PlantSimEngine: A Simulation Engine For The Soil-Plant-Atmosphere System

Rémi Vezy

▶ To cite this version:

PlantSimEngine: A Simulation Engine For The Soil-Plant-Atmosphere System

Rémi Vezy ![ORCID] [1,2]

1 CIRAD, UMR AMAP, F-34398 Montpellier, France. **2** AMAP, Univ Montpellier, CIRAD, CNRS, INRAE, IRD, Montpellier, France.

Summary

`PlantSimEngine` provides a high-level interface for model generation for the simulation and modelling of plants, soil and atmosphere, with a focus on ease of use and computational efficiency. It is designed to help researchers and practitioners prototype, implement and evaluate plant or crop models at any scale, without the hassle of technical computer science details. The package defines a framework for declaring processes and implementing associated models for their simulation, with key features including but not limited to:

- Easy definition of new processes, such as light interception, photosynthesis, growth, soil water transfer, etc…
- Fast, interactive prototyping of models, with constraints to help users avoid errors, but sensible defaults to avoid over-complicating the model writing process
- Automatically manage input and output variables, time-steps, objects, coupling of models with a dependency graph
- Switch between models without changing any code, with a simple syntax to define the model to use for a given process
- Reduce the degrees of freedom by fixing variables, passing measurements, or using a simpler model for a given process
- Fast computation, with 100th of nanoseconds for one model, two coupled models (see this [benchmark script](#)), or the full energy balance of a leaf using [PlantBiophysics.jl](#) ([Vezy & Treillou, 2023](#)), a package that uses PlantSimEngine
- Out of the box sequential, parallel (multi-threaded) or distributed (multi-process) computations over objects, time-steps and independent processes (thanks to [Floops.jl](#))
- Easily scalable, with methods for computing over objects, time-steps and even [Multi-Scale Tree Graphs](#) ([Vezy, 2023](#))
- Composable, allowing the use of any types as inputs such as [Unitful](#) to propagate units, or [MonteCarloMeasurements.jl](#) ([Carlson, 2020](#)) to propagate measurement error

Statement of need

There is a growing demand for robust and efficient tools for simulating and modeling plants, soil and atmosphere in the scientific community. Most often, models are defined by agronomists, plant or soil scientists, who are not necessarily familiar with the intricacies of computer science such as computational performance, parallelization, management of time-steps and objects.

Models are developed either in interpreted programming languages for easier and faster prototyping, or in compiled languages for faster computation. However, interpreted languages are often slow to execute, and compiled languages are slow to prototype in. Furthermore, the development of models is often an iterative, time-consuming process that requires hypothesis testing and frequent switch between models for comparing their different versions. It also

requires unit testing to ensure that the implementation is correct, and integration testing to evaluate the model within its framework.

This process is often tedious and error-prone, and requires a lot of time and effort. The need for a tool that can help researchers and practitioners prototype, implement and evaluate plant or crop models at any scale, without the hassle of technical computer science details, is therefore clear.

PlantSimEngine addresses this need by providing a flexible and user-friendly framework for declaring processes and implementing models for their simulation. The package focuses on key aspects of simulation and modeling, such as ease of definition of new processes, fast and interactive prototyping of models, streamlined management of input and output variables, and the ability to switch between models without changing any underlying code. It also provides a way to test the models or forcing processes to observations to reduce the number of degrees of freedom. It also offers fast computation, scalability, and composability, making it ideal for developing Functional-Structural Plant Models (FSPMs) and crop models. For example the simulation of a toy model for leaf area over a year at daily time-scale took 260 μs to perform (*i.e.* 688 ns per day) on an M1 MacBook Pro 14, and 275 μs (*i.e.* 756 ns per day) when coupled to a model for light interception based on the Beer-Lambert law of light extinction (see provided benchmark script), which shows that PlantSimEngine scales well with several models. These performances are on par with what could be expected if the package was developed using a compiled language such as Fortran or C, and is by far faster than any interpreted language implementation.

The idea behind the innovative features of PlantSimEngine is to enable users to accurately model, predict and analyze the behavior of complex processes to improve decision making and optimize process design.

State of the field

PlantSimEngine is a state-of-the-art plant simulation software that offers significant advantages or different pattern designs over existing tools such as OpenAlea (Pradal et al., 2008), STICS (Brisson et al., 1998), APSIM (Brown et al., 2014; Holzworth et al., 2014) or DSSAT (Jones et al., 2003).

The use of Julia programming language in PlantSimEngine allows for quick and easy prototyping of models compared to compiled languages (STICS, APSIM, DSSAT) and can achieve significantly better performance than typical interpreted languages (*e.g.* Python used in OpenAlea), without the need for translation into another compiled language. For example PlantBiophysics.jl, a Julia package that implements ecophysiological models using PlantSimEngine.jl is 38649 times faster than the same implementation in R in plantecophys (Duursma, 2015).

The package's impressive features are primarily provided by the Julia programming language. Julia is a high-level, high-performance, and dynamic programming language widely used in various scientific fields, and particularly for biology (Roesch et al., 2023). The Julia community has recently developed remarkable tools such as Cropbox.jl (Yun & Kim, 2022), and CliMA Land (Wang et al., 2021).

Cropbox.jl is a declarative crop modeling framework that share a common objective with PlantSimEngine: simplifying the process of defining models by eliminating the complexities of underlying simulations. While both frameworks can be applied across a wide range of scales, including land surface models (LSM), crop models, and functional-structural plant models (FSPM), they differ in their implementation strategies.

Cropbox.jl defines a domain-specific language (DSL) tailored specifically for crop modeling, providing an intuitive approach for model definition. On the other hand, PlantSimEngine prioritizes speed as a fundamental aspect, ensuring efficient computations and enabling scalability right

from the start. This is accomplished through meticulous considerations, such as type stability and pre-allocation to optimize performance. Type stability in PlantSimEngine ensures that variables maintain consistent types throughout computations, which leads to efficient execution. Additionally, the compatibility of PlantSimEngine with MultiScaleTreeGraph.jl (Vezy, 2023) further enhances its scalability and facilitates multi-scale computations. MultiScaleTreeGraph.jl provides a powerful framework for representing and analyzing hierarchical structures, enabling PlantSimEngine to seamlessly handle complex systems spanning multiple scales.

CliMA Land is a next generation LSM that can simulate the Soil-Plant-Atmosphere continuum on a large scale. One interesting feature from PlantSimEngine that could be useful for CliMA Land is the ease of model coupling, including the concepts of hard and soft-coupled models, and automatic computation of the dependency graph.

The automatic computation of the dependency graph of models and the uniform API of PlantSimEngine enables users to switch models without having to modify any underlying code. This is a significant advantage over CliMA Land and other existing tools for the development of complex soil-plant-atmosphere models. Previous initiatives have been implemented in OpenAlea to provide a way to build a dependency graph for soft-dependencies (*i.e.* independent models that are coupled via inputs and outputs) close to PlantSimEngine dependency management, but lacks compatibility with hard-dependencies, *i.e.* models that make an explicit, hard-coded call to another model. PlantSimEngine leverages Julia's multiple-dispatch to automatically compute the full dependency graph at compile time, including hard-dependencies. PlantSimEngine also offers a simple and generic API to define methods for model calibration that are used to programmatically calibrate models and sub-models.

Additionally, PlantSimEngine offers the ability to reduce the number of degrees of freedom by fixing variables, passing measurements, or using a simpler model for a given process, making it an easier and more flexible solution for plant simulation.

PlantSimEngine's approach streamlines the process of model development by automatically managing model coupling, time-steps, parallelization, input and output variables, and the type of objects used for simulations (vectors, dictionaries, and multi-scale tree graphs).

Mention

PlantSimEngine is used in the following packages:

- PlantBiophysics.jl, for the simulation of biophysical processes for plants such as photosynthesis, conductance for heat, water vapor and CO_2, latent, sensible energy fluxes, net radiation and temperature

- XPalm.jl, an experimental crop model for oil palm

References

Brisson, N., Mary, B., Ripoche, D., Jeuffroy, M.-H., Ruget, F., Nicoullaud, B. B., Gate, P., Devienne-Barret, F., Antonioletti, R., Dürr, C., Richard, G., Beaudoin, N., Recous, S., Tayot, X., Plenet, D., Cellier, P., Machet, J. M., Meynard, J. M. J. M., & Delecolle, R. (1998). STICS: A generic model for the simulation of crops and their water and nitrogen balances. I. Theory and parameterization applied to wheat and corn. *Agronomie*, *18*(5-6), 311–346. https://doi.org/10.1051/agro:19980501

Brown, H. E., Huth, N. I., Holzworth, D. P., Teixeira, E. I., Zyskowski, R. F., Hargreaves, J. N. G., & Moot, D. J. (2014). Plant Modelling Framework: Software for building and running crop models on the APSIM platform. *Environmental Modelling & Software*, *62*, 385–398. https://doi.org/10.1016/j.envsoft.2014.09.005

Carlson, F. B. (2020). MonteCarloMeasurements.jl: Nonlinear Propagation of Arbitrary Multivariate Distributions by means of Method Overloading. *CoRR*, *abs/2001.07625*. https://doi.org/10.48550/arXiv.2001.07625

Duursma, R. A. (2015). Plantecophys - An R Package for Analysing and Modelling Leaf Gas Exchange Data. *PLOS ONE*, *10*(11), e0143346. https://doi.org/10.1371/journal.pone.0143346

Holzworth, D. P., Huth, N. I., deVoil, P. G., Zurcher, E. J., Herrmann, N. I., McLean, G., Chenu, K., Oosterom, E. J. van, Snow, V., Murphy, C., Moore, A. D., Brown, H., Whish, J. P. M., Verrall, S., Fainges, J., Bell, L. W., Peake, A. S., Poulton, P. L., Hochman, Z., … Keating, B. A. (2014). APSIM – Evolution towards a new generation of agricultural systems simulation. *Environmental Modelling & Software*, *62*, 327–350. https://doi.org/10.1016/j.envsoft.2014.07.009

Jones, J. W., Hoogenboom, G., Porter, C. H., Boote, K. J., Batchelor, W. D., Hunt, L., Wilkens, P. W., Singh, U., Gijsman, A. J., & Ritchie, J. T. (2003). The DSSAT cropping system model. *European Journal of Agronomy*, *18*(3-4), 235–265.

Pradal, C., Dufour-Kowalski, S., Boudon, F., Fournier, C., & Godin, C. (2008). OpenAlea: A visual programming and component-based software platform for plant modelling. *Functional Plant Biology*, *35*(10), 751–760. https://doi.org/10.1071/FP08084

Roesch, E., Greener, J. G., MacLean, A. L., Nassar, H., Rackauckas, C., Holy, T. E., & Stumpf, M. P. H. (2023). Julia for biologists. *Nature Methods*. https://doi.org/10.1038/s41592-023-01832-z

Vezy, R. (2023). *MultiScaleTreeGraph.jl: Read, Write, Analyze, Compute and Plot Multi-scale Tree Graph Files.* Zenodo. https://doi.org/10.5281/zenodo.7763416

Vezy, R., & Treillou, S. (2023). *VEZY/PlantBiophysics.jl: v0.9.2.* Zenodo. https://doi.org/10.5281/zenodo.7771412

Wang, Y., Köhler, P., He, L., Doughty, R., Braghiere, R. K., Wood, J. D., & Frankenberg, C. (2021). Testing stomatal models at the stand level in deciduous angiosperm and evergreen gymnosperm forests using CliMA Land (v0.1). *Geoscientific Model Development*, *14*(11), 6741–6763. https://doi.org/10.5194/gmd-14-6741-2021

Yun, K., & Kim, S.-H. (2022). Cropbox: A declarative crop modelling framework. *In Silico Plants*, *5*(1). https://doi.org/10.1093/insilicoplants/diac021