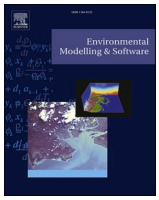
Environmental Modelling and Software 174 (2024) 105973

Contents lists available at ScienceDirect

Environmental Modelling and Software

journal homepage: www.elsevier.com/locate/envsoft



pyMANGA: A modular, open and extendable software platform for modeling of forest and vegetation dynamics

Marie-Christin Wimmler [a,*], Jasper Bathmann [a], Jonas Vollhüter [b], Uta Berger [a]

[a] Institute of Forest Growth and Forest Computer Sciences, TUD Dresden University of Technology, 01062, Dresden, Germany
[b] Institute of Biology, Freie Universität Berlin, 14195, Berlin, Germany

ARTICLE INFO

Handling Editor: Daniel P Ames

Keywords:
Benchmarking
Modularity
Reusability
Building blocks
Individual-based modeling
Ecological modeling

ABSTRACT

Agent-based vegetation models are a widely used tool in ecology, for example, to understand and predict the response of vegetation to environmental change. Models are based on well-established descriptions of processes such as vegetation establishment, growth and mortality. However, they are often developed from scratch, which can be inefficient. Here we present pyMANGA, a free and open-source platform for plant growth modelers. pyMANGA's modular design allows for the combination of different concepts and theories of how plants establish, grow or compete in response to above- and below-ground resource availability. New or alternative modules describing, e.g., competition or facilitation, can be easily added. The interchangeability of modules supports the systematic testing of different hypotheses, e.g., on dominant processes in soil-plant feedback loops. Here we further present the thorough benchmarking strategy to maintain the platform and how pyMANGA can be used to compare models with different levels of abstraction and complexity.

Software and data availability

- Software name: pyMANGA (PYthon Models for AgeNt-based resource GAthering)
- Developer: Jasper Bathmann, Marie-Christin Wimmler, Jonas Vollhüter
- First year available: 2020
- Hardware requirements: PC
- Software requirements: Python 3
- Cost: free
- Availability: https://github.com/pymanga/pyMANGA
- License: GPL-3.0 license
- Program size: 5.05 MB
- Benchmarks and Examples: https://github.com/pymanga/pyMANGA/tree/master/Benchmarks
- Documentation: https://pymanga.github.io/pyMANGA/pyMANGA.html
- Website: http://pymanga.forst.tu-dresden.de/

Data and analysis presented in this manuscript (section 5) are available https://github.com/mcwimm/pym_etal23.

1. Introduction

In forest ecology, computational models are widely used to predict forest development under changing conditions (e.g., abiotic or anthropogenic effects, management) or to better understand certain phenomena and the underlying processes (theory development) (Bugmann and Seidl, 2022). For the latter, so-called agent- or individual -based models (we refer to these terms synonymously as ABMs) have become widely accepted (Zhang and Deangelis, 2020). ABMs describe plants as individual entities with variable and often unique characteristics and developmental trajectories that interact with each other and with their environment. Based on individual-level processes, system-level patterns emerge, that in turn can influence the individual-level processes. All ABMs on forest dynamics typically describe the same processes, namely tree establishment, growth and mortality, and competition for resources. However, the particular descriptions of these processes can vary considerably depending on the type of ABM (e.g., process-based, process-oriented, or phenomenological) or the level of detail. The model structure depends not only on the particular ecosystem-specific processes that are considered important (e.g., seed dispersal, groundwater recharge, or periodic flooding by sea or rivers), but also on the specific concepts or theories used to describe these processes.

* Corresponding author. Forest Biometrics and Systems Analysis, Pienner Straße 8, 01737, Tharandt, Germany.
E-mail address: marie-christin.wimmler@tu-dresden.de (M.-C. Wimmler).

https://doi.org/10.1016/j.envsoft.2024.105973
Received 16 August 2023; Received in revised form 20 December 2023; Accepted 9 February 2024
Available online 10 February 2024

Examples of different concepts to describe competition among neighboring plants are the zone-of-influence (ZOI, Czárán and Bartha, 1989; Wyszomirski et al., 1999), the field-of-neighborhood (FON, Berger et al., 2008; Berger and Hildenbrandt, 2000) or the first-principles competition as introduced by Bathmann et al. (2020). To simulate tree growth, variations such as allometric functions (e.g., Botkin et al., 1972) or mechanistic descriptions that allocate resources to different plant parts (Peters et al., 2014 among others) can be found. The variability of the concepts accounts for the life processes described by ABMs and for the representation of the abiotic environment, including resource availability and fluxes.

Widely used modeling concepts have been reimplemented several times. This is both a sign of the suitability of the concept for a particular purpose and of a certain level of transparency in its documentation, or simply of ease of parameterization. An example is the allometric sigmoid growth function modified by stress factors as introduced for the JABOWA model (Botkin et al., 1972) which was subsequently reused in many gap and agent-based models afterwards (e.g., SOEL model by Kellner and Swihart 2017). Re-implementations generally contribute to the understanding of approaches and to a collective testing of their behavior, thus supporting the development of best-practice software tools in the ecological sciences. On the downside, any reimplementation is error-prone and can be frustrating due to the time spent extracting equations and descriptions from various publications and manuals. It ties up the energies and efforts of scientists (who typically do not have a background in software development and computer science) that could be better spent on their specific research question. There are efforts to address this challenge, e.g., through standardized model descriptions such as the ODD (Grimm et al., 2006, 2020), OPE (Planque et al., 2022) protocol or RBB (Berger et al., 2024. However, even if a transparent model description and the source code are openly available, understanding it and extracting relevant functionality can be challenging as the main focus during model development is not on an efficient and easily understandable model architecture.

Here, we present PYthon Models for AgeNt-based resource GAthering (pyMANGA, v3.1.0), a platform designed to collect common, but also novel, concepts used in modeling vegetation dynamics. The core idea of pyMANGA is its modular structure which provides different concepts, e.g., for plant growth. Concepts are implemented as individual modules that can be swapped or switched on and off according to the modeler's needs. This allows the reuse of well-researched and tested model blocks to focus on new aspects of plant growth dynamics without having to re-implement existing (sub)models. pyMANGA originally emerged from studies of mangrove dynamics (Bathmann et al., 2020, 2021), but due to its modular structure, it can also be used to model other vegetation types such as terrestrial trees, shrubs and grasslands. The platform is free, open-source and version-controlled. We see pyMANGA as a community-based platform that provides the infrastructure to allow model ideas, especially from small projects, to persist and be reused in future projects. In this manuscript, we present the structure of the platform, the main modules and their application, and our benchmarking strategy. We further discuss its benefits and future challenges.

2. Design of the pyMANGA platform

The main idea of pyMANGA is to make models describing plant populations or community dynamics and their descriptions reusable. To achieve this, we have focused on four aspects during the development of the platform: (i) modularity, (ii) transparency and traceability (iii) collaboration and (iv) automation. pyMANGA is publicly available at: https://github.com/pymanga and we use several GitHub (Github, 2023) services to develop and maintain the platform. The clear distinction of terms like model, concept and module is challenging. Therefore, we define the terms we use to describe the design and structure of pyMANGA in Fig. 1a.

2.1. Modularity

We designed pyMANGA as a modular platform to allow for easy integration of new models and components. It is written in the Python 3 programming language (https://www.python.org/, version ≥3.7) and follows object-oriented programming paradigms which support the modular structure of software development best suited to serve the concept of Reusable Building Blocks needed for agent-based modeling (Berger et al., 2024).

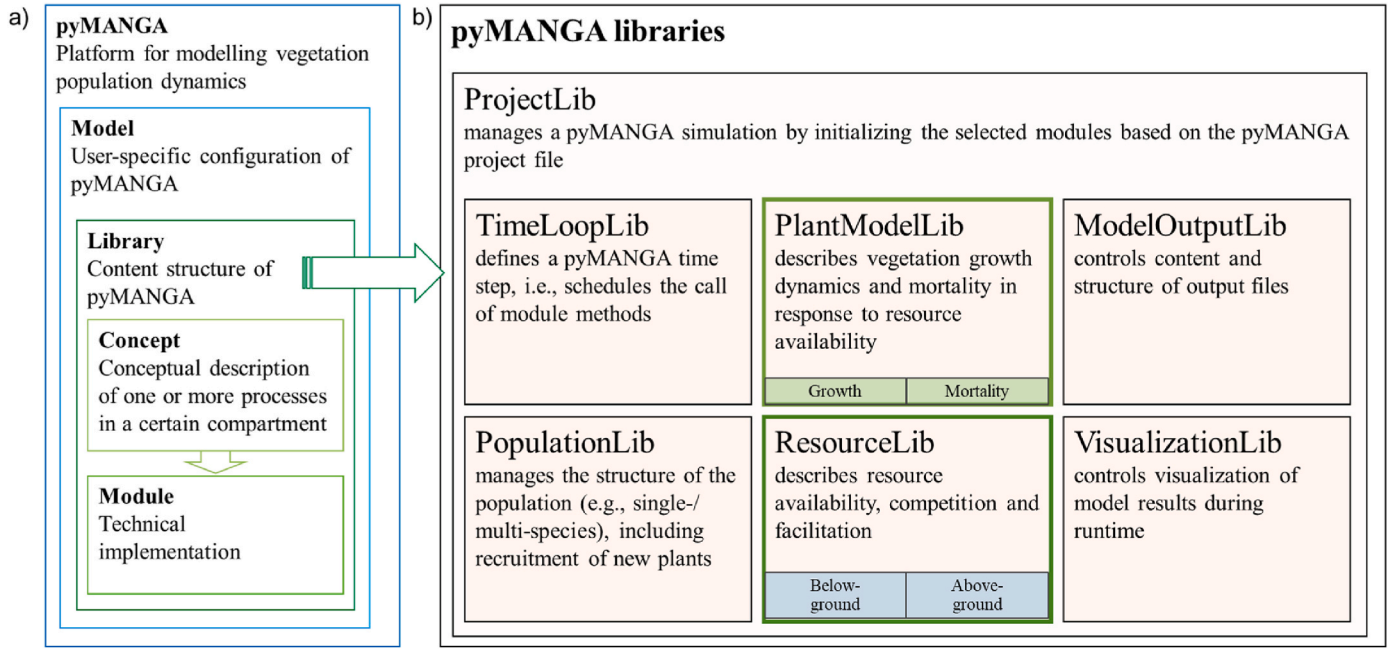


Fig. 1. pyMANGA terminology and structure. (a) Terminology used in pyMANGA. (b) Representation of the pyMANGA libraries. These libraries are each a collection of pyMANGA modules that allow individual model configurations to be assembled. PlantModelLib and ResourceLib are the libraries that manage growth dynamics, mortality and resource availability of the vegetation.

The source code is divided into seven libraries, organized by purpose: *ProjectLib, TimeLoopLib, PopulationLib, PlantModelLib, ResourceLib, ModelOutputLib, VisualizationLib* (Fig. 1b). A library may contain several modules, each representing a different realization of the overarching library purpose. For example, the *ModelOutputLib* contains different realizations of how output is written during a model run, i.e., everything in one file, one file per individual or one file per output step. To allow the existence and interchangeability of different modules, interfaces have been defined. To adapt, for example, different concepts describing the resource availability in the model domain (*ResourceLib*) to the vegetation growth modules (*PlantModelLib*), resource availability is expressed as the ratio of actual to potential availability, resulting in values between 0 and 1 (limiting or competitive conditions) or values > 1 (facilitative conditions). For example, in the case of a pyMANGA application to describe mangrove forests, the salinity of the pore water limits the amount of water accessible to a plant (Bathmann et al., 2020). This limitation is translated into a ratio that describes the availability of resources (in this case water) below the surface. This allows the vegetation growth module to 'interpret' these values independent of how pore water salinity and subsequent water resources are calculated. This way the integration of different concepts to describe processes is possible, regardless of whether they are process-oriented, process-based or phenomenological.

2.2. Transparency and traceability

pyMANGA is free and open-source. To ensure transparency and traceability, we have implemented the 'git' version control system for pyMANGA's source code and documentation. This allows users to easily track changes made to the code and previous versions can be viewed and restored. It also makes code development transparent, as changes are logged and can be discussed (see *2.3 Collaboration*).

All source code is commented and we use pdoc (https://pdoc.dev) for automated source code documentation: https://pymanga.github.io/pyMANGA. This includes technical descriptions of class methods (modules and their functions) but also a user-friendly description of each module. The latter provides information on how to call a particular module in the project file and what parameters to specify. Besides the more technical documentation of the code, we provide several examples and how-tos on our website: http://pymanga.forst.tu-dresden.de/docs/

2.3. Collaboration

Collaboration has been a key aspect in the development of pyMANGA. It is a living platform that collects model descriptions from different projects. It is therefore constantly evolving. To orchestrate this and at the same time work transparently and traceability, we use several GitHub services. The main tool is "GitHub Pull Requests" (also called "merge requests" on other platforms). This allows contributors to share work in progress, ask for help or feedback and discuss code changes with collaborators. Each contribution to the platform must be approved by pyMANGA contributors (i.e., team members) by reviewing a pull request, which includes the discussion of new or extended modules and their technical implementation. This not only helps to identify bugs in the code implementation but also stimulates the exchange and cooperation between contributors.

In addition, "GitHub Issues" provides a communication channel for users to discuss and share ideas, make suggestions for new modules, report bugs or ask for help with a simulation. Issues can be created by anyone and do not require any programming experience.

2.4. Automation

GitHub provides so-called "Actions" to automate workflows. Within the pyMANGA project this is used to run defined module tests (see *4. Module Benchmarks*) and to build the pyMANGA website and documentation based on the current version. Module tests are executed using the unit testing built-in testing framework "unittest" and triggered with every change to the platform (e.g., someone proposes a new vegetation growth module). The output produced by test models (benchmarks) using the current code is compared with reference files. If the tests fail, the change cannot be incorporated into the platform.

2.5. Usage

The usage of pyMANGA requires Python 3 and comes as a command-line tool. A model is configured using a project file (xml-format), in which the realization of each library (module). To use pyMANGA (without code contribution), download a copy from the GitHub page and unzip the folder. From this folder open the prompt and use the following structure to run the project file "example_setup.xml".

py MANGA.py -i path/to/project/file/example_setup.xml.

A detailed guide on how to use pyMANGA can be found on our website (http://pymanga.forst.tu-dresden.de/).

Model in- and output

The minimum input for the model is the pyMANGA project file (xml-format). This file defines which modules are used, including module parameter specifications and paths to additional files (if applicable). For example, the mortality module 'Random' (see Table 1) requires the specification of an annual mortality probability but no further variables. The specification of each module is described in the source code documentation.

The standard model output is a csv file describing the evolution of individuals over time. This can be, for example, the stem radius and height of all trees in the model domain at each time step. Which outputs are written and when must be specified in the project file.

3. Key pyMANGA modules

pyMANGA is designed to assemble different model descriptions that represent various parts of a complete model like building blocks. The key modules are part of the ***PlantModelLib*** and ***ResourceLib*** libraries as they manage plant growth and mortality as well as resource availability below- and above-ground. So far, the plant modules describe tree dynamics. However, the generic structure of the libraries allows the implementation of other vegetation types. These libraries are constantly being extended as new projects require, for example, the adaptation of existing growth concepts or their new development. Both libraries contain modules named *Default*. They allow, for example, to switch-off this library completely (e.g., disable below-ground resource limitation). Besides testing, this might be useful to create very simple models with maximum resource availability. The main modules of these libraries are briefly presented in Table 1.

All modules are implemented using the SI unit system. This allows the user to discretize the model individually in space and time. The model domain and grid cell size can be specified in the project file, as well as the time step length, which is at least 1 s. This is particularly important to distinguish between in-depth studies of individual processes, which require very high resolution, and studies at the population level, where model run times also need to be taken into account.

4. Module benchmarks and automatic testing

pyMANGA contains a collection of benchmarks that serve four purposes; (i) to provide examples for users, and (ii) to ensure the technical functionality of pyMANGA (i.e., test the interfaces of modules) (iii) to test model coherence and plausibility, and (iv) to ensure code functionality after updates. They also allow comparison of pyMANGA results with other implementations of the modules, thus supporting further development of the platform as a toolbox for agent-based models describing forests or plant systems in general.

Table 1
Overview of available modules describing growth dynamics, mortality, resource availability and competition in pyMANGA v3.1.0

Vegetation		Resources	
Growth	Mortality	Below-ground	Above-ground
Bettina: mechanistic concept describing resource allocation in trees and shrubs in order to optimize below- or above-ground resource uptake (Peters et al., 2014; 2018, 2020) **BettinaNetwork**: extension of Bettina enabling resource allocation to root graft development **Kiwi**: phenomenological growth function (Berger and Hildenbrandt, 2000)	**NoGrowth**: plant mortality if maintenance costs exceed resource uptake (mechanistic) **Random**: annual mortality probability (stochastic) **RandomGrowth**: growth(-age)-dependent mortality (mechanistic, stochastic) **Memory**: mortality if relative growth over a certain period falls below a threshold (mechanistic)	**sZOI**: competition based on symmetric zone-of-influence concept (Weiner et al., 2001) **FON**: competition based on field-of-neighborhood concept (Berger and Hildenbrandt, 2000) **FixedSalinity**: resource limitation based on pore-water salinity variable over space (gradual) and time. No feedback between plant water uptake and salinity **OGS**: groundwater flow and density-dependent salt transport with OpenGeoSys (Kolditz et al., 2012). Optional: feedback between plant water uptake and pore water salinity (Bathmann et al., 2020, 2021) **Network**: group formation and resource sharing through root grafts. Related modules: **NetworkFixedSalinity, NetworkOGS** (Wimmler et al., 2022) **Merge:** combine multiple resource modules, e.g., resource limitation due to porewater salinity (FixedSalinity) and competition (sZOI).	**aZOI**: competition based on asymmetric zone-of-influence concept (Weiner et al., 2001)

Our platform contains benchmarks for all modules of *PlantModelLib* and *ResourceLib*, hereafter called module benchmarks. For their systematic construction, we developed a standard benchmark design, following the recommendation by Grimm and Berger (2016) on the design of models and model testing, which states that models should be broken down into their components to assess their robustness. Our module benchmarks are thus as simple as possible and as complex as necessary to trigger potentially critical model behavior. This means that the particular setting (or scenario) is clearly defined and only relevant modules are switched on, avoiding interaction effects.

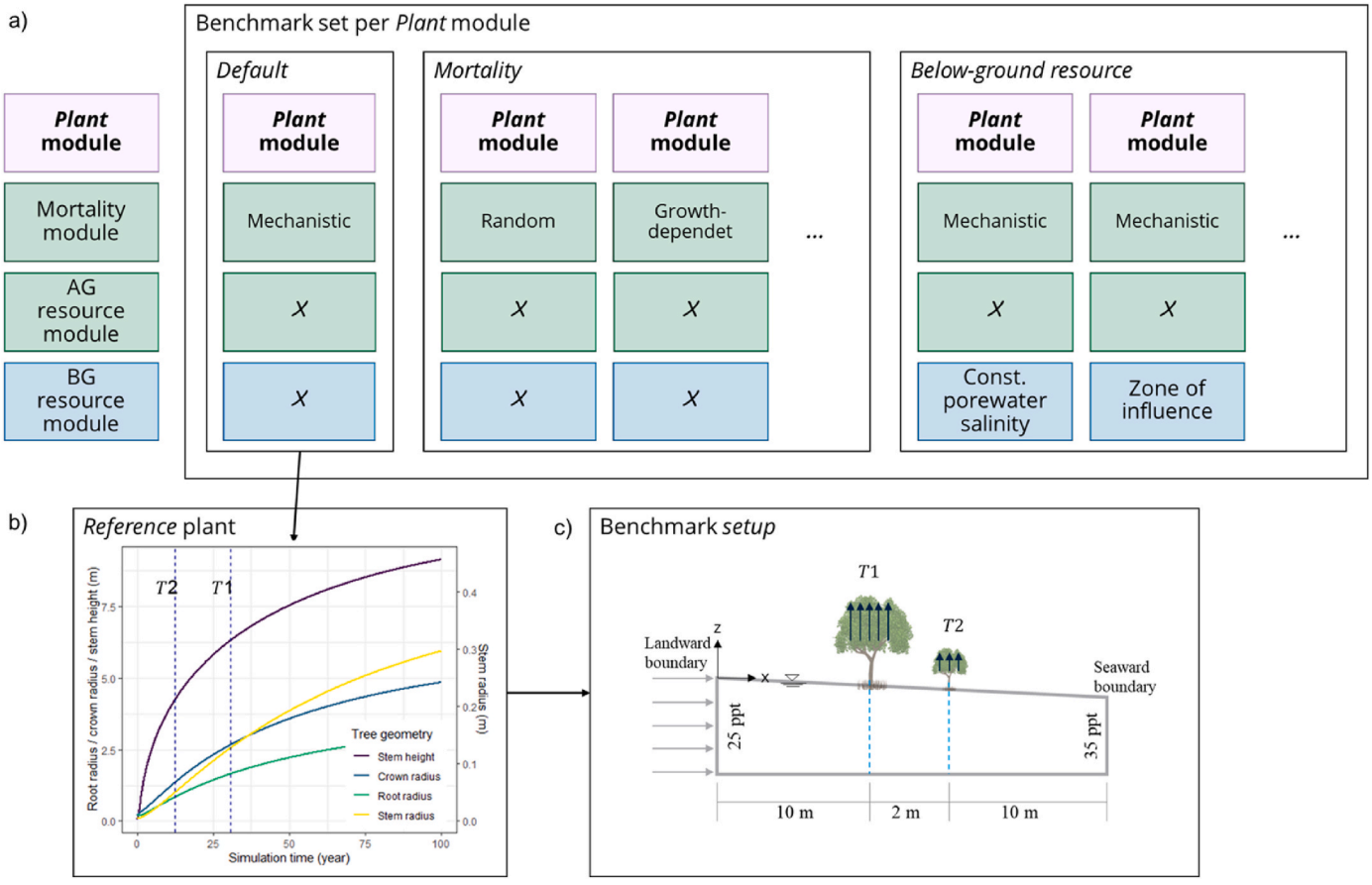


Fig. 2. Benchmark structure. (a) The structure of a set of benchmarks for each plant model. This includes benchmarks in addition to the Default benchmark to test functionality with all appropriate mortality, above-ground (AG) and below-ground (BG) resource modules. (b) Growth curves of the tree geometries simulated using the Bettina approach. The dashed lines mark the tree geometries that serve as initialization for the BG resource benchmarks. (c) Conceptual outline of the model world and its boundary conditions for all benchmarks.

For example, plants are represented or abstracted differently depending on the selected vegetation growth module, hereafter called *plant* module. Their combination with *resource* modules can lead to different model behaviors, requiring an individual set of benchmarks for each plant module (Fig. 2a). This set contains a 'default benchmark', where only the *plant* module under consideration is switched on and *resource* modules are disabled, i.e., plants grow without competition and resource limitation. To avoid negative growth, trees die if resource uptake cannot cover maintenance (we call this mechanistic mortality). Based on this default benchmark, a reference plant for each *plant* module is defined (Fig. 2b). Besides the default benchmark, there are benchmark groups to test mortality, below-ground resource and above-ground resource modules. Within each group modules of other groups are switched off. To benchmark, for example, mortality modules, resource modules are switched off.

If possible and reasonable, each benchmark follows the same experimental design: There are two plants, placed in a 22 × 22 m model domain with 2m distance (Fig. 2c). The initial geometries of those plants are designed to trigger a potentially critical situation for the *resource* modules tested. For below- and above-ground resource concepts this means that crown or root plates of the model plants should overlap, respectively. Thus, we defined separate initial plants with geometries taken from different development stages of the reference tree (vertical dashed lines in Fig. 2b).

All benchmarks have the same time step length of 1e6 seconds (~12 days), a total simulation time of 5e8 seconds (~16 years) and the same grid resolution of 0.25 × 0.25 m^2, if applicable. If land- and seaward boundary conditions need to be defined in a model, salinity is set to 25 ppt at the landward boundary (left) and to 35 ppt at the seaward boundary (right, Fig. 2c). The standard output metrics of benchmarks are tree geometry parameters (e.g., stem diameter, stem height, …). Additional outputs relevant for the tested module might be specified individually.

All module benchmarks are integrated in an automatic testing procedure in pyMANGA (see *2.4 Automation*) using the first timestep of each benchmark. With this automatic test, we ensure that the platform will continue to work according to the defined quality levels and code changes or new developments below that standard will be refused. On the other side, it informs the developers if due to changes in external code, i.e., Python libraries, updates are required.

5. A simple application

5.1. Model setup

Here we present an artificial model study to demonstrate how pyMANGA can be used to experiment with different modules. This is crucial for model development based on the pattern-oriented modeling approach (POM, Grimm et al., 2005) or to easily perform robustness analysis according to Grimm and Berger (2016). In this study, we will test the effect of below-ground and mortality modules with different levels of complexity on the following model outputs: (i) tree density, (ii) tree stem diameter, and (iii) tree age. Complexity in this case means more processes in action (Fig. 3). Data on black mangroves are used to assess the results (Vovides et al. 2018a, 2018b). The dataset includes, inter alia, the position and stem diameter of trees in three 30 × 30m study plots, but no information on tree age (see A1.1 for more details).

The model represents this black mangrove population on a 30 × 30m domain. Tree growth is simulated based on the BETTINA concept (Peters et al., 2014). New trees are recruited as saplings at each time step, i.e., every 3 months, and are randomly distributed across the model domain. Light interception is simulated using the asymmetric zone of influence (ZOI) concept (Weiner et al., 2001). Water accessibility in a mangrove ecosystem is hampered by pore water salinity. In this simulation study, pore water salinity equals 30 ppt and is constant over time. There is no feedback between tree water uptake and porewater salinity. In this study

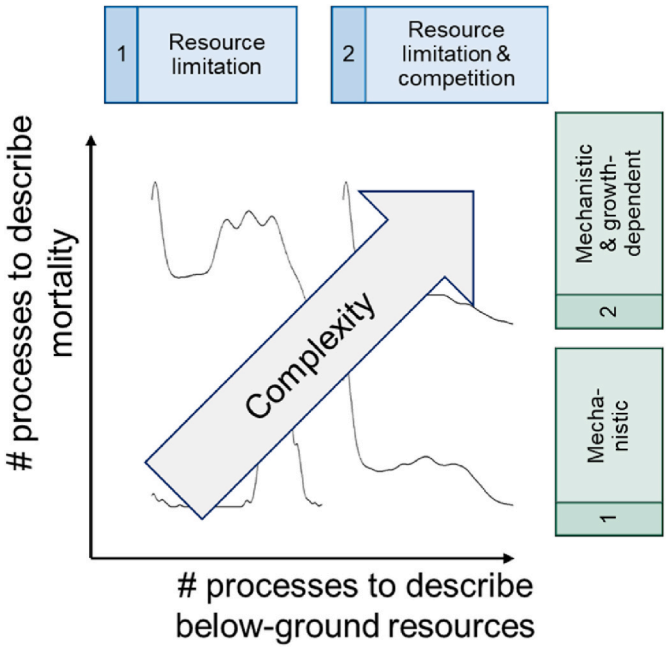


Fig. 3. Application setup. Mortality and below-ground resource availability can be described by one or more processes in pyMANGA, where adding processes increases the complexity of the model and thus the interpretation of model results. In the simple model application presented here, we test whether increasing model complexity has an impact on selected model output parameters. Specifically, we compare model outputs simulated with one or two process describing mortality (green boxes) and below-ground resources (blue boxes), respectively.

we use two different approaches to describe water accessibility: i) water accessibility is only influenced by pore water salinity and ii) water availability is additionally influenced by the presence of neighboring trees. The latter is represented by the symmetric ZOI concept (Weiner et al., 2001). This means that trees occupying the same area share the available water equally. Tree mortality modules are also varied. First, tree mortality is mechanistic, i.e., trees die when resource use cannot cover maintenance costs and second, mortality is additionally caused by a growth-dependent random process. In total, there are four models for this study. Each model setup is repeated 15 times. A detailed description with references to the different parts of the project file and pyMANGA libraries as well as details on the statistical analysis of the results can be found in Appendix A1.

5.2. Results and discussion

The simulated values of tree density and stem diameter are within the range observed in the field, except for the simplest model setup, i.e., resource limitation and mechanistic mortality (Fig. 4). In this model, trees present at the beginning of the simulation rarely die, so newly recruited trees cannot establish. This means that these initial trees can grow unconstrained by competition and become taller than trees in the other models. This also leads to the lowest tree density. On the other hand, in the model with additional below-ground and mortality processes (competition between neighbors and growth-dependent mortality), trees die as growth slows down with age. This leads to a more mixed population where newly recruited trees can establish. Tree density is therefore highest in this configuration.

Enabling growth-dependent mortality has a dominant effect on all presented output metrics, while enabling below-ground competition plays a subordinate role. In the chosen setup, all different module combinations lead to significantly different model outputs ($p < 0.05$, Kruskal-Wallis test, see Appendix A1.4). However, enabling either the

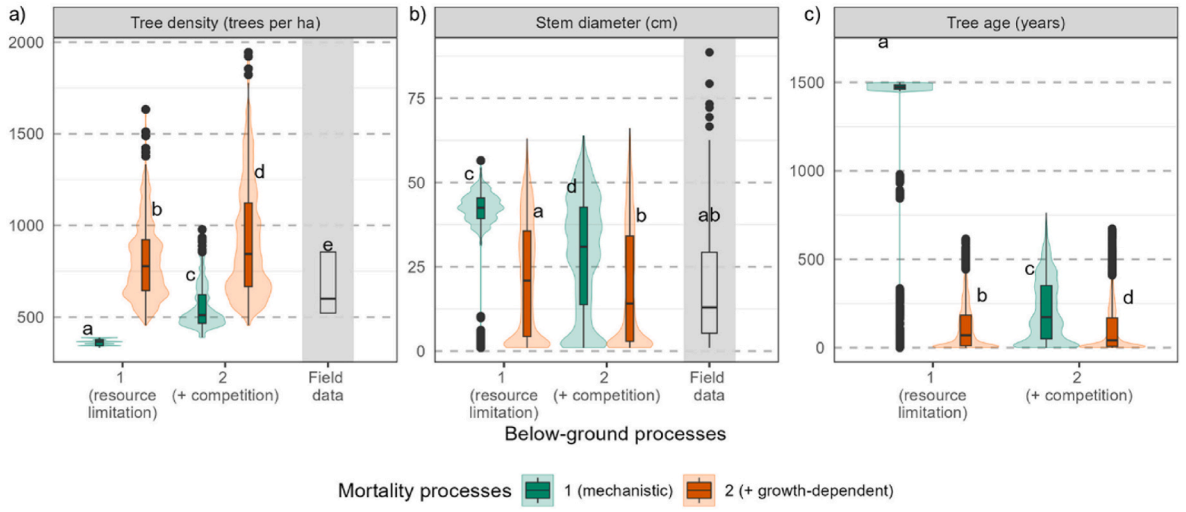


Fig. 4. Results of the application example. Comparison of (a) tree density, (b) stem diameter, and (c) tree age for each of the four models representing different levels of complexity, i.e., number of processes included. Grey shaded boxplots represent the field data reported by Vovides et al. (2018a, b) for Avicennia germinans (Note: data does not include tree age information). Different letters indicate significant differences in the results (Kruskal-Wallis test and Dunn's post-hoc test).

second mortality or the second below-ground process already creates a natural, mixed population. This indicates that not all processes are required, especially in simulation experiments with more processes acting or calibrated to a specific site or species.

6. Conclusion

pyMANGA is a modular platform designed to promote collaboration, transparency, and automation in forest dynamics modeling. Its modular design, which complies with Wilson et al.'s (2014) "Best Practices for Scientific Computing", allows for easy integration of new concepts in vegetation dynamics. A transparent and traceable record of all developments is kept in the GitHub repository.

There are several advantages of collecting model descriptions in a modular platform like pyMANGA rather than working on individual models. Firstly, a modular approach can reduce the time and effort required for reimplementation of model approaches. By developing reusable modules, the potential sources of error associated with reimplementing models from scratch is reduced. Secondly, the modular structure makes it easy to test different modules and hypotheses. As new models and theories emerge, they can be incorporated and compared with previous descriptions. This increases the stability and reliability of the model results. Finally, the defined interfaces make it easy to integrate new ideas into the existing structure of pyMANGA. This allows the developer to focus on a specific module, e.g., resource distribution in the plant, and leave other established processes, e.g., resource availability in the soil, unattended.

By using git and GitHub the development of pyMANGA is made transparent and collaborative (Perez-Riverol et al., 2016; Lowndes et al., 2017). This includes not only the versioning of changes and additions, but also their discussion in pull requests and issues. Throughout the code review process, developers are obligated to follow certain code style rules, making it easier for others to understand their code. It also allows errors to be caught in advance, since at least one other person has to review the code. In this way, we support the principles of open science and studies supported by pyMANGA simulation experiments are reproducible (National Academies, 2019) and transparent. Our benchmarking and testing framework ensure that executability is maintained even if the platform changes or individual researchers leave the project. This makes pyMANGA sustainable in the sense that it is independent of individual research projects and their limited duration.

Of course, there are challenges in building and maintaining such a platform as pyMANGA, two of which we would like to discuss here: complexity and contribution. Complexity refers to the number of modules in each library and how they are interlinked. The large number of modules offers the possibility of creating "integronsters" (Voinov and Shugart 2013), i.e., integrating too many modules and processes into the model configuration. If used incorrectly, this creates complex models that can lead to unreliable interpretations. We therefore always recommend using pyMANGA also as a tool for robustness analysis (Grimm and Berger 2016). In addition, the community needs to decide which modules to include and define which modules can be merged, taking great care not to contradict the ultimate goal of the platform, which is to collect forest models and ensure their applicability. Another challenge is to motivate scientists to use the platform, especially if the models they want to base their work on are not yet or not fully integrated in pyMANGA. Of course, the integration of modules always requires adaptation to the given structure and adherence to interfaces. This also includes making your code readable for others and adapting it if necessary. Another important aspect is to motivate contributors to participate actively, i.e., to review code constructively, but also to be reviewed as the platform thrives with constructive feedback.

Models simplify the real world but often aim to include as many processes as possible. This brings the model closer to reality but makes the interpretation of results more difficult. Testing the robustness of the model by switching off processes one by one is crucial in model development (Grimm and Berger 2016). The 'simple application' example demonstrates pyMANGA's support for this robustness analysis. For example, enabling the second below-ground process (i.e., competition according to Weiner et al., 2001) had only a minor effect when growth-dependent mortality was considered. Depending on the question and therefore the purpose of the model, such a module may be neglected as additional processes also introduce uncertainties due to the necessary parameterization (Edmunds et al. 2019). This demonstrates the benefit of pyMANGA's modular structure, which allows flexibility in model building and testing.

To help expand the collection of concepts covered by the platform, we strongly encourage our colleagues to participate in the pyMANGA project, for example by adding their model concepts of resource distribution, allocation and plant competition.

Funding

MCW was financed by the Volkswagen Foundation (Volkswagen

Stiftung, Project No. 94 844). JB was financed by the German Research Foundation (DFG) in the frame of the MARZIPAN project (project number 398759560). JV is financed by the German Research Foundation (DFG) in the frame of the MASCOT project (project number 492854267).

CRediT authorship contribution statement

Marie-Christin Wimmler: Conceptualization, Data curation, Software, Writing – original draft, Writing – review & editing, Methodology. **Jasper Bathmann:** Conceptualization, Methodology, Software, Writing – review & editing. **Jonas Vollhüter:** Software, Writing – review & editing. **Uta Berger:** Funding acquisition, Project administration, Supervision, Writing – review & editing.

Declaration of Generative AI and AI-assisted technologies in the writing process

During the preparation of this work the author used DeepL (translator and write) in order to help with translations, improve readability and grammar. After using this DeepL, the author(s) reviewed and edited the content as needed and took full responsibility for the content of the publication.

Declaration of competing interest

The authors declare that they have no known competing financial interests or personal relationships that could have appeared to influence the work reported in this paper.

Data availability

All data and code is available in public repositories, which are listed in the manuscript.

Acknowledgement

We would like to thank Ruben M. Jardner from the Freie Universität Berlin for valuable discussions and feedback on the documentation and tutorials.

Appendix A. Supplementary data

Supplementary data to this article can be found online at https://doi.org/10.1016/j.envsoft.2024.105973.

References

Bathmann, J., Peters, R., Naumov, D., Fischer, T., Berger, U., Walther, M., 2020. The MANgrove–GroundwAter feedback model (MANGA) – describing belowground competition based on first principles. Ecol. Model. 420, 108973 https://doi.org/10.1016/j.ecolmodel.2020.108973.

Bathmann, J., Peters, R., Reef, R., Berger, U., Walther, M., Lovelock, C.E., 2021. Modelling mangrove forest structure and species composition over tidal inundation gradients: the feedback between plant water use and porewater salinity in an arid mangrove ecosystem. Agric. For. Meteorol. 308, 309. https://doi.org/10.1016/j.agrformet.2021.108547.

Berger, U., Hildenbrandt, H., 2000. A new approach to spatially explicit modelling of forest dynamics: Spacing, ageing and neighbourhood competition of mangrove trees. Ecol. Model. 132 (3), 287–302. https://doi.org/10.1016/S0304-3800(00)00298-2.

Berger, U., Piou, C., Schiffers, K., Grimm, V., 2008. Competition among plants: concepts, individual-based modelling approaches, and a proposal for a future research strategy. Perspect. Plant Ecol. Evol. Systemat. 9, 121–135. https://doi.org/10.1016/j.ppees.2007.11.002.

Botkin, D.B., Janak, J.F., Wallis, J.R., 1972. Some ecological consequences of a computer model of forest growth. J. Ecol. 60 (3), 849. https://doi.org/10.2307/2258570.

Bugmann, H., Seidl, R., 2022. The evolution, complexity and diversity of models of long-term forest dynamics. J. Ecol. 110 (10), 2288–2307. https://doi.org/10.1111/1365-2745.13989.

Czárán, T., Bartha, S., 1989. The effect of spatial pattern on community dynamics; a comparison of simulated and field data, 83, 229–239.

Edmonds, B., le Page, C., Bithell, M., Chattoe-Brown, E., Grimm, V., Meyer, R., Montañola-Sales, C., Ormerod, P., Root, H., Squazzoni, F., 2019. Different modelling purposes. J. Artif. Soc. Soc. Simulat. 22 (3) https://doi.org/10.18564/jasss.3993.

GitHub, 2023. GitHub: A Collaborative Online Platform to Build Software. Inc. https://github.com/.

Grimm, V., Berger, U., 2016. Robustness analysis: deconstructing computational models for ecological theory and applications. Ecol. Model. 326, 162–167. https://doi.org/10.1016/j.ecolmodel.2015.07.018.

Grimm, V., Berger, U., Bastiansen, F., Eliassen, S., Ginot, V., Giske, J., Goss-Custard, J., Grand, T., Heinz, S.K., Huse, G., Huth, A., Jepsen, J.U., Jørgensen, C., Mooij, W.M., Pe, G., Piou, C., Railsback, S.F., Robbins, A.M., Robbins, M.M., et al., 2006. A Standard Protocol for Describing Individual-Based and Agent-Based Models. https://doi.org/10.1016/j.ecolmodel.2006.04.023.

Grimm, V., Railsback, S.F., Vincenot, C.E., Berger, U., Gallagher, C., Deangelis, D.L., Edmonds, B., Ge, J., Giske, J., Groeneveld, J., Johnston, A.S.A., Milles, A., Nabe-Nielsen, J., Polhill, J.G., Radchuk, V., Rohwäder, M.S., Stillman, R.A., Thiele, J.C., Ayllón, D., 2020. The ODD protocol for describing agent-based and other simulation models: a second update to improve clarity, replication, and structural realism. JASSS 23 (2). https://doi.org/10.18564/jasss.4259.

Grimm, V., Revilla, E., Berger, U., Jeltsch, F., Mooij, W.M., Railsback, S.F., Thulke, H.-H., Weiner, J., Wiegand, T., Deangelis, D.L., 2005. Pattern-oriented modeling of agent-based complex systems: lessons from ecology. Science 310 (5750), 987–991.

Kellner, K.F., Swihart, R.K., 2017. Simulation of oak early life history and interactions with disturbance via an individual-based model, SOEL. PLoS One 12 (6), 1–23. https://doi.org/10.1371/journal.pone.0179643.

Kolditz, O., Bauer, S., Bilke, L., Böttcher, N., Delfs, J.O., Fischer, T., Görke, U.J., Kalbacher, T., Kosakowski, G., McDermott, C.I., Park, C.H., Radu, F., Rink, K., Shao, H., Shao, H.B., Sun, F., Sun, Y.Y., Singh, A.K., Taron, J., et al., 2012. OpenGeoSys: an open-source initiative for numerical simulation of thermo-hydro-mechanical/chemical (THM/C) processes in porous media. Environ. Earth Sci. 67 (2), 589–599. https://doi.org/10.1007/s12665-012-1546-x.

Lowndes, J.S.S., Best, B.D., Scarborough, C., Afflerbach, J.C., Frazier, M.R., O'Hara, C.C., Jiang, N., Halpern, B.S., 2017. Our path to better science in less time using open data science tools. Nature Ecology & Evolution 1 (6), 1–7. https://doi.org/10.1038/s41559-017-0160, 2017 1:6.

National Academies of Sciences, Engineering, and Medicine, 2019. Reproducibility and replicability in science. In: Reproducibility and Replicability in Science. https://doi.org/10.17226/25303.

Perez-Riverol, Y., Gatto, L., Wang, R., Sachsenberg, T., Uszkoreit, J., Leprevost, F. da V., Fufezan, C., Ternent, T., Eglen, S.J., Katz, D.S., Pollard, T.J., Konovalov, A., Flight, R. M., Blin, K., Vizcaíno, J.A., 2016. Ten simple rules for taking advantage of git and GitHub. PLoS Comput. Biol. 12 (7), e1004947 https://doi.org/10.1371/JOURNAL.PCBI.1004947.

Peters, R., Olagoke, A., Berger, U., 2018. A new mechanistic theory of self-thinning: adaptive behaviour of plants explains the shape and slope of self-thinning trajectories. Ecol. Model. 390, 1–9. https://doi.org/10.1016/j.ecolmodel.2018.10.005.

Peters, R., Vovides, A.G., Luna, S., Grüters, U., Berger, U., 2014. Changes in allometric relations of mangrove trees due to resource availability – a new mechanistic modelling approach. Ecol. Model. 283, 53–61. https://doi.org/10.1016/J.ECOLMODEL.2014.04.001.

Peters, R., Walther, M., Lovelock, C., Jiang, J., Berger, U., 2020. The Interplay between Vegetation and Water in Mangroves: New Perspectives for Mangrove Stand Modelling and Ecological Research. Wetlands Ecology and Management. https://doi.org/10.1007/s11273-020-09733-0.

Planque, B., Aarflot, J.M., Buttay, L., Carroll, J.L., Fransner, F., Hansen, C., Husson, B., Langangen, Ø., Lindstrøm, U., Pedersen, T., Primicerio, R., Sivel, E., Skogen, M.D., Strombom, E., Stige, L.C., Varpe, Ø., Yoccoz, N.G., 2022. A standard protocol for describing the evaluation of ecological models. Ecol. Model. 471 https://doi.org/10.1016/j.ecolmodel.2022.110059.

Voinov, A., Shugart, H.H., 2013. 'Integronsters', integral and integrated modeling. Environ. Model. Software 39, 149–158. https://doi.org/10.1016/J.ENVSOFT.2012.05.014.

Vovides, A.G., Berger, U., Grueters, U., Guevara, R., Pommerening, A., Lara-Domínguez, A.L., López-Portillo, J., 2018a. Change in drivers of mangrove crown displacement along a salinity stress gradient. Funct. Ecol. 32 (12), 2753–2765. https://doi.org/10.1111/1365-2435.13218.

Vovides, A.G., Berger, U., López-Portillo-Portillo, J., 2018b. Data from: change in drivers of mangrove crown displacement along a salinity stress gradient. https://doi.org/10.5525/gla.researchdata.657.

Wilson, G., Aruliah, D.A., Brown, C.T., Chue Hong, N.P., Davis, M., Guy, R.T., Haddock, S.H.D., Huff, K.D., Mitchell, I.M., Plumbley, M.D., Waugh, B., White, E.P., Wilson, P., 2014. Best practices for scientific computing. PLoS Biol. 12 (1), e1001745 https://doi.org/10.1371/JOURNAL.PBIO.1001745.

Weiner, J., Stoll, P., Muller-Landau, H., Jasentuliyana, A., 2001. The effects of density, spatial pattern, and competitive symmetry on size variation in simulated plant populations. Am. Nat. 158 (4), 438–450. https://doi.org/10.1086/321988.

Wimmler, M.-C., Vovides, A.G., Peters, R., Walther, M., Nadezhdina, N., Berger, U., 2022. Root grafts matter for inter-tree water exchange – a quantification of water translocation between root grafted mangrove trees using field data and model-based indications. Ann. Bot. 130, 317–330. https://doi.org/10.1093/aob/mcac074.

Wyszomirski, T., Wyszomirska, I., Jarzyna, I., 1999. Simple mechanisms of size distribution dynamics in crowded and uncrowded virtual monocultures. In: Ecological Modelling, vol. 115.

Zhang, B., Deangelis, D.L., 2020. An overview of agent-based models in plant biology and ecology. Ann. Bot. 126, 539–557. https://doi.org/10.1093/aob/mcaa043.

Berger, U., Bell, A., Barton, C.M., Chappin, E., Filatova, T., Fronville, T., Lee, A., van Loon, E., Lorscheid, I., Meyer, M., Piou, C., Radchuk, V., Roxburgh, N., Schüler, L., Troost, C., Wijermans, N., Williams, T.G., Wimmler, M.-C., Grimm, V., 2024. Towards reusable building blocks for agent-based modelling and theory development. Environ. Model. Softw. This special issue.