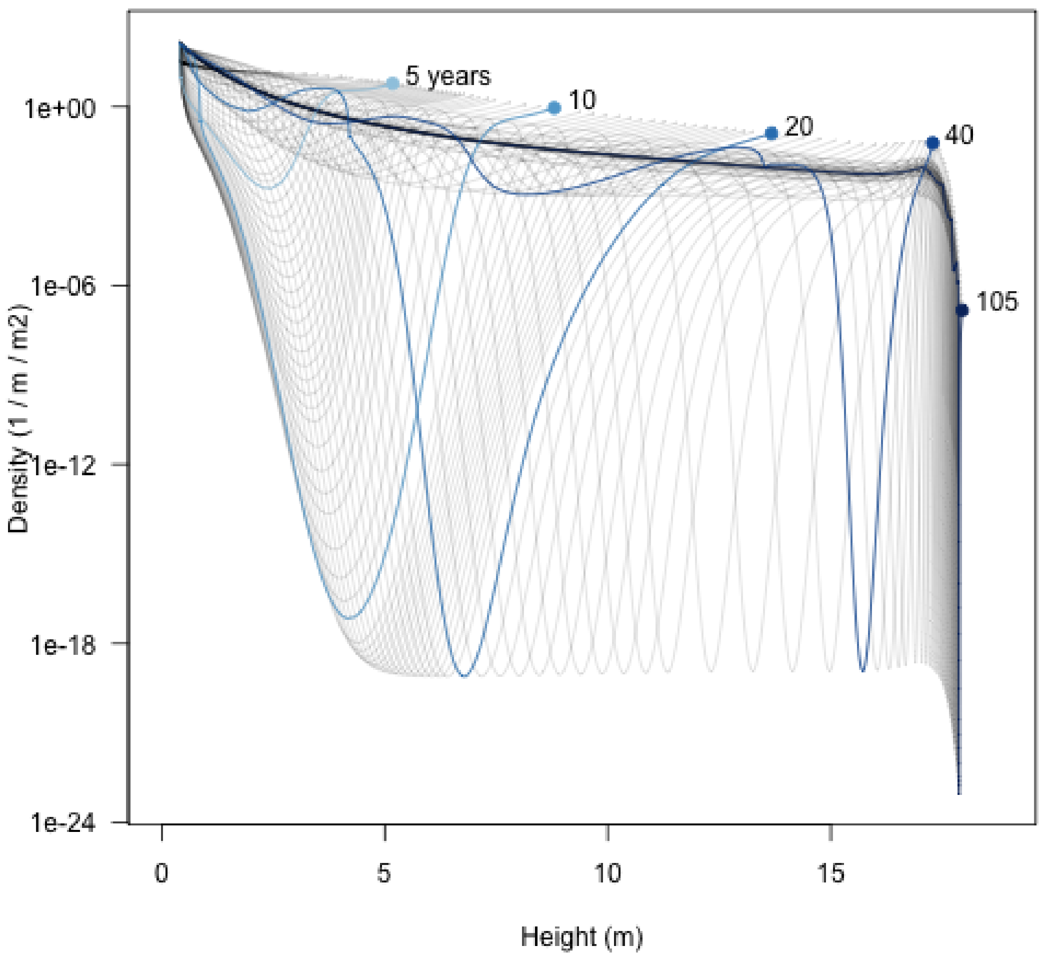

5 years
10
20
40
105
1e+00
1e-06
1e-12
1e-18
1e-24
Density (1 / m / m2)
0
5
10
15
Height (m)

Figure 37

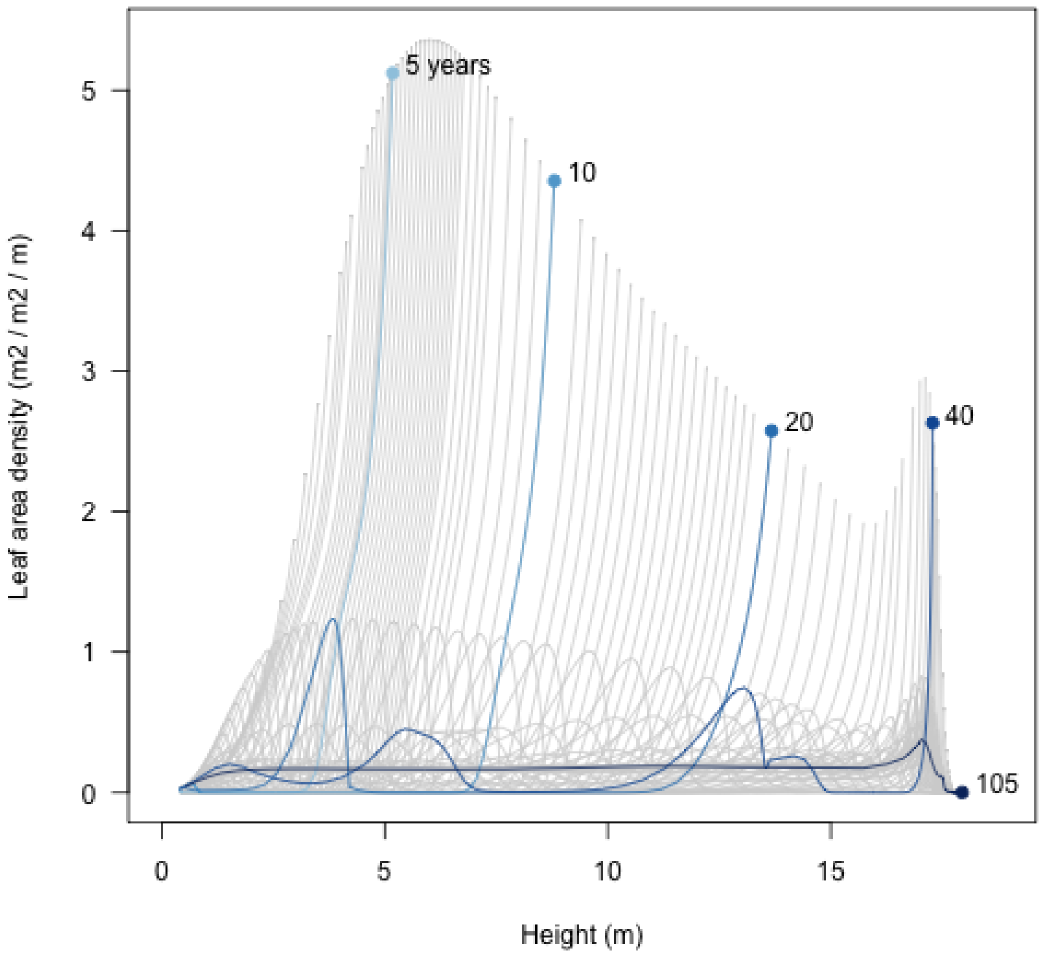

5 years
10
20
40
105
Leaf area density (m2 / m2 / m)
Height (m)
0
1
2
3
4
5
0
5
10
15

Figure 38

```
        ylab="Height growth rate (m / year)", las=1)
matlines(height[, i], growth_rate[, i], col=cols, lty=1, type="l")
points(height[1, i], growth_rate[1, i], pch=19, col=cols)
text(height[1, i] + strwidth("x"), growth_rate[1, i],
     paste0(round(times), c(" years", rep("", length(times) - 1))),
     adj=c(0, 0))
```

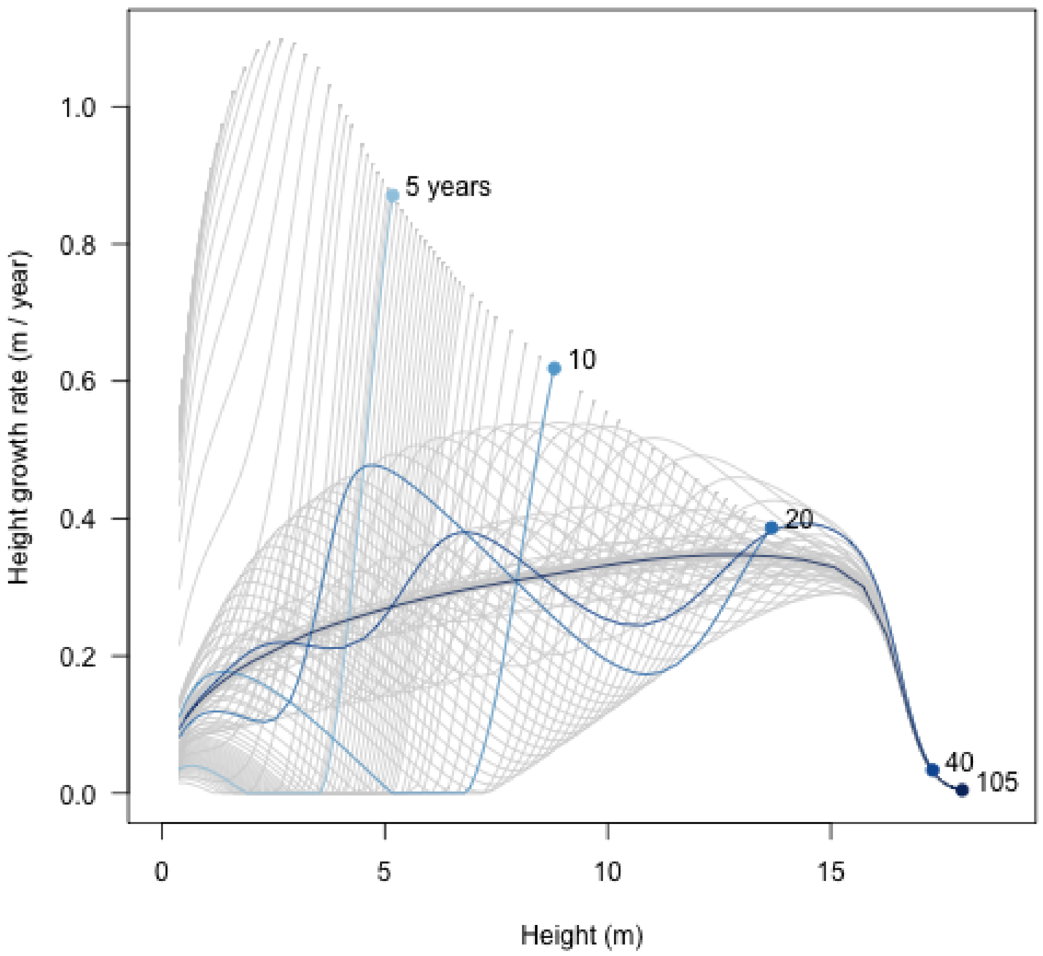


FIGURE 39

The above plots show relationships with patches of a given age What about the average relationship across the entire metapopulation? To get that, we average (integrate) over the distribution over patch (for formula see demography vignette). To achieve this we first need the patch-level relationships to a series of fixed heights

```r
hh <- seq_log_range(range(height, na.rm=TRUE), 500)
```

We'll use a spline interpolation, on log-log-scaled data, clamped so that for x values outside the observed range are set to zero.

```r
f <- function(height, density, hout) {
  r <- range(height, na.rm=TRUE)
  clamp_domain(splinefun_loglog(height, density), r, 0)(hout)
}
```

Now interpolate the height-density relationship in each patch to the series of specified heights

```r
xx <- lapply(seq_along(data1$time),
             function(i) f(height[, i], density[, i], hh))
n_hh <- plant:::pad_list_to_array(xx)
```

For each of these heights, we can now integrate across patches (i.e. across rows), weighting by patch abundance

```r
trapezium <- plant:::trapezium
n_av <- apply(n_hh, 1,
              function(x) trapezium(data1$time, x * data1$patch_density))
```

Add this average to the plot (red line):

```r
xlim <- c(0, max(height, na.rm=TRUE) * 1.05)
matplot(height, density, type="l", lty=1,
        col=make_transparent("black", 0.15),
        xlab="Height (m)", ylab="Density (1 / m / m2)", las=1,
        log="y", xlim=xlim)
matlines(height[, i], density[, i], col=cols, lty=1, type="l")
points(height[1, i], density[1, i], pch=19, col=cols)
text(height[1, i] + strwidth("x"), density[1, i],
     paste0(round(times), c(" years", rep("", length(times) - 1))),
     adj=c(0, 0))
points(hh, n_av, col="red", type='l')
```

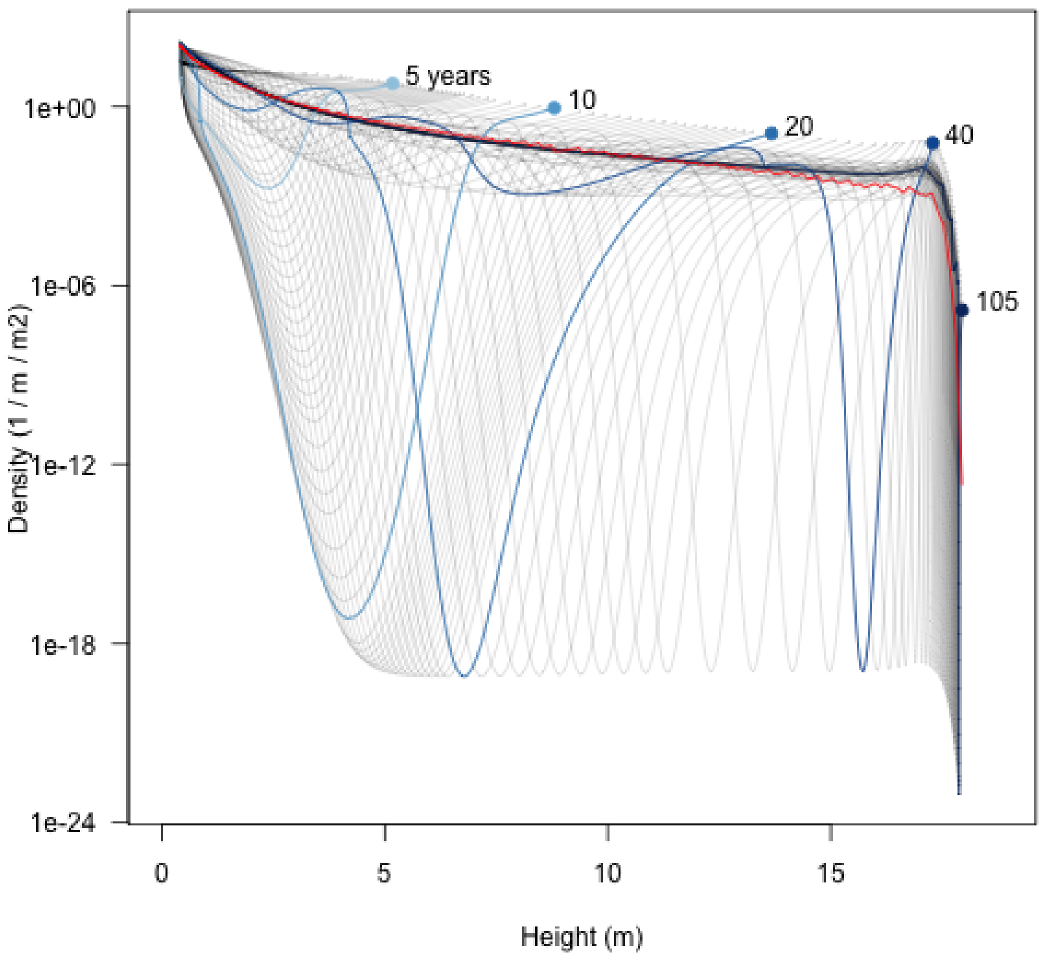

5 years
10
20
40
105
1e+00
1e-06
1e-12
1e-18
1e-24
Density (1 / m / m2)
0
5
10
15
Height (m)

Figure 40

7 CALCULATING FITNESS

```
library(plant)
```

Start by setting a few parameters; this is the base set of parameters we'll use.

```
p0 <- scm_base_parameters("FF16")
p0$control$equilibrium_nsteps <- 30
p0$control$equilibrium_solver_name <- "hybrid"
p0$disturbance_mean_interval <- 30.0
```

First, compute the space that any strategy can exist along the "lma" axis:

```
bounds <- viable_fitness(bounds_infinite("lma"), p0)
bounds

##          lower    upper
## lma 0.02533822 4.989169
```

Generate a set of trait values across this range and compute the fitness landscape:

```
lma <- trait_matrix(seq_log_range(bounds, 101), "lma")
w0 <- fitness_landscape(lma, p0)

plot(lma, w0, type="l", log="x", las=1, ylab="Fitness (empty environment)")
abline(h=0, col="grey")
```

Any trait value along this point can persist, so start with random sample (Set seed for random number generator so that get same results when rerun)

```
set.seed(5)
lma1 <- sort(sample(lma, 4))
```

This function takes an lma value, introduces it to the community, runs that out to equilibrium seed rain:

```
add_eq <- function(x, p) {
  p <- expand_parameters(trait_matrix(x, "lma"), p, mutant=FALSE)
  equilibrium_seed_rain(p)
}
```

If run interactively it will produce a lot of output

```
p1 <- lapply(lma1, add_eq, p0)
```

Then compute fitness landscapes for each of these:

```
w1 <- sapply(p1, function(p) fitness_landscape(lma, p))

matplot(lma, w1, lty=1, type="l", log="x", ylim=c(-5, max(w1)))
abline(h=0, col="grey")
points(lma1, rep(0, 4), col=1:4, pch=19)
```

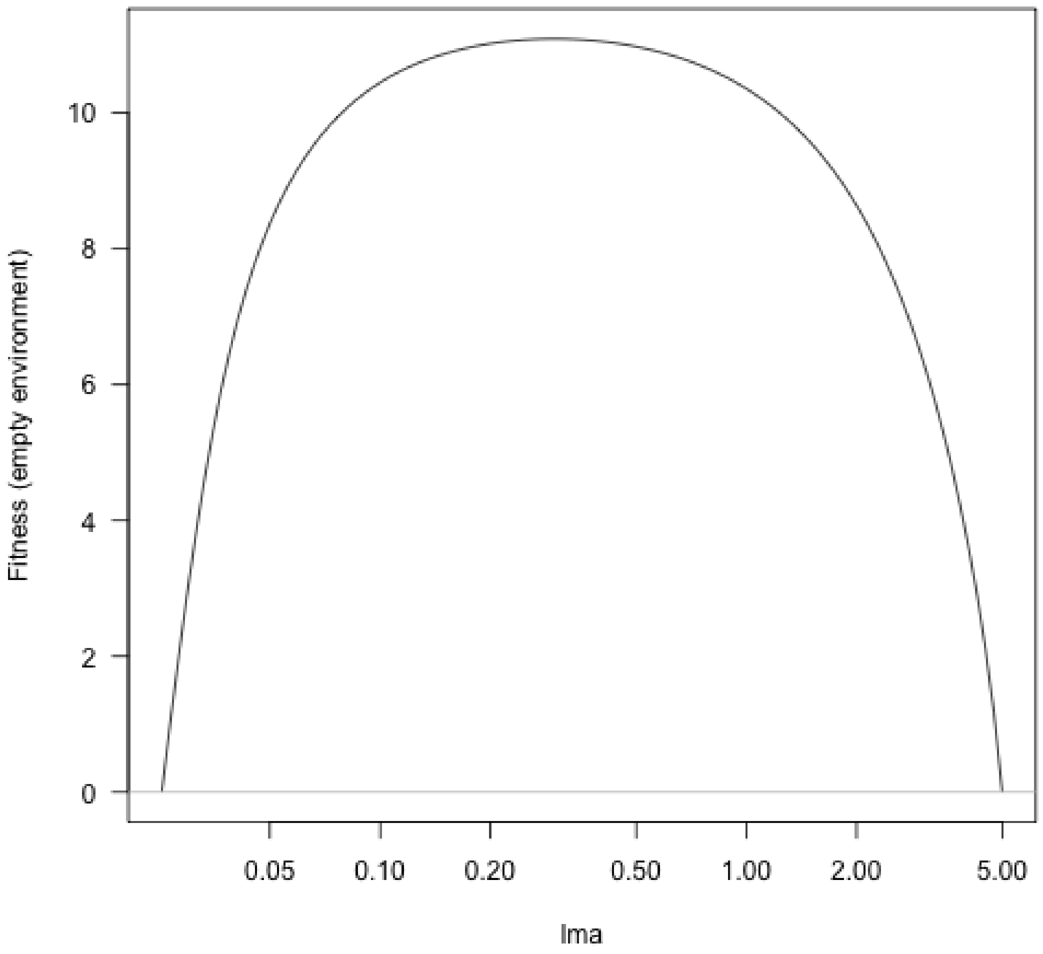

Fitness (empty environment)
0
2
4
6
8
10
0.05
0.10
0.20
0.50
1.00
2.00
5.00
lma

Figure 41

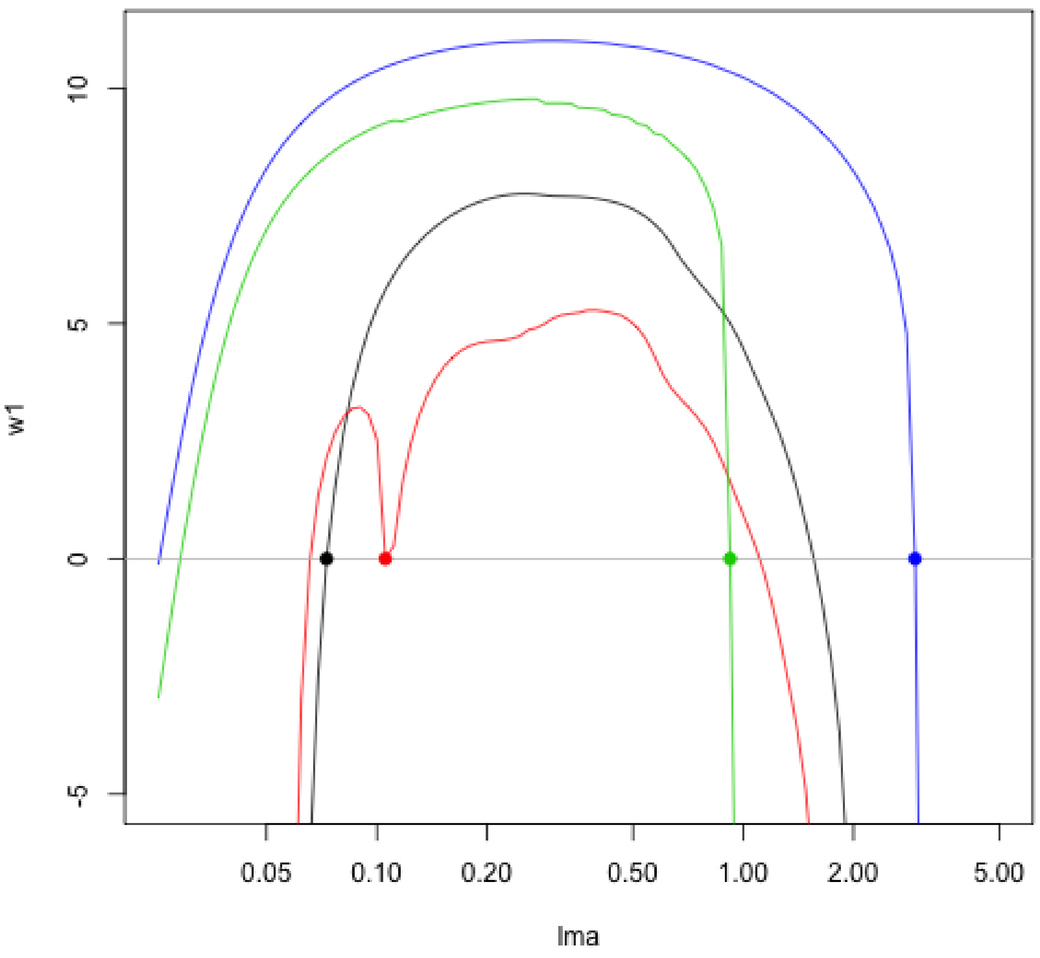


FIGURE 42

For this system, there is an evolutionary attractor around lma 0.0825:

```
lma_b <- 0.0825
p1b <- add_eq(lma_b, p0)

w1b <- fitness_landscape(lma, p1b)
plot(lma, w1b, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness")
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
```

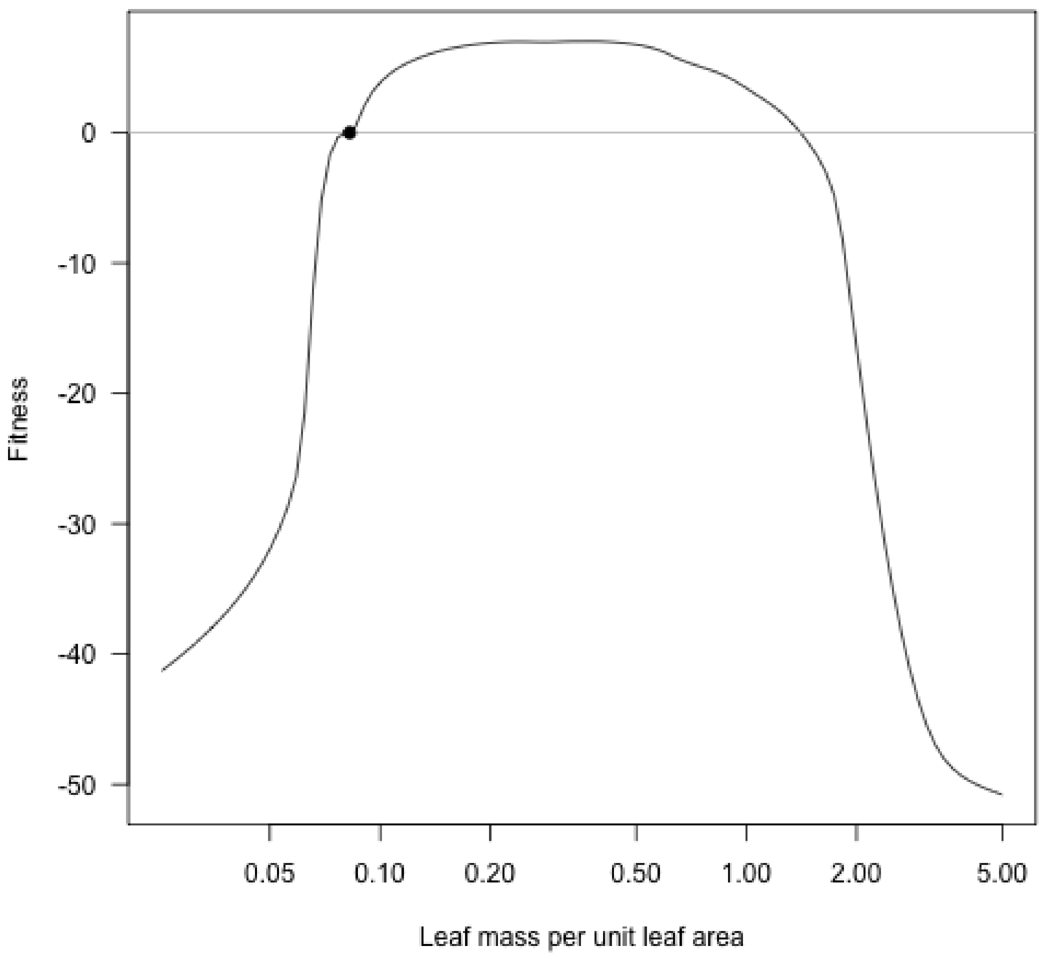


Figure 43

Zooming in in the vicinity of the result shows that this is disruptive selection: fitness increases to both sides of the resident!

```
lma_detail <- trait_matrix(seq_log(lma_b * 0.95, lma_b * 1.05, 51), "lma")
w1b_detail <- fitness_landscape(lma_detail, p1b)
plot(lma_detail, w1b_detail, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness")
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
```

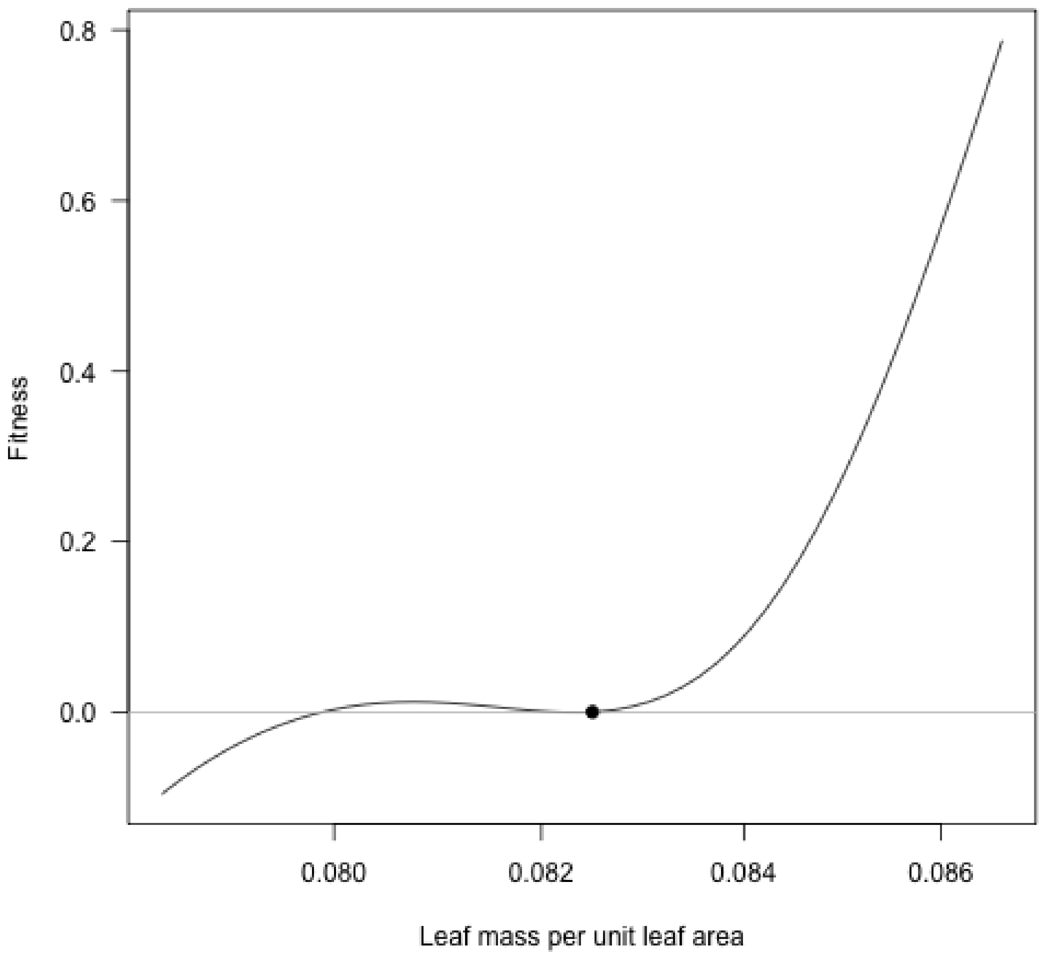


FIGURE 44

Holding the first species at 0.0825 we can introduce additional species (it's close enough to the optimum here, though in general this point might move substantially as new species are introduced).

Consider introducing a new species at the point of maximum fitness:

```
lma_new <- lma[which.max(w1b)]
lma_new
```

```
## [1] 0.3748389
```

```
plot(lma, w1b, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness")
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
abline(v=lma_new, col="red")
```

```
p2 <- add_eq(lma_new, p1b)
w2 <- fitness_landscape(lma, p2)
```

After introducing this species, the fitness landscape is *drawn down* around the second species, with a fitness gradient that points towards increased lma.

```
plot(lma, w1b, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness", col="grey")
lines(lma, w2)
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
points(lma_new, 0, pch=19, col="red")
lma_new2 <- lma[which.max(w2)]
abline(v=lma_new2)
```

At the cost of extremely tedious copy/paste code, here is the result of repeatedly taking the lma value with highest fitness and moving the second species to this point, running to equilibrium, and plotting. For comparison the previous landscapes are retained as dotted lines.

```
p2_2 <- add_eq(lma_new2, p1b)
w2_2 <- fitness_landscape(lma, p2_2)
```

```
plot(lma, w1b, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness", col="grey")
lines(lma, w2, lty=2)
lines(lma, w2_2)
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
points(lma_new, 0)
points(lma_new2, 0, pch=19, col="red")
lma_new3 <- lma[which.max(w2_2)]
abline(v=lma_new3)
```

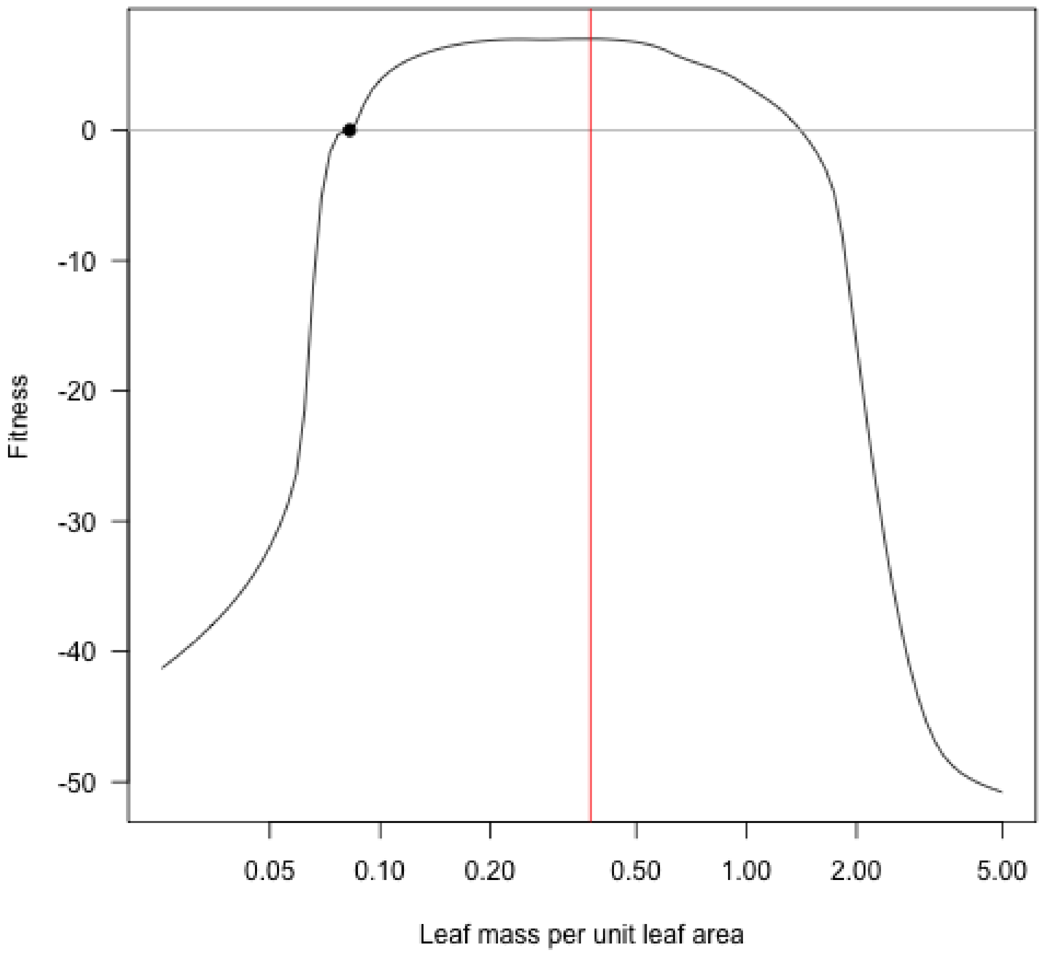

Fitness
0
-10
-20
-30
-40
-50
0.05
0.10
0.20
0.50
1.00
2.00
5.00
Leaf mass per unit leaf area

Figure 45

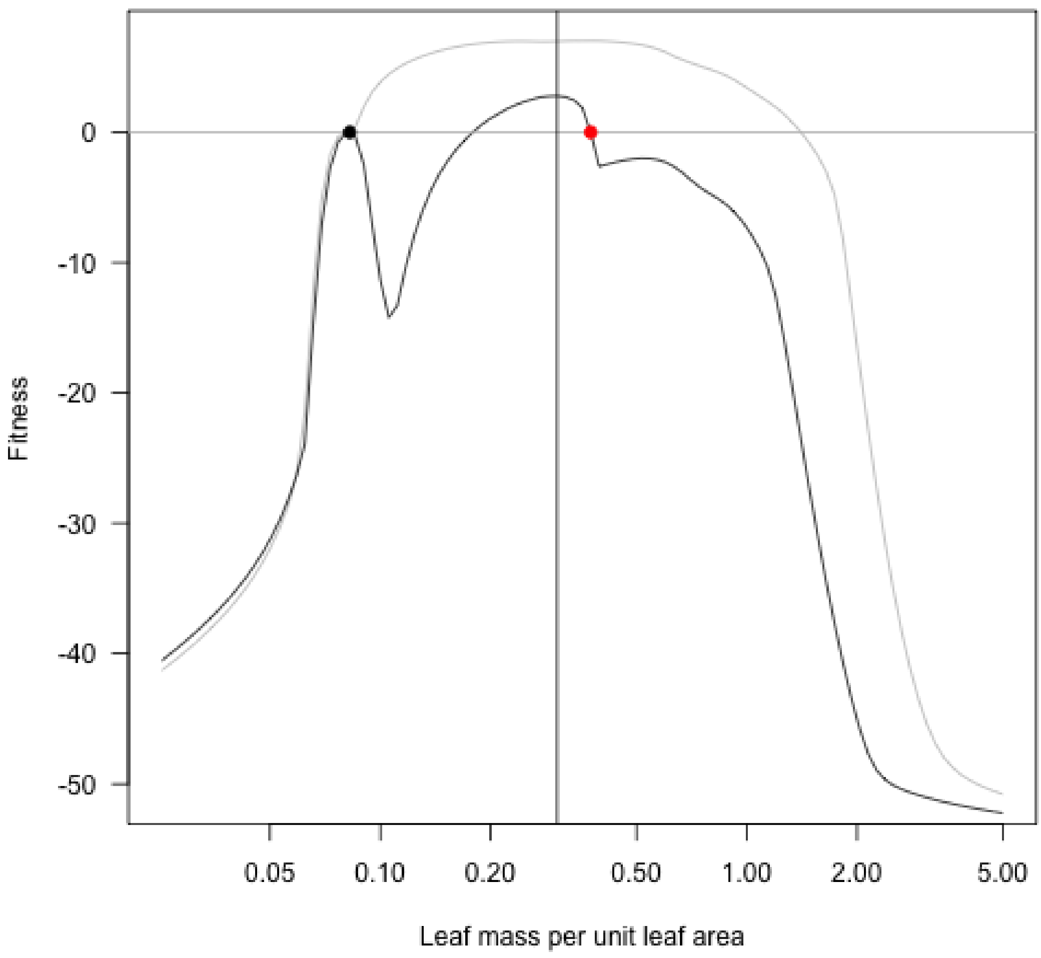

0
-10
-20
-30
-40
-50
Fitness
0.05
0.10
0.20
0.50
1.00
2.00
5.00
Leaf mass per unit leaf area

Figure 46

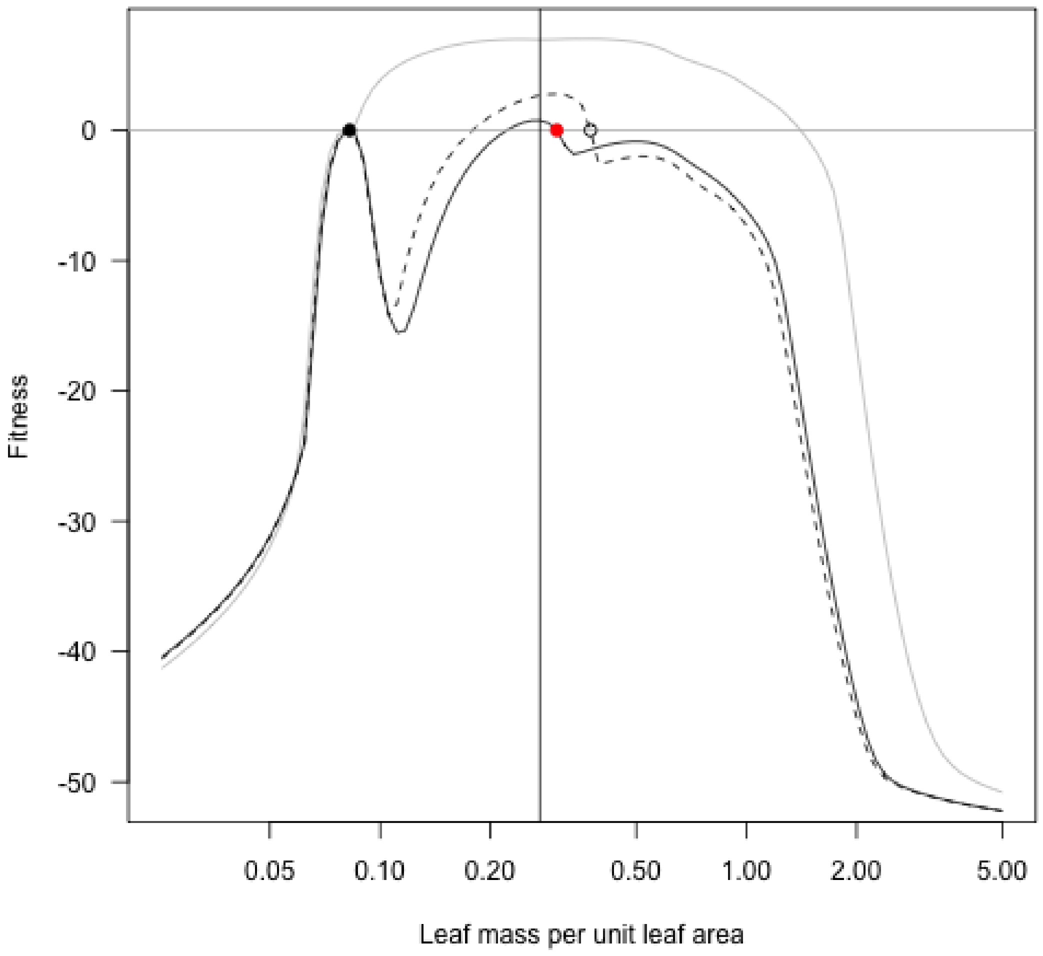
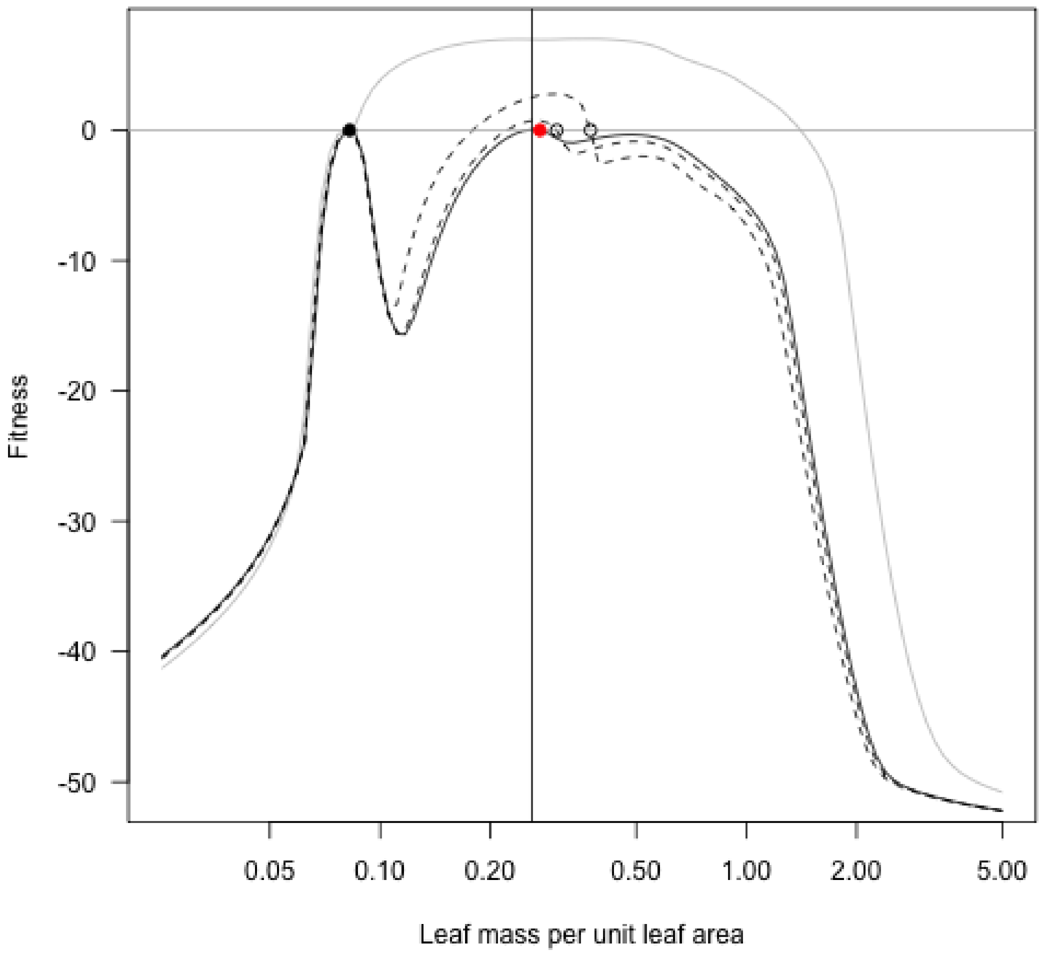

0
-10
-20
-30
-40
-50
Fitness
0.05
0.10
0.20
0.50
1.00
2.00
5.00
Leaf mass per unit leaf area

Figure 47

```
p2_3 <- add_eq(lma_new3, p1b)
w2_3 <- fitness_landscape(lma, p2_3)

plot(lma, w1b, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness", col="grey")
lines(lma, w2, lty=2)
lines(lma, w2_2, lty=2)
lines(lma, w2_3)
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
points(lma_new, 0)
points(lma_new2, 0)
points(lma_new3, 0, pch=19, col="red")
lma_new4 <- lma[which.max(w2_3)]
abline(v=lma_new4)


p2_4 <- add_eq(lma_new4, p1b)
w2_4 <- fitness_landscape(lma, p2_4)

plot(lma, w1b, log="x", type="l", las=1,
     xlab="Leaf mass per unit leaf area", ylab="Fitness", col="grey")
lines(lma, w2, lty=2)
lines(lma, w2_2, lty=2)
lines(lma, w2_3, lty=2)
lines(lma, w2_4)
abline(h=0, col="grey")
points(lma_b, 0, pch=19)
points(lma_new, 0)
points(lma_new2, 0)
points(lma_new3, 0)
points(lma_new4, 0, pch=19, col="red")
lma_new5 <- lma[which.max(w2_4)]
abline(v=lma_new5)
```



0
-10
-20
-30
-40
-50
Fitness
0.05
0.10
0.20
0.50
1.00
2.00
5.00
Leaf mass per unit leaf area

Figure 48

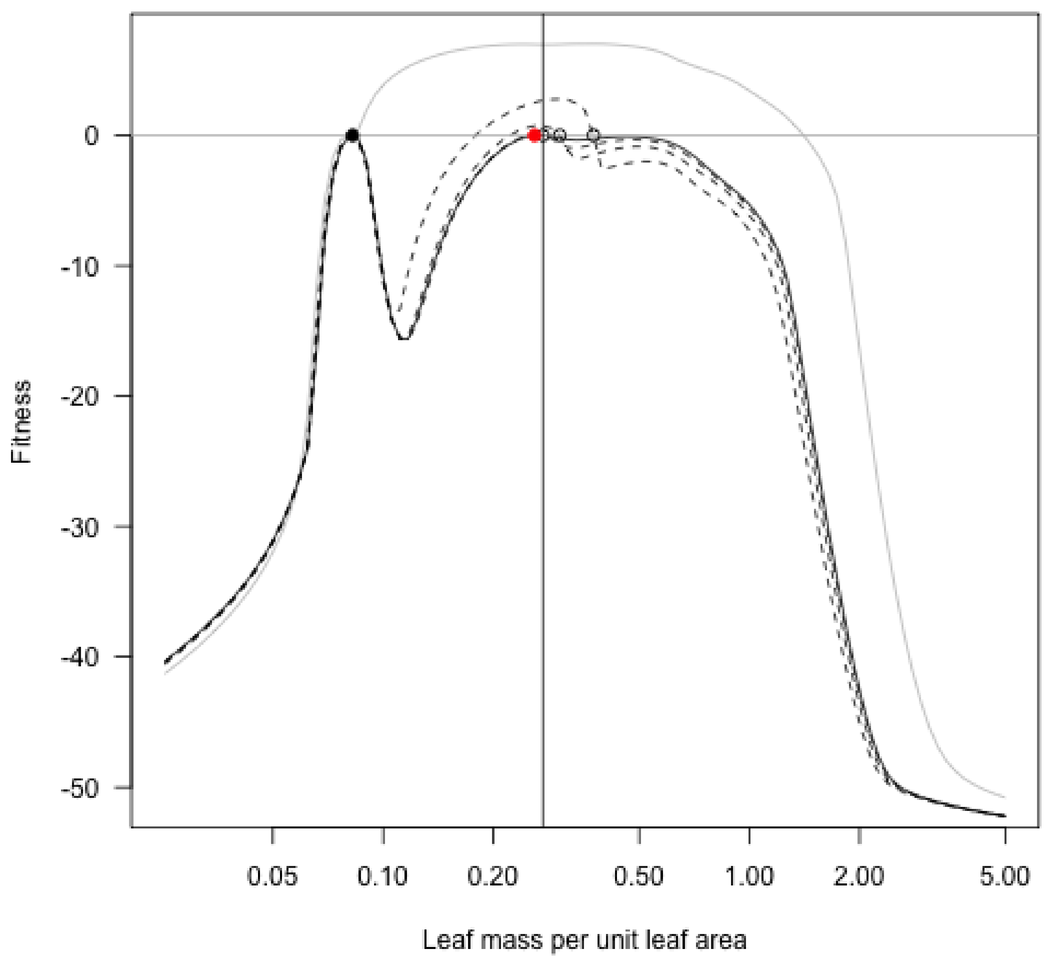

0
-10
-20
-30
-40
-50
Fitness
0.05
0.10
0.20
0.50
1.00
2.00
5.00
Leaf mass per unit leaf area

FIGURE 49

8 MODIFYING PARAMETERS

```
library(plant)
```

There are a large number of parameters to the physiological model, but all are changeable. The default strategy is detailed in the "physiology" vignette:

```
s <- FF16_Strategy()
names(s)
```

```
##  [1] "lma"     "rho"     "hmat"    "omega"   "eta"     "theta"   "a_l1"
##  [8] "a_l2"    "a_r1"    "a_b1"    "r_s"     "r_b"     "r_r"     "r_l"
## [15] "a_y"     "a_bio"   "k_l"     "k_b"     "k_s"     "k_r"     "a_p1"
## [22] "a_p2"    "a_f3"    "a_f1"    "a_f2"    "S_D"     "a_d0"    "d_I"
## [29] "a_dG1"   "a_dG2"   "control"
```

The strategy object here is a special object of class FF16_Strategy. In contrast with most of the "reference" objects used by `plant`, this is simply a list with a class attribute. Some validation will be done on the parameters every time it is passed into the C++ bits of the model.

All parameters except for `control` are floating point (decimal) numbers:

```
unlist(s[names(s) != "control"])
```

```
##          lma          rho         hmat        omega          eta
## 1.978791e-01 6.080000e+02 1.659587e+01 3.800000e-05 1.200000e+01
##        theta         a_l1         a_l2         a_r1         a_b1
## 2.141786e-04 5.440000e+00 3.060000e-01 7.000000e-02 1.700000e-01
##          r_s          r_b          r_r          r_l          a_y
## 6.598684e+00 1.319737e+01 2.170000e+02 1.984545e+02 7.000000e-01
##        a_bio          k_l          k_b          k_s          k_r
## 2.450000e-02 4.565855e-01 2.000000e-01 2.000000e-01 1.000000e+00
##         a_p1         a_p2         a_f3         a_f1         a_f2
## 1.511778e+02 2.047162e-01 1.140000e-04 1.000000e+00 5.000000e+01
##          S_D         a_d0          d_I        a_dG1        a_dG2
## 2.500000e-01 1.000000e-01 1.000000e-02 5.500000e+00 2.000000e+01
```

All of these parameters can be directly changed. For example, we can double the leaf mass per unit leaf area (lma) value:

```
s$lma <- s$lma * 2
```

and then from this construct a plant:

```
pl <- FF16_Plant(s)
pl$strategy$lma
```

```
## [1] 0.3957582
```

`lma` affects a few places in the model; see the source code, but only really for converting from leaf area to leaf mass (leaf mass being leaf area multiplied by leaf mass per unit leaf area). However, as a component of the leaf economic spectrum we imagine LMA as affecting a number of other components of the model.

We capture this through what we call a "hyper-parameterisation"; additional parameters and functions that mean that changing one parameter might affect a number of other lower-level parameters. Our default hyper-parameterisation is via the `FF16_hyperpar` function:

```
FF16_hyperpar

## function (m, s, filter = TRUE)
## {
##     with_default <- function(name, default_value = s[[name]]) {
##         rep_len(if (name %in% colnames(m))
##             m[, name]
##         else default_value, nrow(m))
##     }
##     lma <- with_default("lma")
##     rho <- with_default("rho")
##     omega <- with_default("omega")
##     narea <- with_default("narea", narea)
##     k_l <- B_kl1 * (lma/lma_0)^(-B_kl2)
##     d_I <- B_dI1 * (rho/rho_0)^(-B_dI2)
##     k_s <- B_ks1 * (rho/rho_0)^(-B_ks2)
##     r_s <- B_rs1/rho
##     r_b <- B_rb1/rho
##     a_f3 <- B_f1 * omega
##     assimilation_rectangular_hyperbolae <- function(I, Amax,
##         theta, QY) {
##         x <- QY * I + Amax
##         (x - sqrt(x^2 - 4 * theta * QY * I * Amax))/(2 * theta)
##     }
##     approximate_annual_assimilation <- function(narea, latitude) {
##         E <- seq(0, 1, by = 0.02)
##         D <- seq(0, 365/2, length.out = 10000)
##         I <- PAR_given_solar_angle(solar_angle(D, latitude = abs(latitude)))
##         Amax <- B_lf1 * (narea/narea_0)^B_lf5
##         theta <- B_lf2
##         QY <- B_lf3
##         AA <- NA * E
##         for (i in seq_len(length(E))) {
##             AA[i] <- 2 * trapezium(D, assimilation_rectangular_hyperbolae(k_I *
##                 I * E[i], Amax, theta, QY))
##         }
##         if (all(diff(AA) < 1e-08)) {
##             ret <- c(last(AA), 0)
##             names(ret) <- c("p1", "p2")
##         }
##         else {
##             fit <- nls(AA ~ p1 * E/(p2 + E), data.frame(E = E,
##                 AA = AA), start = list(p1 = 100, p2 = 0.2))
##             ret <- coef(fit)
##         }
##         ret
##     }
##     a_p1 <- a_p2 <- 0 * narea
##     if (length(narea) > 0 || k_I != 0.5) {
##         i <- match(narea, unique(narea))
##         y <- vapply(unique(narea), approximate_annual_assimilation,
##             numeric(2), latitude)
##         a_p1 <- y["p1", i]
##         a_p2 <- y["p2", i]
##     }
##     r_l <- B_lf4 * narea/lma
##     extra <- cbind(k_l, d_I, k_s, r_s, r_b, a_f3, a_p1, a_p2,
##         r_l)
##     overlap <- intersect(colnames(m), colnames(extra))
##     if (length(overlap) > 0L) {
##         stop("Attempt to overwrite generated parameters: ", paste(overlap,
##             collapse = ", "))
##     }
##     if (filter) {
##         if (nrow(extra) == 0L) {
##             extra <- NULL
##         }
##         else {
##             pos <- diff(apply(extra, 2, range)) == 0
##             if (any(pos)) {
##                 eps <- sqrt(.Machine$double.eps)
##                 x1 <- extra[1, pos]
##                 x2 <- unlist(s[names(x1)])
##                 drop <- abs(x1 - x2) < eps & abs(1 - x1/x2) <
##                   eps
##                 if (any(drop)) {
##                   keep <- setdiff(colnames(extra), names(drop)[drop])
##                   extra <- extra[, keep, drop = FALSE]
##                 }
##             }
##         }
##     }
##     if (!is.null(extra)) {
##         m <- cbind(m, extra)
##     }
##     m
## }
## <environment: 0x7fd48e83b0d0>
```

You will rarely need to call this function directly (see below) but note how setting of lma affects parameters `k_l`, `a_p1` and `r_l`

```
FF16_hyperpar(trait_matrix(0.1, "lma"), s)
```

```
##    lma      k_l   r_l
## p1 0.1 1.466783 392.7
```

These are: * `k_l`: Turnover rate for leaves * `a_p1`: Leaf photosynthesis per area * `r_l`: Leaf respiration per mass

This means that it is possible to implement different trade-offs between parameters relatively easily by modifying the hyper-parameterisation function in R, rather than having to modify the underlying physiological model in C++.

The `trait_matrix` function is a simple wrapper that just makes sure the trait matrix has the right format:

```
trait_matrix(0.1, "lma")
```

```
##      lma
## [1,] 0.1
```

```
trait_matrix(c(0.1, 500), "rho")
```

```
##        rho
## [1,]   0.1
## [2,] 500.0
```

To make use of the hyper-parameterisation, the preferred way of setting parameters is through the utility functions `strategy` and `strategy_list`. These take a `Parameters` object:

```
p <- FF16_Parameters()
```

The Parameters object mostly contains information about the patch:

```
names(p)
```

```
##  [1] "k_I"                           "patch_area"
##  [3] "n_patches"                     "disturbance_mean_interval"
##  [5] "strategies"                    "seed_rain"
##  [7] "is_resident"                   "control"
##  [9] "strategy_default"              "cohort_schedule_max_time"
## [11] "cohort_schedule_times_default" "cohort_schedule_times"
## [13] "cohort_schedule_ode_times"     "hyperpar"
```

and it comes pre-set with the hyper-parameterisation function:

```
identical(p$hyperpar, FF16_hyperpar)
```

```
## [1] TRUE
```

`k_I` is the light extinction coefficient, `disturbance_mean_interval` is the mean disturbance interval. The `strategy_default` member is a Strategy:

```
class(p$strategy_default)
```

```
## [1] "FF16_Strategy"
```

This is the Strategy object that all others will be built from by difference. Running

```
s <- strategy(trait_matrix(0.1, "lma"), p)
```

will create a strategy s where lma is set but also all the parameters that *depend* on lma. The function strategy_list can be used to create a list of strategies:

```
lma <- trait_matrix(seq(0.1, 0.5, length.out=5), "lma")
FF16_hyperpar(trait_matrix(lma, "lma"), s)
```

```
##      lma        k_l     r_l
## [1,] 0.1 1.46678341 392.700
## [2,] 0.2 0.44833712 196.350
## [3,] 0.3 0.22412414 130.900
## [4,] 0.4 0.13703875  98.175
## [5,] 0.5 0.09356799  78.540
```

```
ss <- strategy_list(lma, p)
length(ss)
```

```
## [1] 5
```

We can then use standard R commands to extract variable from this list

```
sapply(ss, function(x) x$lma)
```

```
## [1] 0.1 0.2 0.3 0.4 0.5
```

```
sapply(ss, function(x) x$k_l)
```

```
## [1] 1.46678341 0.44833712 0.22412414 0.13703875 0.09356799
```

```
sapply(ss, function(x) x$a_p1)
```

```
## [1] 151.1778 151.1778 151.1778 151.1778 151.1778
```

```
sapply(ss, function(x) x$r_l)
```

```
## [1] 392.700 196.350 130.900  98.175  78.540
```

There's a convenience function plant_list that returns a set of *plants* based on a vector of traits:

```
pp <- plant_list(lma, p)
```

```
sapply(pp, function(p) p$area_leaf_above(0))
```

```
## [1] 1.713949e-04 1.201090e-04 9.206433e-05 7.449914e-05 6.250239e-05
```

In addition to the physiological parameters there are large number of "control" parameters that affect the behaviour of the various numerical algorithms used (note that in contrast to the physiological parameters these have a variety of types)

```
p$control
```

```
## $plant_assimilation_adaptive
## [1] TRUE
##
## $plant_assimilation_over_distribution
## [1] FALSE
##
## $plant_assimilation_tol
## [1] 1e-06
##
## $plant_assimilation_iterations
## [1] 1000
##
## $plant_assimilation_rule
## [1] 21
##
## $plant_seed_tol
## [1] 1e-08
##
## $plant_seed_iterations
## [1] 1000
##
## $cohort_gradient_eps
## [1] 1e-06
##
## $cohort_gradient_direction
## [1] 1
##
## $cohort_gradient_richardson
## [1] FALSE
##
## $cohort_gradient_richardson_depth
## [1] 4
##
## $environment_light_tol
## [1] 1e-06
##
## $environment_light_nbase
## [1] 17
##
## $environment_light_max_depth
## [1] 16
##
## $environment_light_rescale_usually
## [1] FALSE
##
## $ode_step_size_initial
## [1] 1e-06
##
## $ode_step_size_min
## [1] 1e-06
##
## $ode_step_size_max
## [1] 0.1
##
## $ode_tol_rel
## [1] 1e-06
##
## $ode_tol_abs
## [1] 1e-06
##
## $ode_a_y
## [1] 1
##
## $ode_a_dydt
## [1] 0
##
## $schedule_nsteps
## [1] 20
##
## $schedule_eps
## [1] 0.001
##
## $schedule_verbose
## [1] FALSE
##
## $schedule_patch_survival
## [1] 6.253026e-05
##
## $equilibrium_nsteps
## [1] 20
##
## $equilibrium_eps
## [1] 1e-05
##
## $equilibrium_large_seed_rain_change
## [1] 10
##
## $equilibrium_verbose
## [1] TRUE
##
## $equilibrium_solver_name
## [1] "iteration"
##
## $equilibrium_extinct_seed_rain
## [1] 0.001
##
## $equilibrium_nattempts
## [1] 5
```

```
##
## $equilibrium_solver_logN
## [1] TRUE
##
## $equilibrium_solver_try_keep
## [1] TRUE
##
## attr(,"class")
## [1] "Control"
```

The defaults are rather too slow for many uses, so `scm_base_parameters` provides a faster set by using `fast_control` to set many of these to less accurate values.

```
p2 <- scm_base_parameters("FF16")
p2$control[unlist(p$control) != unlist(p2$control)]
```

```
## $plant_assimilation_adaptive
## [1] FALSE
##
## $plant_assimilation_tol
## [1] 1e-04
##
## $cohort_gradient_direction
## [1] -1
##
## $environment_light_tol
## [1] 1e-04
##
## $environment_light_rescale_usually
## [1] TRUE
##
## $ode_step_size_max
## [1] 5
##
## $ode_tol_rel
## [1] 1e-04
##
## $ode_tol_abs
## [1] 1e-04
##
## $schedule_eps
## [1] 0.005
##
## $schedule_verbose
## [1] TRUE
##
## $equilibrium_eps
## [1] 0.001
```