MEAGAN LANG

CROPS IN SILICO HACKATHON 2021

BUT FIRST…

NOTEBOOK PREP

PREP CHECKLIST

1. SIGN-UP FOR GITHUB

   We will be using GitHub Issues to debug

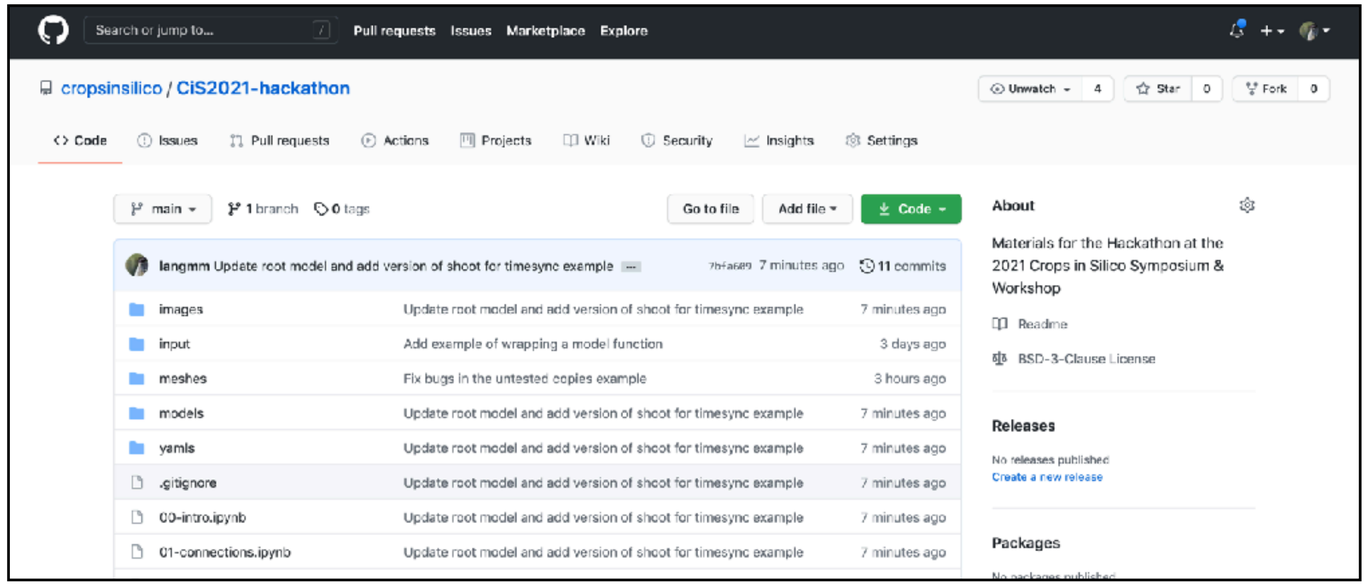
2. OPEN THE PROJECT MATERIALS
3. START A MYBINDER INSTANCE

   We will be using Jupyter notebooks via MyBinder to avoid local installations (instructions for installing locally are found in the README)

Materials: https://github.com/cropsinsilico/CiS2021-hackathon

Launch binder in a **new** window/tab

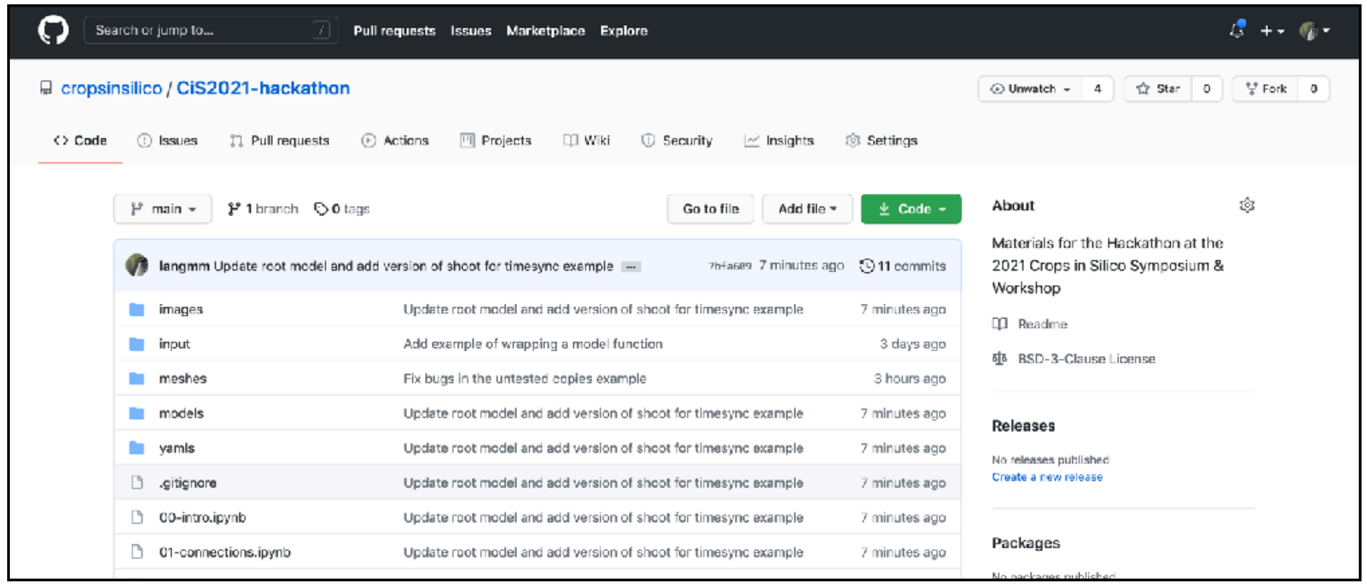
Keep the materials repo open, we will need it later



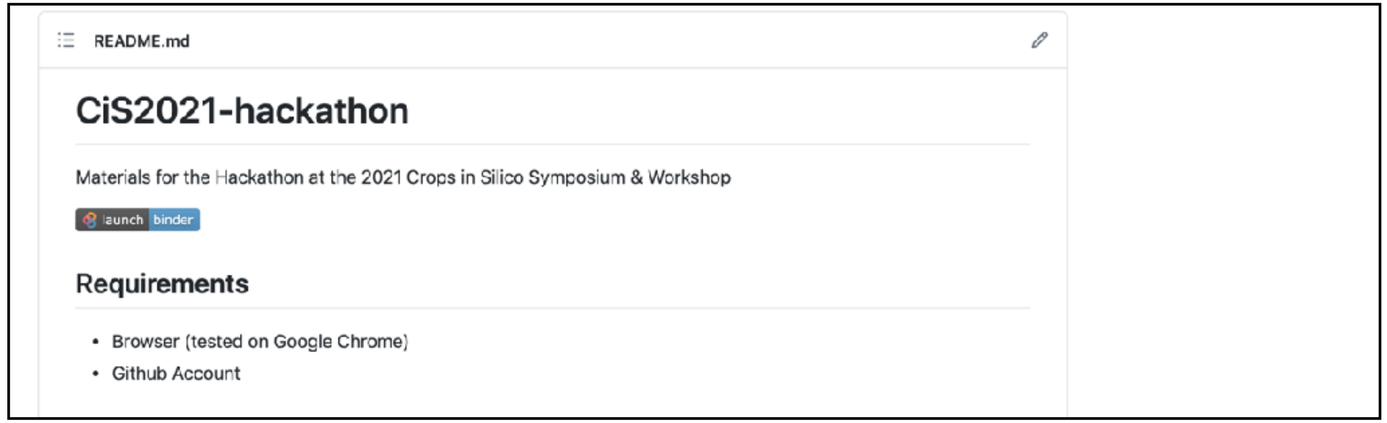
Materials: https://github.com/cropsinsilico/CiS2021-hackathon

Launch binder in a **new** window/tab

Keep the materials repo open, we will need it later



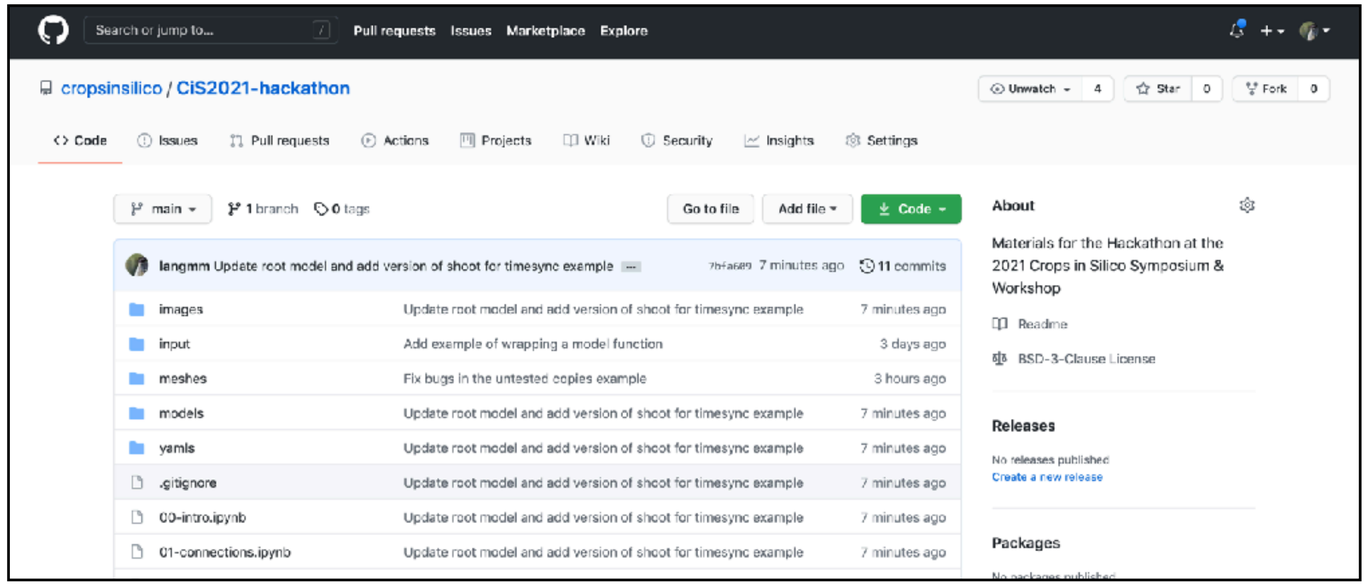
SCROLL



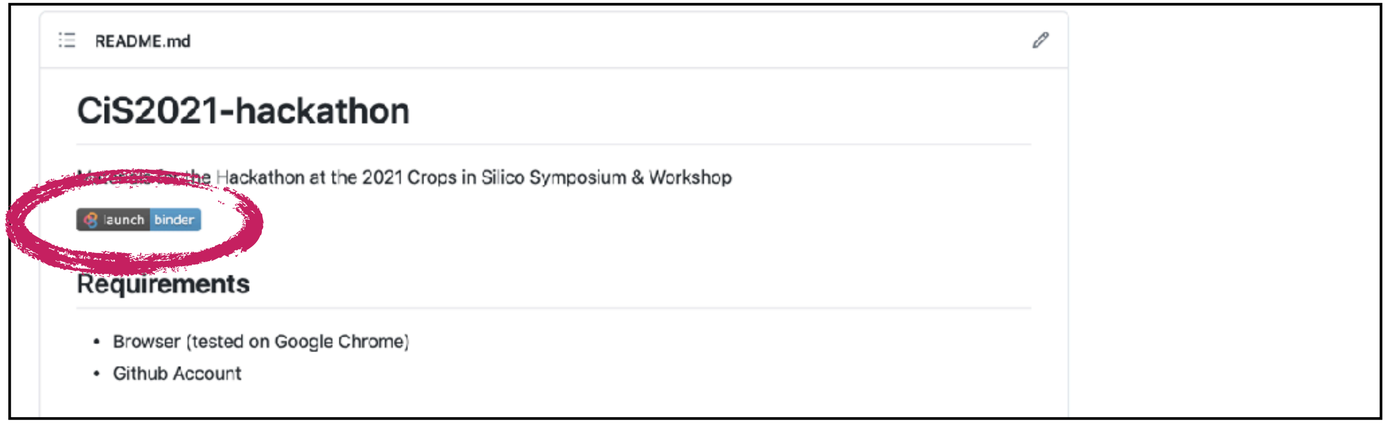
Materials: https://github.com/cropsinsilico/CiS2021-hackathon

Launch binder in a **new** window/tab

Keep the materials repo open, we will need it later



SCROLL



Materials: https://github.com/cropsinsilico/CiS2021-hackathon

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessary. We will be using Github Issues to track problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder icon (or the link below).

  It may take a few moments to initialize. If you encounter an error, open an issue and try with another browser. If you still cannot launch the binder, you can install the materials on your machine by following the instructions at one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

DOCUMENTATION

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessary. We will be using Github Issues to track problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder icon (or the link below).

  It may take a few moments to initialize. If you encounter an error, open an issue and try with another browser. If you still cannot launch the binder, you can install the materials on your machine by following the instructions at one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

DOCUMENTATION

DOCUMENTATION

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessa
  problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder

  It may take a few moments to initialize. If you encounter an error, ope
  you still cannot launch the binder, you can install the materials on you
  one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

Notes on Autowrapping C/C++ Model Functions
Conditional I/O
Transformed I/O
Timestep Synchronization
YAML Files
Configuration Files
Units
C-Style Format Strings
Debugging
OpenMP Threading in Models
Examples
Advanced
Development
Publications
Welcome to the 2018 Crops in Silico Hackathon!
Welcome to the 2019 Crops in Silico Hackathon!
Welcome to the 2021 Crops in Silico Hackathon!
Setup
Debugging
Introduction to Yggdrasil
Introduction to Jupyter Notebooks
Command Line Interfaces (CLIs)

» Welcome to the 2021 Crops in Silico Hackathon! View page source

Welcome to the 2021 Crops in Silico Hackathon!

These materials walk through some of the basics to using yggdrasil to connect models using Jupyter notebooks. These exercises were created for the 2021 Crops in Silico hackathon, but can be used by anyone who would like to learn how to use yggdrasil.

Contents:

- Setup
  - MyBinder
  - Local Install
  - Docker Container
- Debugging
- Introduction to Yggdrasil
  - What
  - Why
  - How
  - Who
- Introduction to Jupyter Notebooks
  - Some notes about notebooks
- Command Line Interfaces (CLIs)
  - Running Integrations
  - Validating Integration YAML(s)

LIVE FEEDBACK

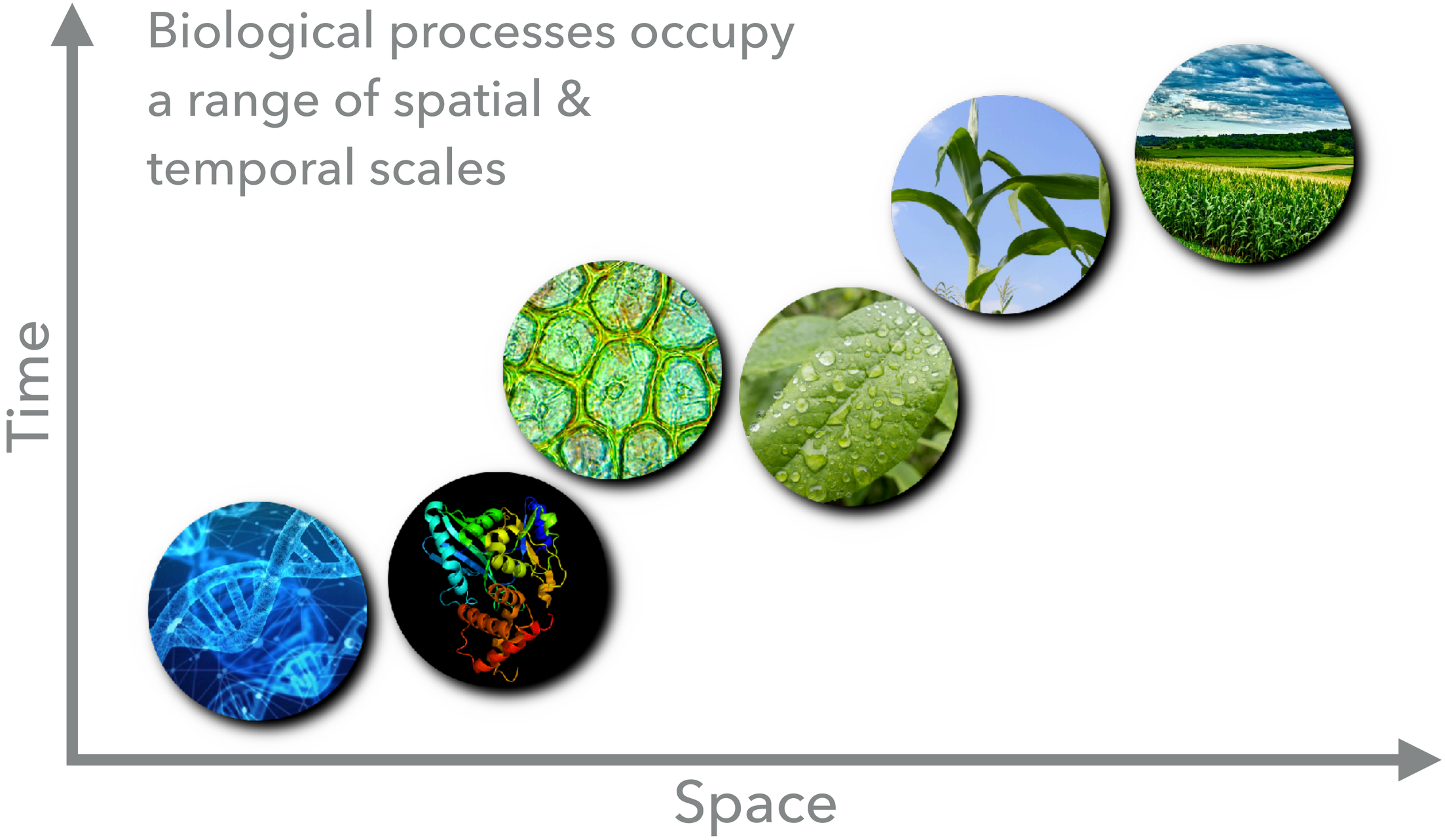
“Slow down I’m still waiting for a previous step to finish!”

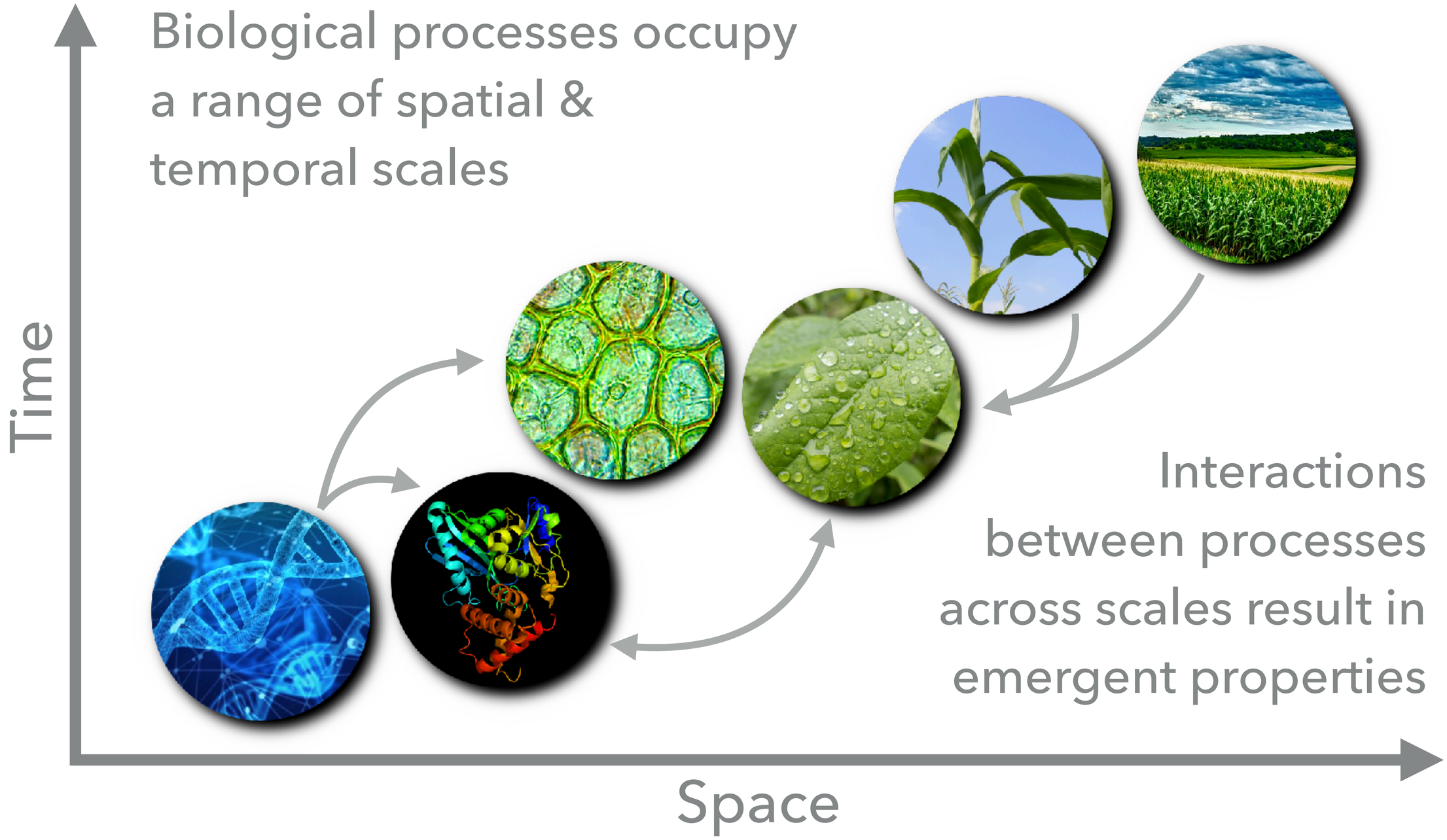
“I’ve caught up" or “ I have finished the task you are asking about”

OK BACK TO THE…

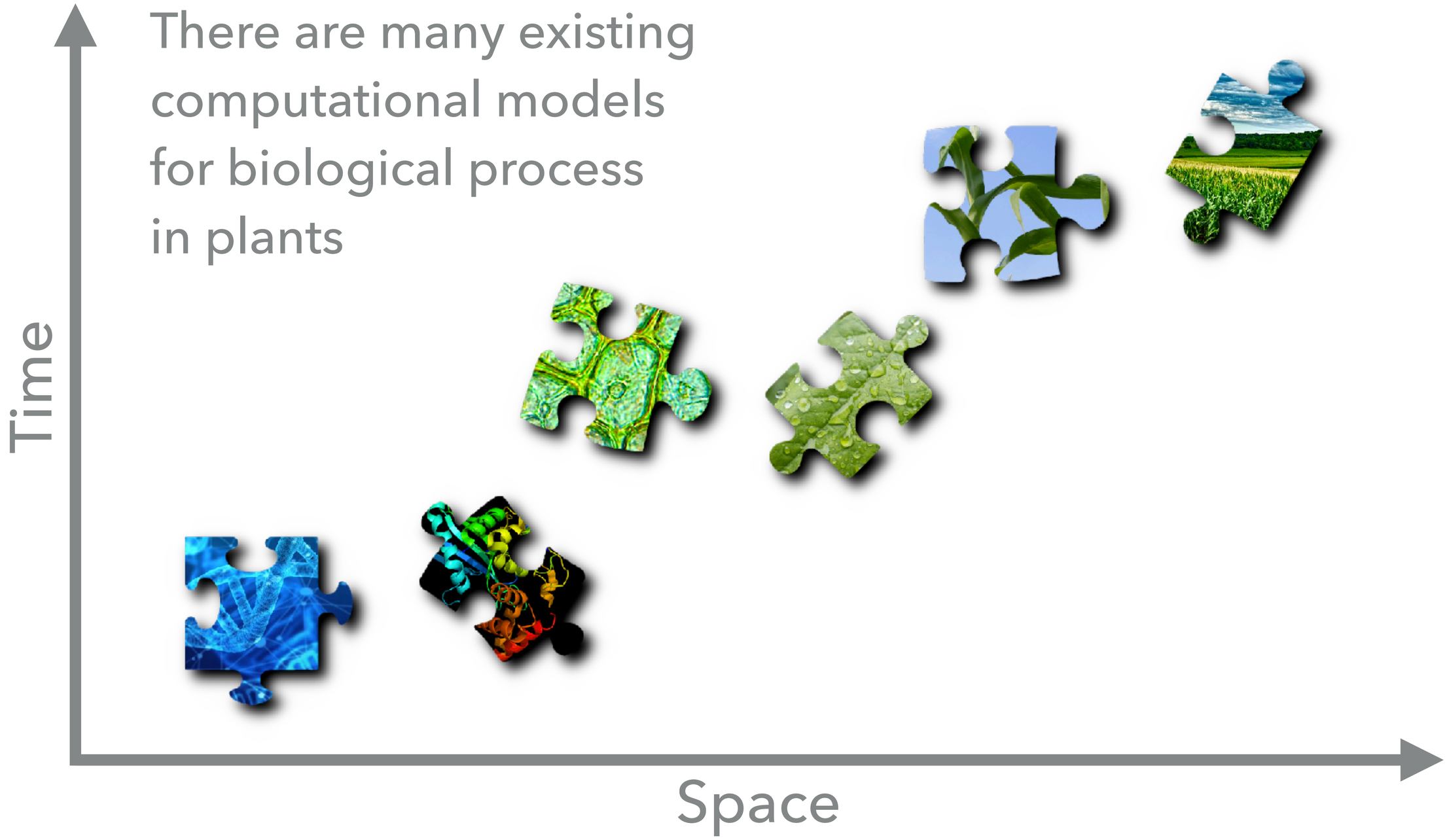
INTRODUCTION

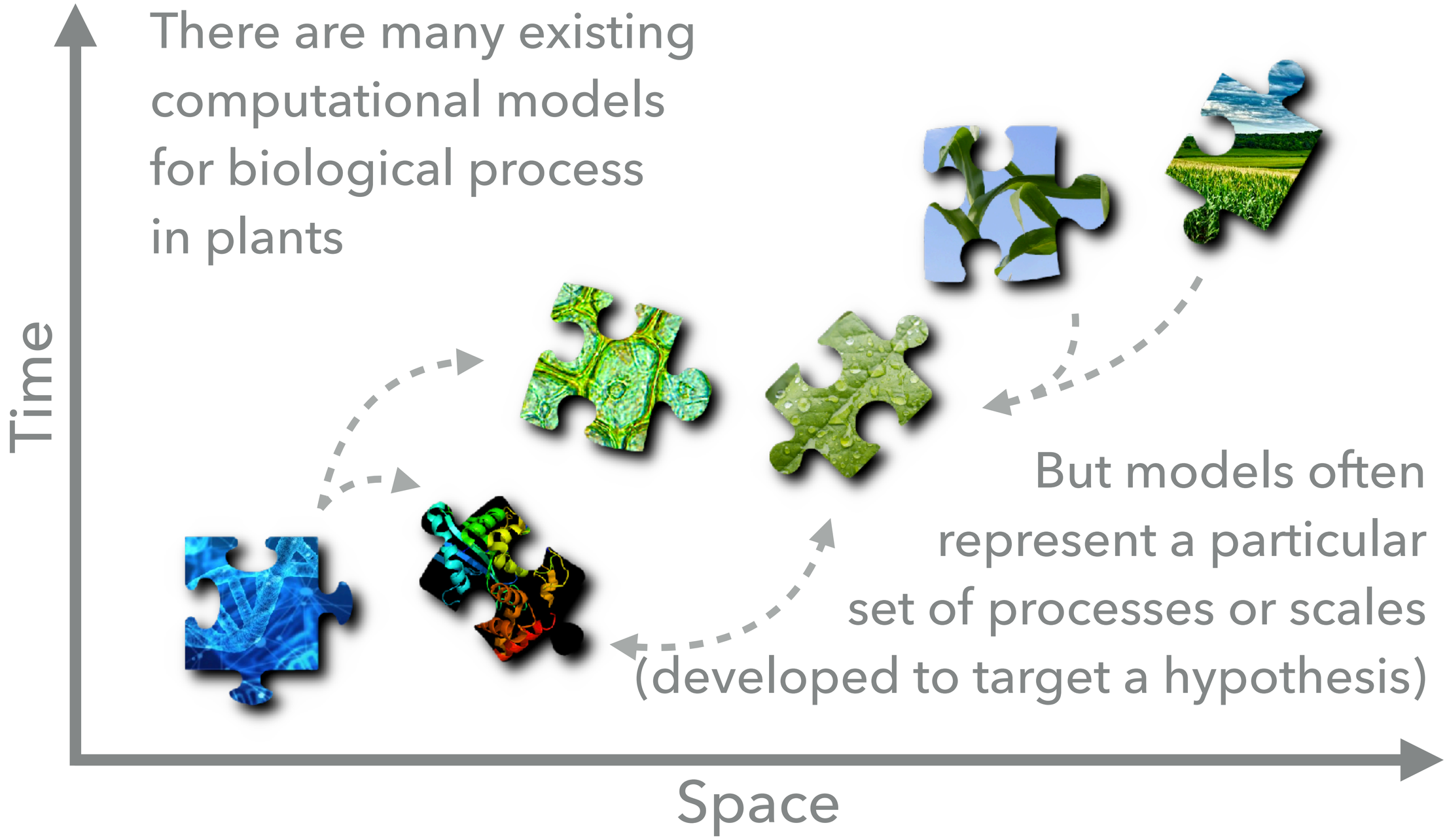
Biological processes occupy a range of spatial & temporal scales




Biological processes occupy
a range of spatial &
temporal scales
Time
Space
Interactions
between processes
across scales result in
emergent properties

There are many existing computational models for biological process in plants

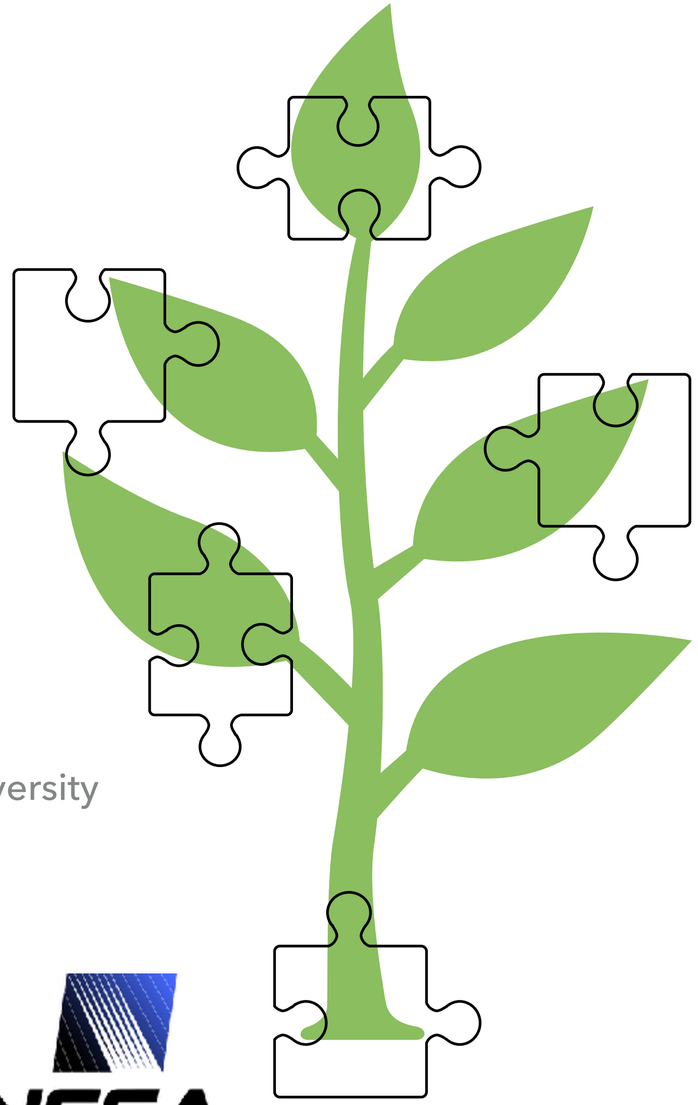



There are many existing computational models for biological process in plants
But models often represent a particular set of processes or scales (developed to target a hypothesis)
Time
Space

Crops[1] *in silico*

ILLINOIS

In partnership with
Oxford University
Pennsylvania State University
Purdue University
University of Nebraska



Funded by

FFAR ISEE NCSA

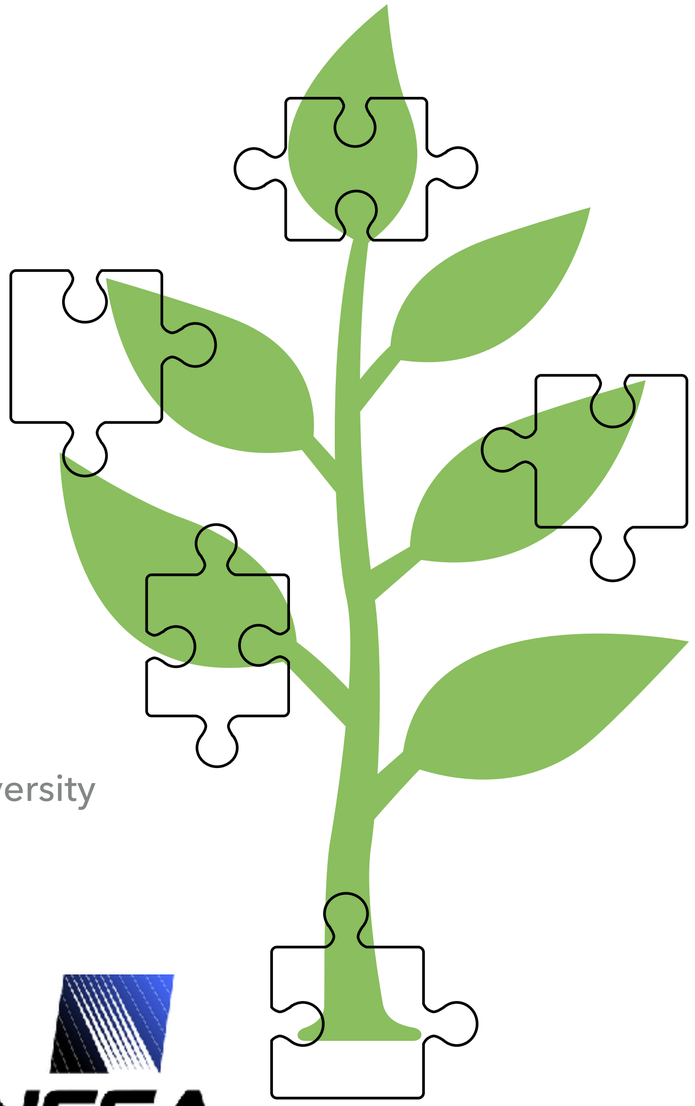
Models are pieces from different puzzles

We need an adapter

Crops[1] in silico

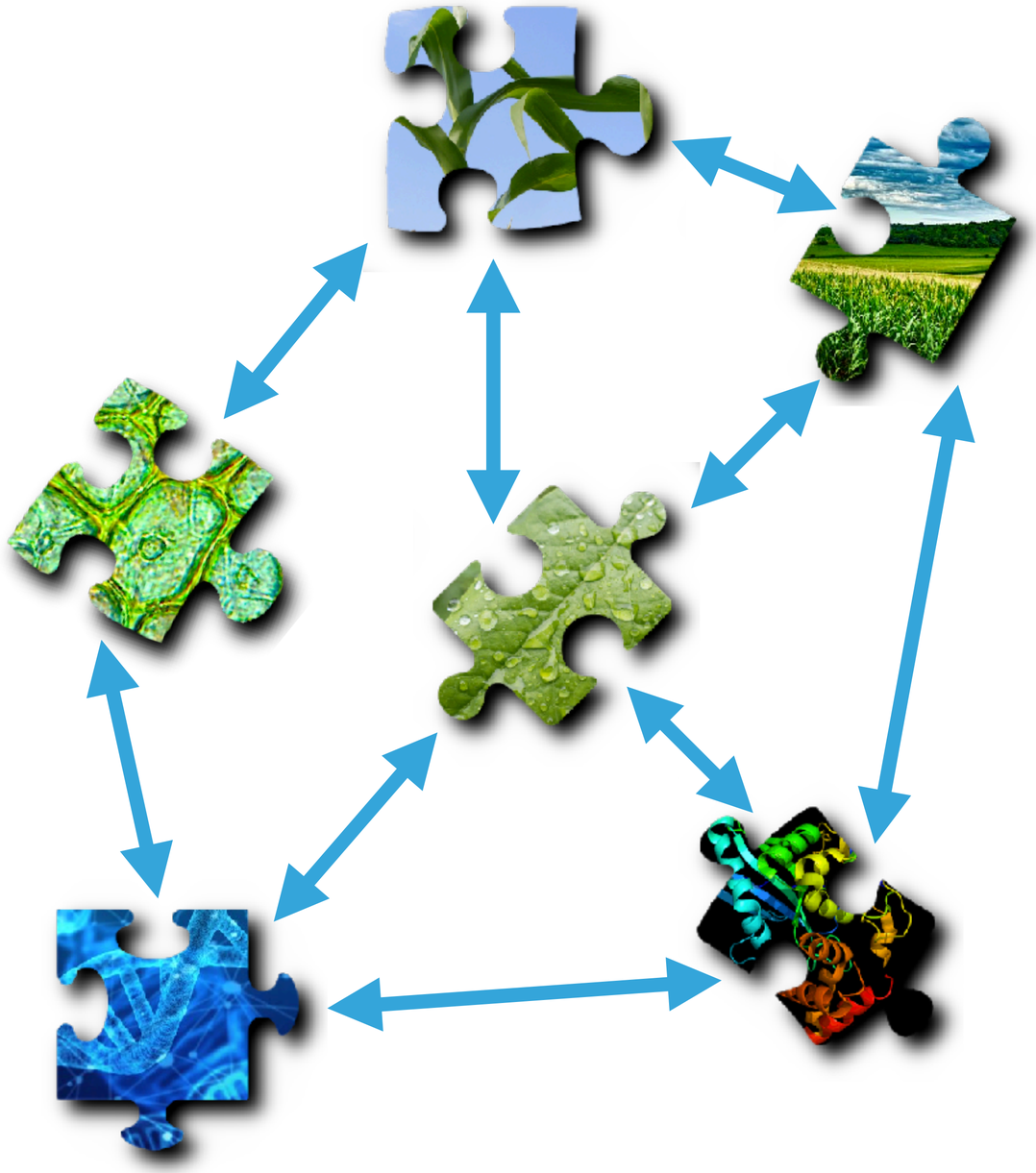
ILLINOIS

In partnership with
Oxford University
Pennsylvania State University
Purdue University
University of Nebraska



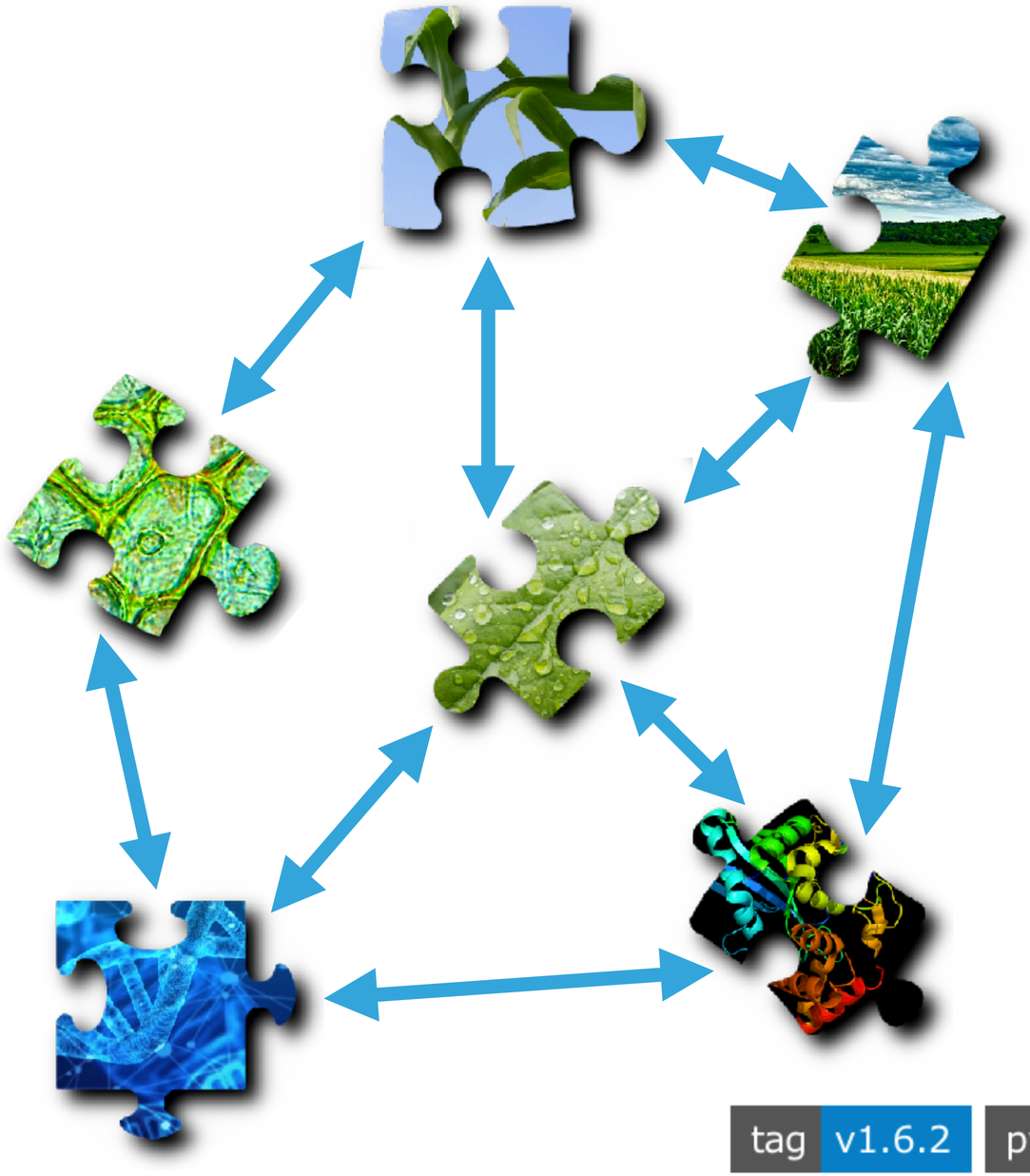
Funded by

FFAR ISEE NCSA



YGGDRASIL:

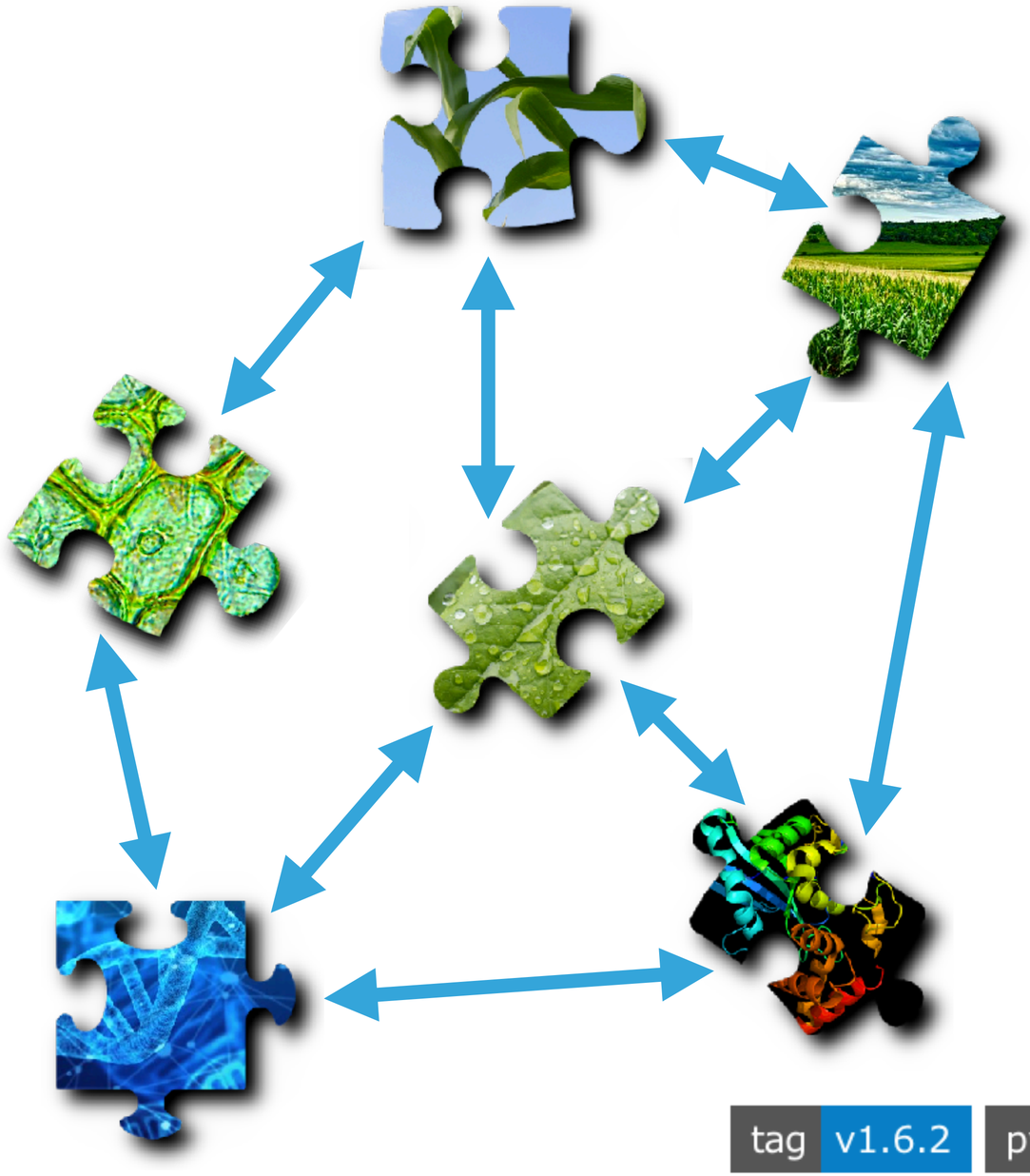
OPEN SOURCE PYTHON PACKAGE FOR CONNECTING MODELS ACROSS SCALES AND LANGUAGES



YGGDRASIL:

OPEN SOURCE PYTHON PACKAGE FOR CONNECTING MODELS ACROSS SCALES AND LANGUAGES

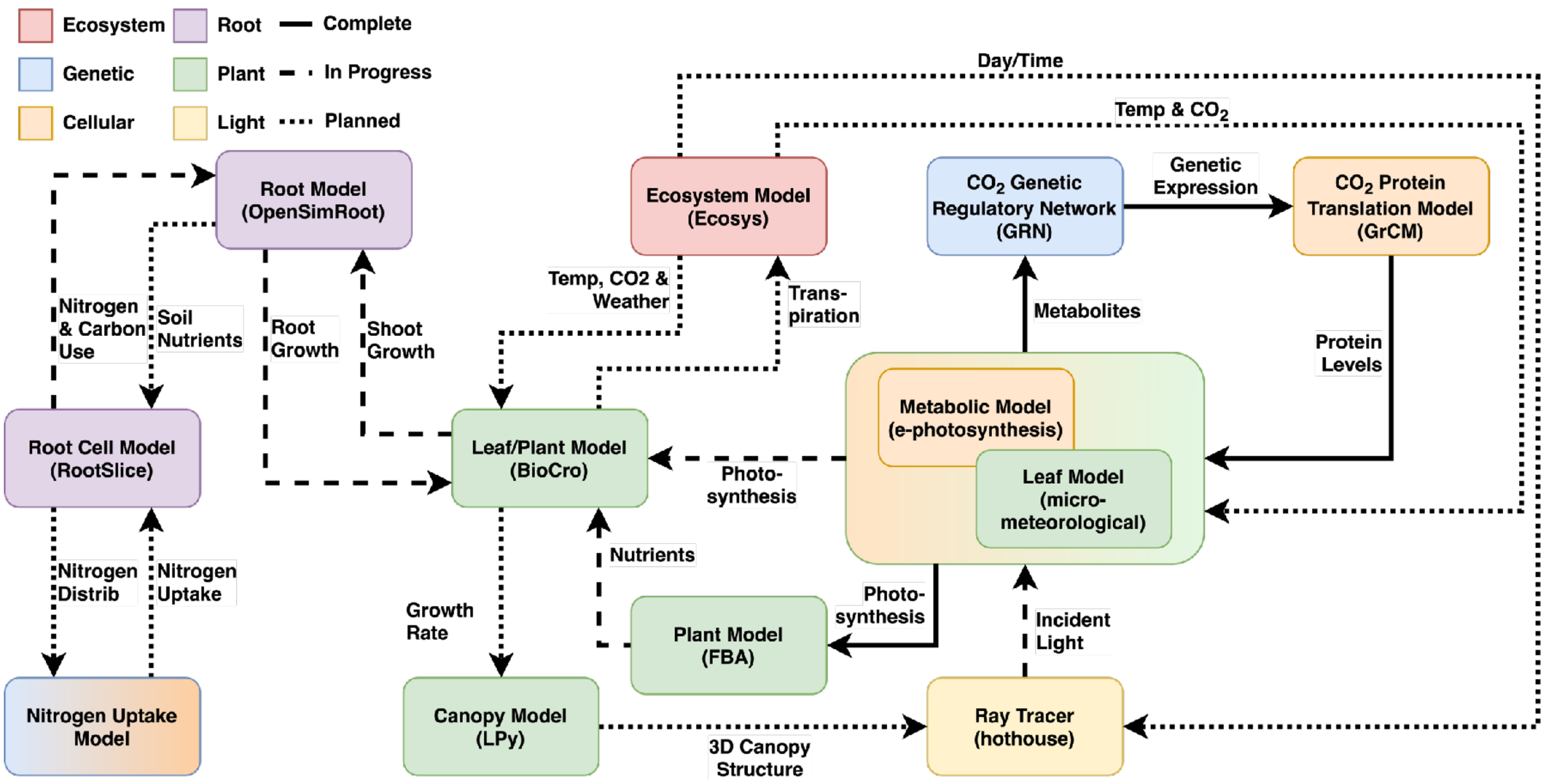
tag v1.6.2 | pypi v1.6.2 | build passing | coverage 100% | code style pep8

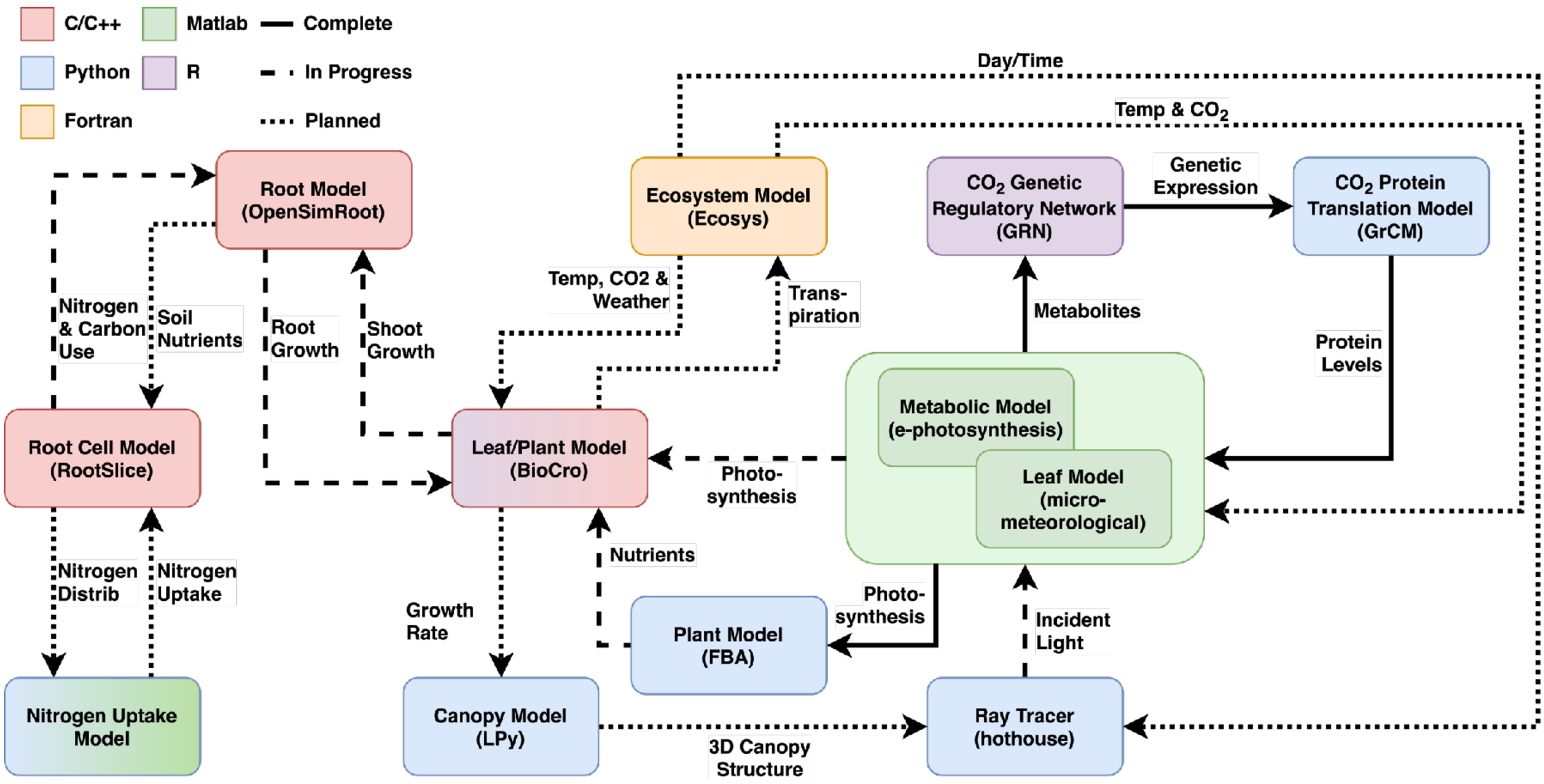


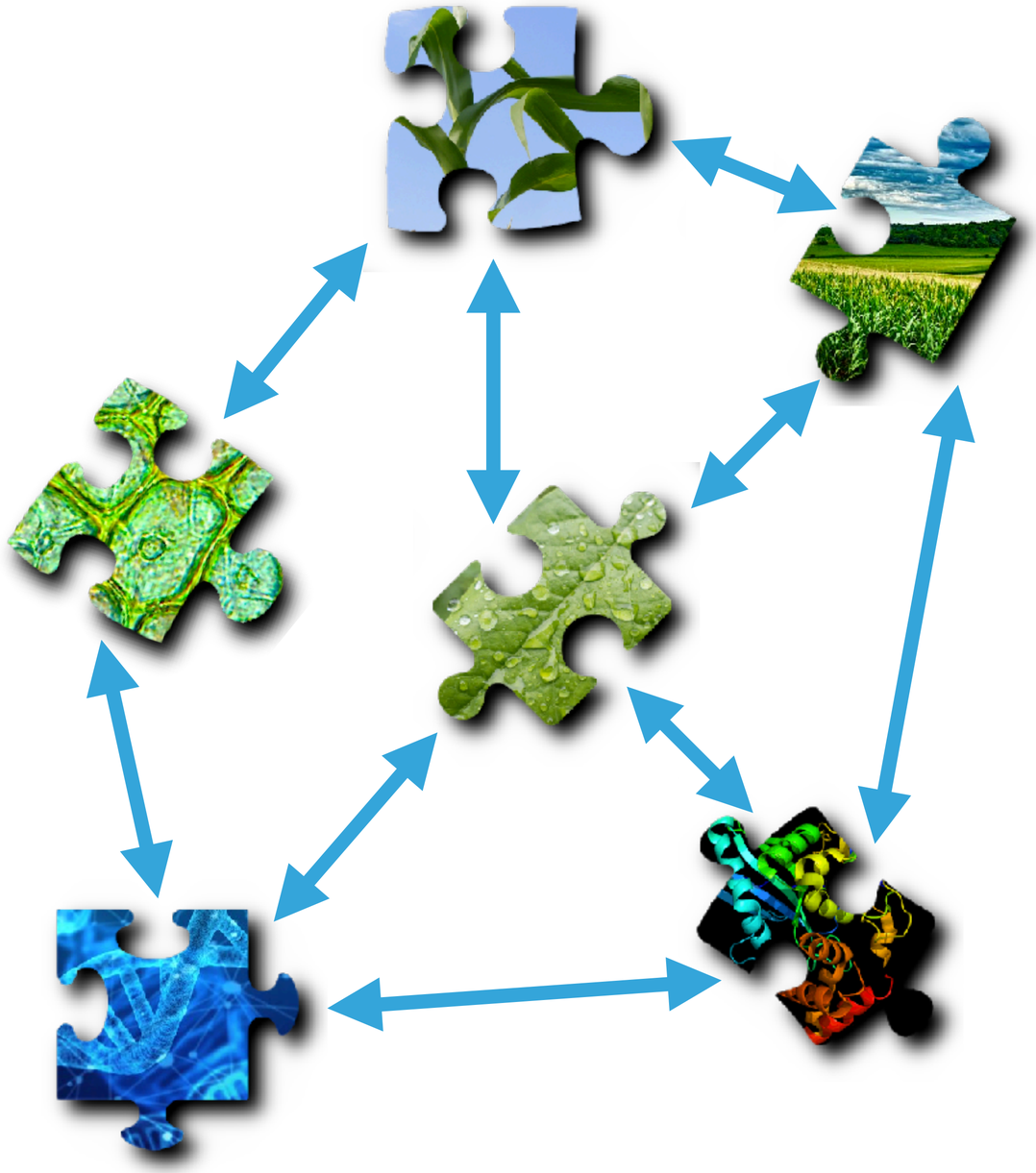
YGGDRASIL:

OPEN SOURCE PYTHON PACKAGE FOR CONNECTING MODELS ACROSS SCALES AND LANGUAGES

tag	v1.6.2	pypi	v1.6.2	build	passing	coverage	100%	code style	pep8
license	BSD	conda platforms	linux-64 | win-64 | osx-64						


Ecosystem
Root
Complete
Genetic
Plant
In Progress
Cellular
Light
Planned
Day/Time
Temp & CO_2
Root Model (OpenSimRoot)
Ecosystem Model (Ecosys)
CO_2 Genetic Regulatory Network (GRN)
Genetic Expression
CO_2 Protein Translation Model (GrCM)
Nitrogen & Carbon Use
Soil Nutrients
Root Growth
Shoot Growth
Temp, CO2 & Weather
Trans-piration
Metabolites
Protein Levels
Metabolic Model (e-photosynthesis)
Leaf Model (micro-meteorological)
Root Cell Model (RootSlice)
Leaf/Plant Model (BioCro)
Photo-synthesis
Nutrients
Nitrogen Distrib
Nitrogen Uptake
Growth Rate
Plant Model (FBA)
Photo-synthesis
Incident Light
Nitrogen Uptake Model
Canopy Model (LPy)
3D Canopy Structure
Ray Tracer (hothouse)


C/C++
Matlab
Complete
Python
R
In Progress
Fortran
Planned
Day/Time
Temp & CO_2
Root Model (OpenSimRoot)
Ecosystem Model (Ecosys)
CO_2 Genetic Regulatory Network (GRN)
Genetic Expression
CO_2 Protein Translation Model (GrCM)
Nitrogen & Carbon Use
Soil Nutrients
Root Growth
Shoot Growth
Temp, CO2 & Weather
Trans-piration
Metabolites
Protein Levels
Metabolic Model (e-photosynthesis)
Leaf Model (micro-meteorological)
Root Cell Model (RootSlice)
Leaf/Plant Model (BioCro)
Photo-synthesis
Nutrients
Nitrogen Distrib
Nitrogen Uptake
Growth Rate
Photo-synthesis
Incident Light
Plant Model (FBA)
Nitrogen Uptake Model
Canopy Model (LPy)
3D Canopy Structure
Ray Tracer (hothouse)



LANGUAGES

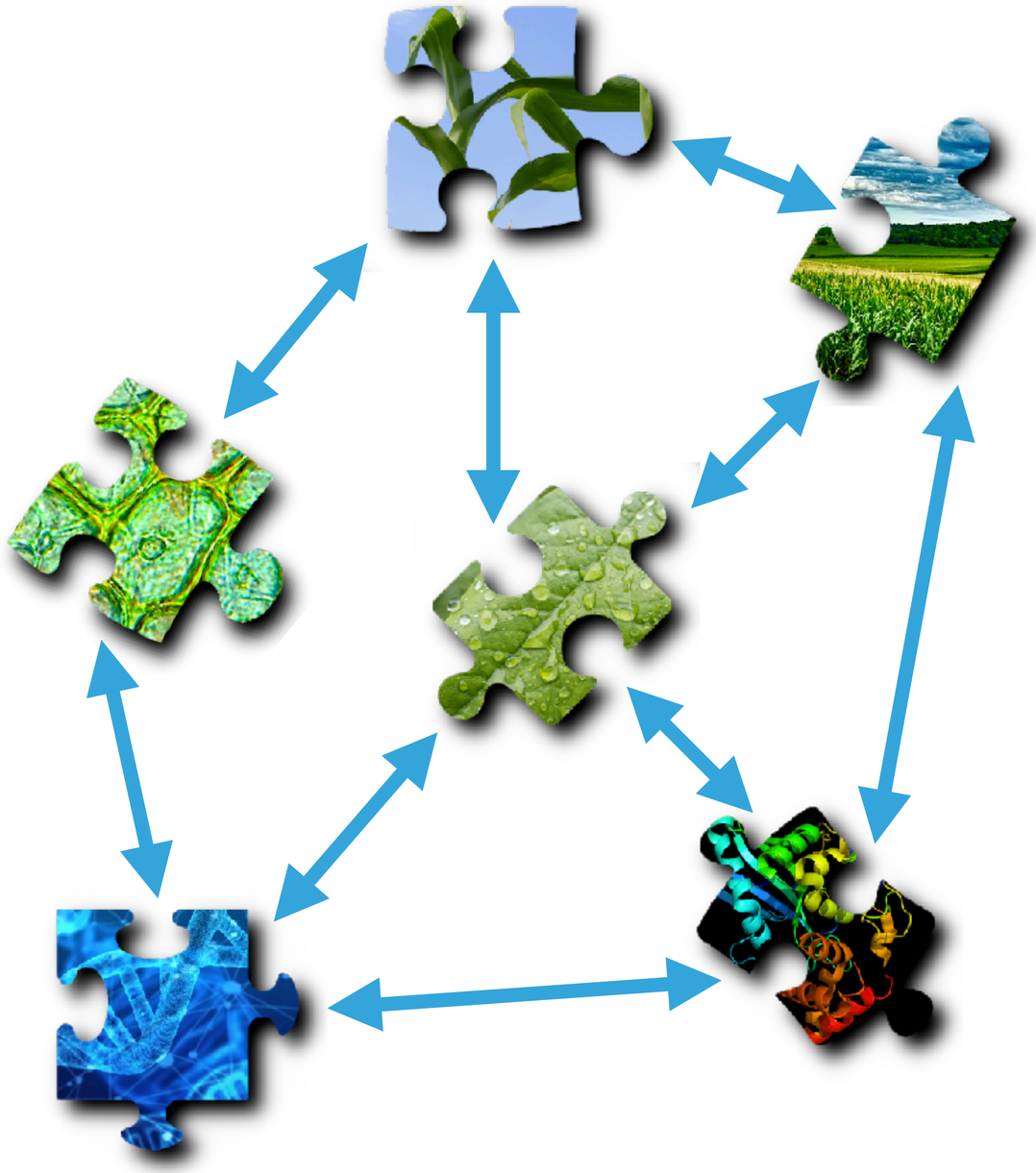
PYTHON

MATLAB

R

C/C++

FORTRAN ('03)



LANGUAGES

PYTHON

MATLAB

R

C/C++

FORTRAN ('03)

DOMAIN SPECIFIC LANGUAGES

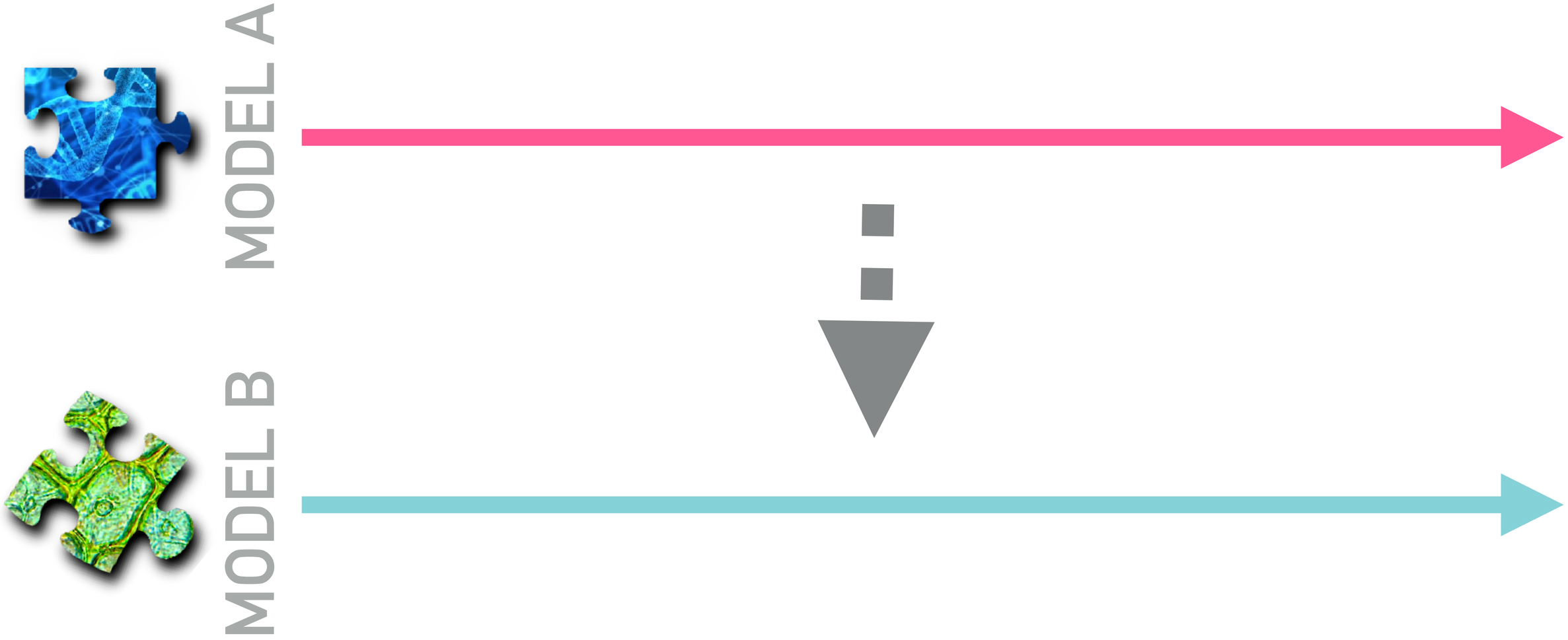
SBML

OPENSIMROOT XML

COMMUNICATION

PARALLEL EXECUTION

PARALLEL EXECUTION



ASYNCHRONOUS COMMUNICATION

(SEND FIRST)

MODEL A

MODEL B

ASYNCHRONOUS COMMUNICATION

(SEND FIRST)

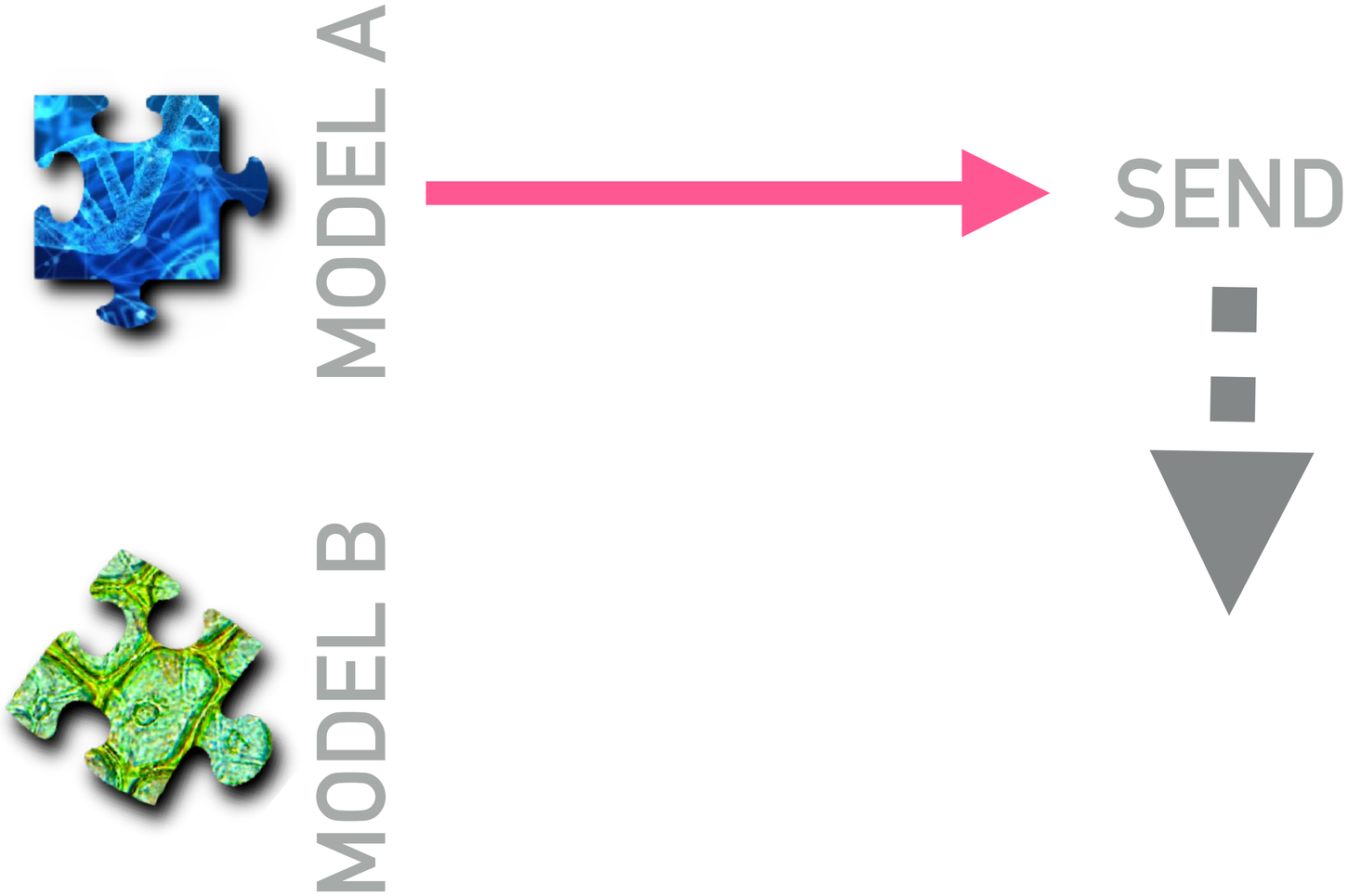
MODEL A

SEND

MODEL B

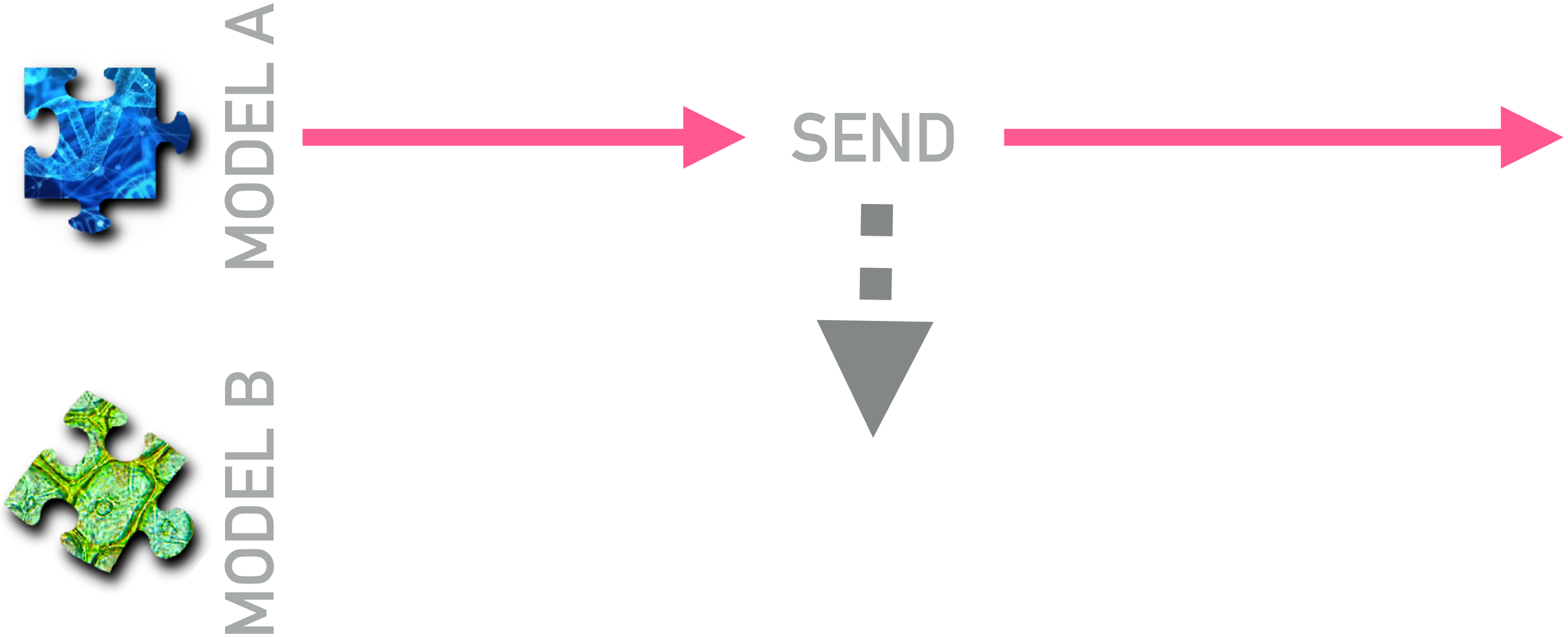
ASYNCHRONOUS COMMUNICATION

(SEND FIRST)



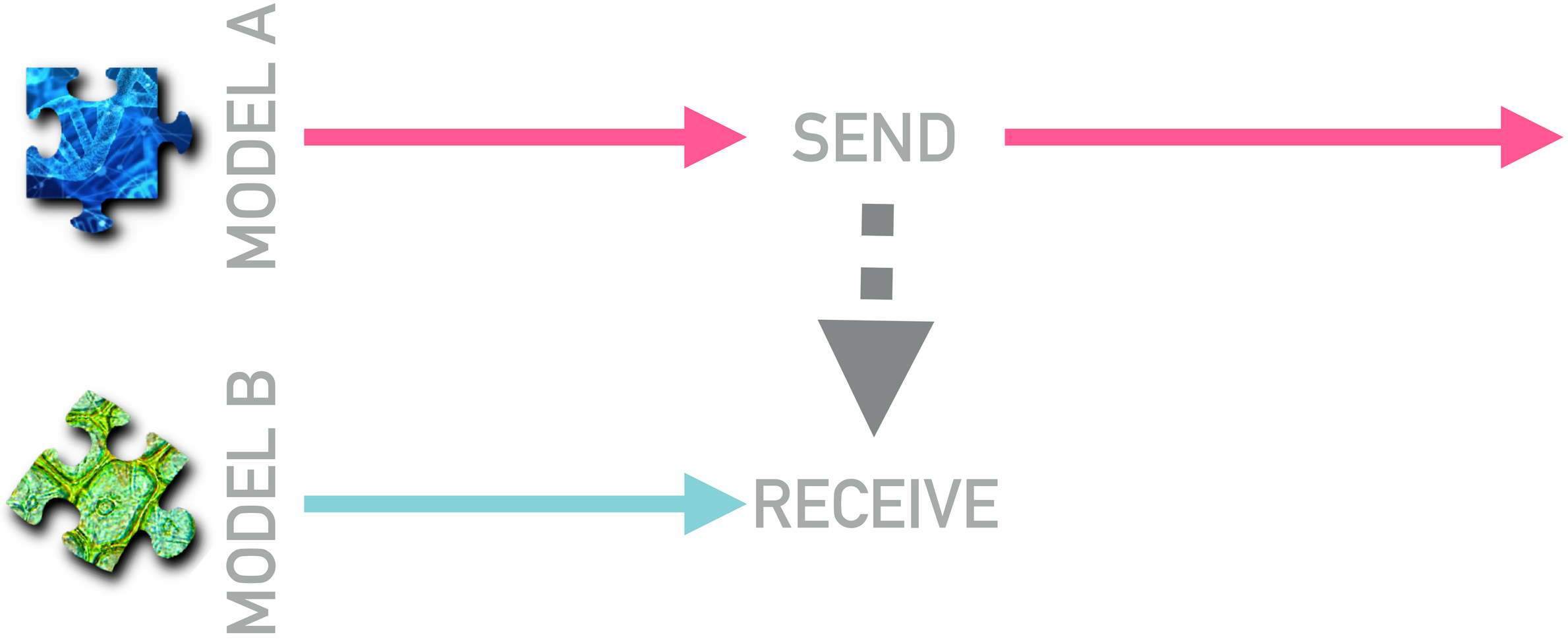
ASYNCHRONOUS COMMUNICATION

(SEND FIRST)



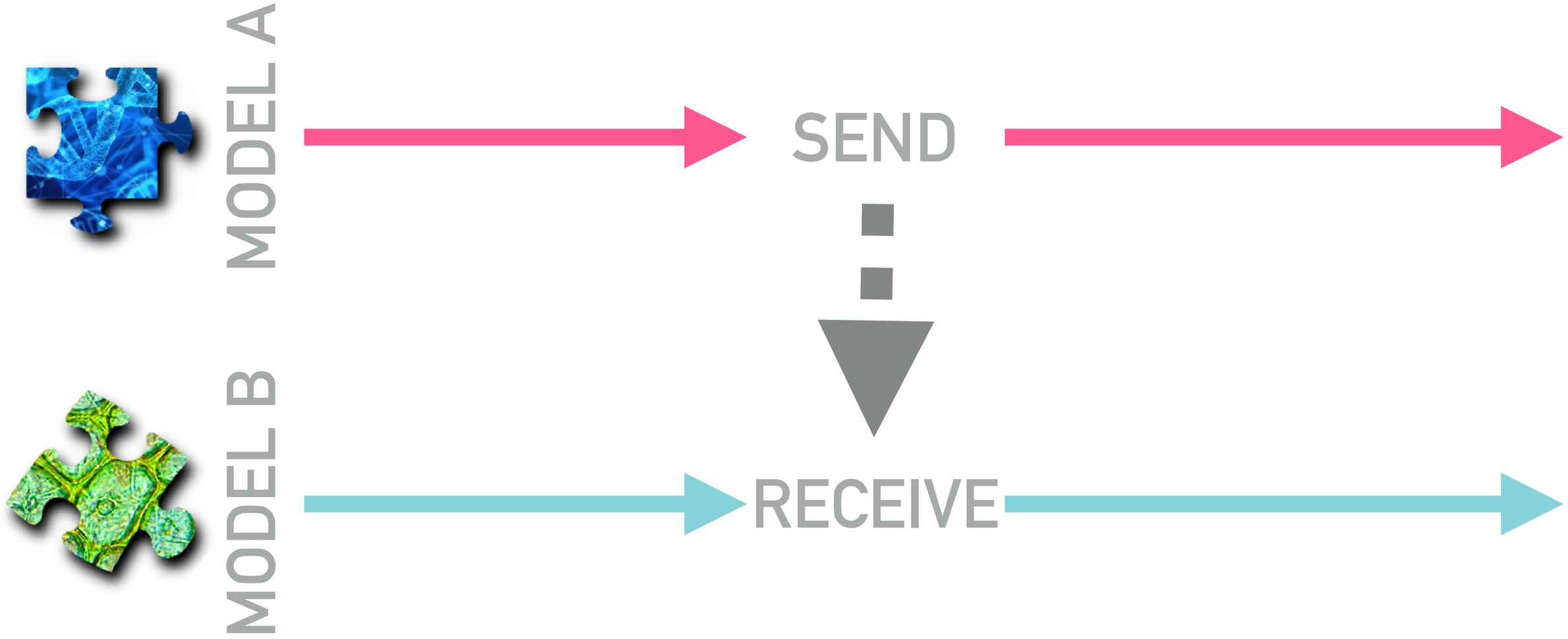
ASYNCHRONOUS COMMUNICATION

(SEND FIRST)



ASYNCHRONOUS COMMUNICATION

(SEND FIRST)



ASYNCHRONOUS COMMUNICATION

(RECEIVE FIRST)

MODEL A

MODEL B

ASYNCHRONOUS COMMUNICATION

(RECEIVE FIRST)

MODEL A

MODEL B

RECEIVE

ASYNCHRONOUS COMMUNICATION

(RECEIVE FIRST)

MODEL A

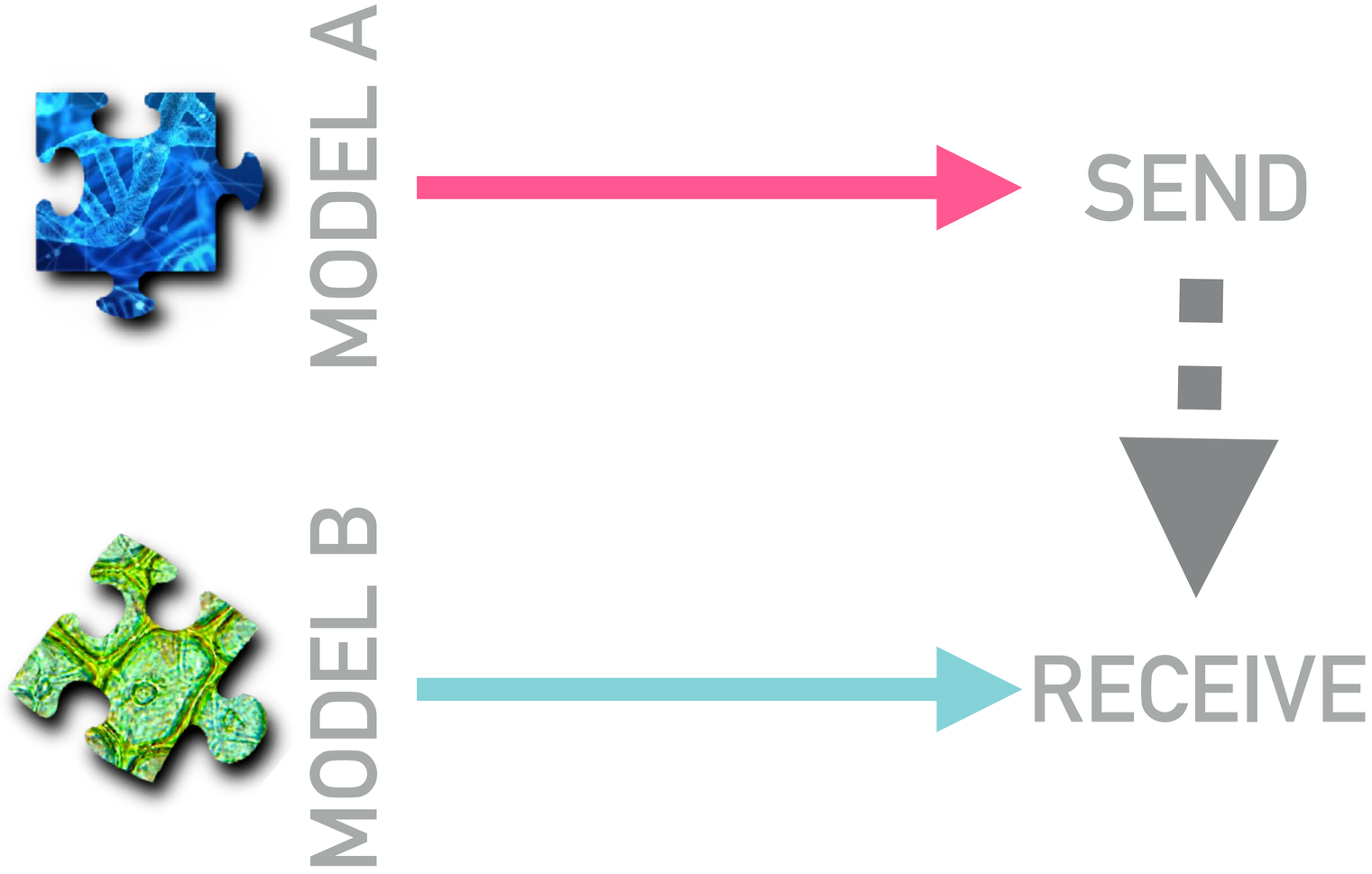
SEND

MODEL B

RECEIVE

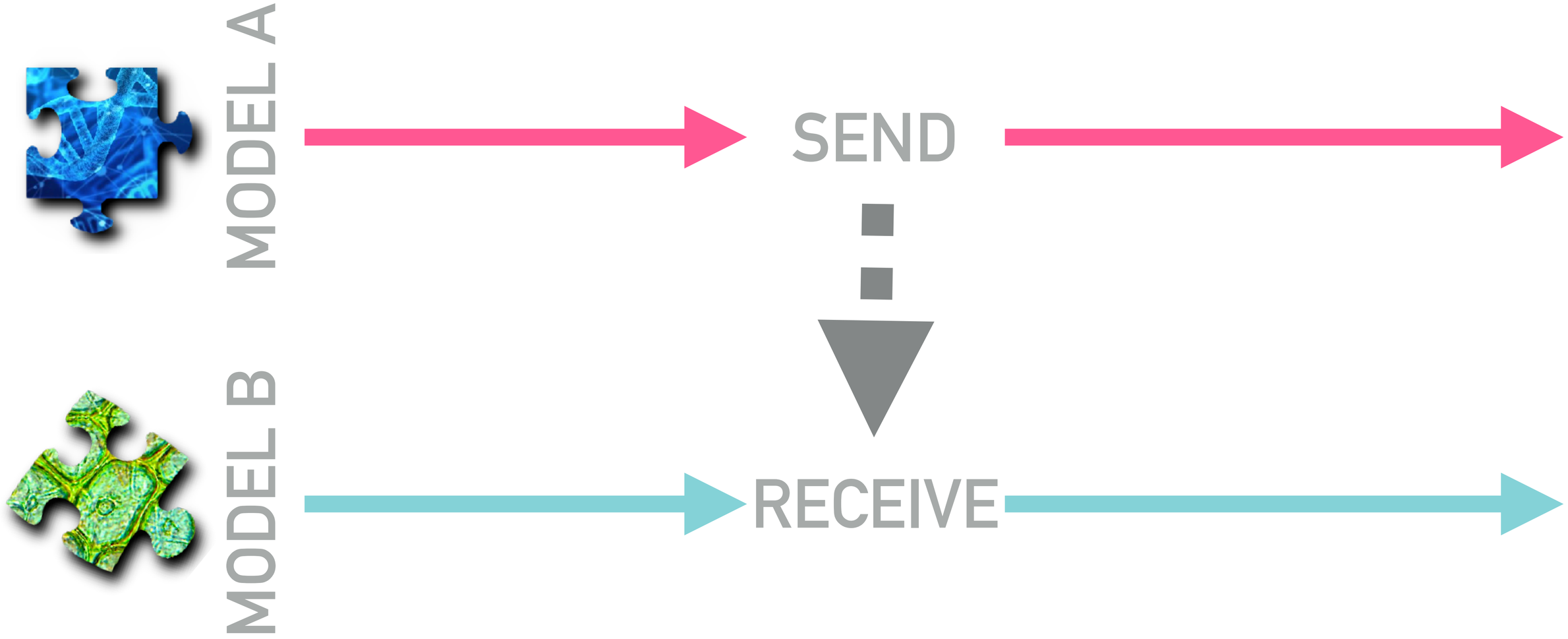
ASYNCHRONOUS COMMUNICATION

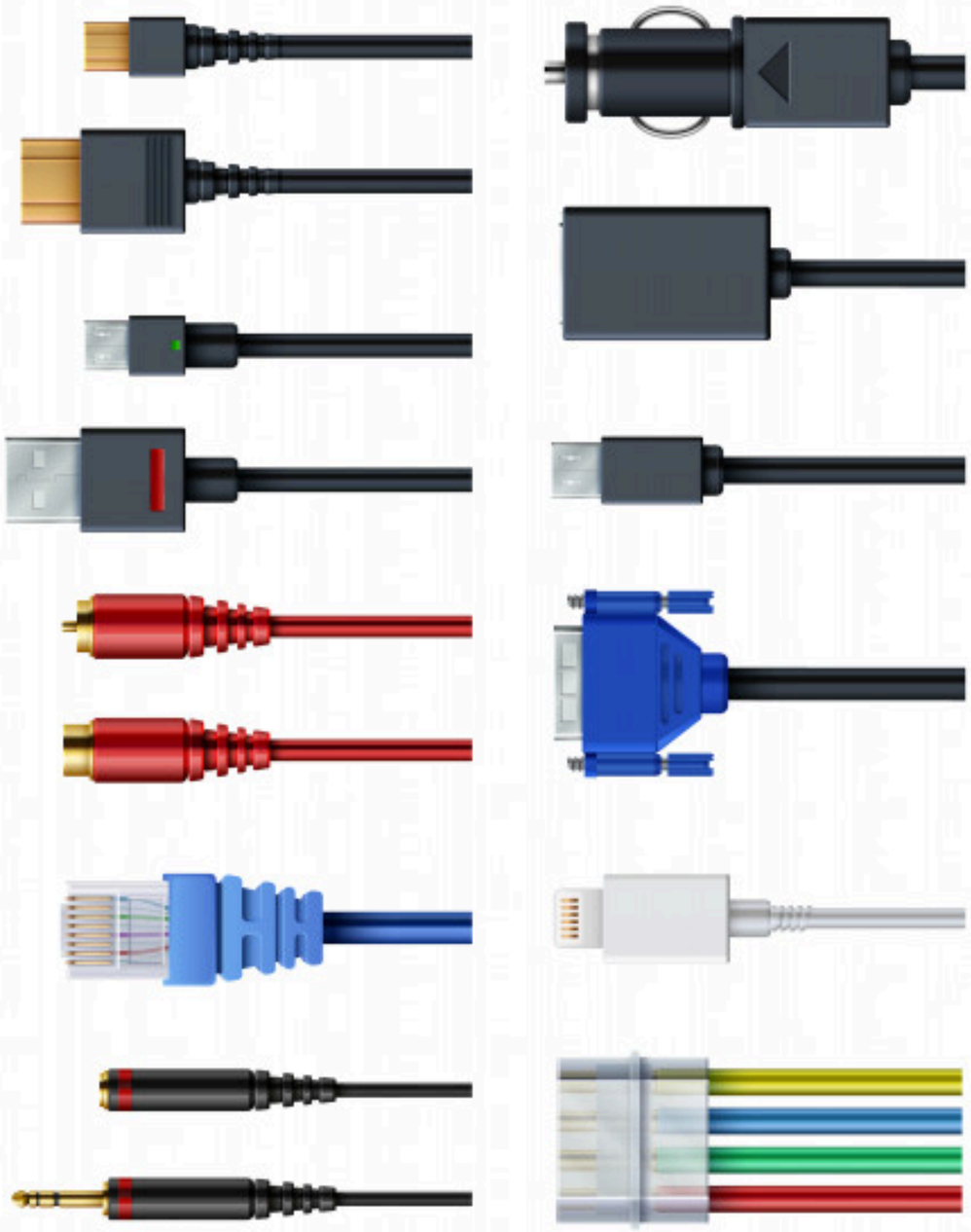
(RECEIVE FIRST)



ASYNCHRONOUS COMMUNICATION

(RECEIVE FIRST)





COMM:

COMMUNICATION OBJECT ALLOWING MODELS TO SEND/RECEIVE MESSAGES TO/FROM OTHER MODELS

COMM CLASSES

INPUT

Receive messages from another model

OUTPUT

Send messages to another model

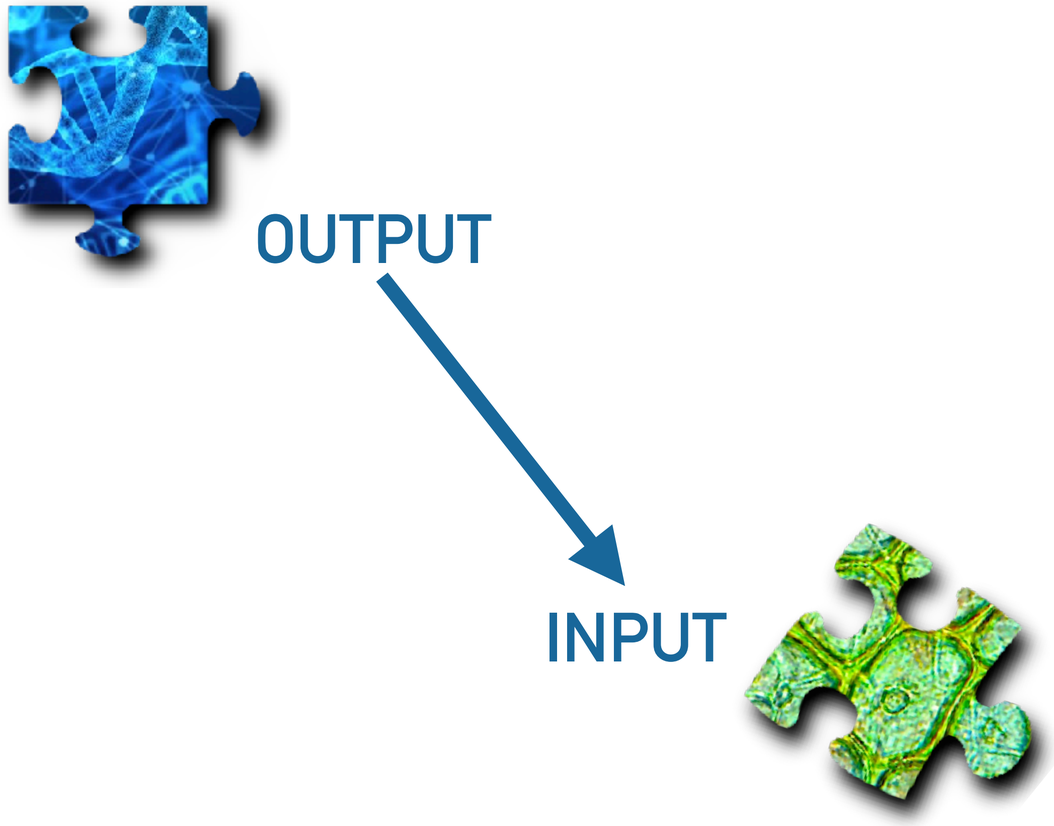
COMM CLASSES

INPUT

Receive messages from another model

OUTPUT

Send messages to another model



COMM CLASSES

INPUT

Receive messages from another model

OUTPUT

Send messages to another model

SERVER

Receive requests from client models and send responses

CLIENT

Send requests to a server model and receive messages (“call” a server model)

COMM CLASSES

CLIENT

INPUT

Receive messages from another model

OUTPUT

Send messages to another model

SERVER

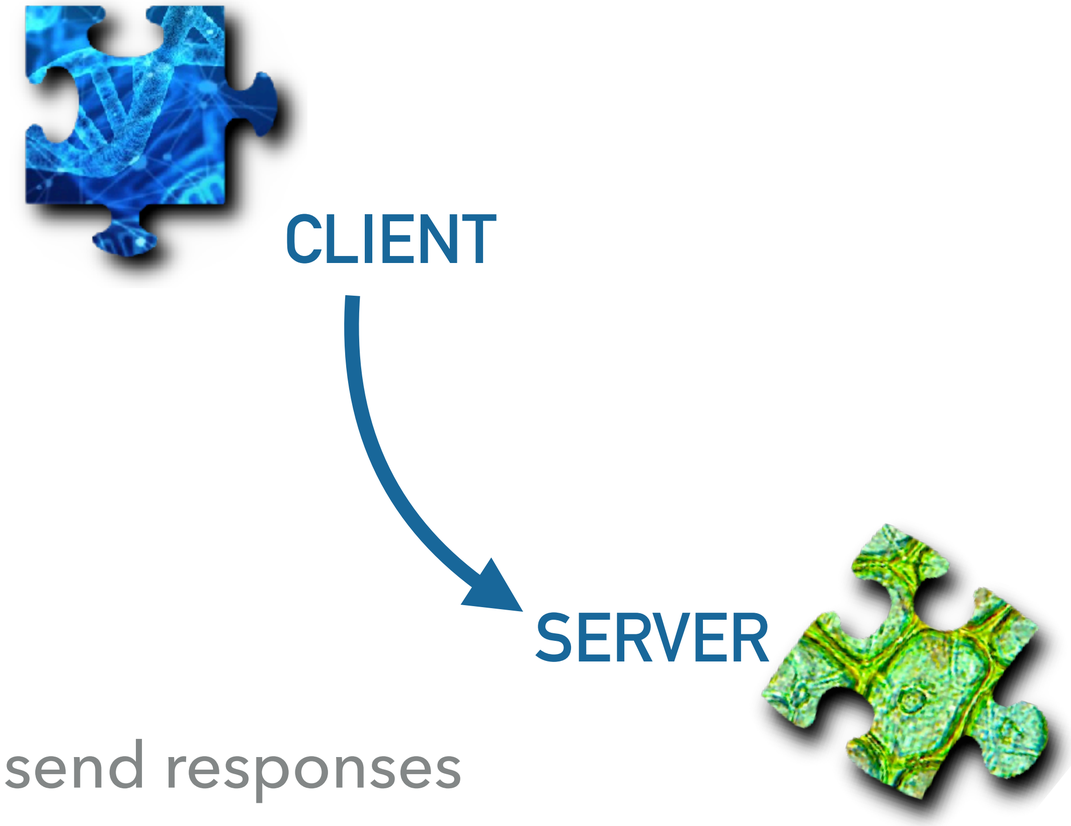
SERVER

Receive requests from client models and send responses

CLIENT

Send requests to a server model and receive messages (“call” a server model)

COMM CLASSES



INPUT

Receive messages from another model

OUTPUT

Send messages to another model

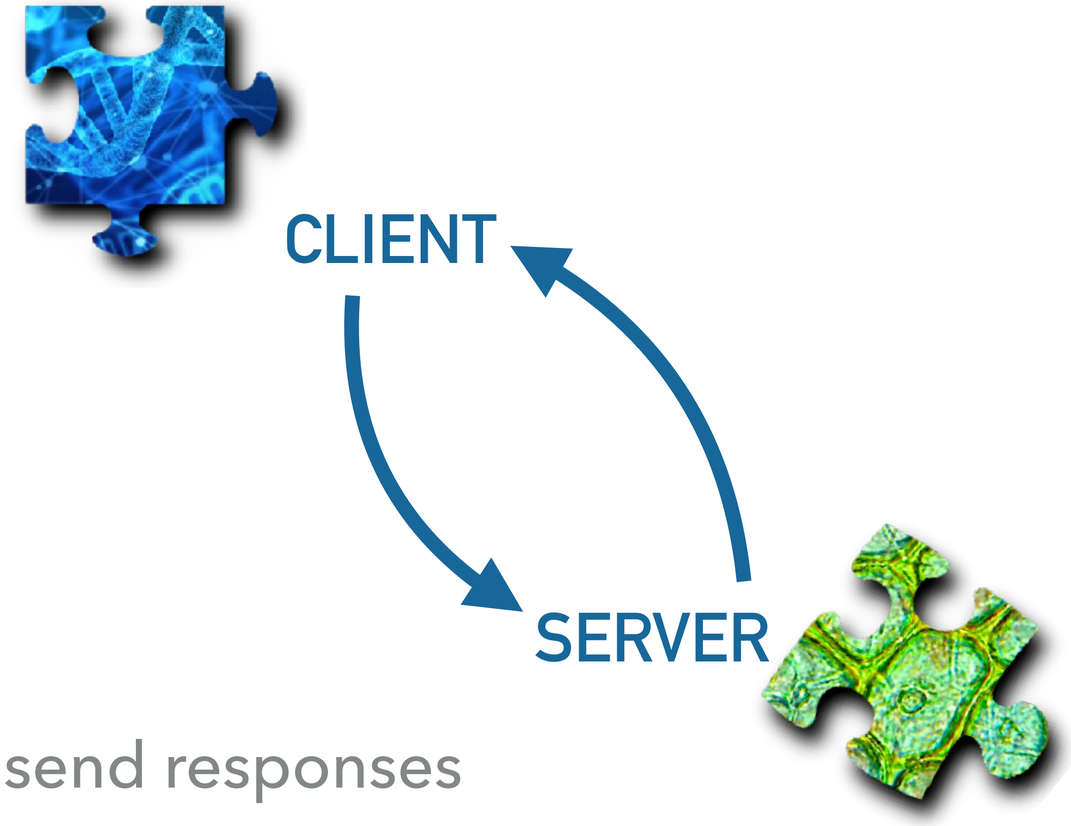
SERVER

Receive requests from client models and send responses

CLIENT

Send requests to a server model and receive messages (“call” a server model)

COMM CLASSES



INPUT

Receive messages from another model

OUTPUT

Send messages to another model

SERVER

Receive requests from client models and send responses

CLIENT

Send requests to a server model and receive messages (“call” a server model)

COMM CLASSES

INPUT

Receive messages from another model

OUTPUT

Send messages to another model

SERVER

Receive requests from client models and send responses

CLIENT

Send requests to a server model and receive messages (“call” a server model)

TIMESYNC

Send requests to a set of time-dependent models and receive time-dependent variables (“call” a time step synchronization)

COMM CLASSES

TIMESYNC

INPUT

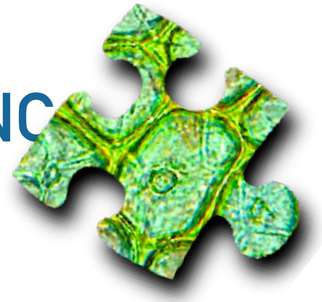
Receive messages from another model

SYNC

OUTPUT

Send messages to another model

TIMESYNC



SERVER

Receive requests from client models and send responses

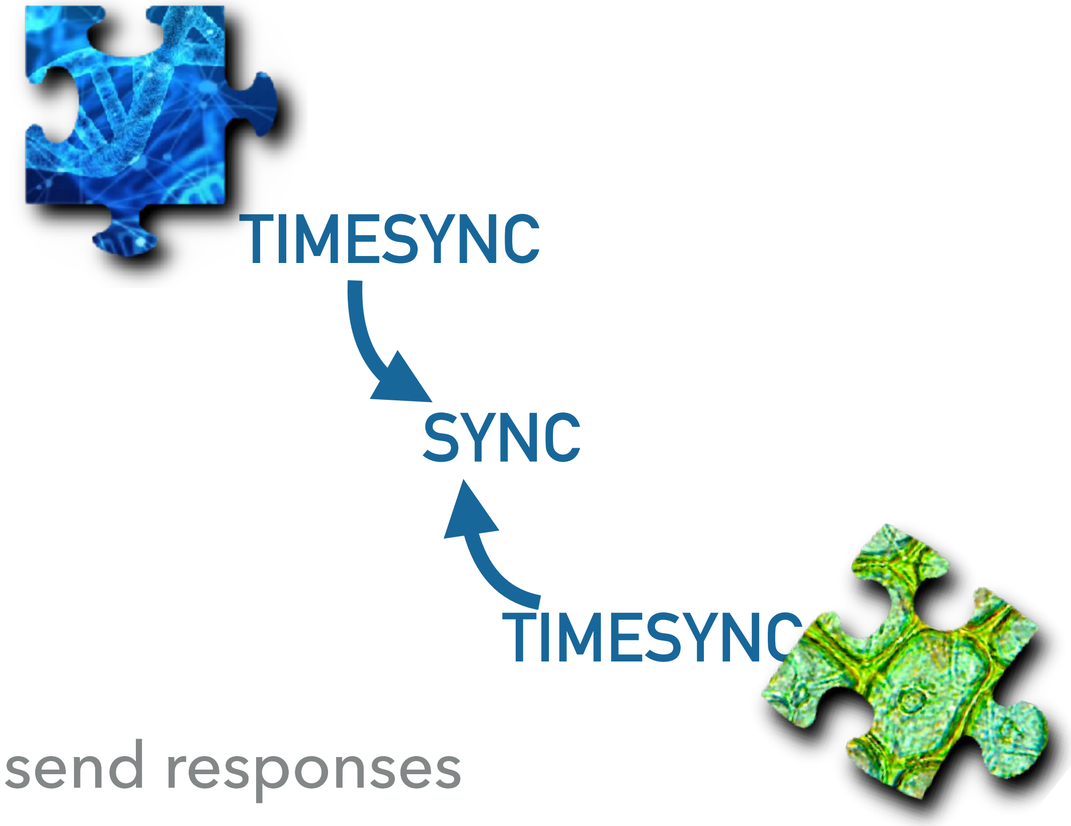
CLIENT

Send requests to a server model and receive messages (“call” a server model)

TIMESYNC

Send requests to a set of time-dependent models and receive time-dependent variables (“call” a time step synchronization)

COMM CLASSES



INPUT

Receive messages from another model

OUTPUT

Send messages to another model

SERVER

Receive requests from client models and send responses

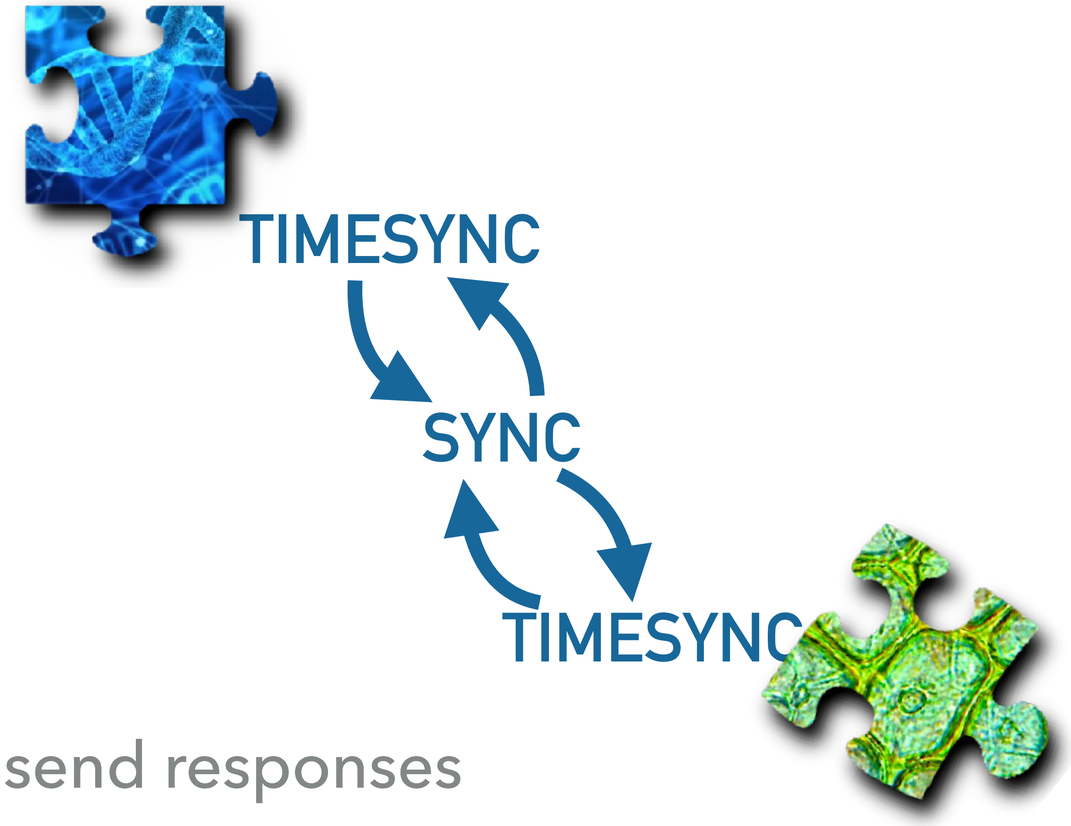
CLIENT

Send requests to a server model and receive messages ("call" a server model)

TIMESYNC

Send requests to a set of time-dependent models and receive time-dependent variables ("call" a time step synchronization)

COMM CLASSES



INPUT

Receive messages from another model

OUTPUT

Send messages to another model

SERVER

Receive requests from client models and send responses

CLIENT

Send requests to a server model and receive messages (“call” a server model)

TIMESYNC

Send requests to a set of time-dependent models and receive time-dependent variables (“call” a time step synchronization)

COMM METHODS

IPC

Interprocess communication, only available on Unix (Linux & Mac)

ZEROMQ

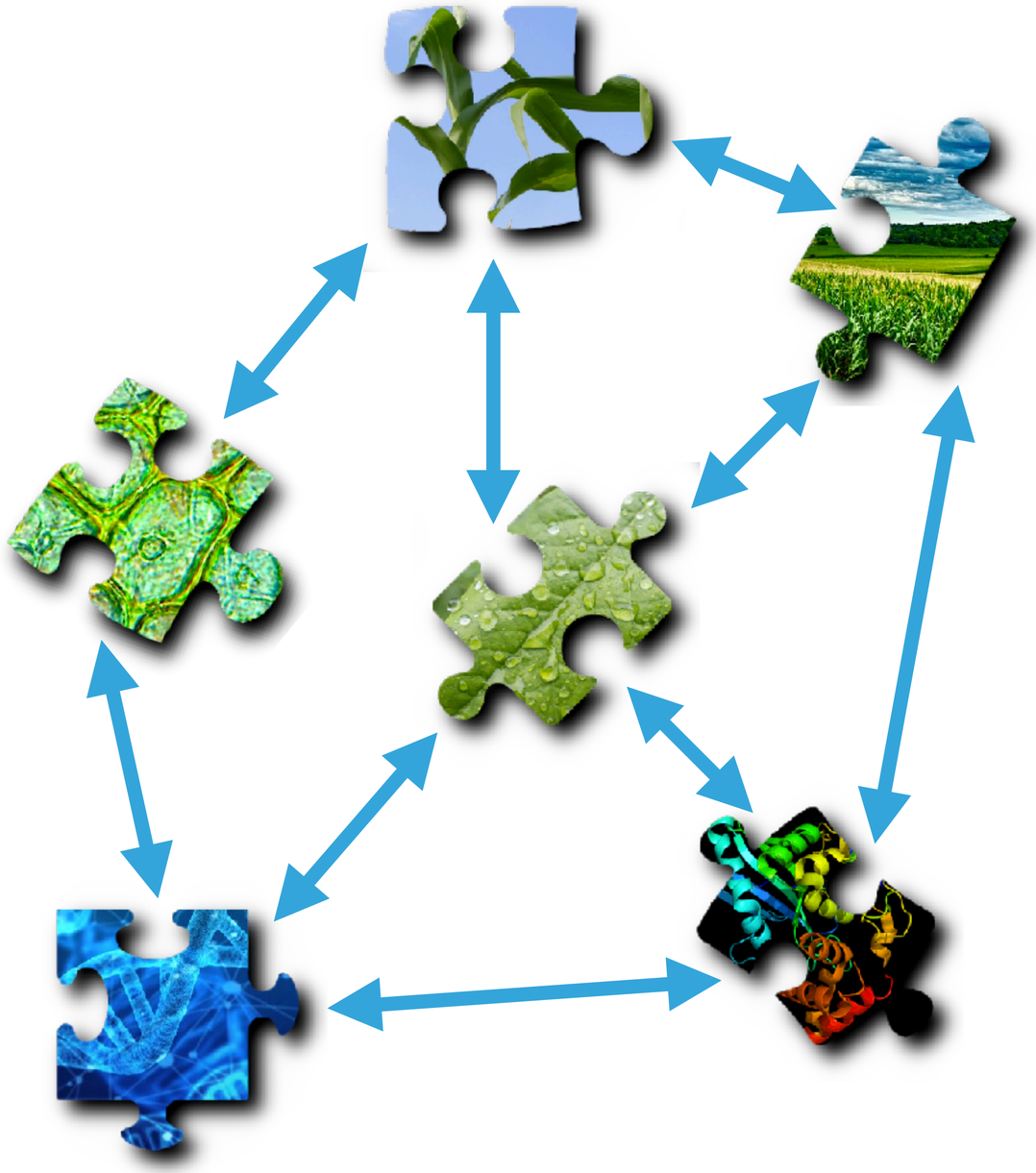
Broker-less communication sockets via TCP, IPC, UDP, inproc, etc.; available on all OSs

RABBITMQ

Brokered communication; requires a RabbitMQ server, but allows for more reliable communication with remote machines

FILES

Slow due to interaction with the disk, but useful for input/output of parameters in a generic way that allows the same model to be used w/ files or other models

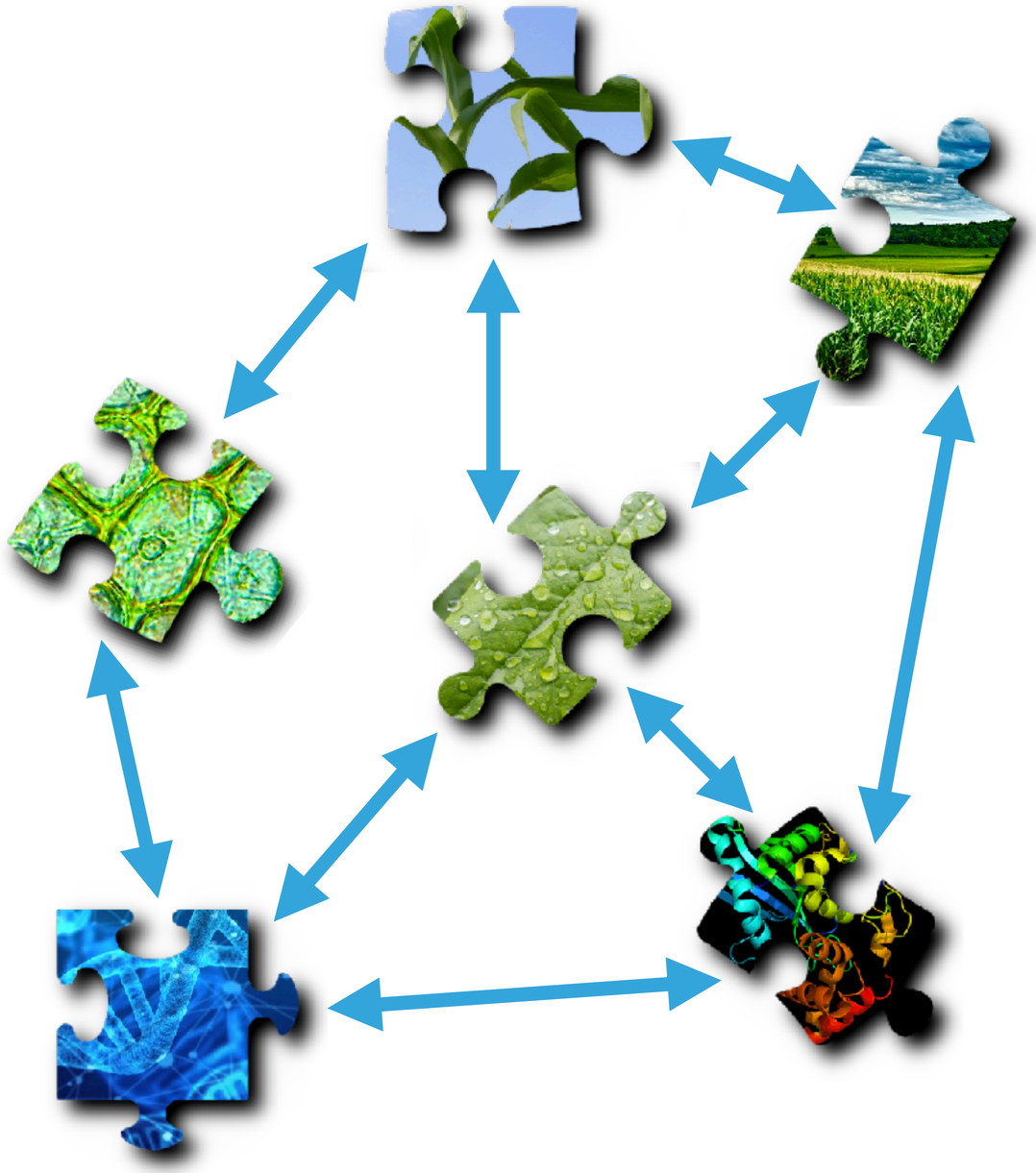


UNITS (AUTOMATED CONVERSION)

UNYT (PYTHON)

MATLAB SYMBOLIC UNITS

R UNITS



UNITS (AUTOMATED CONVERSION)

UNYT (PYTHON)

MATLAB SYMBOLIC UNITS

R UNITS

DATA FORMATS

SCALARS/ARRAYS

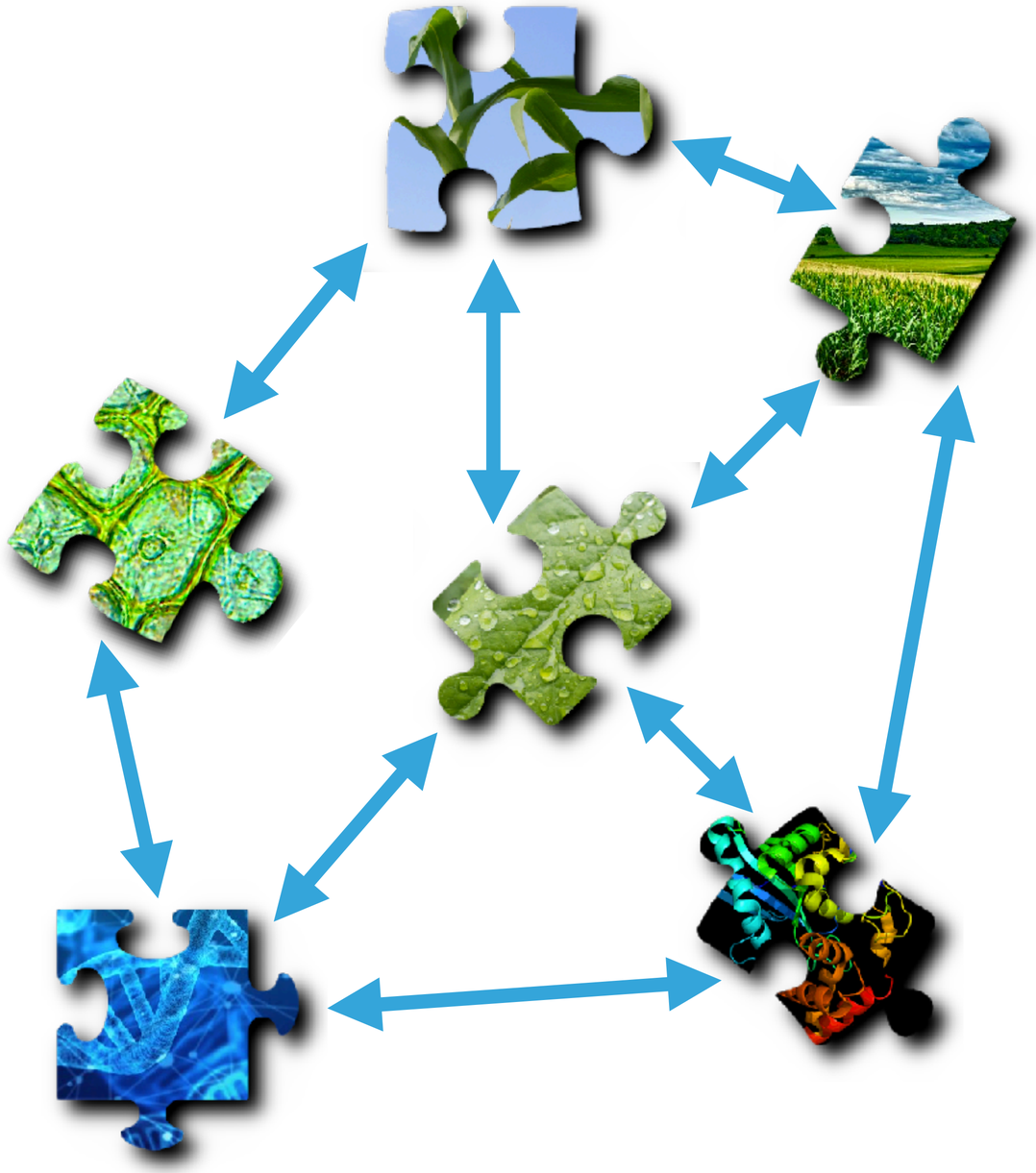
DELIMITED TABLES

PANDAS/R DATA FRAMES

PLY/OBJ

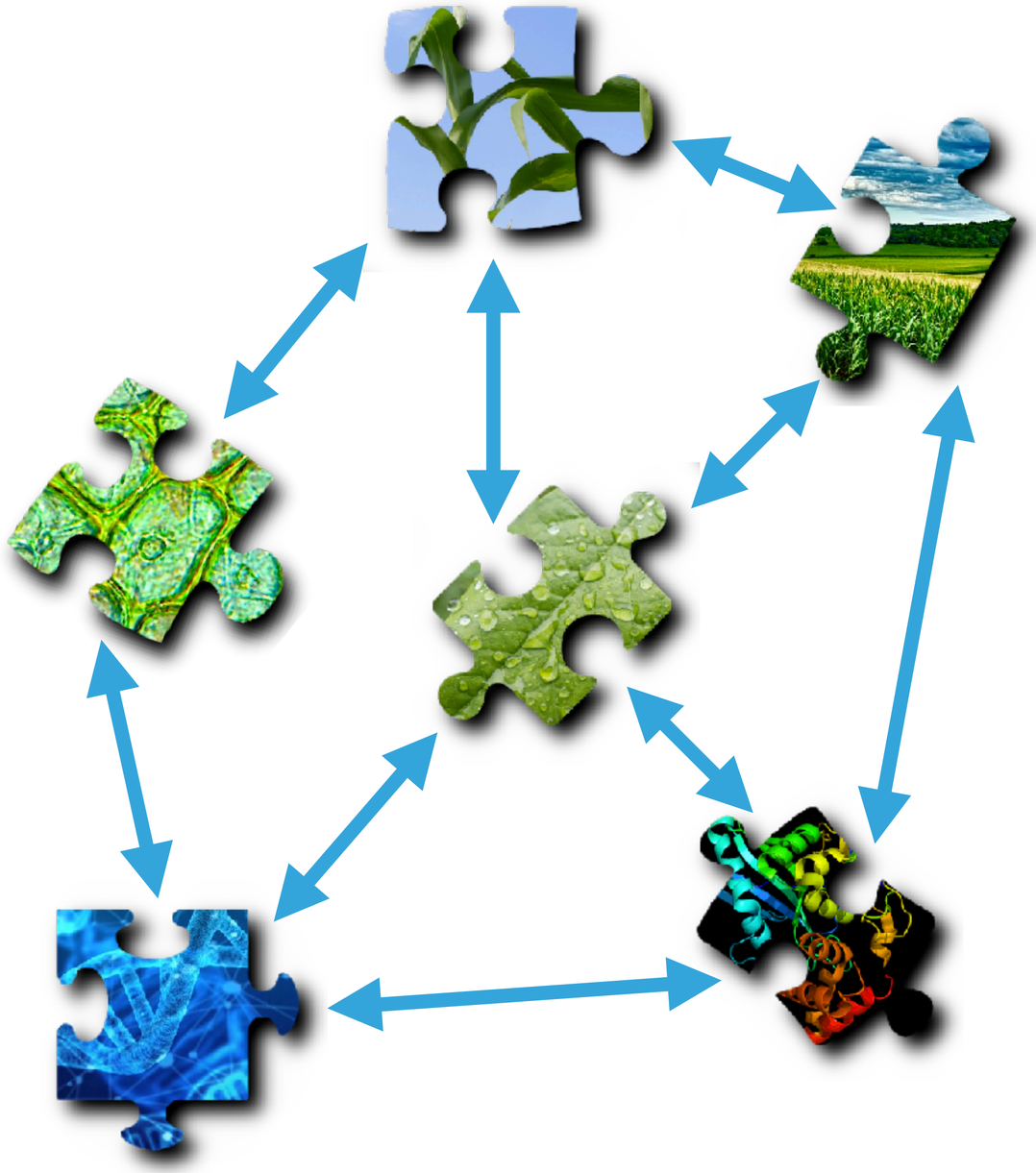
PYTHON PICKLES/
R .RDATA/MATLAB .MAT

EXECUTION



INTEGRATION:

NETWORK OF MODELS RUN USING YGGDRASIL



YAML SPECIFICATION:

INPUT FILE(S) TO YGGDRASIL CONTAINING INFO ON MODELS & CONNECTIONS

YAML SPECIFICATION:

INPUT FILE(S) TO YGGDRASIL CONTAINING INFO ON MODELS & CONNECTIONS

```yaml
model:
  name: GrowthModel
  language: python
  args: ./src/growth.py
  inputs:
    - light
  outputs:
    - growth_rate
```

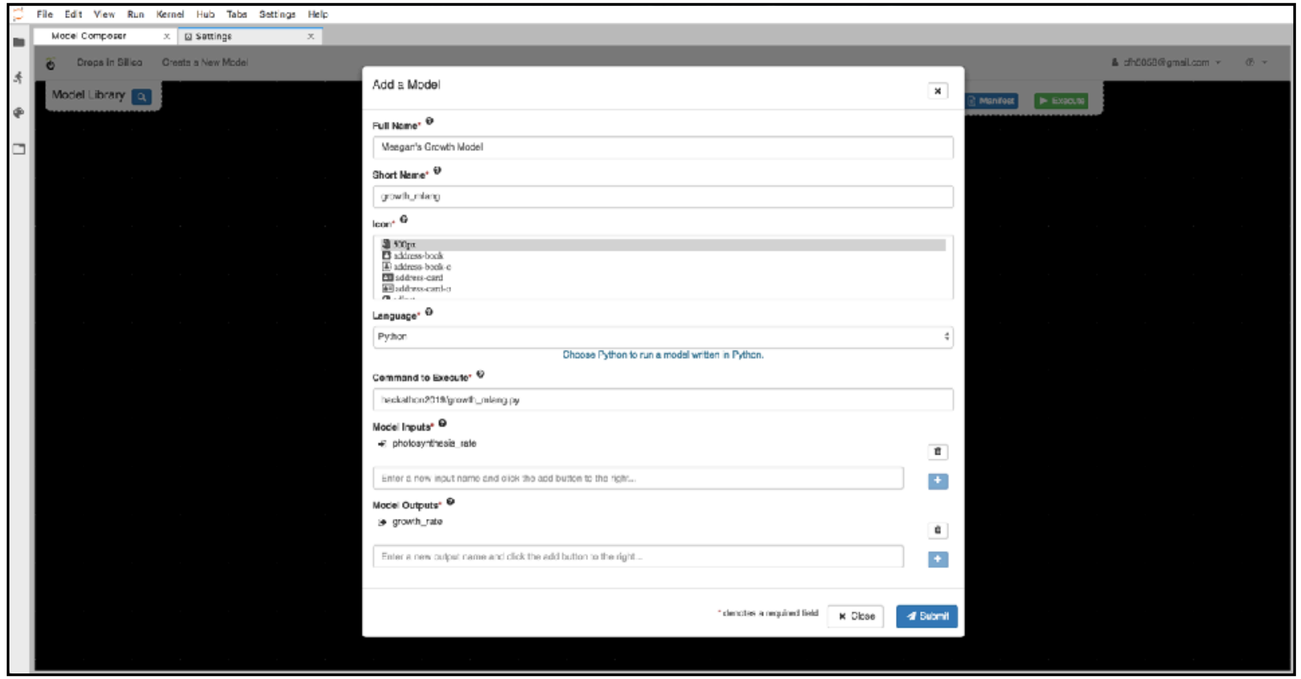
YAML SPECIFICATION:

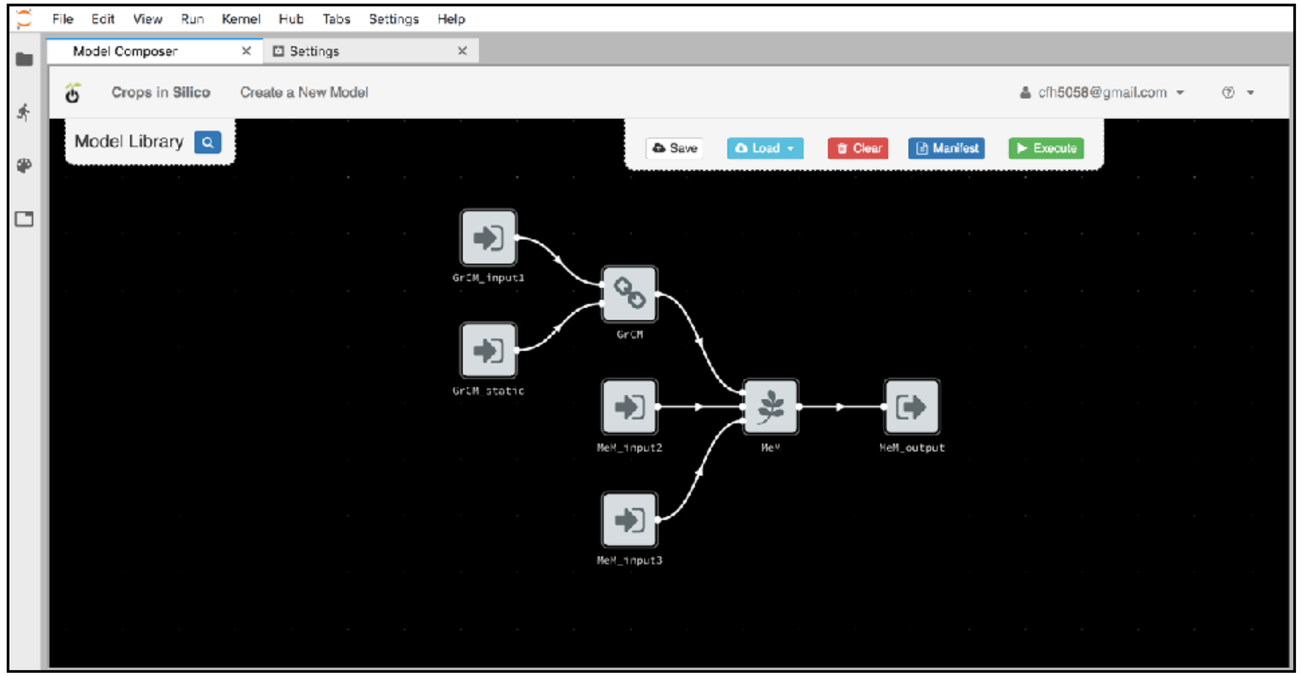
INPUT FILE(S) TO YGGDRASIL CONTAINING INFO ON MODELS & CONNECTIONS

```yaml
model:
  name: GrowthModel
  language: python
  args: ./src/growth.py
  inputs:
    - light
  outputs:
    - growth_rate

connections:
  - input: LightModel:light
    output: GrowthModel:light
  - input: growth_rate
    output: ./Output/growth.txt
    filetype: table
    field_names: growth_rate
```

YAML SPECIFICATION:

INPUT FILE(S) TO YGGDRASIL CONTAINING INFO ON MODELS & CONNECTIONS

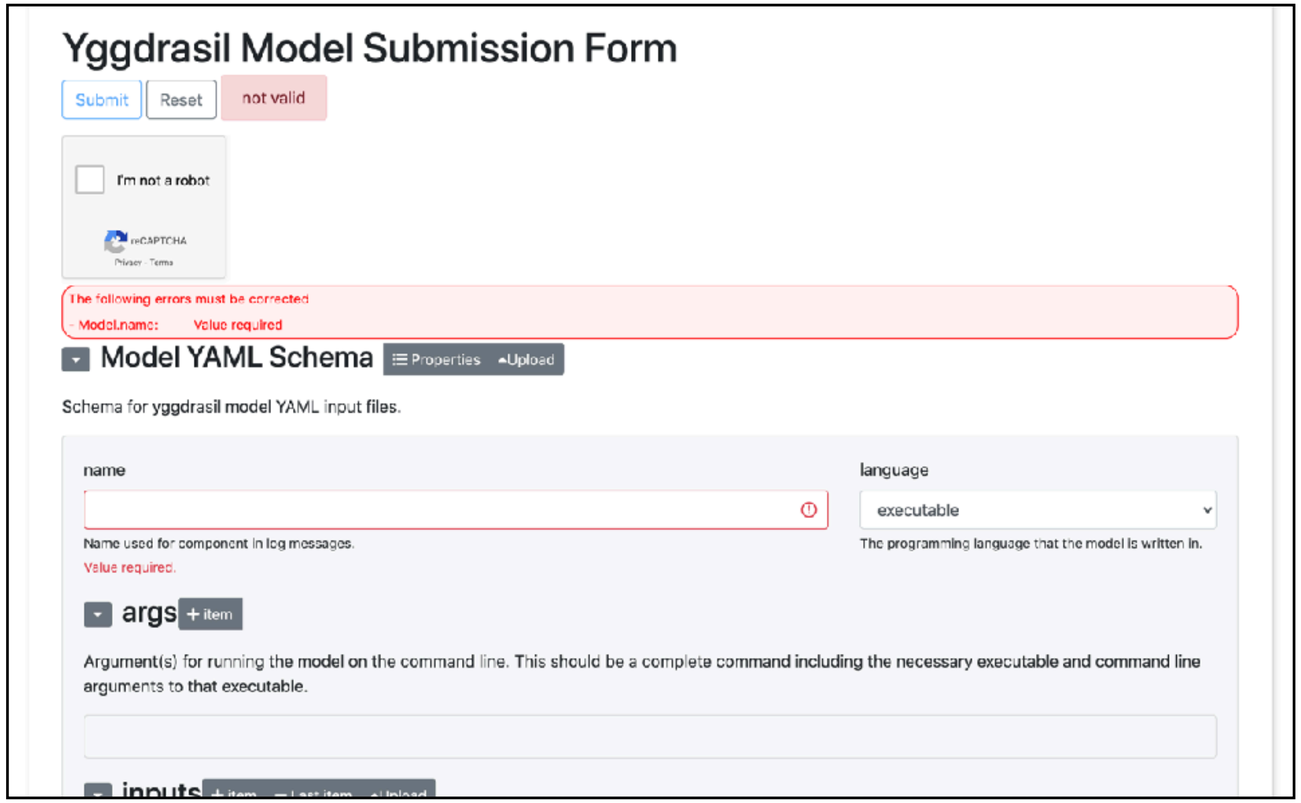




GUI:

UPDATE TO GRAPHICAL USER INTERFACE (GUI) IN DEVELOPMENT FOR YGGDRASIL CAN GENERATE THE YAML

Work by Doug Friedel



MODEL REPOSITORY:

FORM FOR ENTERING MODEL INFORMATION TO VALIDATE, GENERATE & SHARE MODEL YAMLS WITH OTHERS

https://yggdrasil-models.herokuapp.com/

NEW FEATURES

OPENMP THREADING

Support for threaded interface calls

MODEL COPIES

Improved duplication of models via a YAML parameter including outside server/client communication patterns

VALUE COMM

Comm that provides a constant value defined via the YAML that can be used for parameters

STREAM-LINED CONNECTIONS

Remove the use of intermediate comms to improve performance

IN PROGRESS FEATURES

MPI PARALLELISM

Support for running integrations on distributed compute resources via MPI

JULIA

Support for models written in Julia

DIRECT CONNECTIONS

Further improved performance by eliminating connection proxies when possible

MIGRATION TO C++

Move most algorithms into C++ for improved performance, particularly in RapidJSON serialization/deserialization

DEBUGGING

DEBUGGING

1. TRY ANOTHER BROWSER (IF USING MYBINDER)
2. IF IN A NOTEBOOK CELL, TRY RE-RUNNING PREVIOUS CELLS
3. CHECK FOR SIMILAR ISSUES ON GITHUB

Materials: https://github.com/cropsinsilico/CiS2021-hackathon

GITHUB ISSUES

Search or jump to... / Pull requests Issues Marketplace Explore

cropsinsilico / CiS2021-hackathon

Unwatch 4 Star 1 Fork 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 2 branches 0 tags Go to file Add file Code

langmm Fixed temp cmake that was added ··· a38cd3f 16 hours ago 56 commits

.github	Update issue template and fix typo in README	7 days ago
images	Fixed temp cmake that was added	16 hours ago
input	Add example of wrapping a model function	16 days ago
meshes	Fix bugs in the untested copies example	13 days ago
models	Fixed temp cmake that was added	16 hours ago
yamls	Fixed temp cmake that was added	16 hours ago
.gitignore	Update root model and add version of shoot for timesync example	13 days ago
00-intro.ipynb	Fixed temp cmake that was added	16 hours ago

About

Materials for the Hackathon at the 2021 Crops in Silico Symposium & Workshop

Readme

BSD-3-Clause License

Releases

No releases published

Create a new release

Packages

No packages published

GITHUB ISSUES

Search or jump to... / Pull requests Issues Marketplace Explore

cropsinsilico / CiS2021-hackathon

Unwatch 4 Star 1 Fork 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

main 2 branches 0 tags Go to file Add file Code

langmm Fixed temp cmake that was added ··· a38cd3f 16 hours ago 56 commits

.github	Update issue template and fix typo in README	7 days ago
images	Fixed temp cmake that was added	16 hours ago
input	Add example of wrapping a model function	16 days ago
meshes	Fix bugs in the untested copies example	13 days ago
models	Fixed temp cmake that was added	16 hours ago
yamls	Fixed temp cmake that was added	16 hours ago
.gitignore	Update root model and add version of shoot for timesync example	13 days ago
00-intro.ipynb	Fixed temp cmake that was added	16 hours ago

About

Materials for the Hackathon at the 2021 Crops in Silico Symposium & Workshop

Readme

BSD-3-Clause License

Releases

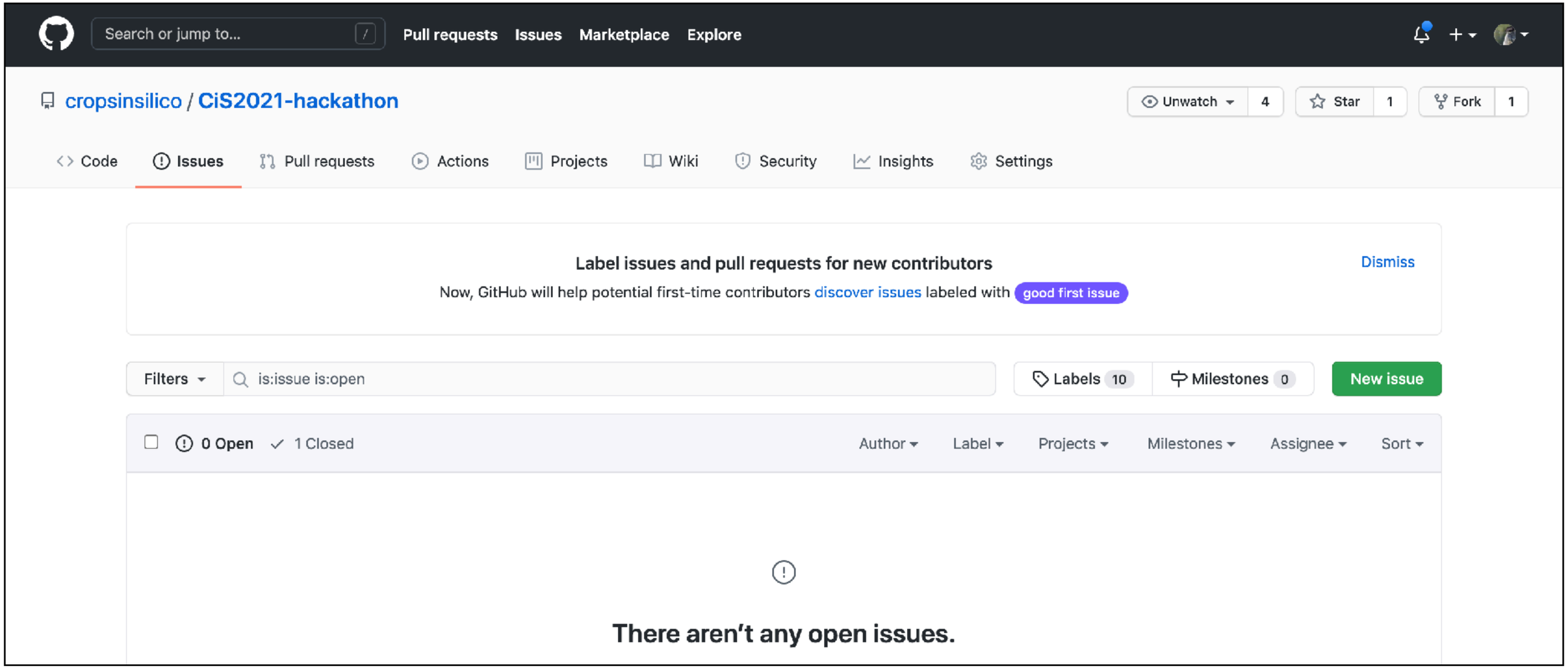
No releases published

Create a new release

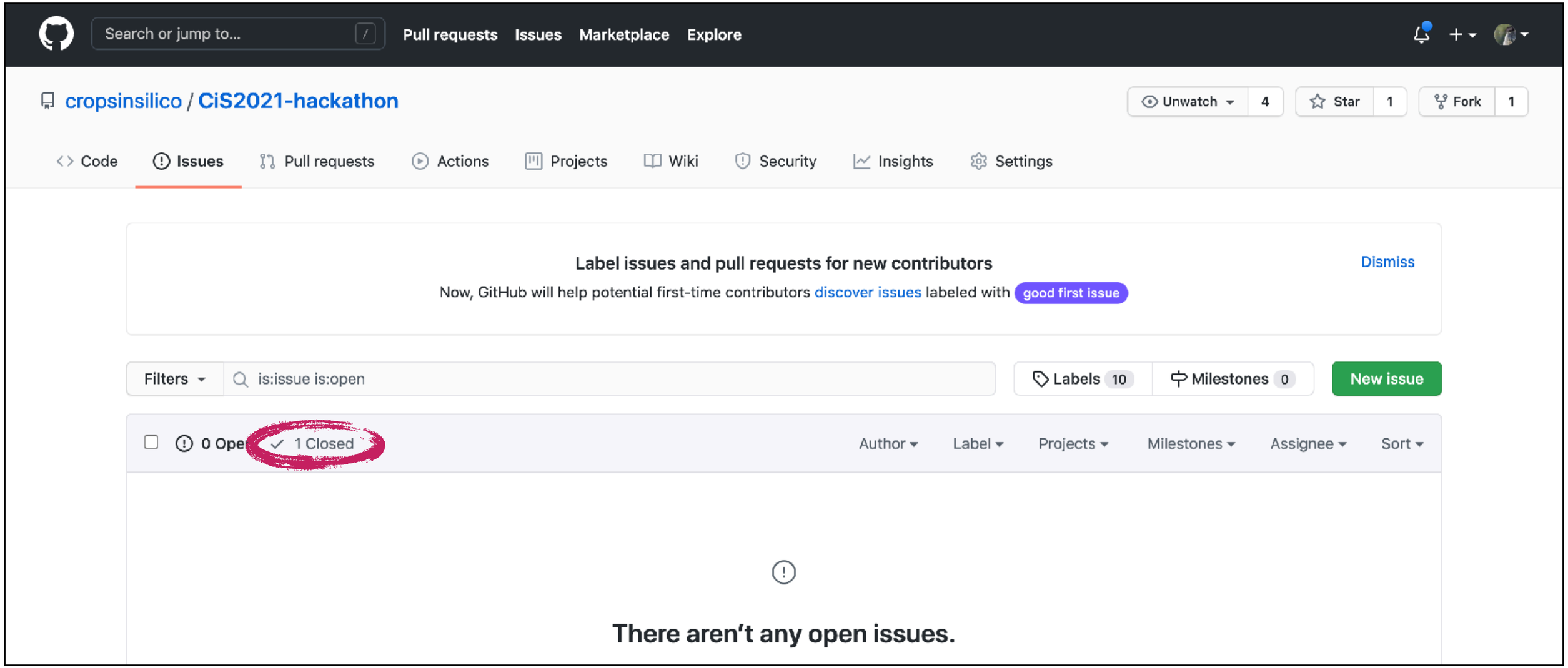
Packages

No packages published

GITHUB ISSUES



GITHUB ISSUES



GITHUB ISSUES

Search or jump to... / Pull requests Issues Marketplace Explore

cropsinsilico / CiS2021-hackathon

Unwatch 4 Star 1 Fork 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Label issues and pull requests for new contributors

Now, GitHub will help potential first-time contributors discover issues labeled with good first issue

Dismiss

Filters is:issue is:closed Labels 10 Milestones 0 New issue

Clear current search query, filters, and sorts

0 Open 1 Closed Author Label Projects Milestones Assignee Sort

"Binder not found" mybinder 1

#2 by langmm was closed 21 hours ago 2 of 7

ProTip! Find everything you created by searching author:langmm.

DEBUGGING

1. TRY ANOTHER BROWSER (IF USING MYBINDER)
2. IF IN A NOTEBOOK CELL, TRY RE-RUNNING PREVIOUS CELLS
3. CHECK FOR SIMILAR ISSUES ON GITHUB
4. IF USING A LOCAL INSTALL, CHECK THE DEBUGGING DOCS FOR YGGDRASIL

Materials: https://github.com/cropsinsilico/CiS2021-hackathon

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessary. We will be using Github Issues to track problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder icon (or the link below).

  It may take a few moments to initialize. If you encounter an error, open an issue and try with another browser. If you still cannot launch the binder, you can install the materials on your machine by following the instructions at one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

OTHER RESOURCES

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessary. We will be using Github Issues to track problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder icon (or the link below).

  It may take a few moments to initialize. If you encounter an error, open an issue and try with another browser. If you still cannot launch the binder, you can install the materials on your machine by following the instructions at one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

OTHER RESOURCES

OTHER RESOURCES

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessa
  problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder

  It may take a few moments to initialize. If you encounter an error, ope
  you still cannot launch the binder, you can install the materials on you
  one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

yggdrasil
1.6.1

Search docs

CONTENTS:

Overview
Installation
Getting started
Formatted I/O
Server/Client I/O
Autowrapping Model Functions
Notes on Autowrapping C/C++ Model Functions
Conditional I/O
Transformed I/O
Timestep Synchronization
YAML Files
Configuration Files
Units
C-Style Format Strings
Debugging
Tips for Debugging
Possible Errors
OpenMP Threading in Models
Examples

» Debugging View page source

Debugging

Tips for Debugging

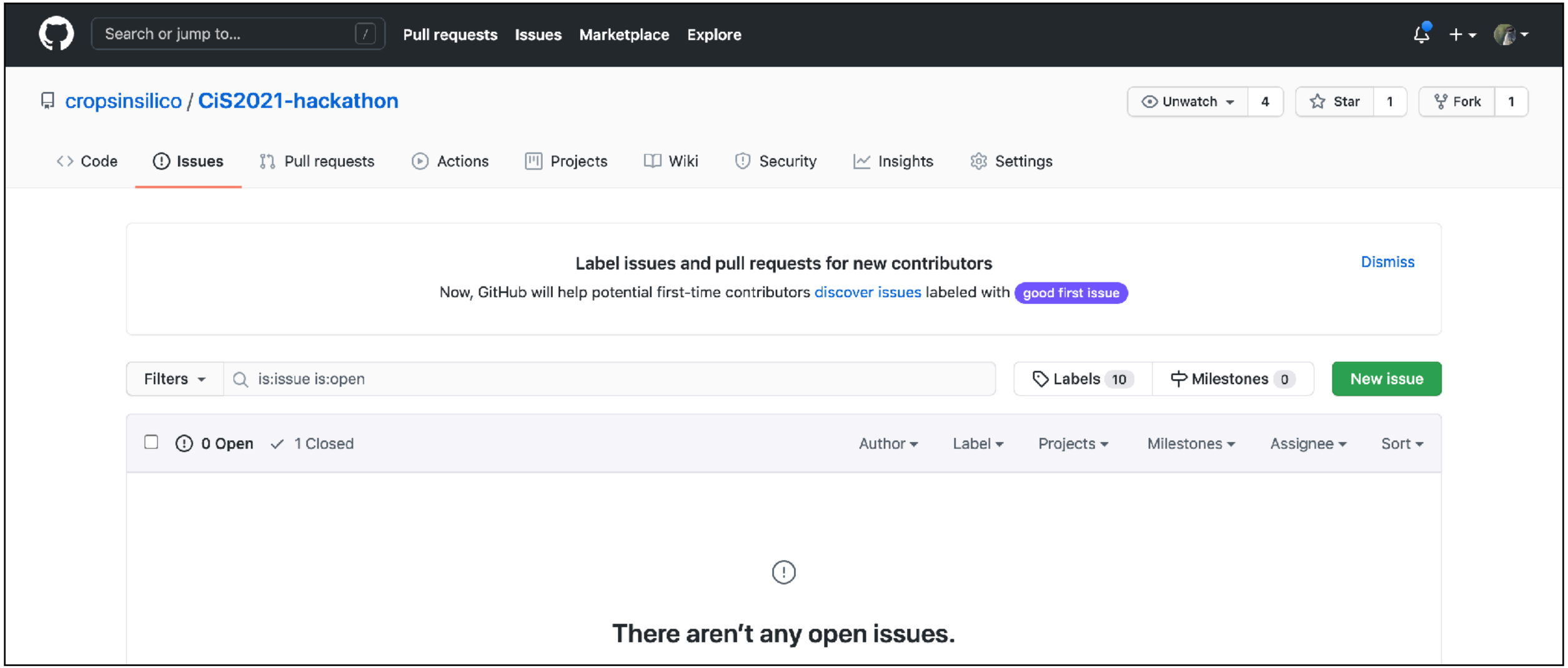
1. *Look at the full output.* The final error raised by yggdrasil may not contain all of the informations provided by errors that were raised within a model due to limitations of error forwarding between the different languages. It is important to look at the full output from a failed run. Usually the first error encountered or the error raised within the model's language will be the most relevant and be the most useful for debugging.
2. *Check for known errors.* The list below includes several errors that have already been encountered by yggdrasil users and the method used to solve the issue.
3. *Turn on debugging log messages.* This will increase the number of log messages greatly and help you track down any issues. Debug messages can be enabled by setting the `ygg` and `client` debug options in your config file to `DEBUG` (see Configuration Options for details on the location of the user config file and additional logging options).
4. *Trace the flow of data.* Use the debug messages to trace the flow of data from one model to the next and determine where the failure is occuring.
5. *Check |yggdrasil| summary.* yggdrasil includes a command line utility, `yggdrasil` that will print out relevent information about yggdrasil, the languages it supports, and the operating system. This information can be useful for running down conflicting dependencies or determining why yggdrasil thinks a language is or isn't install. Additional information about the system can be display by adding the `--verbose` flag, including the current conda environment information (if you are using a conda environment) and informaiton about the current R installation (if R is installed). This information should be included in any Github issues opened related to bugs in order to help us assist you.

DEBUGGING

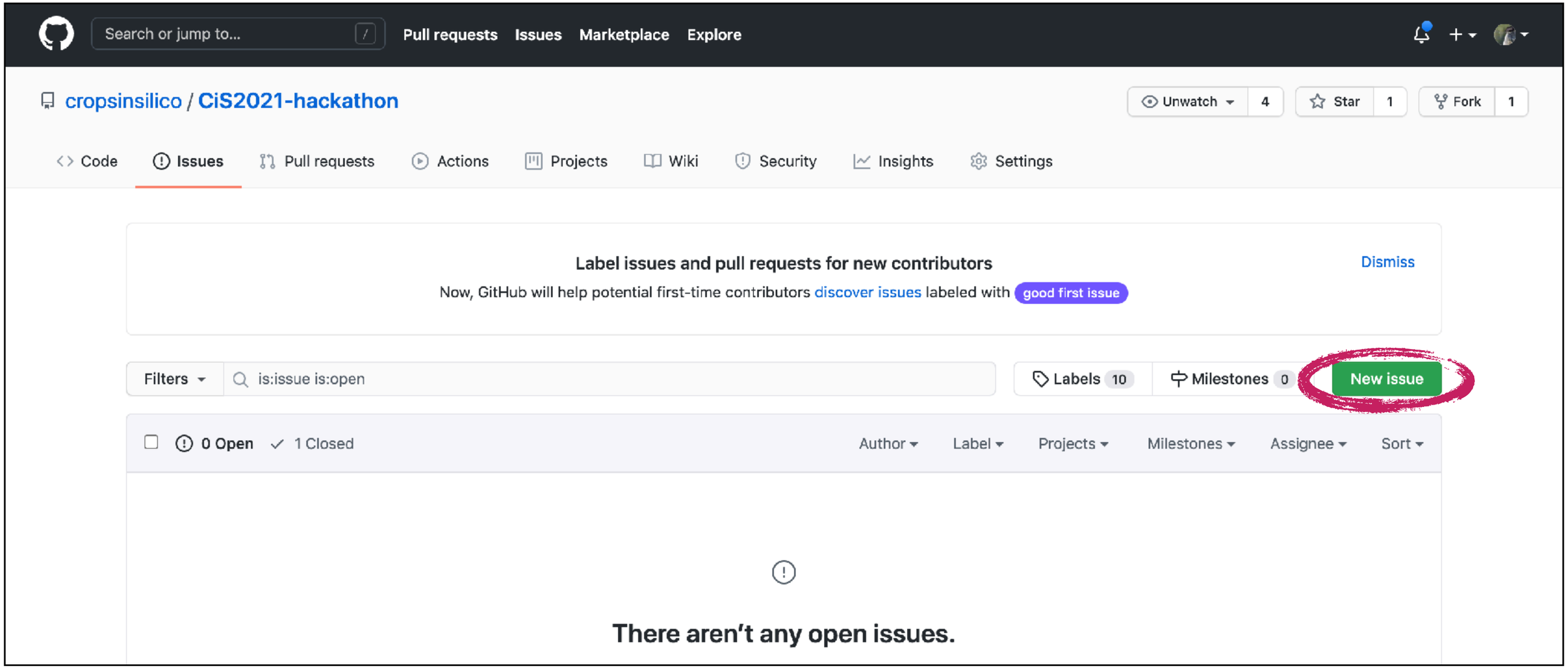
1. TRY ANOTHER BROWSER (IF USING MYBINDER)
2. IF IN A NOTEBOOK CELL, TRY RE-RUNNING PREVIOUS CELLS
3. CHECK FOR SIMILAR ISSUES ON GITHUB
4. IF USING A LOCAL INSTALL, CHECK THE DEBUGGING DOCS FOR YGGDRASIL
5. OPEN A NEW GITHUB ISSUE

Materials: https://github.com/cropsinsilico/CiS2021-hackathon

GITHUB ISSUES



GITHUB ISSUES



GITHUB ISSUES

Search or jump to... / Pull requests Issues Marketplace Explore

cropsinsilico / CiS2021-hackathon

Unwatch 4 Star 1 Fork 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Title

Write Preview

<!--- Provide some information about how you are accessing the hackathon materials. -->
<!--- To mark a check box, replace the space inside the brackets with an X, e.g. [X] -->
Context (Environment)
* [ ] MyBinder instance
* [ ] Local install
* [ ] Docker container

<!--- What operating system are you on? (e.g. Windows, Mac, Linux) -->
OS:

<!--- What web browser are you using to access the notebooks? -->
Browser:

<!--- Tell us what type of issue you are having -->

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Linked pull requests
Successfully merging a pull request may close this issue.
None yet

GITHUB ISSUES

Search or jump to... / Pull requests Issues Marketplace Explore

cropsinsilico / CiS2021-hackathon

Unwatch 4 Star 1 Fork 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Title

Write Preview

<!--- Provide some information about how you are accessing the hackathon materials. -->
<!--- To mark a check box, replace the space inside the brackets with an X, e.g. [X] -->
Context (Environment)
* [ ] MyBinder instance
* [ ] Local install
* [ ] Docker container

<!--- What operating system are you on? (e.g. Windows, Mac, Linux) -->
OS:

<!--- What web browser are you using to access the notebooks? -->
Browser:

<!--- Tell us what type of issue you are having -->

Assignees
No one—assign yourself

Labels
None yet

Projects
None yet

Milestone
No milestone

Linked pull requests
Successfully merging a pull request may close this issue.
None yet

GITHUB ISSUES

Search or jump to... /

Pull requests Issues Marketplace Explore

cropsinsilico / CiS2021-hackathon

Unwatch 4 Star 1 Fork 1

Code Issues Pull requests Actions Projects Wiki Security Insights Settings

Title

Write Preview

Context (Environment)

- [ ] MyBinder instance
- [ ] Local install
- [ ] Docker container

OS:

Browser:

Type of Issue

- [ ] Jupyter notebook failed to open

Assignees

No one—assign yourself

Labels

None yet

Projects

None yet

Milestone

No milestone

Linked pull requests

Successfully merging a pull request may close this issue.

None yet

DEMO TIME!

Thanks to Google Cloud, OVH, GESIS Notebooks and the Turing Institute for supporting us! 🎉

binder

Starting repository: cropsinsilico/CiS2021-hackathon/HEAD

You can connect with the Binder community in the Jupyter community forum.

Build logs hide

```
libgfortran-ng-9.3.0 | 22 KB     | ########## | 100%
gitdb-4.0.7          | 46 KB     | ########## | 100%
r-jsonlite-1.7.2     | 462 KB    | ########## | 100%
smmap-3.0.5          | 22 KB     | ########## | 100%
czmq-4.2.1           | 540 KB    | ########## | 100%
networkx-2.5.1       | 1.2 MB    | ########## | 100%
gcc_linux-64-9.3.0   | 23 KB     | ########## | 100%
openjpeg-2.4.0       | 444 KB    | ########## | 100%
r-rappdirs-0.3.3     | 50 KB     | ########## | 100%
```

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

NOTEBOOK INTRO

jupyter 00-intro Last Checkpoint: 8 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Help

Not Trusted

Python 3

Run

Markdown

Introduction

(NOTE: This notebook is intended for use with the slides found here).

This is a Jupyter notebook. It allows us to run code (in this case Python) alongside text in different "cells". This cell is a markdown cell that can display text and html, the next cell is a code cell.

In the code cells (prefixed by `In [ ]:`), you can assign variables, perform calculations or call external functions/classes. You can run code cells by selecting the cell (so that a blue or green box appears around it) and then clicking the run button (located at the top of the page) or pressing `Shift+Enter` together. Then a number will appear inside the brackets indicating the order of when the cell was executed.

Output from the cell will be displayed below it with the `Out[#]:` prefix where the number in the brackets indicates the input cell that generated it.

In [ ]:
```
x = 1
y = 3
z = (x + y)**3
z
```

Any Python code can be used, and we can import external packages as well just like in Python scripts. Cells can also use any variables created in any previously executed cell. The cell below imports some tools that will be used in the rest of this notebook.

jupyter 00-intro Last Checkpoint: 8 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Help

Not Trusted

Python 3

Run

Markdown

TEXT CELL

Introduction

(NOTE: This notebook is intended for use with the slides found here).

This is a Jupyter notebook. It allows us to run code (in this case Python) alongside text in different "cells". This cell is a markdown cell that can display text and html, the next cell is a code cell.

In the code cells (prefixed by `In [ ]:`), you can assign variables, perform calculations or call external functions/classes. You can run code cells by selecting the cell (so that a blue or green box appears around it) and then clicking the run button (located at the top of the page) or pressing `Shift+Enter` together. Then a number will appear inside the brackets indicating the order of when the cell was executed.

Output from the cell will be displayed below it with the `Out[#]:` prefix where the number in the brackets indicates the input cell that generated it.

In [ ]:
```
x = 1
y = 3
z = (x + y)**3
z
```

Any Python code can be used, and we can import external packages as well just like in Python scripts. Cells can also use any variables created in any previously executed cell. The cell below imports some tools that will be used in the rest of this notebook.

jupyter 00-intro Last Checkpoint: 8 minutes ago (autosaved)

Logout

File Edit View Insert Cell Kernel Help Not Trusted Python 3

Run Markdown

Introduction

(NOTE: This notebook is intended for use with the slides found here).

This is a Jupyter notebook. It allows us to run code (in this case Python) alongside text in different "cells". This cell is a markdown cell that can display text and html, the next cell is a code cell.

In the code cells (prefixed by `In [ ]:`), you can assign variables, perform calculations or call external functions/classes. You can run code cells by selecting the cell (so that a blue or green box appears around it) and then clicking the run button (located at the top of the page) or pressing `Shift+Enter` together. Then a number will appear inside the brackets indicating the order of when the cell was executed.

Output from the cell will be displayed below it with the `Out[#]:` prefix where the number in the brackets indicates the input cell that generated it.

CODE CELL

In [ ]:
```
x = 1
y = 3
z = (x + y)**3
z
```

Any Python code can be used, and we can import external packages as well just like in Python scripts. Cells can also use any variables created in any previously executed cell. The cell below imports some tools that will be used in the rest of this notebook.

Code cells can contain any valid Python code
Run cells by holding shift and pressing enter
(shift + enter)

```
In [ ]: x = 1
        y = 3
        z = (x + y)**3
        z
```

Code cells can contain any valid Python code
Run cells by holding shift and pressing enter
(shift + enter)

```
In [1]: x = 1
        y = 3
        z = (x + y)**3
        z

Out[1]: 64
```

Code cells can contain any valid Python code
Run cells by holding shift and pressing enter
(shift + enter)

```
In [1]: x = 1
        y = 3
        z = (x + y)**3
        z

Out[1]: 64
```

Output appears below
Number in bracket is the order of execution
("*" indicates the cell is still running)

We need some tools!

trimesh - package for loading/displaying meshes in the notebook
yggdrasil - the method for running yggdrasil integration

```
In [2]: from yggdrasil import tools  # Displaying syntax
        from yggdrasil.runner import run  # Running integ
        import trimesh  # Load & display 3D meshes
```

We need some tools!

trimesh - package for loading/displaying meshes in the notebook
yggdrasil - the method for running yggdrasil integration

```
In [2]: from yggdrasil import tools  # Displaying syntax
        from yggdrasil.runner import run  # Running integ
        import trimesh  # Load & display 3D meshes
```

No output, so nothing appears below

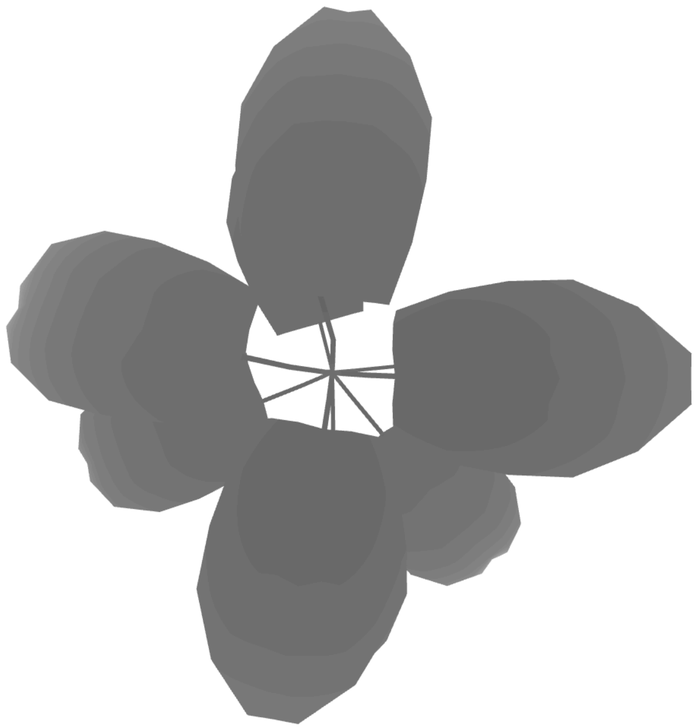
Lets load & display some 3D mesh data

```
In [ ]: fname = 'meshes/plants-2.obj'
        mesh = trimesh.load_mesh(fname)
        mesh.show()
```

Lets load & display some 3D mesh data

In [3]:
```
fname = 'meshes/plants-2.obj'
mesh = trimesh.load_mesh(fname)
mesh.show()
```

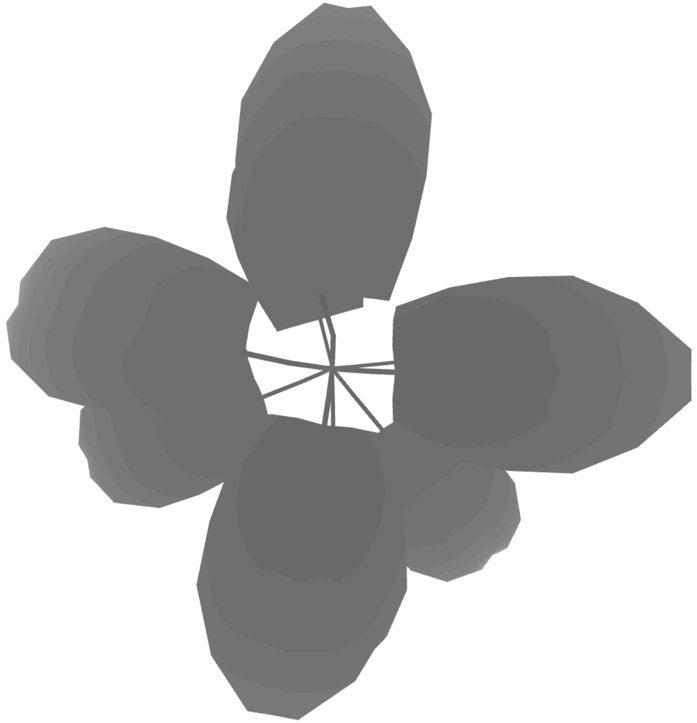
Out[3]:



Lets load & display some 3D mesh data

In [3]:
```
fname = 'meshes/plants-2.obj'
mesh = trimesh.load_mesh(fname)
mesh.show()
```

Out[3]:



Click and drag
to move mesh

BREAK! (10 MIN)

INTEGRATING MODELS AS FUNCTIONS

In [4]:
```
tools.display_source('models/light_v0.py', number_lines=True)
```

In [4]:
```python
tools.display_source('models/light_v0.py', number_lines=True)
```

file: models/light_v0.py
==========

```python
 1: import numpy as np
 2: from yggdrasil import units
 3:
 4:
 5: def light(doy, height):
 6:     r"""Compute the intensity of light.
 7:
 8:     Args:
 9:        doy (float): Day of year.
10:        height (float): Distance from ground in cm.
11:
12:     Returns:
13:        float: Intensity of light in ergs cm^-2 s^-1.
14:
15:     """
16:     # Define parameters that are static across a run
17:     amplitude = units.add_units(80.0, 'ergs cm^-3 s^-1')
18:     doy_offset = units.add_units(0.0, 'days')
19:
20:     # Calculate intensity
21:     intensity = (
22:         amplitude * height *
23:         (1.0 + np.sin(2.0 * np.pi * (doy - doy_offset) /
24:                       units.add_units(365.0, 'days'))))
25:
26:     return intensity
```

In [5]:
```
tools.display_source('yamls/light_v0_python.yml', number_lines=True)
```

```
In [5]: tools.display_source('yamls/light_v0_python.yml', number_lines=True)
```

YAMLs provide info needed to run model

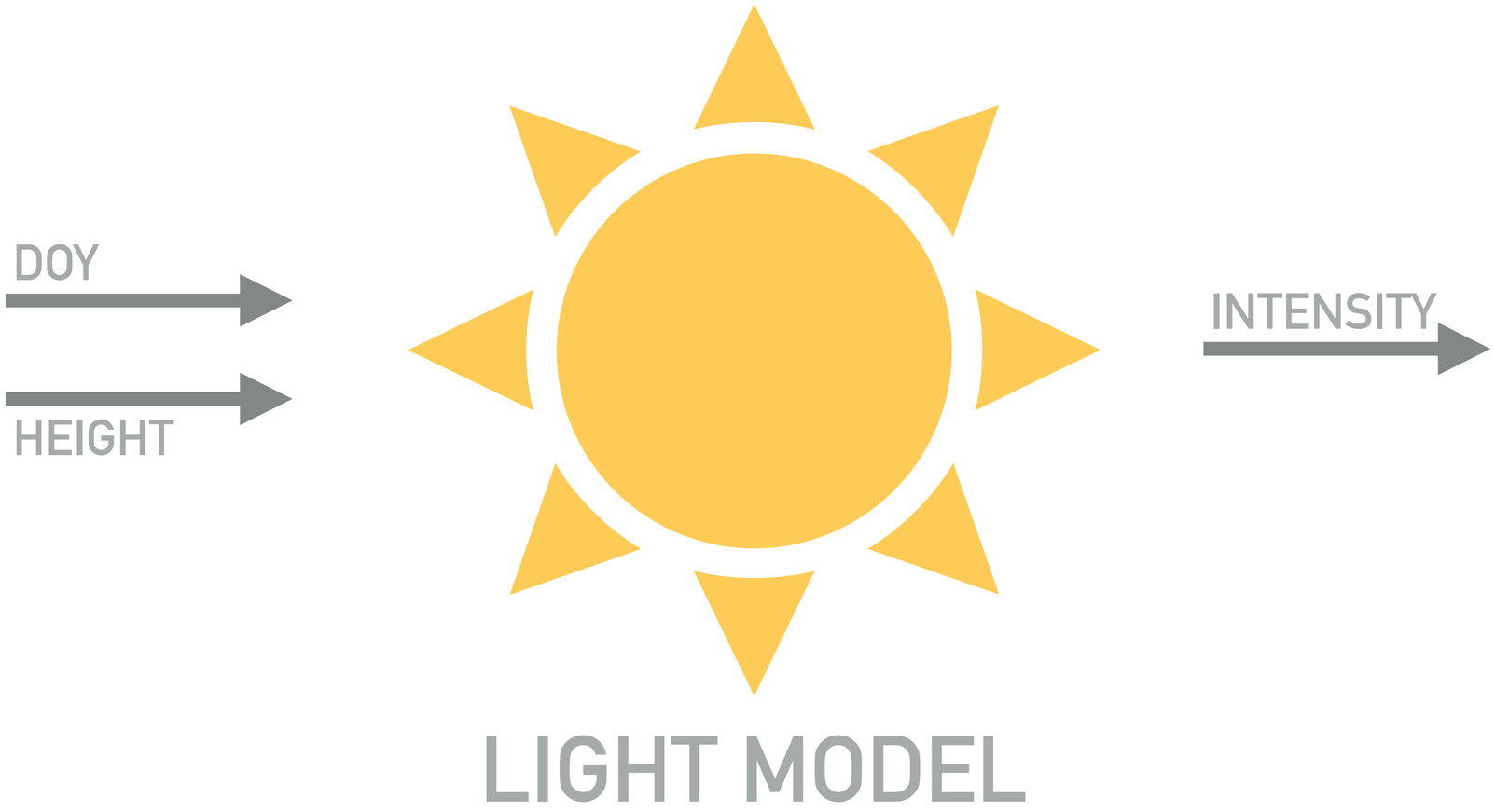
function - name of the function that yggdrasil should wrap

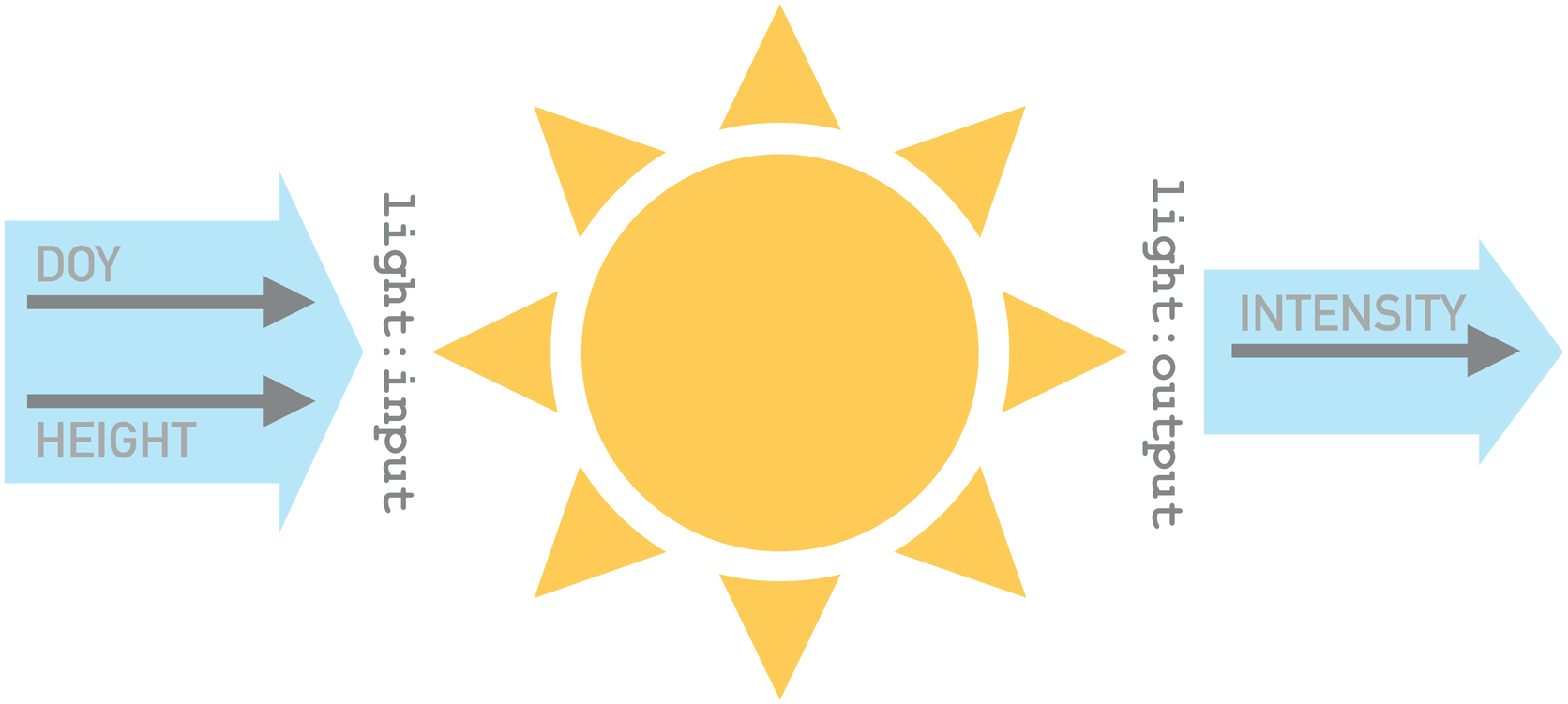
```
In [5]: tools.display_source('yamls/light_v0_python.yml', number_lines=True)
```

```
file: yamls/light_v0_python.yml
================================
 1: model:
 2:   name: light
 3:   language: python
 4:   args: ../models/light_v0.py
 5:   function: light
```

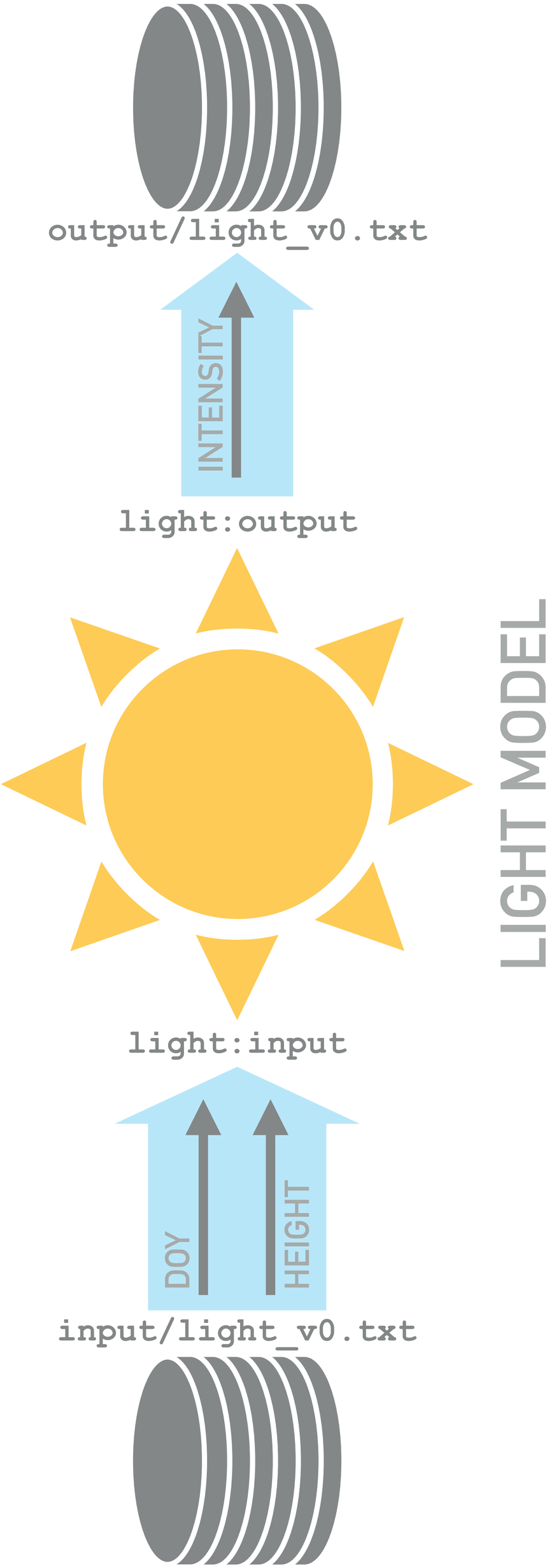
YAMLs provide info needed to run model

function - name of the function that yggdrasil should wrap


DOY
HEIGHT
INTENSITY
LIGHT MODEL



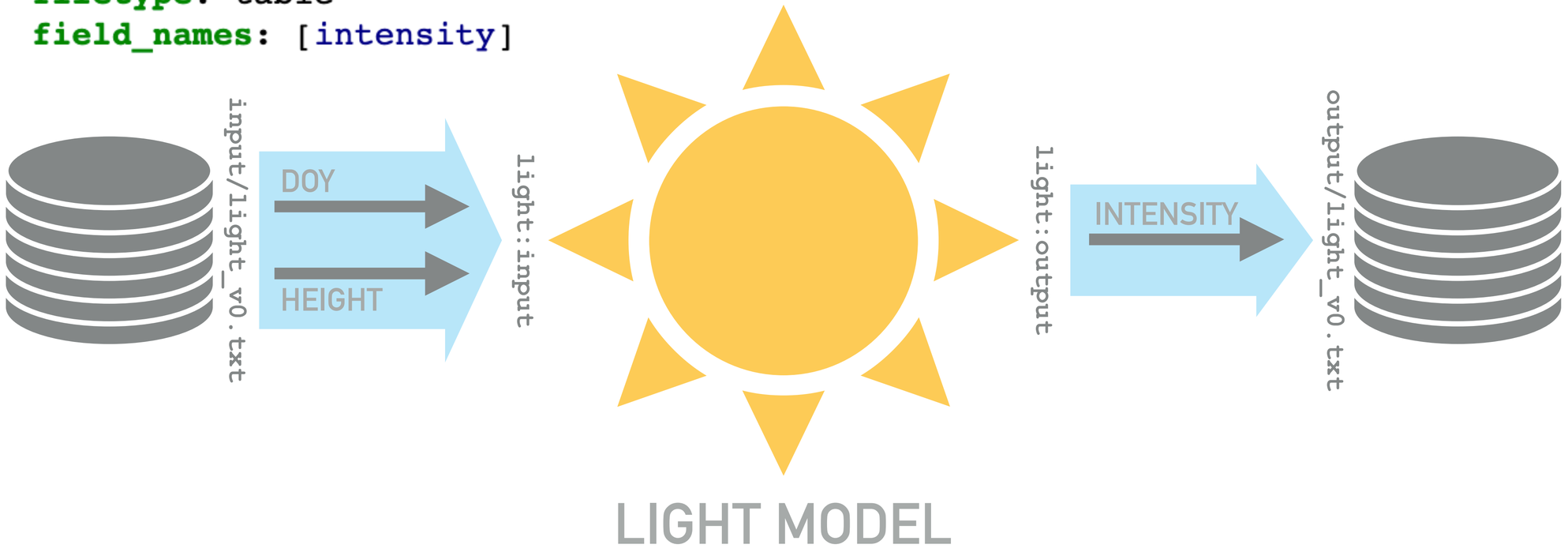
LIGHT MODEL


output/light_v0.txt
INTENSITY
light:output
LIGHT MODEL
light:input
DOY
HEIGHT
input/light_v0.txt

input/light_v0.txt
DOY
HEIGHT
light:input
light:output
INTENSITY
output/light_v0.txt
LIGHT MODEL

In [6]:
```
tools.display_source('yamls/connections_v0.yml', number_lines=True)
```

```
file: yamls/connections_v0.yml
===============================
 1: connections:
 2:   - input:
 3:       name: ../input/light_v0.txt
 4:       filetype: table
 5:     output: light:input
 6:   - input: light:output
 7:     output:
 8:       name: ../output/light_v0.txt
 9:       filetype: table
10:       field_names: [intensity]
```



In [7]:
```
run(['yamls/light_v0_python.yml', 'yamls/connections_v0.yml'], production_run=True)
```

In [7]:
```
run(['yamls/light_v0_python.yml', 'yamls/connections_v0.yml'], production_run=True)
```

```
INFO:88383:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
End of input from temp_doy.
INFO:88383:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:88383:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:88383:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:88383:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:88383:runner.run[374]:YggRunner(runner):         load drivers      0.309536
INFO:88383:runner.run[374]:YggRunner(runner):        start drivers      0.091199
INFO:88383:runner.run[374]:YggRunner(runner):           run models      5.400952
INFO:88383:runner.run[374]:YggRunner(runner):              at exit      0.023104
INFO:88383:runner.run[376]:YggRunner(runner): ========================================
INFO:88383:runner.run[377]:YggRunner(runner):                Total      5.824792
```

In [7]:
```
run(['yamls/light_v0_python.yml', 'yamls/connections_v0.yml'], production_run=True)
```

```
INFO:88383:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
End of input from temp_doy.
INFO:88383:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:88383:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:88383:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:88383:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:88383:runner.run[374]:YggRunner(runner):          load drivers      0.309536
INFO:88383:runner.run[374]:YggRunner(runner):         start drivers      0.091199
INFO:88383:runner.run[374]:YggRunner(runner):            run models      5.400952
INFO:88383:runner.run[374]:YggRunner(runner):               at exit      0.023104
INFO:88383:runner.run[376]:YggRunner(runner): ========================================
INFO:88383:runner.run[377]:YggRunner(runner):                 Total      5.824792
```

In [8]:
```
tools.display_source('output/light_v0.txt', number_lines=True)
```

In [7]:
```
run(['yamls/light_v0_python.yml', 'yamls/connections_v0.yml'], production_run=True)
```

```
INFO:88383:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
End of input from temp_doy.
INFO:88383:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:88383:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:88383:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:88383:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:88383:runner.run[374]:YggRunner(runner):          load drivers      0.309536
INFO:88383:runner.run[374]:YggRunner(runner):         start drivers      0.091199
INFO:88383:runner.run[374]:YggRunner(runner):            run models      5.400952
INFO:88383:runner.run[374]:YggRunner(runner):               at exit      0.023104
INFO:88383:runner.run[376]:YggRunner(runner): ========================================
INFO:88383:runner.run[377]:YggRunner(runner):                 Total      5.824792
```

In [8]:
```
tools.display_source('output/light_v0.txt', number_lines=True)
```

```
file: output/light_v0.txt
=========================
 1: # intensity
 2: # erg/(cm**2*s)
 3: # %g
 4: 0
 5: 40.6885
 6: 82.7537
 7: 168.259
 8: 342.017
 9: 434.386
10: 617.737
11: 896.166
```

SOME NOTES

Usually run via the command line

jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2d9y79yl26:~$ yggrun -h
usage: Run an integration. [-h] [--loglevel LOGLEVEL] [--rmq-loglevel RMQ_LOGLEVEL] [--client-loglevel CLIENT_LOGLEVEL]
                           [--validate-components] [--validate-messages {False,True,First}] [--namespace NAMESPACE]
                           [--host HOST] [--vhost VHOST] [--user USER] [--password PASSWORD] [--cluster CLUSTER]
                           [--default-comm DEFAULT_COMM] [--production-run] [--debug]
                           yamlfile [yamlfile ...]

positional arguments:
  yamlfile              One or more yaml specification files.

optional arguments:
  -h, --help            show this help message and exit
  --loglevel LOGLEVEL   Logging level for yggdrasil operations.
  --rmq-loglevel RMQ_LOGLEVEL, --rmqloglevel RMQ_LOGLEVEL
                        Logging level for RabbitMQ operations.
  --client-loglevel CLIENT_LOGLEVEL, --clientloglevel CLIENT_LOGLEVEL
                        Logging level for yggdrasil operations on model processes.
  --validate-components, --validatecomponents
                        Validate components on creation using their JSON schema (Decreases performance).
  --validate-messages {False,True,First}, --validatemessages {False,True,First}
                        Which messages should be validated during communication. 'True': all messages (decreases
                        performance), 'False': no messages, or 'First': only the first message a comm sends/receives.
  --namespace NAMESPACE
                        RabbitMQ namespace.
  --host HOST           RabbitMQ host address.
  --vhost VHOST         RabbitMQ virtual host address.
  --user USER           RabbitMQ username.
  --password PASSWORD   RabbitMQ password.
  --cluster CLUSTER     Cluster that should be used.
  --default-comm DEFAULT_COMM, --defaultcomm DEFAULT_COMM
                        Comm type that should be used by default.
```

SOME NOTES

```
In [7]: run(['yamls/light_v0_python.yml', 'yamls/connections_v0.yml', production_run=True)
```

```
INFO:88383:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
End of input from temp_doy.
INFO:88383:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:88383:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:88383:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:88383:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:88383:runner.run[374]:YggRunner(runner):          load drivers      0.309536
INFO:88383:runner.run[374]:YggRunner(runner):         start drivers      0.091199
INFO:88383:runner.run[374]:YggRunner(runner):            run models      5.400952
INFO:88383:runner.run[374]:YggRunner(runner):               at exit      0.023104
INFO:88383:runner.run[376]:YggRunner(runner): ========================================
INFO:88383:runner.run[377]:YggRunner(runner):                 Total      5.824792
```

"production_run" flag turns off checks to improve performance and should only be used after testing

In [9]:
```
tools.display_source('models/light_v0.cpp', number_lines=True)
tools.display_source('yamls/light_v0_cpp.yml', number_lines=True)
run(['yamls/light_v0_cpp.yml', 'yamls/connections_v0.yml'], production_run=True)
tools.display_source('output/light_v0.txt', number_lines=True)
```

OTHER LANGUAGES

OTHER LANGUAGES

```
In [9]: tools.display_source('models/light_v0.cpp', number_lines=True)
        tools.display_source('yamls/light_v0_cpp.yml', number_lines=True)
        run(['yamls/light_v0_cpp.yml', 'yamls/connections_v0.yml'], production_run=True)
        tools.display_source('output/light_v0.txt', number_lines=True)
```

```
file: models/light_v0.cpp
=========================
 1: #define _USE_MATH_DEFINES  // Required to use M_PI with MSVC
 2: #include <math.h>
 3:
 4: /*!
 5:   @brief Compute the intensity of light.
 6:
 7:   @param[in] doy Day of year.
 8:   @param[in] height Distance from ground in cm.
 9:
10:   @returns intensity Intensity of light in ergs cm^-2 s^-1.
11:  */
12: double light(double doy, double height) {
13:   // Define parameters that are static across a run
14:   double amplitude = 80.0;
15:   double doy_offset = 0.0;
16:
17:   // Calculate intensity
18:   double intensity = amplitude * height * (1.0 + sin(2.0 * M_PI * (doy - doy_offset) / 365));
19:
20:   return intensity;
21: }
```

OTHER LANGUAGES

```
In [9]: tools.display_source('models/light_v0.cpp', number_lines=True)
        tools.display_source('yamls/light_v0_cpp.yml', number_lines=True)
        run(['yamls/light_v0_cpp.yml', 'yamls/connections_v0.yml'], production_run=True)
        tools.display_source('output/light_v0.txt', number_lines=True)
```

```
file: yamls/light_v0_cpp.yml
============================
 1: model:
 2:   name: light
 3:   language: c++
 4:   args: ../models/light_v0.cpp
 5:   function: light
 6:   inputs:
 7:     - name: input
 8:       vars: [doy, height]
 9:       datatype:
10:         type: array
11:         items:
12:           - type: float
13:             units: day
14:           - type: float
15:             units: cm
16:   output:
17:     - name: output
18:       datatype:
19:         type: float
20:         units: ergs/(cm**2*s)
```

OTHER LANGUAGES

```
In [9]: tools.display_source('models/light_v0.cpp', number_lines=True)
        tools.display_source('yamls/light_v0_cpp.yml', number_lines=True)
        run(['yamls/light_v0_cpp.yml', 'yamls/connections_v0.yml'], production_run=True)
        tools.display_source('output/light_v0.txt', number_lines=True)
```

```
file: yamls/light_v0_cpp.yml
============================
 1: model:
 2:   name: light
 3:   language: c++
 4:   args: ../models/light_v0.cpp
 5:   function: light
 6:   inputs:
 7:     - name: input
 8:       vars: [doy, height]
 9:       datatype:
10:         type: array
11:         items:
12:           - type: float
13:             units: day
14:           - type: float
15:             units: cm
16:   output:
17:     - name: output
18:       datatype:
19:         type: float
20:         units: ergs/(cm**2*s)
```

Inputs & outputs are explicit in the compiled languages so that units can be specified

In [9]:
```
tools.display_source('models/light_v0.cpp', number_lines=True)
tools.display_source('yamls/light_v0_cpp.yml', number_lines=True)
run(['yamls/light_v0_cpp.yml', 'yamls/connections_v0.yml'], production_run=True)
tools.display_source('output/light_v0.txt', number_lines=True)
```

```
INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg_light_v0_cpp_clang++x_clang++x.out
End of input from &doy, &height.
INFO:91854:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:91854:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:91854:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:91854:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:91854:runner.run[374]:YggRunner(runner):          load drivers      3.856998
INFO:91854:runner.run[374]:YggRunner(runner):         start drivers      0.148603
INFO:91854:runner.run[374]:YggRunner(runner):            run models      0.533877
INFO:91854:runner.run[374]:YggRunner(runner):               at exit      0.075296
INFO:91854:runner.run[376]:YggRunner(runner): ========================================
INFO:91854:runner.run[377]:YggRunner(runner):                 Total      4.614775
```

OTHER LANGUAGES

In [9]:
```
tools.display_source('models/light_v0.cpp', number_lines=True)
tools.display_source('yamls/light_v0_cpp.yml', number_lines=True)
run(['yamls/light_v0_cpp.yml', 'yamls/connections_v0.yml'], production_run=True)
tools.display_source('output/light_v0.txt', number_lines=True)
```

```
INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg_light_v0_cpp_clang++x_clang++x.out
End of input from &doy, &height.
INFO:91854:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:91854:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:91854:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:91854:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:91854:runner.run[374]:YggRunner(runner):          load drivers      3.856998
INFO:91854:runner.run[374]:YggRunner(runner):         start drivers      0.148603
INFO:91854:runner.run[374]:YggRunner(runner):            run models      0.533877
INFO:91854:runner.run[374]:YggRunner(runner):               at exit      0.075296
INFO:91854:runner.run[376]:YggRunner(runner): ========================================
INFO:91854:runner.run[377]:YggRunner(runner):                 Total      4.614775

file: output/light_v0.txt
=========================
 1: # intensity
 2: # ergs/(cm**2*s)
 3: # %g
 4: 0
 5: 40.6885
 6: 82.7537
 7: 168.259
 8: 342.017
 9: 434.386
10: 617.737
11: 896.166
```

OTHER LANGUAGES

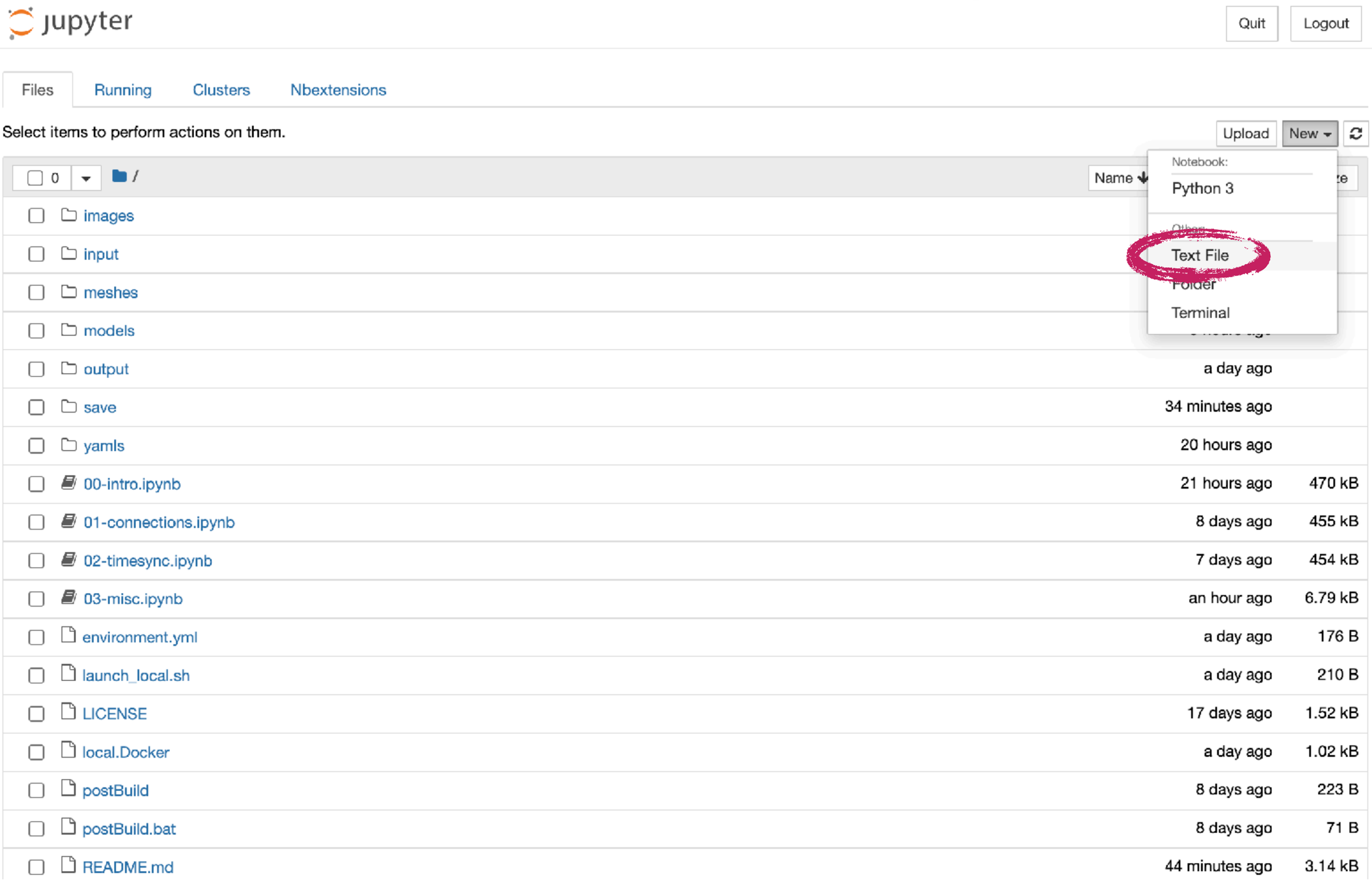
Test your knowledge #1

1. Given the model located at `models/weather.py`, write a YAML to run the model in isolation, taking input from the `input/intensity.txt` file and outputing to `output/temp.txt` as a table.
2. Run the model in the empty code cell below using the `run` method.
3. Write a function in your favorite programming language (out of Python, R, Fortran, C, or C++), write a YAML to run it in isolation, and run you model in the cell below.

Tip: You can open and create files from the browser page

TEST YOUR KNOWLEDGE

Need to be in desired directory



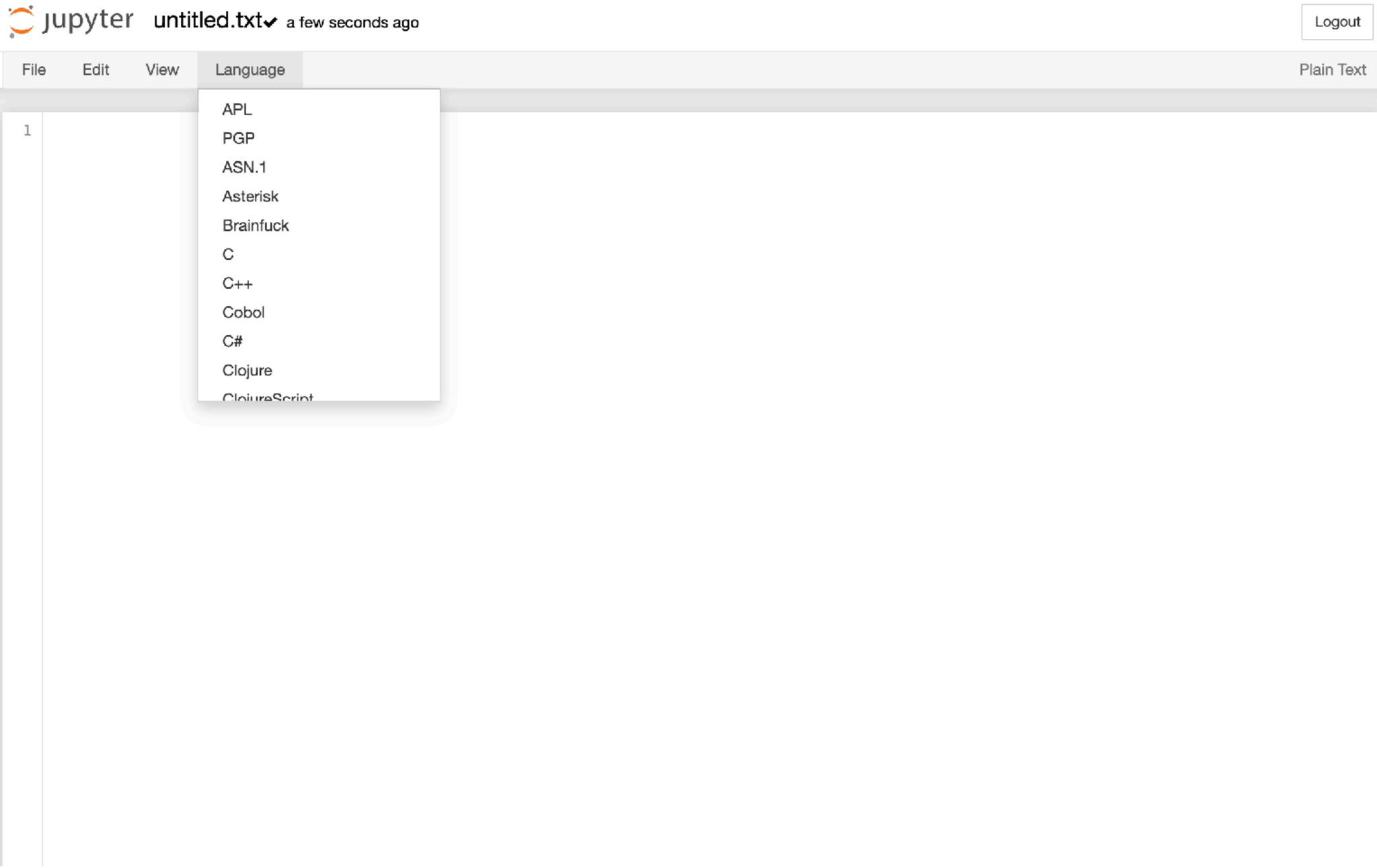
jupyter untitled.txt a few seconds ago

Logout

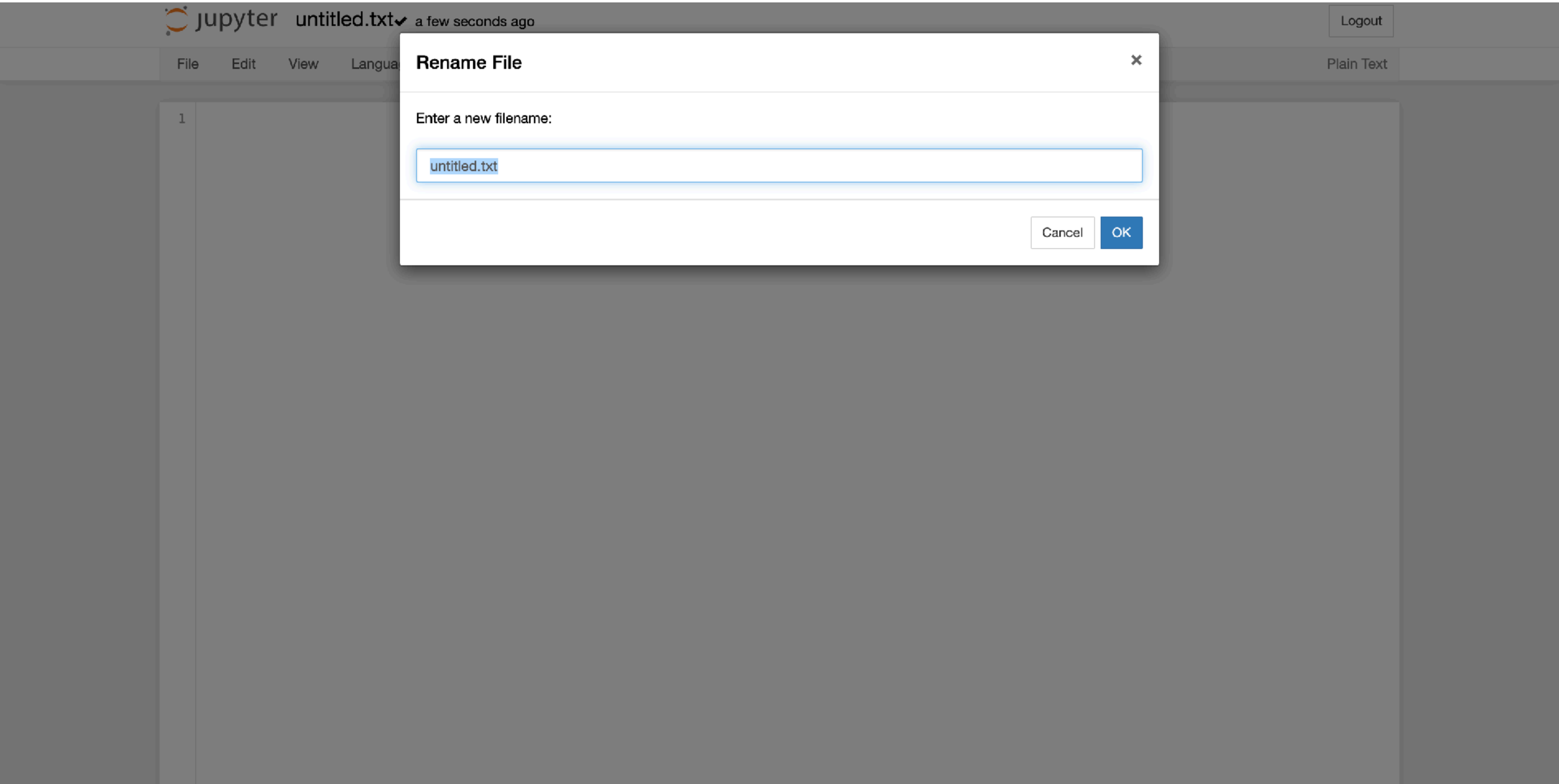
File Edit View Language Plain Text

1

Select the language you want to write in



Rename the file



SOLUTIONS

Test your knowledge #1

1. Given the model located at `models/weather.py`, write a YAML to run the model in isolation, taking input from the `input/intensity.txt` file and outputing to `output/temp.txt` as a table.
2. Run the model in the empty code cell below using the `run` method.
3. Write a function in your favorite programming language (out of Python, R, Fortran, C, or C++), write a YAML to run it in isolation, and run you model in the cell below.

Tip: You can open and create files from the browser page

```
In [ ]: %load solutions/tyk1/run.py
```

SOLUTIONS

Test your knowledge #1

1. Given the model located at `models/weather.py`, write a YAML to run the model in isolation, taking input from the `input/intensity.txt` file and outputing to `output/temp.txt` as a table.
2. Run the model in the empty code cell below using the `run` method.
3. Write a function in your favorite programming language (out of Python, R, Fortran, C, or C++), write a YAML to run it in isolation, and run you model in the cell below.

Tip: You can open and create files from the browser page

In [ ]:
```
# %load solutions/tyk1/run.py
import os
from yggdrasil import tools
from yggdrasil.runner import run
tyk_dir = 'tyk1'

try:
    # Change to the solution directory
    old_dir = os.getcwd()
    if not old_dir.endswith(tyk_dir):
        os.chdir(os.path.join('solutions', tyk_dir))
    if not os.path.isdir('output'):
        os.mkdir('output')

    # Part 1: YAML
    tools.display_source('yaml/weather.yml', number_lines=True)
```

Test your knowledge #1

1. Given the model located at `models/weather.py`, write a YAML to run the model in isolation, taking input from the `input/intensity.txt` file and outputing to `output/temp.txt` as a table.
2. Run the model in the empty code cell below using the `run` method.
3. Write a function in your favorite programming language (out of Python, R, Fortran, C, or C++), write a YAML to run it in isolation, and run you model in the cell below.

Tip: You can open and create files from the browser page

TEST YOUR KNOWLEDGE (15 MIN)

Test your knowledge #1

1. Given the model located at `models/weather.py`, write a YAML to run the model in isolation, taking input from the `input/intensity.txt` file and outputing to `output/temp.txt` as a table.

In [2]:
```
from yggdrasil import tools
from yggdrasil.runner import run
# Part 1: YAML
tools.display_source('solutions/tyk1/yamls/weather.yml', number_lines=True)
# Part 2: Run
run(['solutions/tyk1/yamls/weather.yml'], production_run=True)
```

```
file: solutions/tyk1/yamls/weather.yml
======================================
 1: model:
 2:   name: weather
 3:   language: python
 4:   args: ../models/weather.py
 5:   function: temp
 6:
 7: connections:
 8:   - input:
 9:       name: ../input/intensity.txt
10:       filetype: table
11:     output: weather:input
12:   - input: weather:output
13:     output:
14:       name: ../output/temp.txt
15:       filetype: table
16:       field_names: [temp]
```

Test your knowledge #1

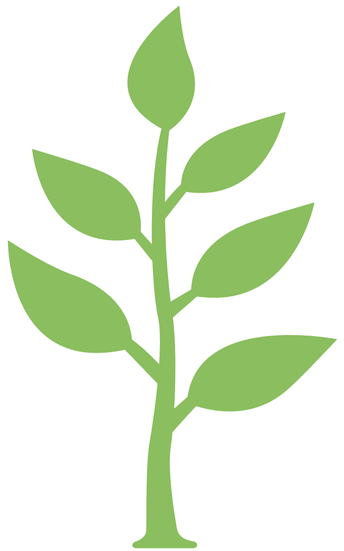
2. Run the model in the empty code cell below using the `run` method.

In [2]:
```
from yggdrasil import tools
from yggdrasil.runner import run
# Part 1: YAML
tools.display_source('solutions/tyk1/yamls/weather.yml', number_lines=True)
# Part 2: Run
run(['solutions/tyk1/yamls/weather.yml'], production_run=True)
```

```
INFO:60480:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk1/models/ygg_weather.py
End of input from intensity.
INFO:60480:runner.waitModels[553]:YggRunner(runner): weather finished running.
INFO:60480:runner.waitModels[559]:YggRunner(runner): weather finished exiting.
INFO:60480:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:60480:runner.run[374]:YggRunner(runner):                 init      0.000000
INFO:60480:runner.run[374]:YggRunner(runner):         load drivers      0.562023
INFO:60480:runner.run[374]:YggRunner(runner):        start drivers      0.134939
INFO:60480:runner.run[374]:YggRunner(runner):           run models      11.390312
INFO:60480:runner.run[374]:YggRunner(runner):              at exit      0.064951
INFO:60480:runner.run[376]:YggRunner(runner): ========================================
INFO:60480:runner.run[377]:YggRunner(runner):                Total      12.152225
```

INTEGRATING MODELS VIA INTERFACE

```
In [12]: tools.display_source('models/shoot_v0.py', number_lines=True)
```

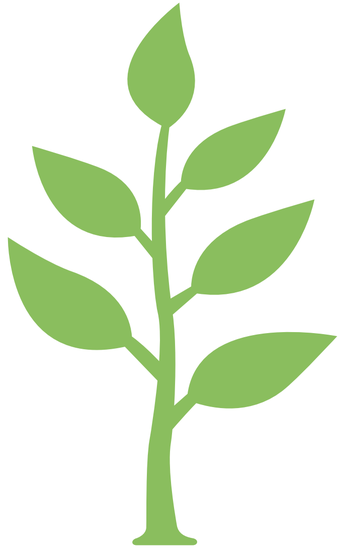


SHOOT
MODEL

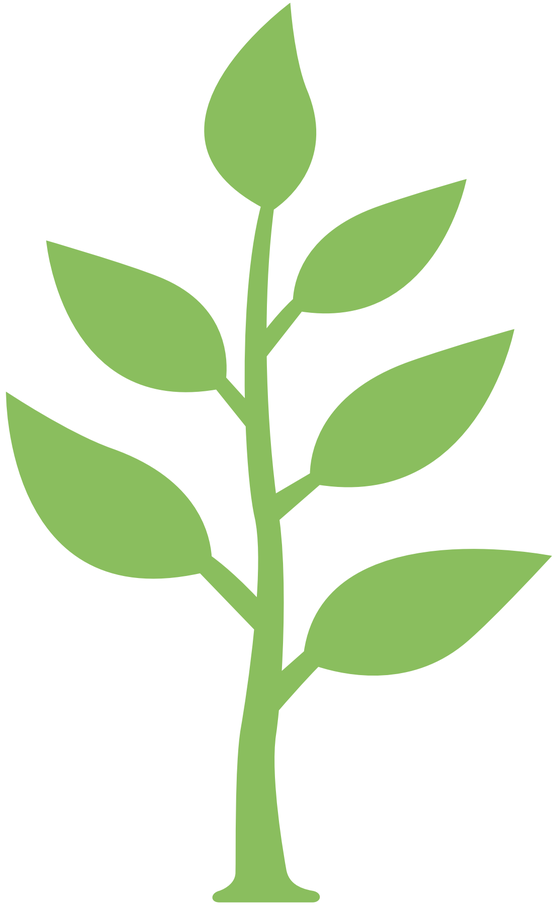
```
In [12]: tools.display_source('models/shoot_v0.py', number_lines=True)
```

file: models/shoot_v0.py

```
 1: import os
 2: import trimesh
 3: import argparse
 4:
 5: _dir = os.path.dirname(os.path.realpath(__file__))
 6:
 7: # Parse command-line arguments
 8: parser = argparse.ArgumentParser("Simulate a shoot's growth over time.")
 9: parser.add_argument('tmin', help='Starting time (in hours)', type=float)
10: parser.add_argument('tmax', help='Ending time (in hours)', type=float)
11: parser.add_argument('tstep', help='Time step (in hours)', type=float)
12: parser.add_argument('--meshfile', help='Path to file where mesh is stored.',
13:                     default='../meshes/plants-2.obj')
14: args = parser.parse_args()
15: tmin = args.tmin
16: tmax = args.tmax
17: tstep = args.tstep
18: mesh = trimesh.load_mesh(args.meshfile)
19:
20: # Set initial conditions
21: mass = 2000.0
22: t = tmin
23: i = 0
24:
25: # Continue simulation until time limit is reached
26: while t <= tmax:
27:
28:     # Compute the scale factor
29:     # (pretend this is a biologically complex calculation)
30:     scale = mass / 4.5e4
31:
32:     # Grow the shoot
33:     # (pretend this is a biologically complex calculation)
34:     mesh.vertices[:, 2] += mesh.vertices[:, 2] * scale
35:     mass += mass * scale
36:
37:     # Save mesh for this timestep
38:     filename_mesh = os.path.join(_dir, f'../output/mesh_{i:03d}.obj')
39:     with open(filename_mesh, 'w') as fd:
40:         mesh.export(fd, 'obj')
41:
42:     # Advance time step
43:     t += tstep
44:     i += 1
```



SHOOT
MODEL



SHOOT
MODEL

In [13]:
```
tools.display_source('yamls/shoot_v0.yml', number_lines=True)
run('yamls/shoot_v0.yml', production_run=True)
```

In [13]:
```
tools.display_source('yamls/shoot_v0.yml', number_lines=True)
run('yamls/shoot_v0.yml', production_run=True)
```

```
file: yamls/shoot_v0.yml
========================
 1: model:
 2:   name: shoot
 3:   language: python
 4:   args: [../models/shoot_v0.py, 0.0, 48.0, 6.0]
```

In [13]:
```
tools.display_source('yamls/shoot_v0.yml', number_lines=True)
run('yamls/shoot_v0.yml', production_run=True)
```

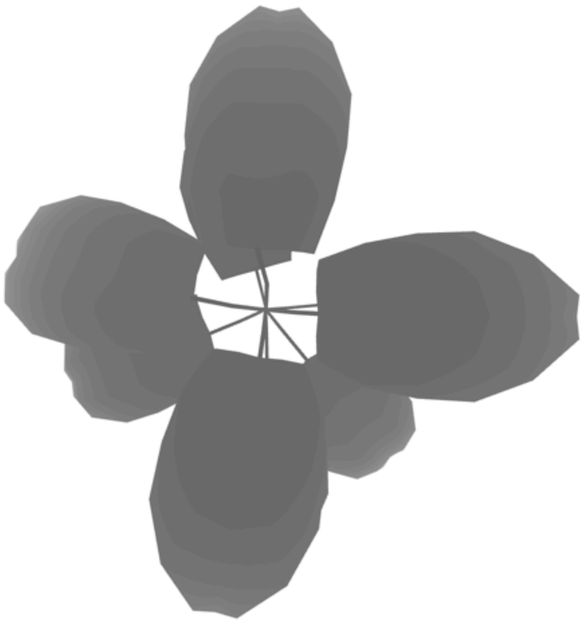
```
file: yamls/shoot_v0.yml
========================
 1: model:
 2:   name: shoot
 3:   language: python
 4:   args: [../models/shoot_v0.py, 0.0, 48.0, 6.0]

INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v0.py 0.0 48.0 6.0
INFO:91854:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:91854:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:91854:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:91854:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:91854:runner.run[374]:YggRunner(runner):         load drivers      0.000918
INFO:91854:runner.run[374]:YggRunner(runner):        start drivers      0.042213
INFO:91854:runner.run[374]:YggRunner(runner):           run models      0.856992
INFO:91854:runner.run[374]:YggRunner(runner):              at exit      0.000524
INFO:91854:runner.run[376]:YggRunner(runner): ========================================
INFO:91854:runner.run[377]:YggRunner(runner):                Total      0.900648
```

In [13]:
```
tools.display_source('yamls/shoot_v0.yml', number_lines=True)
run('yamls/shoot_v0.yml', production_run=True)
```

```
file: yamls/shoot_v0.yml
========================
 1: model:
 2:   name: shoot
 3:   language: python
 4:   args: [../models/shoot_v0.py, 0.0, 48.0, 6.0]

INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v0.py 0.0 48.0 6.0
INFO:91854:runner.waitModels[553]:Y
INFO:91854:runner.waitModels[559]:Y
INFO:91854:runner.waitModels[573]:Y
INFO:91854:runner.run[374]:YggRunne
INFO:91854:runner.run[374]:YggRunne
INFO:91854:runner.run[374]:YggRunne
INFO:91854:runner.run[374]:YggRunne
INFO:91854:runner.run[374]:YggRunne
INFO:91854:runner.run[376]:YggRunne
INFO:91854:runner.run[377]:YggRunne
```

In [14]:
```
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

Out[14]:



In [15]:
```
tools.display_source_diff('models/shoot_v0.py', 'models/shoot_v1.py', number_lines=True)
```

```
In [15]: tools.display_source_diff('models/shoot_v0.py', 'models/shoot_v1.py', number_lines=True)
```

```diff
file1: models/shoot_v0.py
file2: models/shoot_v1.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse
 4:
 5:   _dir = os.path.dirname(os.path.realpath(__file__))
 6:
 7:   # Parse command-line arguments
 8:   parser = argparse.ArgumentParser("Simulate a shoot's growth over time.")
 9:   parser.add_argument('tmin', help='Starting time (in hours)', type=float)
10:   parser.add_argument('tmax', help='Ending time (in hours)', type=float)
11:   parser.add_argument('tstep', help='Time step (in hours)', type=float)
12:   parser.add_argument('--meshfile', help='Path to file where mesh is stored.',
13:                        default='../meshes/plants-2.obj')
14:   args = parser.parse_args()
15:   tmin = args.tmin
16:   tmax = args.tmax
17:   tstep = args.tstep
18:   mesh = trimesh.load_mesh(args.meshfile)
19:
20:   # Set initial conditions
21:   mass = 2000.0
22:   t = tmin
23:   i = 0
24:
25: + # Check if model is running as a part of an yggdrasil integration
26: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
27: +
28: + # If the model is running as part of an yggdrasil integration, import
29: + # the relevant yggdrasil routines and use the interface routine to
30: + # complete the connection defined in the YAML
31: + if with_yggdrasil:
32: +     from yggdrasil import units
33: +     from yggdrasil.languages.Python.YggInterface import YggOutput
34: +     height_out = YggOutput('height')
35: +
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39: +     # If running as part an yggdrasil integration, send the time and
40: +     # maximum height of the mesh to the height channel with units
41: +     if with_yggdrasil:
42: +         flag = height_out.send(
43: +             [units.add_units(t, 'hrs'),
44: +              units.add_units(max(mesh.vertices[:, 2]), 'm')])
45: +         if not flag:
46: +             raise Exception("Error sending height to output")
47: +
48:       # Compute the scale factor
49:       # (pretend this is a biologically complex calculation)
50:       scale = mass / 4.5e4
51:
52:       # Grow the shoot
53:       # (pretend this is a biologically complex calculation)
54:       mesh.vertices[:, 2] += mesh.vertices[:, 2] * scale
55:       mass += mass * scale
56:
57:       # Save mesh for this timestep
58:       filename_mesh = os.path.join(_dir, f'../output/mesh_{i:03d}.obj')
59:       with open(filename_mesh, 'w') as fd:
60:           mesh.export(fd, 'obj')
61:
62:       # Advance time step
63:       t += tstep
64:       i += 1
```

Code to call yggdrasil inside “if blocks” so that model runs exactly the same without yggdrasil

```
In [15]: tools.display_source_diff('models/shoot_v0.py', 'models/shoot_v1.py', number_lines=True)
```

```
file1: models/shoot_v0.py
file2: models/shoot_v1.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse
 4:
 5:   _dir = os.path.dirname(os.path.realpath(__file__))
 6:
 7:   # Parse command-line arguments
 8:   parser = argparse.ArgumentParser("Simulate a shoot's growth over time.")
 9:   parser.add_argument('tmin', help='Starting time (in hours)', type=float)
10:   parser.add_argument('tmax', help='Ending time (in hours)', type=float)
11:   parser.add_argument('tstep', help='Time step (in hours)', type=float)
12:   parser.add_argument('--meshfile', help='Path to file where mesh is stored.',
13:                        default='../meshes/plants-2.obj')
14:   args = parser.parse_args()
15:   tmin = args.tmin
16:   tmax = args.tmax
17:   tstep = args.tstep
18:   mesh = trimesh.load_mesh(args.meshfile)
19:
20:   # Set initial conditions
21:   mass = 2000.0
22:   t = tmin
23:   i = 0
24:
25: + # Check if model is running as a part of an yggdrasil integration
26: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
27: +
28: + # If the model is running as part of an yggdrasil integration, import
29: + # the relevant yggdrasil routines and use the interface routine to
30: + # complete the connection defined in the YAML
31: + if with_yggdrasil:
32: +     from yggdrasil import units
33: +     from yggdrasil.languages.Python.YggInterface import YggOutput
34: +     height_out = YggOutput('height')
35: +
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39: +     # If running as part an yggdrasil integration, send the time and
40: +     # maximum height of the mesh to the height channel with units
41: +     if with_yggdrasil:
42: +         flag = height_out.send(
43: +             [units.add_units(t, 'hrs'),
44: +              units.add_units(max(mesh.vertices[:, 2]), 'm')])
45: +         if not flag:
46: +             raise Exception("Error sending height to output")
47: +
48:       # Compute the scale factor
49:       # (pretend this is a biologically complex calculation)
50:       scale = mass / 4.5e4
51:
52:       # Grow the shoot
53:       # (pretend this is a biologically complex calculation)
54:       mesh.vertices[:, 2] += mesh.vertices[:, 2] * scale
55:       mass += mass * scale
56:
57:       # Save mesh for this timestep
58:       filename_mesh = os.path.join(_dir, f'../output/mesh_{i:03d}.obj')
59:       with open(filename_mesh, 'w') as fd:
60:           mesh.export(fd, 'obj')
61:
62:       # Advance time step
63:       t += tstep
64:       i += 1
```

Import yggdrasil functions and connect to the channel that will be listed in the YAML.

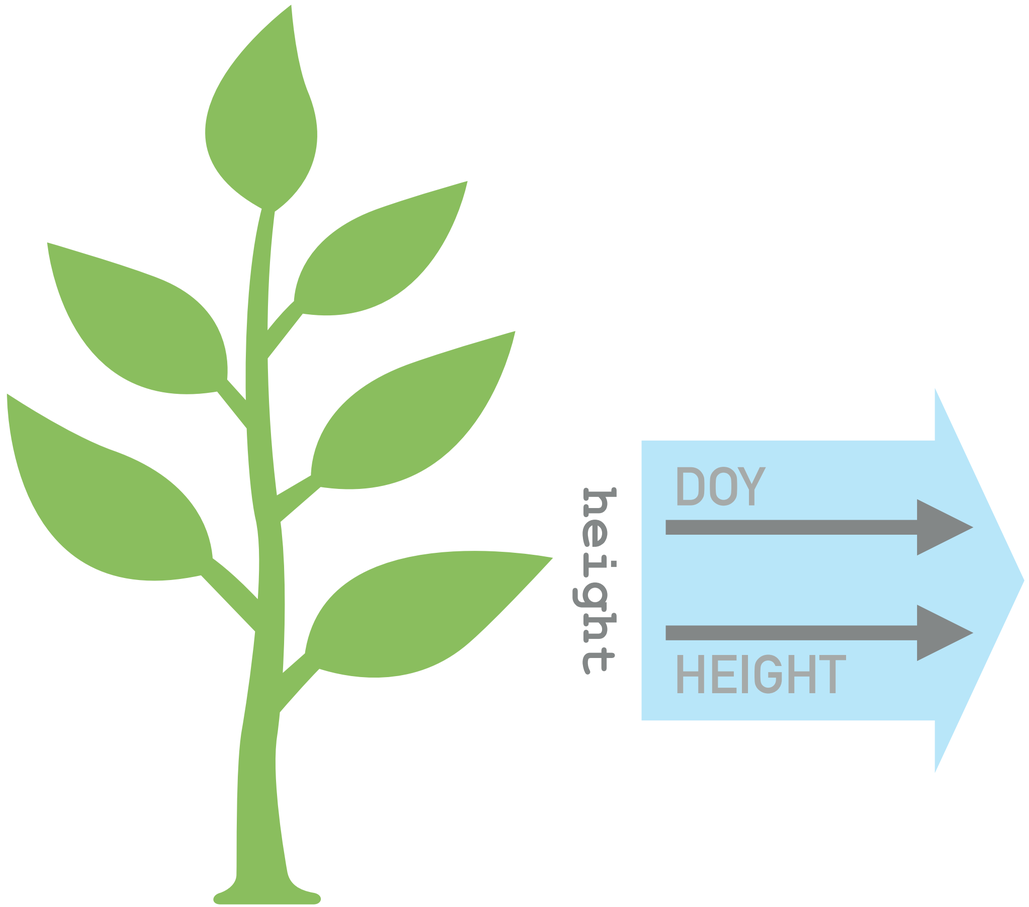
Code to call yggdrasil inside “if blocks” so that model runs exactly the same without yggdrasil

```python
In [15]: tools.display_source_diff('models/shoot_v0.py', 'models/shoot_v1.py', number_lines=True)
```

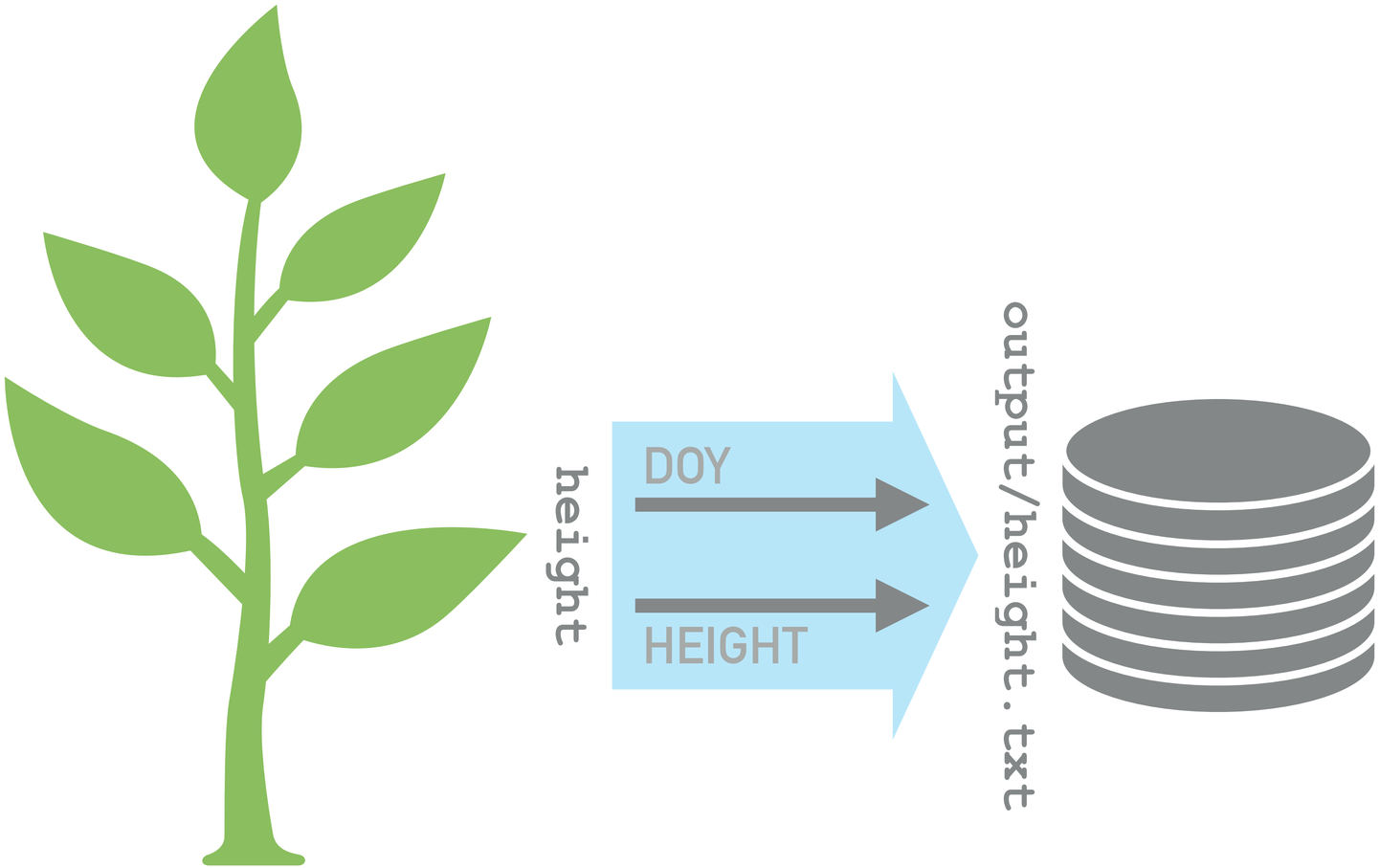
```diff
file1: models/shoot_v0.py
file2: models/shoot_v1.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse
 4:
 5:   _dir = os.path.dirname(os.path.realpath(__file__))
 6:
 7:   # Parse command-line arguments
 8:   parser = argparse.ArgumentParser("Simulate a shoot's growth over time.")
 9:   parser.add_argument('tmin', help='Starting time (in hours)', type=float)
10:   parser.add_argument('tmax', help='Ending time (in hours)', type=float)
11:   parser.add_argument('tstep', help='Time step (in hours)', type=float)
12:   parser.add_argument('--meshfile', help='Path to file where mesh is stored.',
13:                        default='../meshes/plants-2.obj')
14:   args = parser.parse_args()
15:   tmin = args.tmin
16:   tmax = args.tmax
17:   tstep = args.tstep
18:   mesh = trimesh.load_mesh(args.meshfile)
19:
20:   # Set initial conditions
21:   mass = 2000.0
22:   t = tmin
23:   i = 0
24:
25: + # Check if model is running as a part of an yggdrasil integration
26: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
27: +
28: + # If the model is running as part of an yggdrasil integration, import
29: + # the relevant yggdrasil routines and use the interface routine to
30: + # complete the connection defined in the YAML
31: + if with_yggdrasil:
32: +     from yggdrasil import units
33: +     from yggdrasil.languages.Python.YggInterface import YggOutput
34: +     height_out = YggOutput('height')
35: +
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39: +     # If running as part an yggdrasil integration, send the time and
40: +     # maximum height of the mesh to the height channel with units
41: +     if with_yggdrasil:
42: +         flag = height_out.send(
43: +             [units.add_units(t, 'hrs'),
44: +              units.add_units(max(mesh.vertices[:, 2]), 'm')])
45: +         if not flag:
46: +             raise Exception("Error sending height to output")
47: +
48:       # Compute the scale factor
49:       # (pretend this is a biologically complex calculation)
50:       scale = mass / 4.5e4
51:
52:       # Grow the shoot
53:       # (pretend this is a biologically complex calculation)
54:       mesh.vertices[:, 2] += mesh.vertices[:, 2] * scale
55:       mass += mass * scale
56:
57:       # Save mesh for this timestep
58:       filename_mesh = os.path.join(_dir, f'../output/mesh_{i:03d}.obj')
59:       with open(filename_mesh, 'w') as fd:
60:           mesh.export(fd, 'obj')
61:
62:       # Advance time step
63:       t += tstep
64:       i += 1
```

Send height to output channel

Code to call yggdrasil inside “if blocks” so that model runs exactly the same without yggdrasil

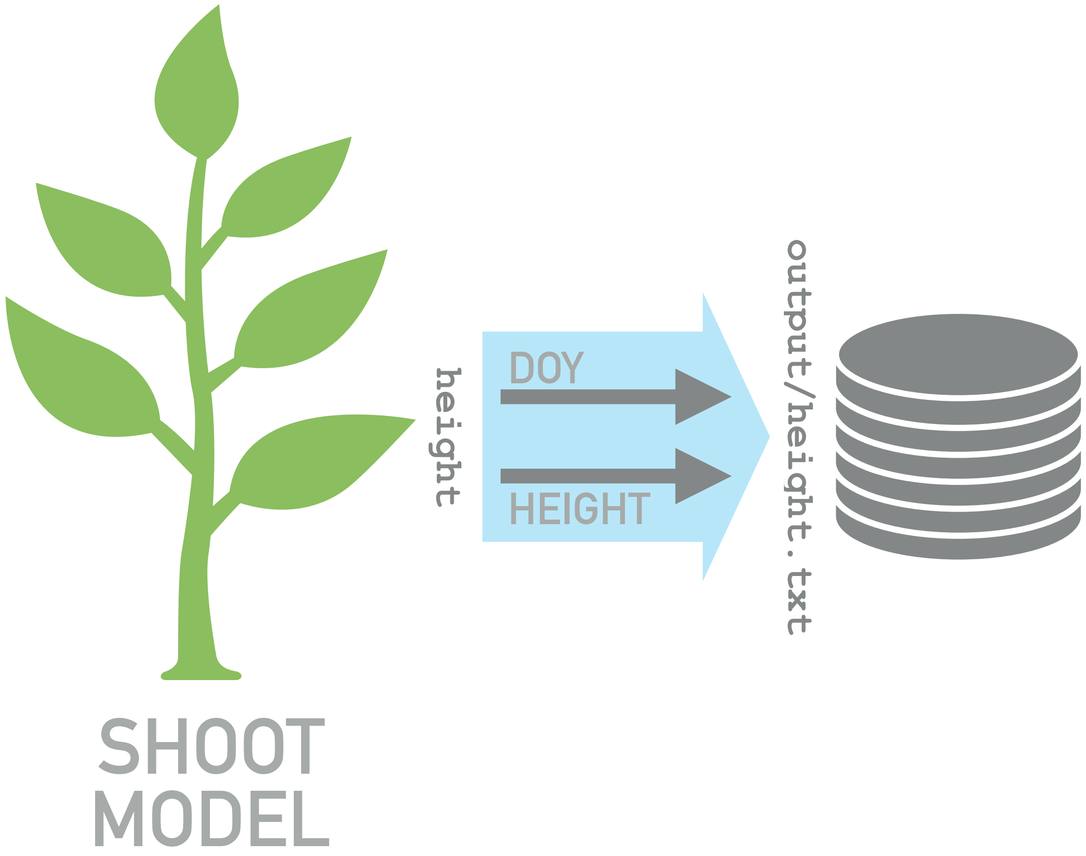


SHOOT
MODEL



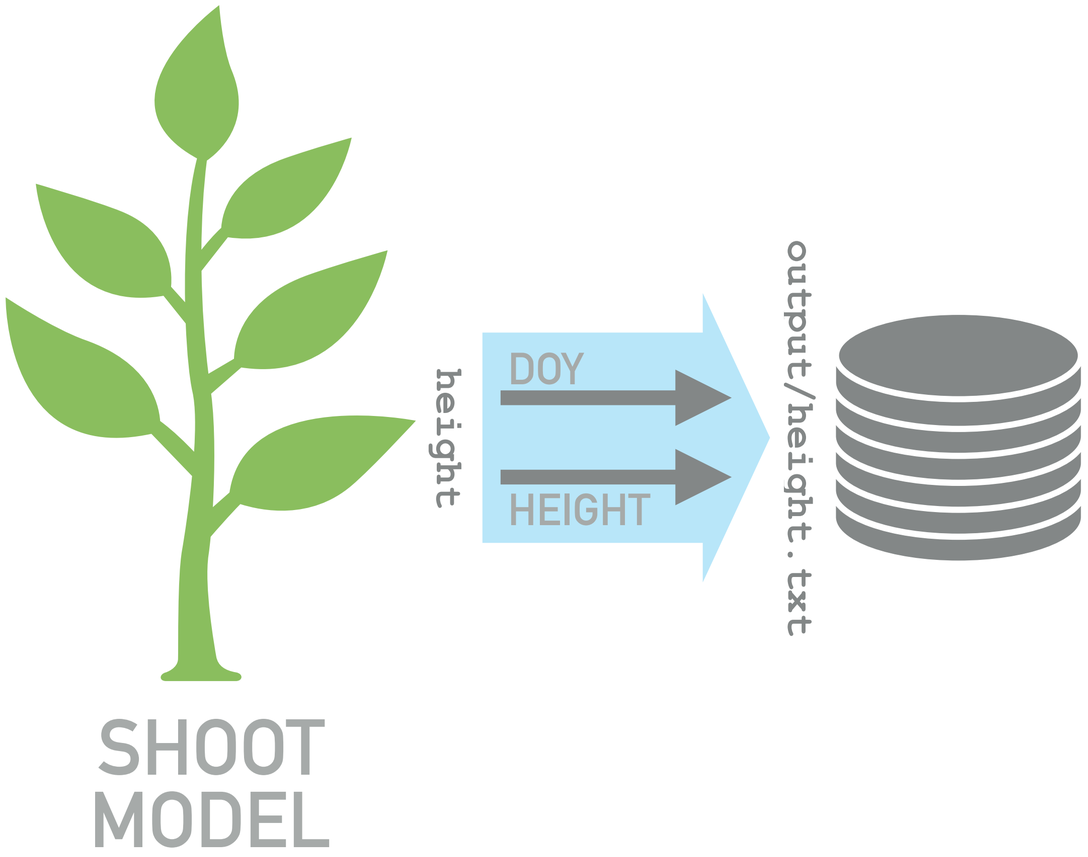
SHOOT
MODEL

```
In [16]: tools.display_source_diff('yamls/shoot_v0.yml', 'yamls/shoot_v1.yml', number_lines=True)
```



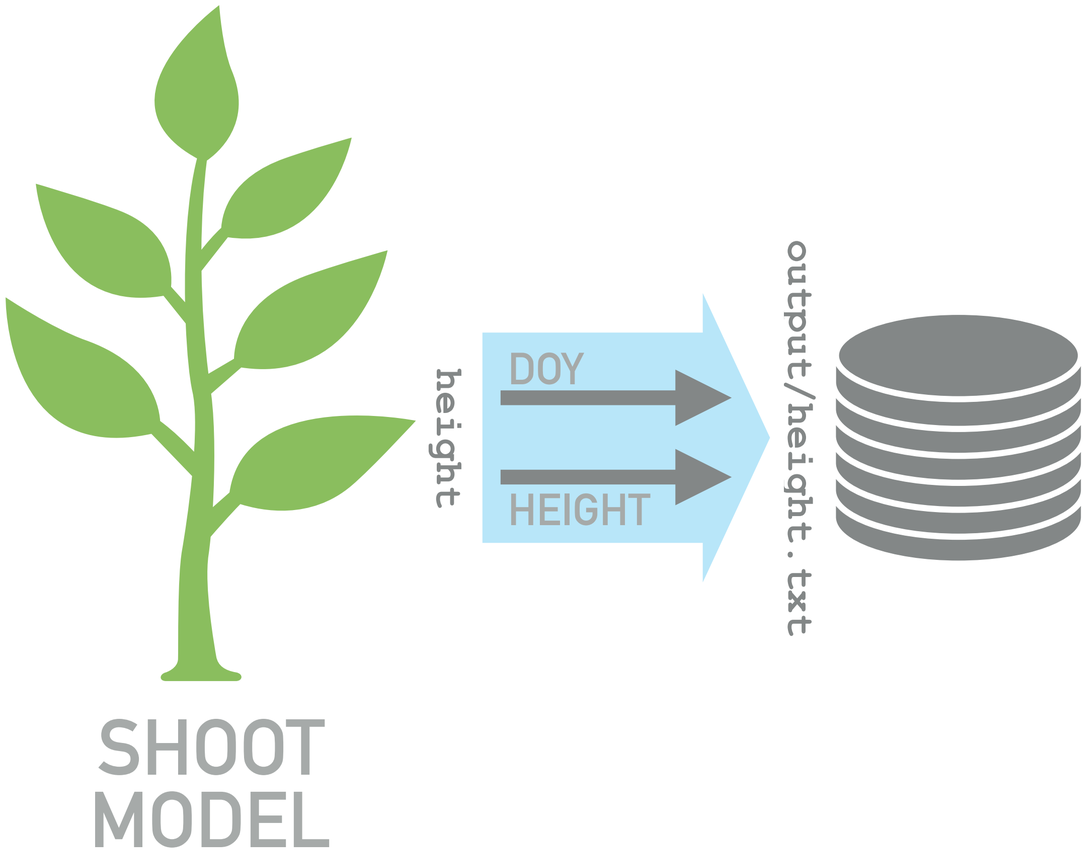
```
In [16]: tools.display_source_diff('yamls/shoot_v0.yml', 'yamls/shoot_v1.yml', number_lines=True)
```

```
file1: yamls/shoot_v0.yml
file2: yamls/shoot_v1.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v0.py, 0.0, 48.0, 6.0]
    ?

 4: +   args: [../models/shoot_v1.py, 0.0, 48.0, 6.0]
    ?

 5: +   outputs:
 6: +     - name: height
 7: +       default_file:
 8: +         name: ../output/height.txt
 9: +         filetype: table
```



```
In [16]: tools.display_source_diff('yamls/shoot_v0.yml', 'yamls/shoot_v1.yml', number_lines=True)
```

```
file1: yamls/shoot_v0.yml
file2: yamls/shoot_v1.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v0.py, 0.0, 48.0, 6.0]
    ?

 4: +   args: [../models/shoot_v1.py, 0.0, 48.0, 6.0]
    ?

 5: +   outputs:
 6: +     - name: height
 7: +       default_file:
 8: +         name: ../output/height.txt
 9: +         filetype: table
```



Declare one output with a default file that is only used if no other connection connects to it.

In [17]:
```
run(['yamls/shoot_v1.yml'], production_run=True)
```

In [17]:
```
run(['yamls/shoot_v1.yml'], production_run=True)
```

```
INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v1.py 0.0 48.0 6.0
INFO:91854:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:91854:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:91854:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:91854:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:91854:runner.run[374]:YggRunner(runner):         load drivers      0.006655
INFO:91854:runner.run[374]:YggRunner(runner):        start drivers      0.060119
INFO:91854:runner.run[374]:YggRunner(runner):           run models      5.849015
INFO:91854:runner.run[374]:YggRunner(runner):              at exit      0.003885
INFO:91854:runner.run[376]:YggRunner(runner): ========================================
INFO:91854:runner.run[377]:YggRunner(runner):                Total      5.919675
```

In [17]:
```
run(['yamls/shoot_v1.yml'], production_run=True)
```

```
INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v1.py 0.0 48.0 6.0
INFO:91854:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:91854:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:91854:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:91854:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:91854:runner.run[374]:YggRunner(runner):         load drivers      0.006655
INFO:91854:runner.run[374]:YggRunner(runner):        start drivers      0.060119
INFO:91854:runner.run[374]:YggRunner(runner):            run models      5.849015
INFO:91854:runner.run[374]:YggRunner(runner):               at exit      0.003885
INFO:91854:runner.run[376]:YggRunner(runner): ========================================
INFO:91854:runner.run[377]:YggRunner(runner):                 Total      5.919675
```

In [18]:
```
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

Out[18]:

```
In [17]: run(['yamls/shoot_v1.yml'], production_run=True)
```

```
INFO:91854:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v1.py 0.0 48.0 6.0
INFO:91854:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:91854:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:91854:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:91854:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:91854:runner.run[374]:YggRunner(runner):         load drivers      0.006655
INFO:91854:runner.run[374]:YggRunner(runner):        start drivers      0.060119
INFO:91854:runner.run[374]:YggRunner(runner):            run models      5.849015
INFO:91854:runner.run[374]:YggRunner(runner):               at exit      0.003885
INFO:91854:runner.run[376]:YggRunner(runner): ========================================
INFO:91854:runner.run[377]:YggRunner(runner):                Total      5.919675
```

```
In [18]: mesh = trimesh.load_mesh('output/mesh_008.obj')
         mesh.show()
Out[18]:
```

```
In [19]: tools.display_source('output/height.txt')
```

```
file: output/height.txt
========================
# hr     m
# %g     %g
0        77.2603
6        80.6941
12       84.4399
18       88.5415
24       93.0513
30       98.0321
36       103.561
42       109.73
48       116.656
```

Test your knowledge #2

1. Add interface calls to the `models/co2.py` model to receive height data from the `input/height.txt` file and send output to the `output/co2.txt` file in the pattern defined by the `yamls/co2.yml` YAML and run it in the cell below
2. Create a YAML for the model with input/output interface calls located at `models/humidity.py` that takes inputs from `input/height.txt` and `input/co2.txt`, sends output to the `output/humidity.txt` file, and run it in the cell below.
3. Try adding interface calls to the `models/reflectance.py` model and writing a YAML to receive input from the `input/temp.txt` file and send output to the `output/reflect.txt` file and run it in the cell below.

TEST YOUR KNOWLEDGE (15 MIN)

Test your knowledge #2

1. Add interface calls to the `models/co2.py` model to receive height data from the `input/height.txt` file and send output to the `output/co2.txt` file in the pattern defined by the `yamls/co2.yml` YAML and run it in the cell below

In [2]:
```
from yggdrasil import tools
from yggdrasil.runner import run

# Part 1: Add interface calls
tools.display_source_diff('models/co2.py', 'solutions/tyk2/models/co2.py', number_lines=True)
run(['solutions/tyk2/yamls/co2.yml'], production_run=True)

# Part 2: Write a YAML
tools.display_source('solutions/tyk2/yamls/humidity.yml', number_lines=True)
run(['solutions/tyk2/yamls/humidity.yml'], production_run=True)

# Part 3: Add interface calls and write a YAML
tools.display_source_diff('models/reflectance.py', 'solutions/tyk2/models/reflectance.py', number_lines=True)
tools.display_source('solutions/tyk2/yamls/reflectance.yml', number_lines=True)
run(['solutions/tyk2/yamls/reflectance.yml'], production_run=True)
```

Test your knowledge #2

1. Add interface calls to the `models/co2.py` model to receive height data from the `input/height.txt` file and send output to the `output/co2.txt` file in the pattern defined by the `yamls/co2.yml` YAML and run it in the cell below

```
file1: models/co2.py
file2: solutions/tyk2/models/co2.py
===================================
 1:   import argparse
 2:   import numpy as np
    -
 3: + import os
 4:
 5:   def calculate_concentration(doy, dist, height, offset=60.0):
 6:       r"""Function that calculates the concentration of CO2.
 7:
 8:       Args:
 9:           doy (float): Day of year.
10:           dist (float): Distance from the plant in cm.
11:           height (float): Distance from the ground in cm.
12:           offset (float, optional): Offset in the year in days. Defaults to 60.
13:
14:       Returns:
15:           float: CO2 concentration in cm^-3
16:
17:       """
18:       return np.sin(2.0 * np.pi * (doy + offset) / 365) / (dist * dist * height)
```

Test your knowledge #2

1. Add interface calls to the `models/co2.py` model to receive height data from the `input/height.txt` file and send output to the `output/co2.txt` file in the pattern defined by the `yamls/co2.yml` YAML and run it in the cell below

```
+ # Check if model is running as a part of an yggdrasil integration
+ with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
+
+ # If the model is running as part of an yggdrasil integration, import
+ # the relevant yggdrasil routines and use the interface routine to
+ # complete the connection defined in the YAML
+ if with_yggdrasil:
+     from yggdrasil import units
+     from yggdrasil.languages.Python.YggInterface import YggInput, YggOutput
+     height_in = YggInput('height')
+     conc_out = YggOutput('co2')
+
+     # Add units to parameters
+     dist = units.add_units(dist, 'cm')
+     offset = units.add_units(offset, 'days')
+
+     # Loop over input
+     while True:
+         # Receive height
+         flag, height_data = height_in.recv()
+         if not flag:
+             print('End of height input')
+             break
+         [doy, height] = height_data[:]
```

Test your knowledge #2

1. Add interface calls to the `models/co2.py` model to receive height data from the `input/height.txt` file and send output to the `output/co2.txt` file in the pattern defined by the `yamls/co2.yml` YAML and run it in the cell below

```
62: +         # Send output
63: +         flag = conc_out.send(doy, conc)
64: +         if not flag:
65: +             raise Exception("Error sending concentration to output")
66: +
67: + else:
68: +
69: +     # Compute concentration
70: +     conc = calculate_concentration(doy, dist, height)
    - print('Concentration', conc)
71: +     print('Concentration', conc)
    ? ++++
```

Test your knowledge #2

1. Add interface calls to the `models/co2.py` model to receive height data from the `input/height.txt` file and send output to the `output/co2.txt` file in the pattern defined by the `yamls/co2.yml` YAML and run it in the cell below

```
INFO:66078:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk2/models/co2.py 5.0 23.0 126.0
Concentration -0.00017474583950314156 1/(cm**2*m)
Concentration -0.00011830989676120096 1/(cm**2*m)
Concentration -6.503028644669879e-05 1/(cm**2*m)
Concentration -1.5550498522274506e-05 1/(cm**2*m)
Concentration 2.9576127062090165e-05 1/(cm**2*m)
Concentration 6.989267563644088e-05 1/(cm**2*m)
Concentration 0.00010504269272568111 1/(cm**2*m)
Concentration 0.00013477610239339267 1/(cm**2*m)
Concentration 0.0001589460536630261 1/(cm**2*m)
End of height input
INFO:66078:runner.waitModels[553]:YggRunner(runner): co2 finished running.
INFO:66078:runner.waitModels[559]:YggRunner(runner): co2 finished exiting.
INFO:66078:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:66078:runner.run[374]:YggRunner(runner):                  init      0.000023
INFO:66078:runner.run[374]:YggRunner(runner):          load drivers      0.515482
INFO:66078:runner.run[374]:YggRunner(runner):         start drivers      0.165137
INFO:66078:runner.run[374]:YggRunner(runner):            run models      14.999156
INFO:66078:runner.run[374]:YggRunner(runner):               at exit      0.011509
INFO:66078:runner.run[376]:YggRunner(runner): ========================================
INFO:66078:runner.run[377]:YggRunner(runner):                 Total      15.691307
```

Test your knowledge #2

2. Create a YAML for the model with input/output interface calls located at `models/humidity.py` that takes inputs from `input/height.txt` and `input/co2.txt`, sends output to the `output/humidity.txt` file, and run it in the cell below.

```
file: solutions/tyk2/yamls/humidity.yml
========================================
 1: model:
 2:   name: humidity
 3:   language: python
 4:   args: ../models/humidity.py
 5:   inputs:
 6:     - name: height
 7:       default_file:
 8:         name: ../input/height.txt
 9:         filetype: table
10:     - name: co2
11:       default_file:
12:         name: ../input/co2.txt
13:         filetype: table
14:   outputs:
15:     - name: humidity
16:       default_file:
17:         name: ../output/humidity.txt
18:         filetype: table
19:         field_names: [doy,height,co2,humidity]
```

Test your knowledge #2

2. Create a YAML for the model with input/output interface calls located at `models/humidity.py` that takes inputs from `input/height.txt` and `input/co2.txt`, sends output to the `output/humidity.txt` file, and run it in the cell below.

```
INFO:66078:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk2/models/humidity.py
Humidity 0.0 hr 77.2603 m -0.000174746 1/(cm**2*m) -372.25868669129795 dimensionless
Humidity 6.0 hr 80.6941 m -0.00011831 1/(cm**2*m) -482.58123682403436 dimensionless
Humidity 12.0 hr 84.4399 m -6.50303e-05 1/(cm**2*m) -766.2072892988516 dimensionless
Humidity 18.0 hr 88.5415 m -1.55505e-05 1/(cm**2*m) -2779.073771287703 dimensionless
Humidity 24.0 hr 93.0513 m 2.95761e-05 1/(cm**2*m) 1258.7766079118592 dimensionless
Humidity 30.0 hr 98.0321 m 6.98927e-05 1/(cm**2*m) 455.49537215991813 dimensionless
Humidity 36.0 hr 103.561 m 0.000105043 1/(cm**2*m) 257.05179156367245 dimensionless
Humidity 42.0 hr 109.73 m 0.000134776 1/(cm**2*m) 168.39743098226793 dimensionless
Humidity 48.0 hr 116.656 m 0.000158946 1/(cm**2*m) 118.82093626705479 dimensionless
No more inputs
INFO:66078:runner.waitModels[553]:YggRunner(runner): humidity finished running.
INFO:66078:runner.waitModels[559]:YggRunner(runner): humidity finished exiting.
INFO:66078:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:66078:runner.run[374]:YggRunner(runner):                 init      0.000002
INFO:66078:runner.run[374]:YggRunner(runner):         load drivers      0.039554
INFO:66078:runner.run[374]:YggRunner(runner):        start drivers      0.386276
INFO:66078:runner.run[374]:YggRunner(runner):            run models      14.085093
INFO:66078:runner.run[374]:YggRunner(runner):               at exit      0.029636
INFO:66078:runner.run[376]:YggRunner(runner): ========================================
INFO:66078:runner.run[377]:YggRunner(runner):                Total      14.540561
```

Test your knowledge #2

3. Try adding interface calls to the `models/reflectance.py` model and writing a YAML to receive input from the `input/temp.txt` file and send output to the `output/reflect.txt` file and run it in the cell below.

```
...
13: + # If the model is running as part of an yggdrasil integration, import
14: + # the relevant yggdrasil routines and use the interface routine to
15: + # complete the connection defined in the YAML
16: + if with_yggdrasil:
17: +     from yggdrasil import units
18: +     from yggdrasil.languages.Python.YggInterface import YggInput, YggOutput
19: +     temp_in = YggInput('temperature')
20: +     refl_out = YggOutput('reflectance')
21: +
22: +     # Receive the temperatures
23: +     flag, temp = temp_in.recv()
24: +     if not flag:
25: +         raise Exception("Error receiving temperature.")
26: +
27: +     # Ensure correct units
28: +     temp = units.convert_to(temp[0], 'K')  # list of one table column
29: +     temp = units.get_data(temp)
30:
31:

...
39: +     if with_yggdrasil:
40: +         flag = refl_out.send(temp[i], *R)  # expand array elements
41: +         if not flag:
42: +             raise Exception("Error sending reflectance")
```

Test your knowledge #2

3. Try adding interface calls to the `models/reflectance.py` model and writing a YAML to receive input from the `input/temp.txt` file and send output to the `output/reflect.txt` file and run it in the cell below.

```
file: solutions/tyk2/yamls/reflectance.yml
==========================================
 1: model:
 2:   name: reflectance
 3:   language: python
 4:   args: [../models/reflectance.py, 5.0, 23.0, 126.0]
 5:   inputs:
 6:     - name: temperature
 7:       default_file:
 8:         name: ../input/temp.txt
 9:         filetype: table
10:         as_array: true
11:   outputs:
12:     - name: reflectance
13:       default_file:
14:         name: ../output/reflectance.txt
15:         filetype: table
```

Test your knowledge #2

3. Try adding interface calls to the `models/reflectance.py` model and writing a YAML to receive input from the `input/temp.txt` file and send output to the `output/reflect.txt` file and run it in the cell below.

```
INFO:66078:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk2/models/reflectance.py 5.0 23.0 126.0
Reflectance at 430.183 K: [0.25001185 0.255132   0.27101694 0.29929321 0.34284125 0.40601749
 0.49487502 0.61723752 0.78238144 1.        ]
Reflectance at 435.352 K: [0.28099013 0.28628736 0.30266084 0.33159306 0.37566956 0.43871488
 0.5258929  0.64364631 0.79928558 1.        ]
Reflectance at 440.79 K: [0.31138953 0.31680738 0.33349893 0.36280239 0.40701953 0.46948672
 0.5545903  0.66762184 0.81432816 1.        ]
Reflectance at 446.523 K: [0.34113075 0.34662053 0.36348437 0.39291992 0.436961   0.49850023
 0.5812456  0.68952543 0.82783217 1.        ]
Reflectance at 452.583 K: [0.37017845 0.37569824 0.39260998 0.42197684 0.46558275 0.52591956
 0.60610392 0.70965502 0.84005162 1.        ]
Reflectance at 459.005 K: [0.39851231 0.40402609 0.42087953 0.45000892 0.4929673  0.55188628
 0.62936731 0.7282479  0.85118311 1.        ]
Reflectance at 465.833 K: [0.42614334 0.43162009 0.44832444 0.47707409 0.51921015 0.57654142
 0.65122045 0.74550882 0.86138918 1.        ]
Reflectance at 473.116 K: [0.45308501 0.45849799 0.47497546 0.50322512 0.54439462 0.60000431
 0.67181572 0.76160308 0.87079848 1.        ]
Reflectance at 480.912 K: [0.47936135 0.48468741 0.50087112 0.52851878 0.56860219 0.62238463

 0.69128718 0.77667121 0.87951744 1.        ]
INFO:66078:runner.waitModels[553]:YggRunner(runner): reflectance finished running.
INFO:66078:runner.waitModels[559]:YggRunner(runner): reflectance finished exiting.
INFO:66078:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:66078:runner.run[374]:YggRunner(runner):                 init      0.000008
INFO:66078:runner.run[374]:YggRunner(runner):         load drivers      0.028545
INFO:66078:runner.run[374]:YggRunner(runner):        start drivers      0.169874
INFO:66078:runner.run[374]:YggRunner(runner):            run models      14.868390
INFO:66078:runner.run[374]:YggRunner(runner):               at exit      0.008931
INFO:66078:runner.run[376]:YggRunner(runner): ========================================
INFO:66078:runner.run[377]:YggRunner(runner):                Total      15.075748
```

NEW NOTEBOOK!

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

jupyter

Visit repo | Copy Binder link | Quit

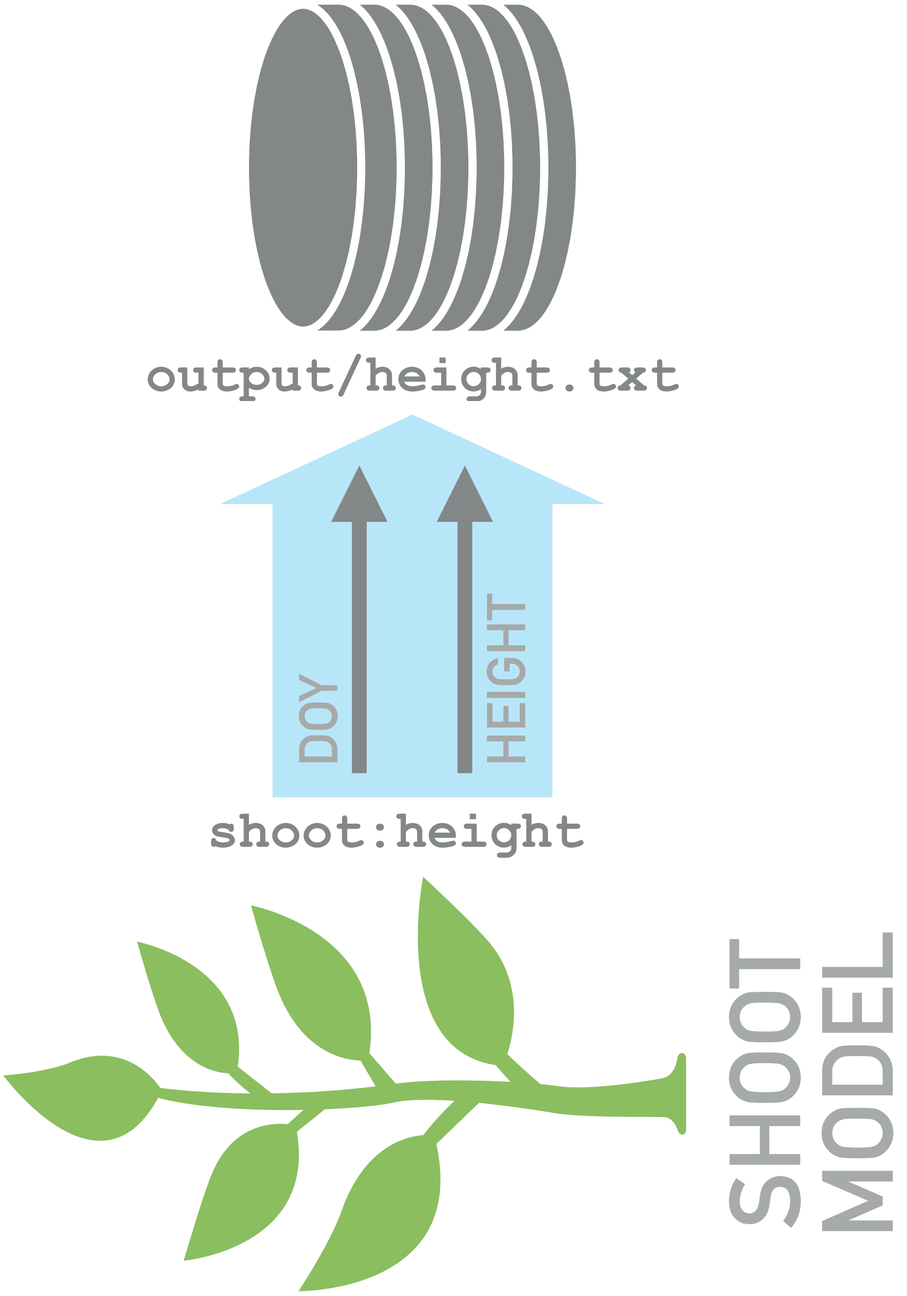
Files | Running | Clusters | Nbextensions

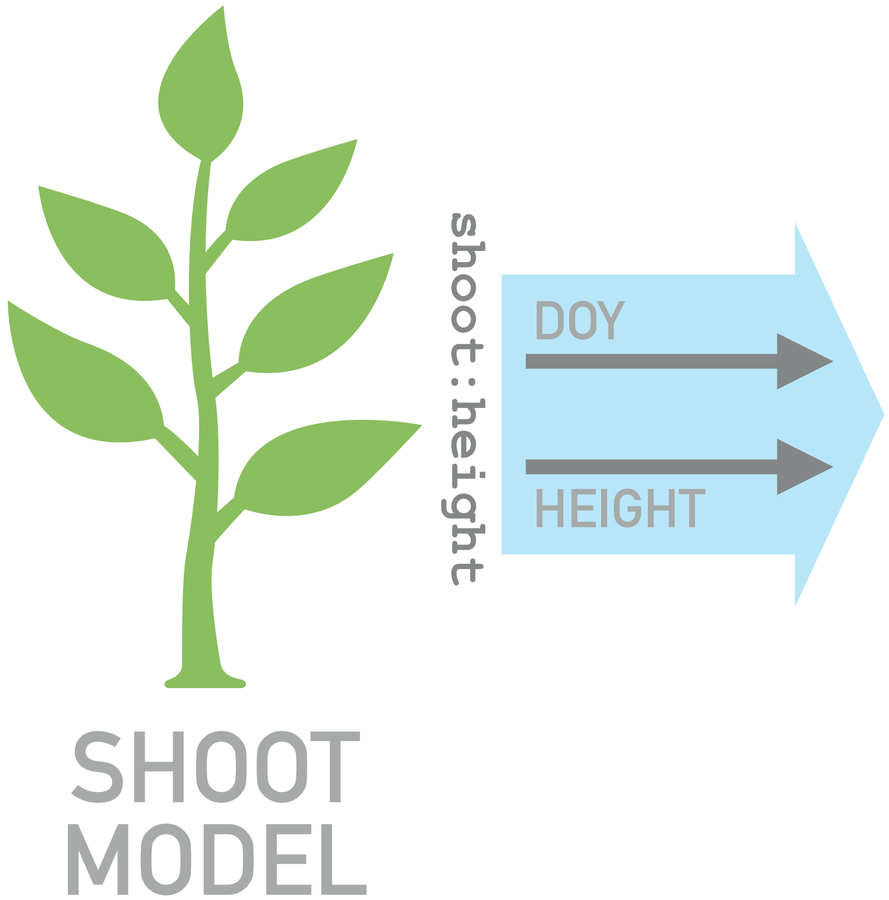
Select items to perform actions on them.

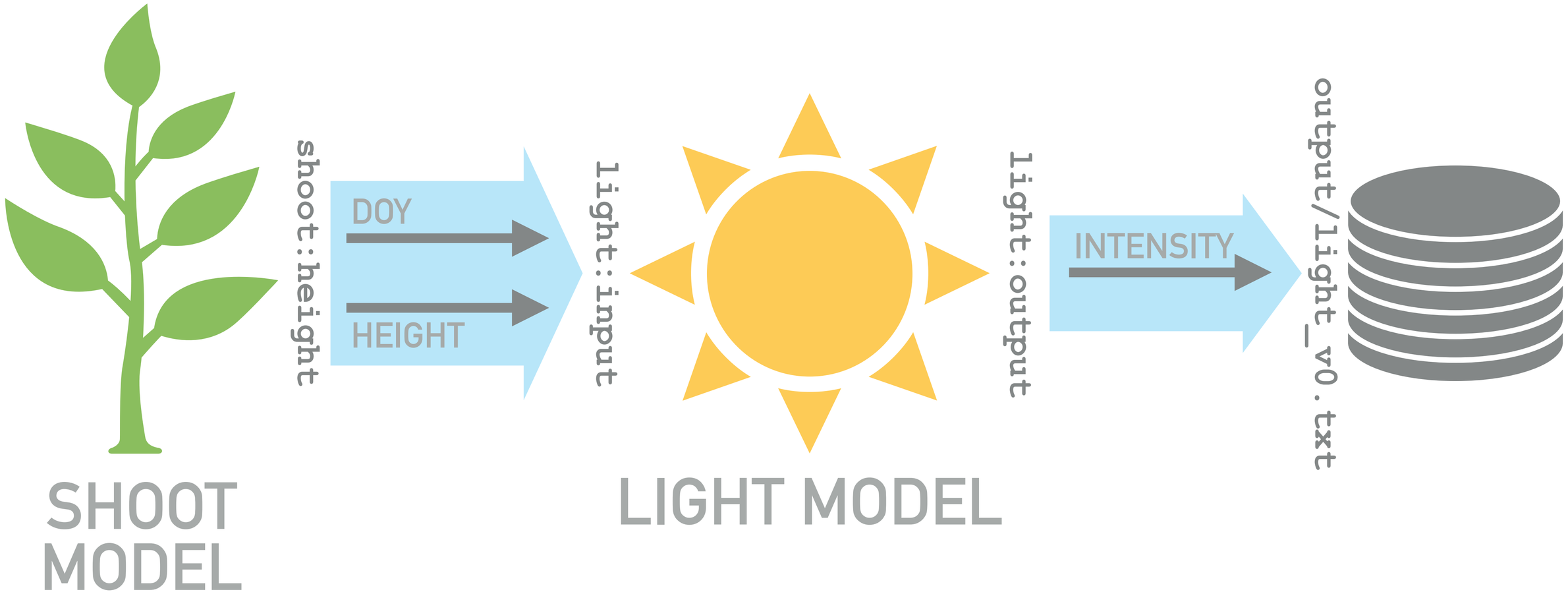
Upload | New

0 /	Name	Last Modified	File size
images		33 minutes ago	
input		33 minutes ago	
meshes		33 minutes ago	
models		33 minutes ago	
yamls		33 minutes ago	
00-intro.ipynb		33 minutes ago	457 kB
01-connections.ipynb		33 minutes ago	470 kB
02-timesync.ipynb		33 minutes ago	298 kB
03-misc.ipynb		33 minutes ago	3.56 kB

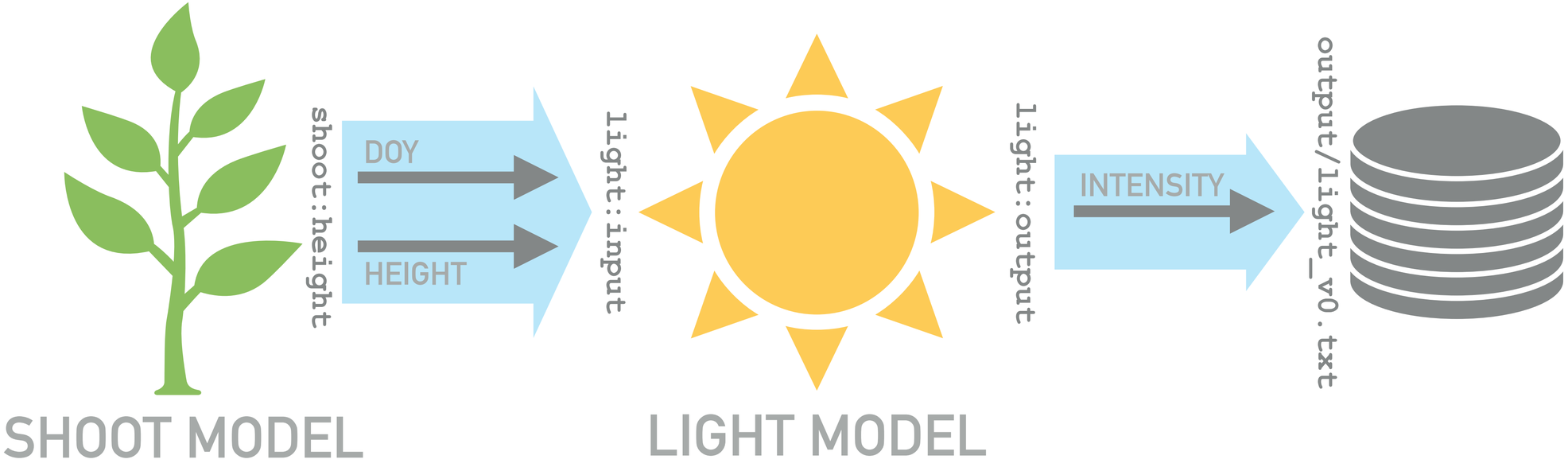
ONE WAY MODEL-TO-MODEL CONNECTION


output/height.txt
DOY
HEIGHT
shoot:height
SHOOT
MODEL


SHOOT
MODEL
shoot:height
DOY
HEIGHT

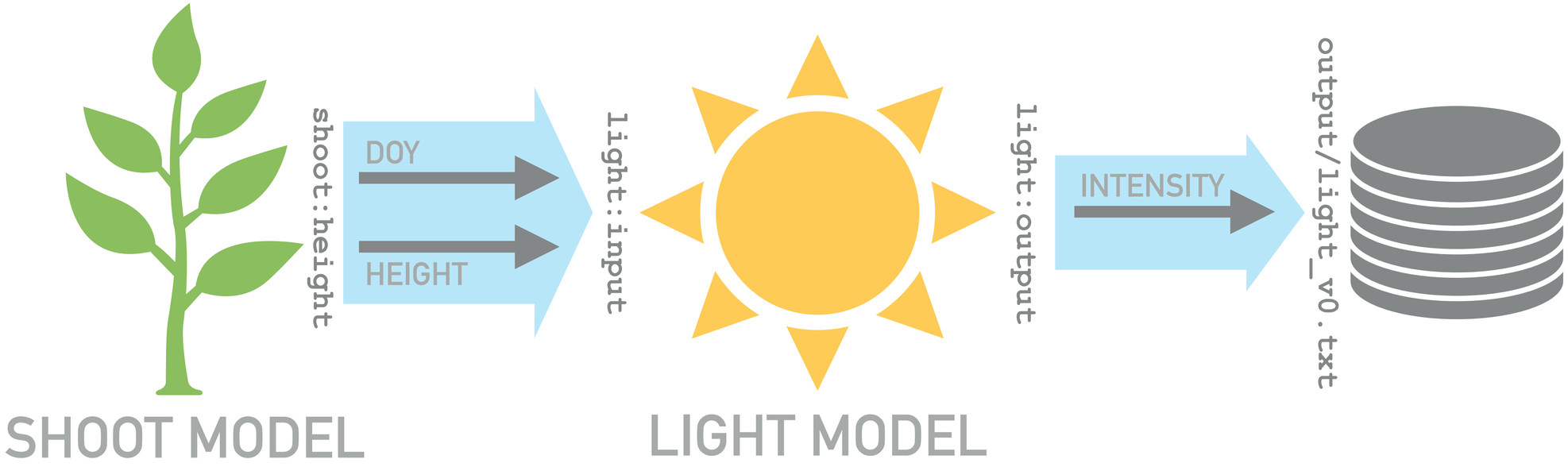

SHOOT MODEL
shoot:height
DOY
HEIGHT
light:input
LIGHT MODEL
light:output
INTENSITY
output/light_v0.txt

```
In [2]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_v1.yml', number_lines=True)
```



```
In [2]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_v1.yml', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_v1.yml
================================
 1:   connections:
    -   - input:
 2: +   - input: shoot:height
    ?                              +++++++++++++

    -       name: ../input/light_v0.txt
    -       filetype: table
 3:       output: light:input
 4:     - input: light:output
 5:       output:
 6:         name: ../output/light_v0.txt
 7:         filetype: table
 8:         field_names: [intensity]
```

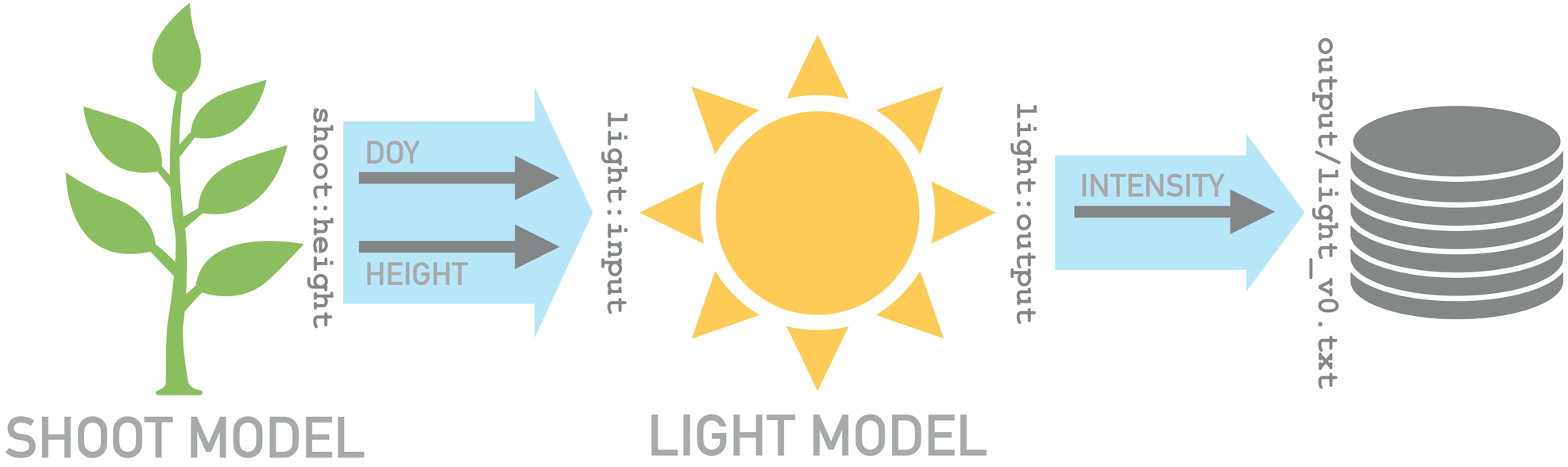


```
In [2]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_v1.yml', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_v1.yml
================================
 1:   connections:
    -   - input:
 2: +   - input: shoot:height
    ?                            +++++++++++++

    -       name: ../input/light_v0.txt
    -       filetype: table
 3:       output: light:input
 4:     - input: light:output
 5:       output:
 6:         name: ../output/light_v0.txt
 7:         filetype: table
 8:         field_names: [intensity]
```

Connect light input with shoot output channel "`height`"

The prefix "`shoot:`" indicates the height channel for the shoot model



In [3]:
```
run(['yamls/light_v0_python.yml', 'yamls/shoot_v1.yml', 'yamls/connections_v1.yml'], production_run=True)
```

In [3]:
```
run(['yamls/light_v0_python.yml', 'yamls/shoot_v1.yml', 'yamls/connections_v1.yml'], production_run=True)
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v1.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                  init      0.000000
INFO:93696:runner.run[374]:YggRunner(runner):         load drivers      0.282206
INFO:93696:runner.run[374]:YggRunner(runner):        start drivers      0.090545
INFO:93696:runner.run[374]:YggRunner(runner):           run models      6.751301
INFO:93696:runner.run[374]:YggRunner(runner):              at exit      0.021096
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                Total      7.145148
```

```
In [3]: run(['yamls/light_v0_python.yml', 'yamls/shoot_v1.yml', 'yamls/connections_v1.yml'], production_run=True)
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v1.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                  init      0.000000
INFO:93696:runner.run[374]:YggRunner(runner):         load drivers      0.282206
INFO:93696:runner.run[374]:YggRunner(runner):        start drivers      0.090545
INFO:93696:runner.run[374]:YggRunner(runner):           run models      6.751301
INFO:93696:runner.run[374]:YggRunner(runner):              at exit      0.021096
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                Total      7.145148
```

```
In [4]: mesh = trimesh.load_mesh('output/mesh_000.obj')
        mesh.show()
```

Out[4]:

```
In [3]: run(['yamls/light_v0_python.yml', 'yamls/shoot_v1.yml', 'yamls/connections_v1.yml'], production_run=True)
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v1.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                  init      0.000000
INFO:93696:runner.run[374]:YggRunner(runner):          load drivers      0.282206
INFO:93696:runner.run[374]:YggRunner(runner):         start drivers      0.090545
INFO:93696:runner.run[374]:YggRunner(runner):            run models      6.751301
INFO:93696:runner.run[374]:YggRunner(runner):               at exit      0.021096
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                 Total      7.145148
```

```
In [4]: mesh = trimesh.load_mesh('output/mesh_000.obj')
        mesh.show()
```

Out[4]:

```
In [5]: tools.display_source('output/light_v0.txt')
```

```
file: output/light_v0.txt
=========================
# intensity
# erg/(cm**2*s)
# %g
618082
648331
681333
717477
757224
801131
849874
904281
965376
```

Test your knowledge #3

1. Write a YAML to connect the `models/light_v0.py` model to the `models/weather.py` model from "Test your knowledge #1" and run the integration in the cell below.
2. Write a YAML to connect the `models/shoot_v1.py` model to the `models/co2.py` model from "Test your knowledge #2" and run the integration in the cell below.
3. Write a YAML to connect the `models/shoot_v1.py` model to both the `models/light_v0.py` model and the `models/co2.py` model and run the integration in the cell below.

Tip: Connections can have more than one input and/or output

TEST YOUR KNOWLEDGE (10 MIN)

Test your knowledge #3

1. Write a YAML to connect the `models/light_v0.py` model to the `models/weather.py` model from "Test your knowledge #1" and run the integration in the cell below.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run

# Part 1: light-to-weather
tools.display_source('solutions/tyk3/yamls/light_to_weather.yml', number_lines=True)
run(['solutions/tyk3/yamls/light_to_weather.yml', 'solutions/tyk3/yamls/light_v0_python.yml'],
    production_run=True)

# Part 2: shoot-to-co2
tools.display_source('solutions/tyk3/yamls/shoot_to_co2.yml', number_lines=True)
run(['solutions/tyk3/yamls/shoot_to_co2.yml', 'solutions/tyk3/yamls/shoot_v1.yml', 'solutions/tyk3/yamls/co2.yml'],
    production_run=True)

# Part 3: shoot-to-light&co2
tools.display_source('solutions/tyk3/yamls/shoot_to_light_and_co2.yml', number_lines=True)
run(['solutions/tyk3/yamls/shoot_to_light_and_co2.yml', 'solutions/tyk3/yamls/shoot_v1.yml',
     'solutions/tyk3/yamls/co2.yml', 'solutions/tyk3/yamls/light_v0_python.yml'], production_run=True)
```

Test your knowledge #3

1. Write a YAML to connect the `models/light_v0.py` model to the `models/weather.py` model from "Test your knowledge #1" and run the integration in the cell below.

```
file: solutions/tyk3/yamls/light_to_weather.yml
===============================================
model:
  name: weather
  language: python
  args: ../models/weather.py
  function: temp

connections:
  - input:
      name: ../input/light_v0.txt
      filetype: table
    output: light:input
  - input: light:output
    output: weather:input
  - input: weather:output
    output:
      name: ../output/temp.txt
      filetype: table
      field_names: [temp]
```

Test your knowledge #3

1. Write a YAML to connect the `models/light_v0.py` model to the `models/weather.py` model from "Test your knowledge #1" and run the integration in the cell below.

```
INFO:60837:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/ygg_weather.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/ygg_light_v0.py
End of input from temp_doy.
End of input from intensity.
INFO:60837:runner.waitModels[553]:YggRunner(runner): weather finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): weather finished exiting.
INFO:60837:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:60837:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:60837:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:60837:runner.run[374]:YggRunner(runner):         load drivers      0.469073
INFO:60837:runner.run[374]:YggRunner(runner):        start drivers      0.235939
INFO:60837:runner.run[374]:YggRunner(runner):           run models      12.208608
INFO:60837:runner.run[374]:YggRunner(runner):              at exit      0.080839
INFO:60837:runner.run[376]:YggRunner(runner): ========================================
INFO:60837:runner.run[377]:YggRunner(runner):                Total      12.994460
```

Test your knowledge #3

2. Write a YAML to connect the `models/shoot_v1.py` model to the `models/co2.py` model from "Test your knowledge #2" and run the integration in the cell below.

```
file: solutions/tyk3/yamls/shoot_to_co2.yml
============================================
connections:
  - input: shoot:height
    output: co2:height
```

Test your knowledge #3

2. Write a YAML to connect the `models/shoot_v1.py` model to the `models/co2.py` model from "Test your knowledge #2" and run the integration in the cell below.

```
INFO:60837:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/shoot_v1.py 0.0 48.0 6.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/co2.py 5.0 23.0 126.0
Concentration -0.00017474590966360063 1/(cm**2*m)
Concentration -0.00011830995729510536 1/(cm**2*m)
Concentration -6.503031956767388e-05 1/(cm**2*m)
Concentration -1.555050119909338e-05 1/(cm**2*m)
Concentration 2.9576139878533936e-05 1/(cm**2*m)
Concentration 6.989264979460234e-05 1/(cm**2*m)
Concentration 0.00010504317937369221 1/(cm**2*m)
Concentration 0.00013477608202178796 1/(cm**2*m)
Concentration 0.0001589453863753824 1/(cm**2*m)
End of height input
INFO:60837:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:60837:runner.waitModels[553]:YggRunner(runner): co2 finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): co2 finished exiting.
INFO:60837:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:60837:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:60837:runner.run[374]:YggRunner(runner):         load drivers      0.030084
INFO:60837:runner.run[374]:YggRunner(runner):        start drivers      0.128300
INFO:60837:runner.run[374]:YggRunner(runner):           run models      10.649023
INFO:60837:runner.run[374]:YggRunner(runner):              at exit      0.010026
INFO:60837:runner.run[376]:YggRunner(runner): ========================================
INFO:60837:runner.run[377]:YggRunner(runner):                Total      10.817434
```

Test your knowledge #3

3. Write a YAML to connect the `models/shoot_v1.py` model to both the `models/light_v0.py` model and the `models/co2.py` model and run the integration in the cell below.

```
file: solutions/tyk3/yamls/shoot_to_light_and_co2.yml
=====================================================
connections:
  - input: shoot:height
    outputs:
      - light:input
      - co2:height
  - input: light:output
    output:
      name: ../output/light_v0.txt
      filetype: table
      field_names: [intensity]
```

Test your knowledge #3

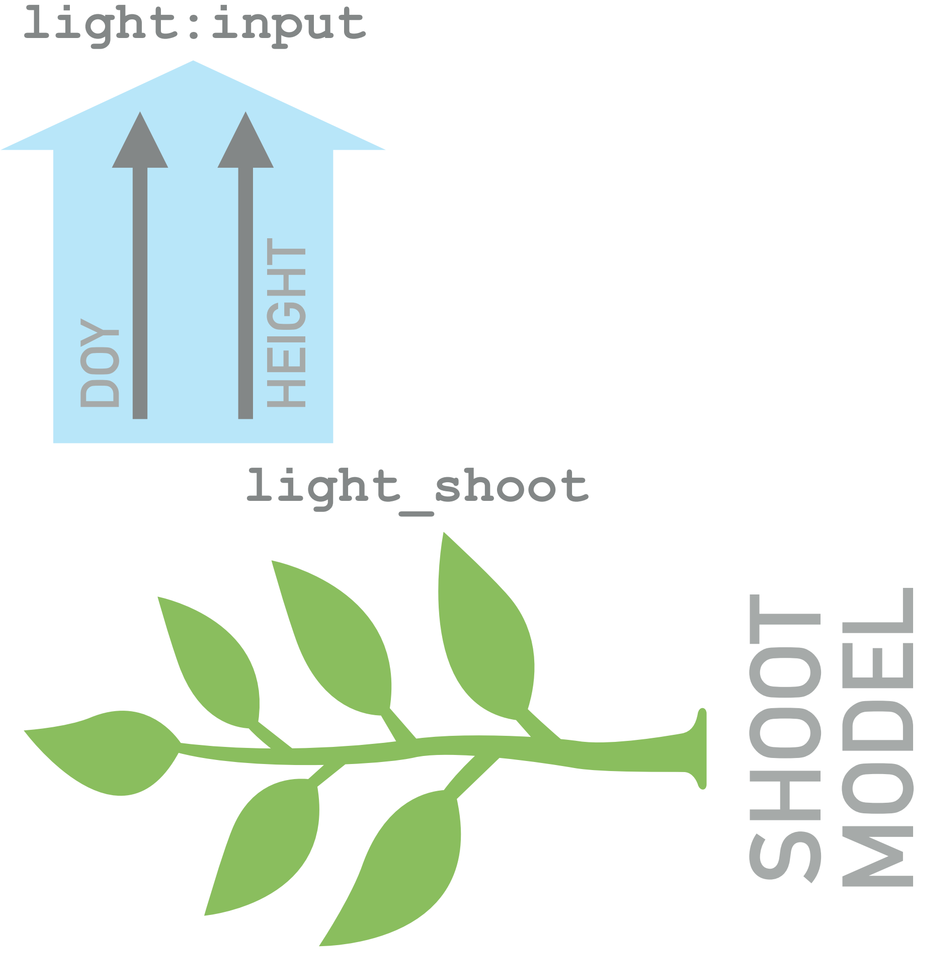
3. Write a YAML to connect the `models/shoot_v1.py` model to both the `models/light_v0.py` model and the `models/co2.py` model and run the integration in the cell below.

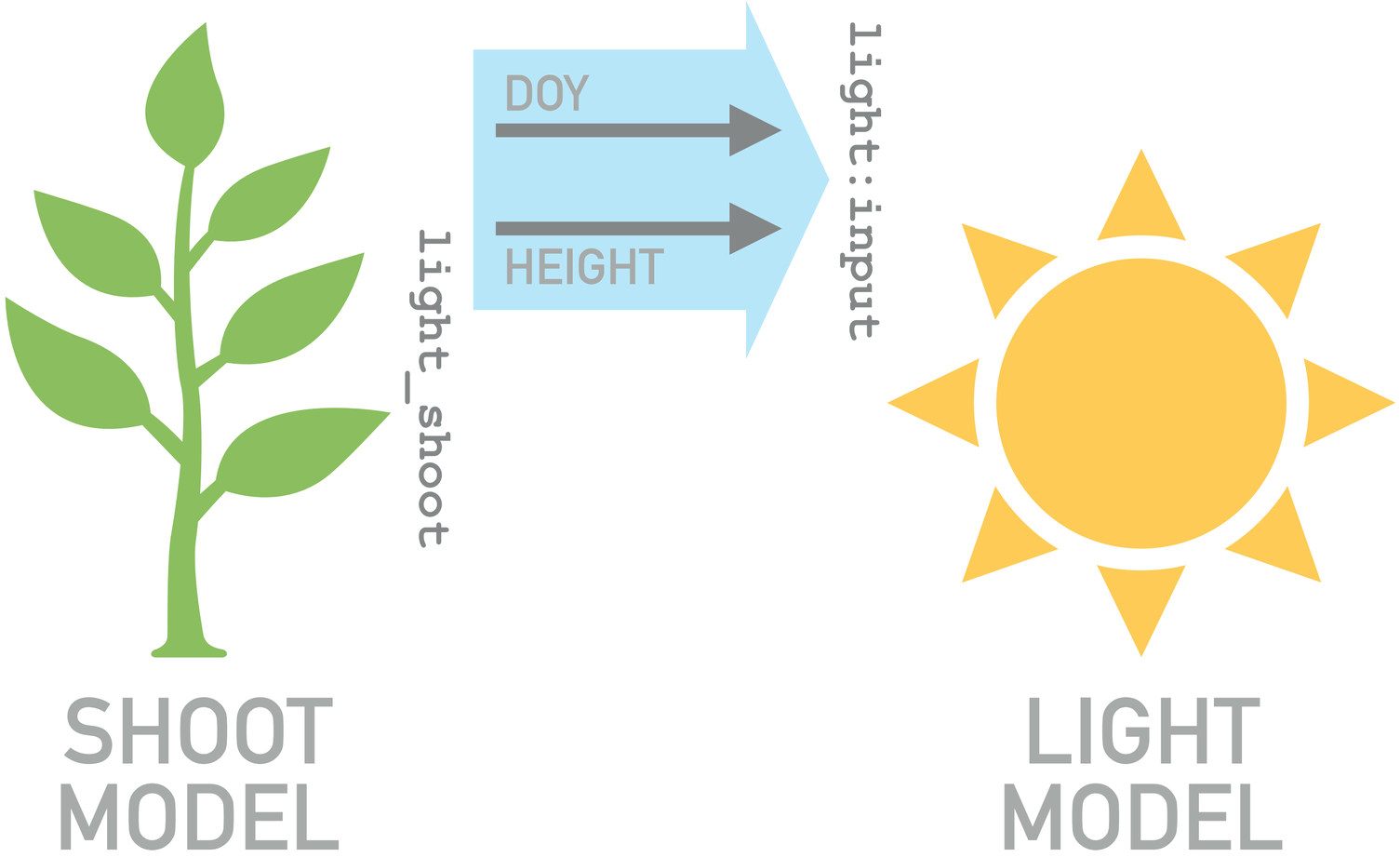
```
INFO:60837:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/shoot_v1.py 0.0 48.0 6.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/co2.py 5.0 23.0 126.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk3/models/ygg_light_v0.py
End of input from temp_doy.
Concentration -0.00017474590966360063 1/(cm**2*m)
Concentration -0.00011830995729510536 1/(cm**2*m)
Concentration -6.503031956767388e-05 1/(cm**2*m)
Concentration -1.555050119909338e-05 1/(cm**2*m)
Concentration 2.9576139878533936e-05 1/(cm**2*m)
Concentration 6.989264979460234e-05 1/(cm**2*m)
End of height input
INFO:60837:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:60837:runner.waitModels[553]:YggRunner(runner): co2 finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): co2 finished exiting.
INFO:60837:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:60837:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:60837:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:60837:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:60837:runner.run[374]:YggRunner(runner):         load drivers      0.056683
INFO:60837:runner.run[374]:YggRunner(runner):        start drivers      0.230479
INFO:60837:runner.run[374]:YggRunner(runner):           run models      22.343147
INFO:60837:runner.run[374]:YggRunner(runner):              at exit      0.023549
INFO:60837:runner.run[376]:YggRunner(runner): ========================================
INFO:60837:runner.run[377]:YggRunner(runner):                Total      22.653859
```

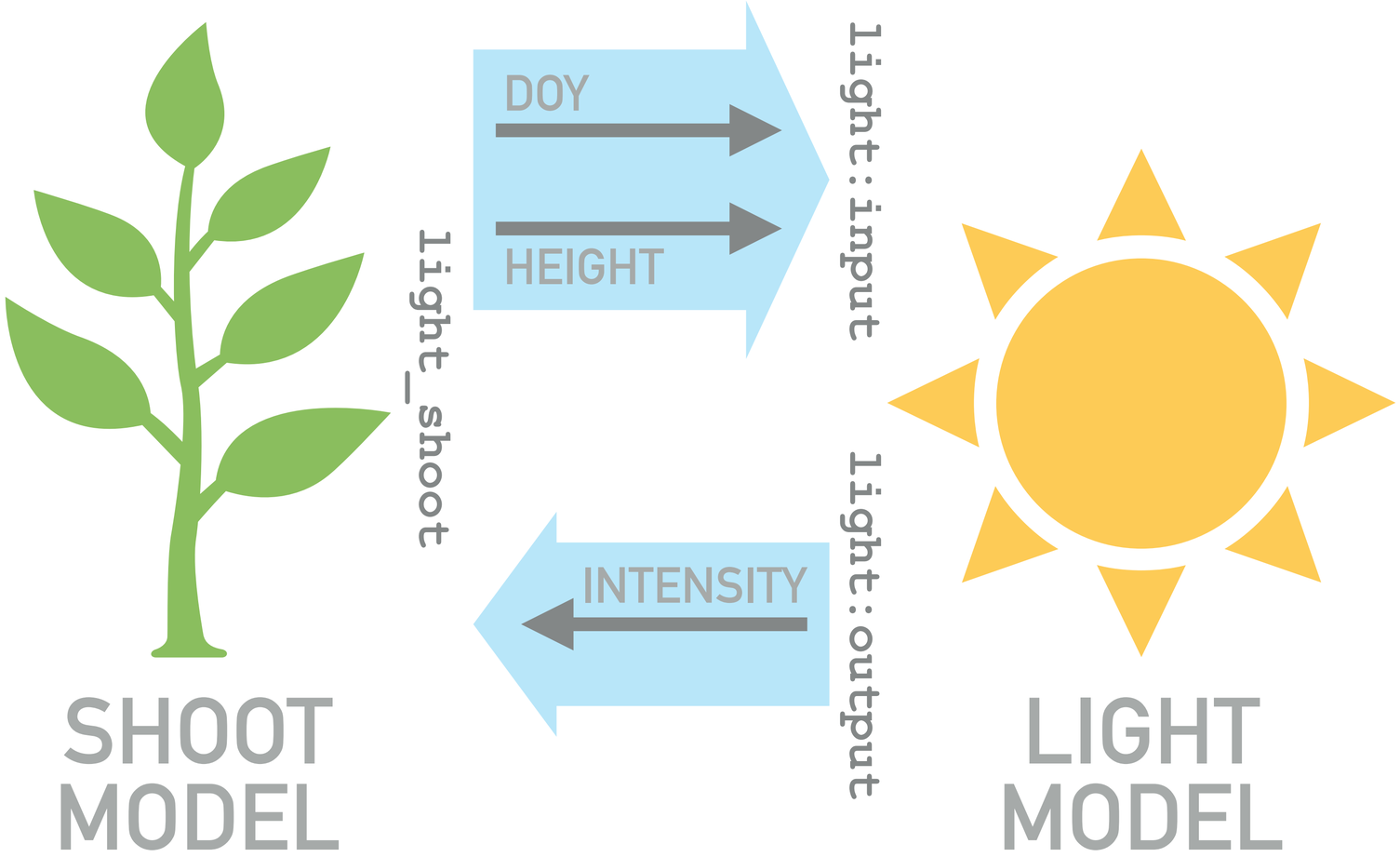
REMOTE PROCEDURE CALL (RPC)

light_shoot

SHOOT
MODEL


light:input
DOY
HEIGHT
light_shoot
SHOOT
MODEL


DOY
HEIGHT
light_shoot
light:input
SHOOT MODEL
LIGHT MODEL


DOY
HEIGHT
light:input
light_shoot
INTENSITY
light:output
SHOOT MODEL
LIGHT MODEL

In [6]:
```
tools.display_source_diff('models/shoot_v1.py', 'models/shoot_v2.py', number_lines=True)
```

```
In [6]: tools.display_source_diff('models/shoot_v1.py', 'models/shoot_v2.py', number_lines=True)
```

```
file1: models/shoot_v1.py
file2: models/shoot_v2.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse

...

28:   # If the model is running as part of an yggdrasil integration, import
29:   # the relevant yggdrasil routines and use the interface routine to
30:   # complete the connection defined in the YAML
31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggOutput
    ?
^^ ---

33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient
    ?
^^^^^^^^

    -     height_out = YggOutput('height')
    ?     ^^     ^^^                      ^^^ ^                            ^^

34: +     light_rpc = YggRpcClient('light_shoot')
    ?     ^     ^^^                      ^ ^^^^^^                           ^   ++++++

35:
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39:       # If running as part an yggdrasil integration, send the time and
40:       # maximum height of the mesh to the height channel with units
41:       if with_yggdrasil:
    -         flag = height_out.send(
    ?                              ^^     ^^^                 ^^^^

42: +         flag, intensity = light_rpc.call(
    ?             ++++++++++                 ^     ^^^                 ^^^^

43:               [units.add_units(t, 'hrs'),
44:                units.add_units(max(mesh.vertices[:, 2]), 'm')])

...

    -     # Compute the scale factor
47: +
48: +         # Compute the scale factor using intensity, stripping units
49: +         # of the result to allow use with trimesh
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++

51: +         scale = units.get_data(
52: +             units.add_units(mass, 'g') * intensity /
53: +             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
54: +
55: +     else:
56: +         # Compute the scale factor
57: +         # (pretend this is a biologically complex calculation)
    -     scale = mass / 4.5e4
58: +         scale = mass / 4.5e4
    ? ++++
```

```
In [6]: tools.display_source_diff('models/shoot_v1.py', 'models/shoot_v2.py', number_lines=True)
```

```
file1: models/shoot_v1.py
file2: models/shoot_v2.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse

...

28:   # If the model is running as part of an yggdrasil integration, import
29:   # the relevant yggdrasil routines and use the interface routine to
30:   # complete the connection defined in the YAML
31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggOutput
    ?
^^ ---

33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient
    ?
^^^^^^^^

    -     height_out = YggOutput('height')
    ?     ^^     ^^^                     ^^^ ^                            ^^

34: +     light_rpc = YggRpcClient('light_shoot')
    ?     ^      ^^^                     ^ ^^^^^^                         ^   ++++++

35:
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39:       # If running as part an yggdrasil integration, send the time and
40:       # maximum height of the mesh to the height channel with units
41:       if with_yggdrasil:
    -         flag = height_out.send(
    ?                              ^^     ^^^                 ^^^^

42: +         flag, intensity = light_rpc.call(
    ?             ++++++++++                 ^     ^^^                 ^^^^

43:               [units.add_units(t, 'hrs'),
44:                units.add_units(max(mesh.vertices[:, 2]), 'm')])

...

    -     # Compute the scale factor
47: +
48: +         # Compute the scale factor using intensity, stripping units
49: +         # of the result to allow use with trimesh
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++

51: +         scale = units.get_data(
52: +             units.add_units(mass, 'g') * intensity /
53: +             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
54: +
55: +     else:
56: +         # Compute the scale factor
57: +         # (pretend this is a biologically complex calculation)
    -     scale = mass / 4.5e4
58: +         scale = mass / 4.5e4
    ? ++++
```

Replace the interface function

```
In [6]: tools.display_source_diff('models/shoot_v1.py', 'models/shoot_v2.py', number_lines=True)
```

```
file1: models/shoot_v1.py
file2: models/shoot_v2.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse

...

28:   # If the model is running as part of an yggdrasil integration, import
29:   # the relevant yggdrasil routines and use the interface routine to
30:   # complete the connection defined in the YAML
31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggOutput
    ?
^^ ---

33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient
    ?
^^^^^^^^

    -     height_out = YggOutput('height')
    ?     ^^     ^^^                     ^^^ ^                            ^^

34: +     light_rpc = YggRpcClient('light_shoot')
    ?     ^      ^^^                     ^ ^^^^^^                          ^   ++++++

35:
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39:       # If running as part an yggdrasil integration, send the time and
40:       # maximum height of the mesh to the height channel with units
41:       if with_yggdrasil:
    -         flag = height_out.send(
    ?                              ^^     ^^^                 ^^^^

42: +         flag, intensity = light_rpc.call(
    ?             ++++++++++                  ^     ^^^                 ^^^^
43:               [units.add_units(t, 'hrs'),
44:                units.add_units(max(mesh.vertices[:, 2]), 'm')])

...
```

```
    -     # Compute the scale factor
47: +
48: +         # Compute the scale factor using intensity, stripping units
49: +         # of the result to allow use with trimesh
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++

51: +         scale = units.get_data(
52: +             units.add_units(mass, 'g') * intensity /
53: +             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
54: +
55: +     else:
56: +         # Compute the scale factor
57: +         # (pretend this is a biologically complex calculation)
    -     scale = mass / 4.5e4
58: +         scale = mass / 4.5e4
    ? ++++
```

Replace the send with a call

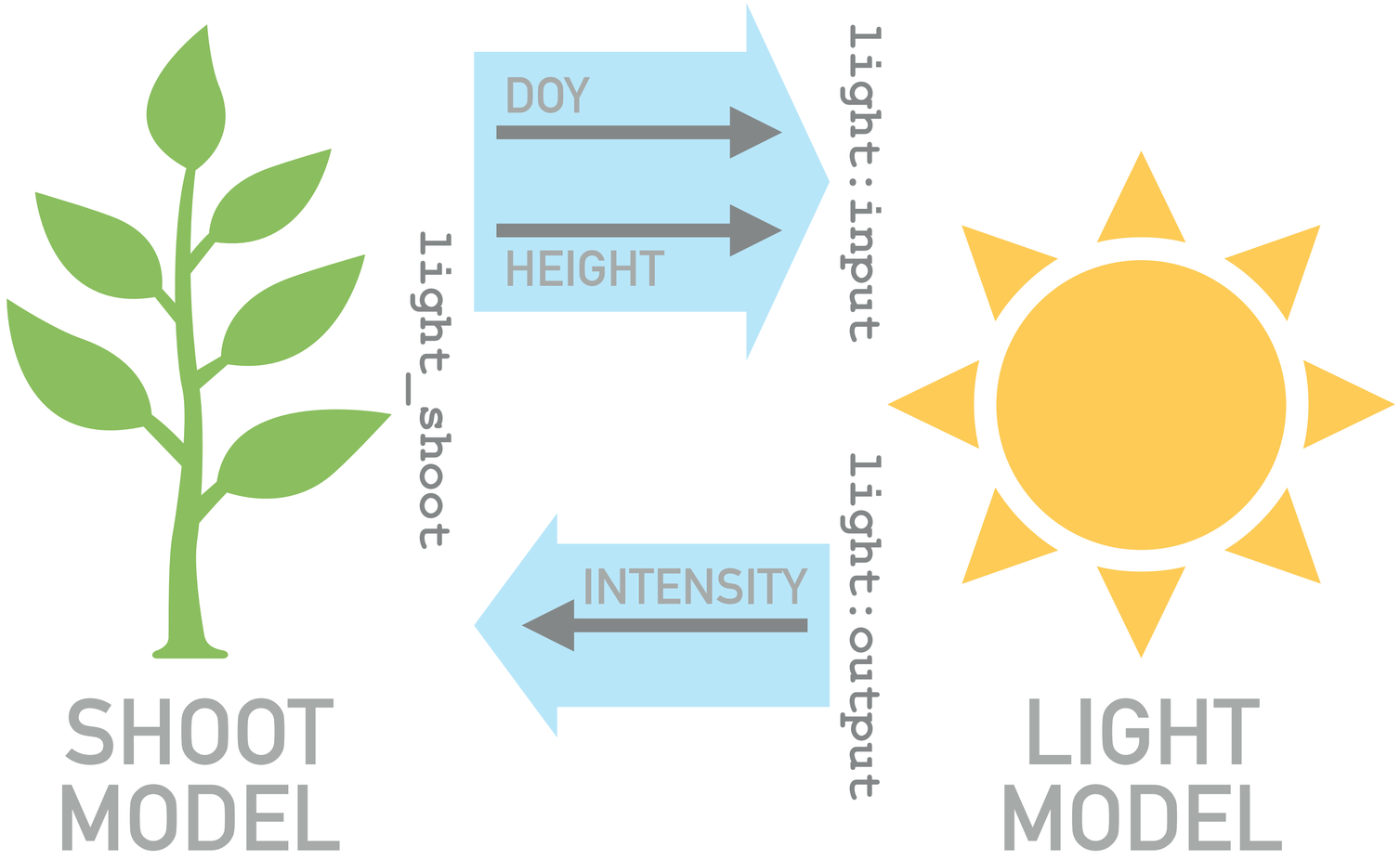
```
In [6]: tools.display_source_diff('models/shoot_v1.py', 'models/shoot_v2.py', number_lines=True)
```

```
file1: models/shoot_v1.py
file2: models/shoot_v2.py
=========================
 1:   import os
 2:   import trimesh
 3:   import argparse

...

28:   # If the model is running as part of an yggdrasil integration, import
29:   # the relevant yggdrasil routines and use the interface routine to
30:   # complete the connection defined in the YAML
31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggOutput
    ?
^^ ---

33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient
    ?
^^^^^^^^

    -     height_out = YggOutput('height')
    ?     ^^     ^^^                     ^^^ ^                            ^^

34: +     light_rpc = YggRpcClient('light_shoot')
    ?     ^      ^^^                     ^ ^^^^^^                          ^   ++++++

35:
36:   # Continue simulation until time limit is reached
37:   while t <= tmax:
38:
39:       # If running as part an yggdrasil integration, send the time and
40:       # maximum height of the mesh to the height channel with units
41:       if with_yggdrasil:
    -         flag = height_out.send(
    ?                              ^^     ^^^                 ^^^^

42: +         flag, intensity = light_rpc.call(
    ?             ++++++++++                  ^     ^^^                 ^^^^
43:               [units.add_units(t, 'hrs'),
44:                units.add_units(max(mesh.vertices[:, 2]), 'm')])

...
```

```
    -     # Compute the scale factor
47: +
48: +         # Compute the scale factor using intensity, stripping units
49: +         # of the result to allow use with trimesh
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++

51: +         scale = units.get_data(
52: +             units.add_units(mass, 'g') * intensity /
53: +             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
54: +
55: +     else:
56: +         # Compute the scale factor
57: +         # (pretend this is a biologically complex calculation)
    -     scale = mass / 4.5e4
58: +         scale = mass / 4.5e4
    ? ++++
```

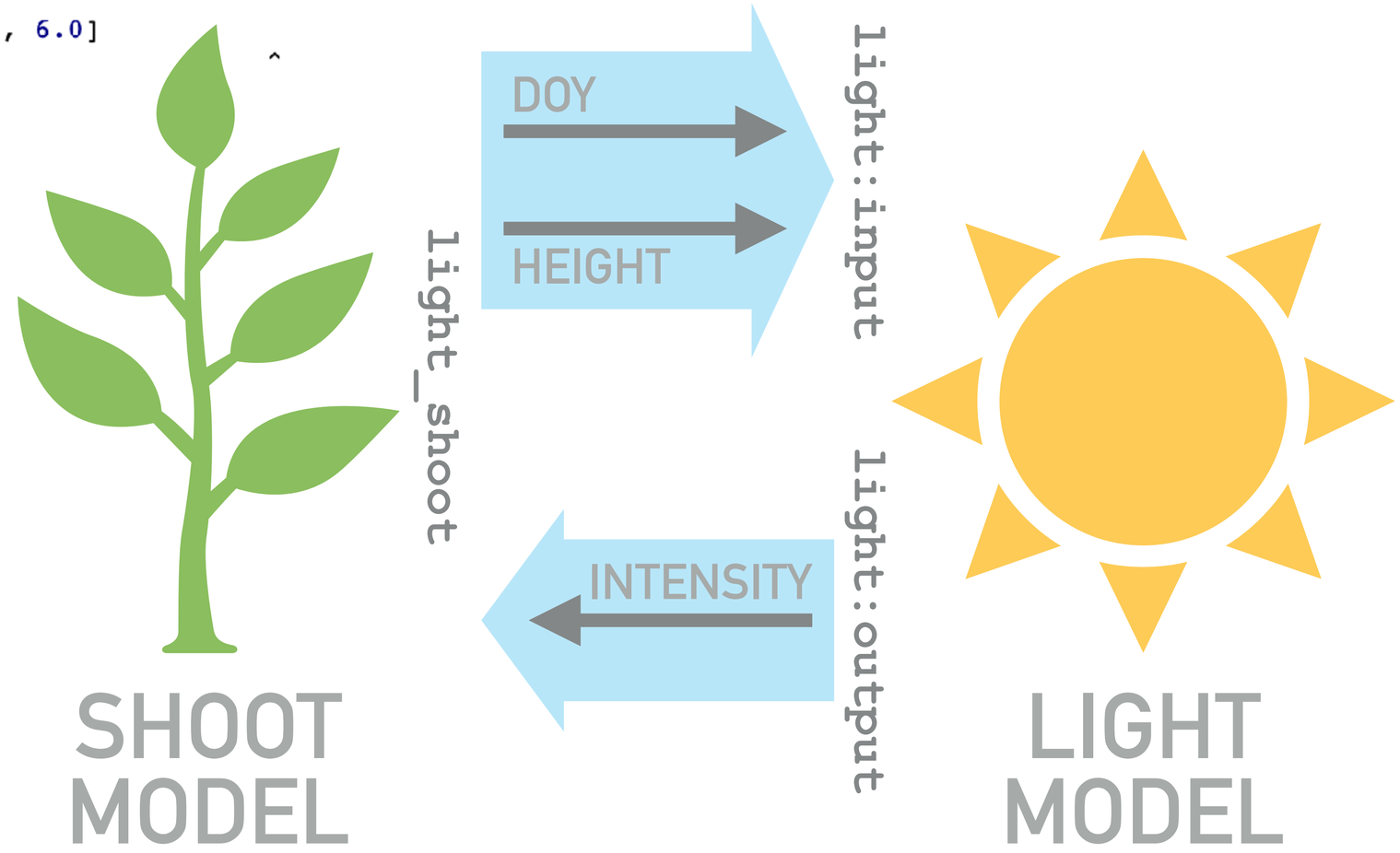
Change how scale is computed when yggdrasil used

```
In [7]: tools.display_source_diff('yamls/shoot_v1.yml', 'yamls/shoot_v2.yml', number_lines=True)
        tools.display_source_diff('yamls/light_v0_python.yml', 'yamls/light_v1_python.yml', number_lines=True)
```



```
In [7]: tools.display_source_diff('yamls/shoot_v1.yml', 'yamls/shoot_v2.yml', number_lines=True)
        tools.display_source_diff('yamls/light_v0_python.yml', 'yamls/light_v1_python.yml', number_lines=True)
```

```
file1: yamls/shoot_v1.yml
file2: yamls/shoot_v2.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v1.py, 0.0, 48.0, 6.0]
    ?                                                 ^

 4: +   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?                                                 ^

    -   outputs:
    -     - name: height
    ?  ----               ^^^          ^^

 5: +   client_of: light
    ?                ++++ ^^^^          ^

    -       default_file:
    -         name: ../output/height.txt
    -         filetype: table

file1: yamls/light_v0_python.yml
file2: yamls/light_v1_python.yml
================================
 1:   model:
 2:     name: light
 3:     language: python
 4:     args: ../models/light_v0.py
 5:     function: light
 6: +   is_server: true
```

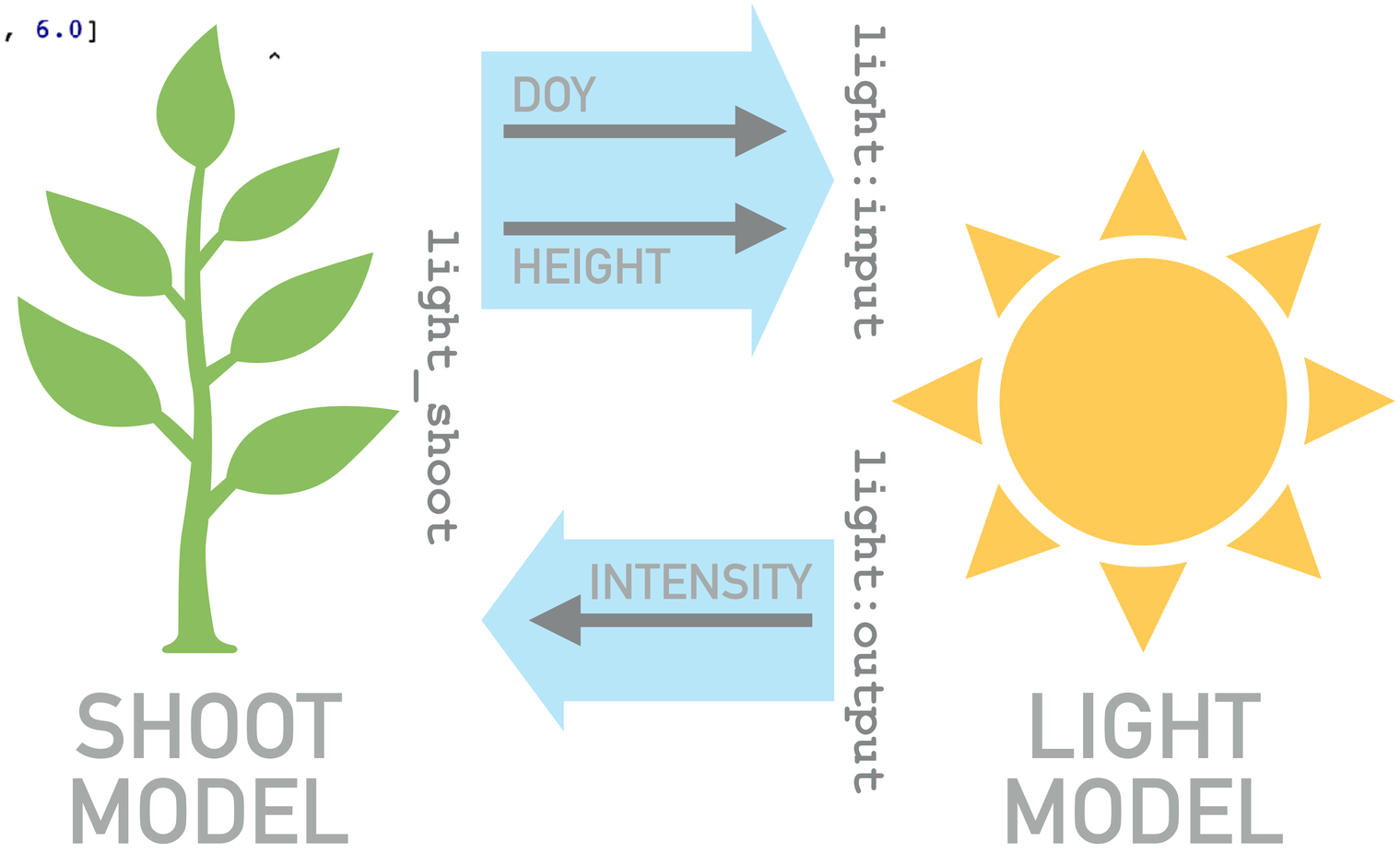


```
In [7]: tools.display_source_diff('yamls/shoot_v1.yml', 'yamls/shoot_v2.yml', number_lines=True)
        tools.display_source_diff('yamls/light_v0_python.yml', 'yamls/light_v1_python.yml', number_lines=True)
```

```
file1: yamls/shoot_v1.yml
file2: yamls/shoot_v2.yml
==========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v1.py, 0.0, 48.0, 6.0]
    ?                                                ^

 4: +   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?                                                ^

    -   outputs:
    -     - name: height
    ?  ----               ^^^          ^^

 5: +   client_of: light
    ?                +++++ ^^^^          ^

    -       default_file:
    -         name: ../output/height.txt
    -         filetype: table
```

```
file1: yamls/light_v0_python.yml
file2: yamls/light_v1_python.yml
================================
 1:   model:
 2:     name: light
 3:     language: python
 4:     args: ../models/light_v0.py
 5:     function: light
 6: +   is_server: true
```

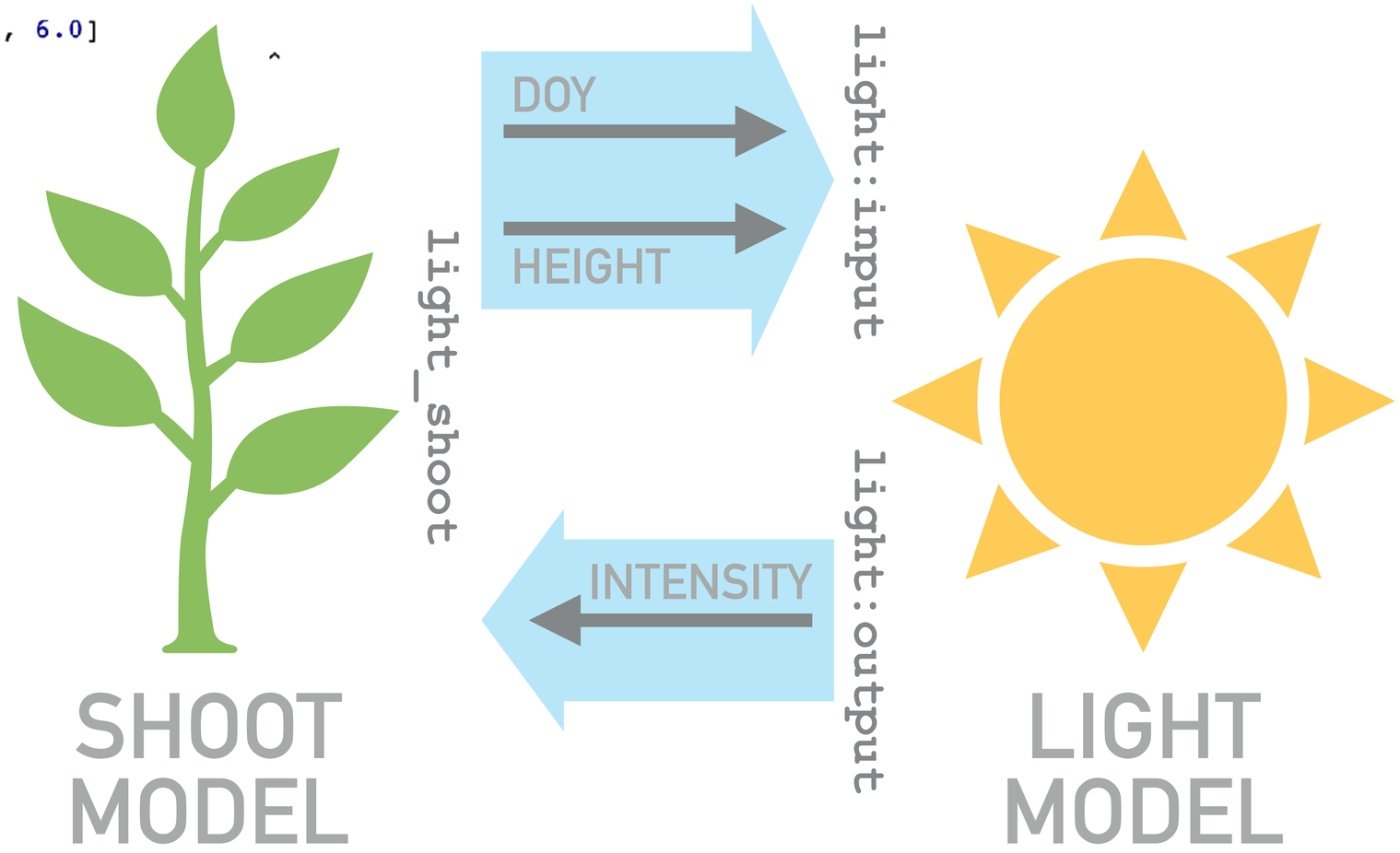
Remove output & add "client_of" to shoot model



```
In [7]: tools.display_source_diff('yamls/shoot_v1.yml', 'yamls/shoot_v2.yml', number_lines=True)
        tools.display_source_diff('yamls/light_v0_python.yml', 'yamls/light_v1_python.yml', number_lines=True)
```

```
file1: yamls/shoot_v1.yml
file2: yamls/shoot_v2.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v1.py, 0.0, 48.0, 6.0]
    ?                                                ^

 4: +   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?                                                ^

    -   outputs:
    -     - name: height
    ?  ----               ^^^          ^^

 5: +   client_of: light
    ?                +++++ ^^^^          ^

    -       default_file:
    -         name: ../output/height.txt
    -         filetype: table

file1: yamls/light_v0_python.yml
file2: yamls/light_v1_python.yml
================================
 1:   model:
 2:     name: light
 3:     language: python
 4:     args: ../models/light_v0.py
 5:     function: light
 6: +   is_server: true
```

Add "`is_server`" to light model



In [8]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2.yml'], production_run=True)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

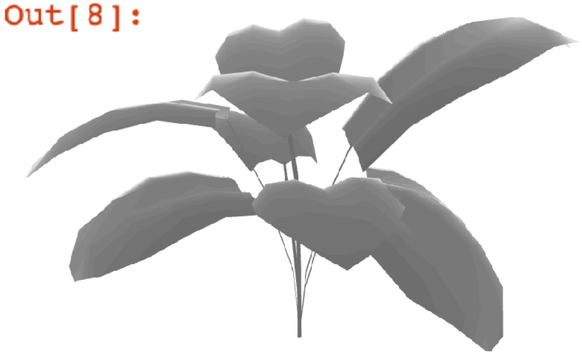
In [8]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2.yml'], production_run=True)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v2.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:93696:runner.run[374]:YggRunner(runner):         load drivers      0.010089
INFO:93696:runner.run[374]:YggRunner(runner):        start drivers      0.090419
INFO:93696:runner.run[374]:YggRunner(runner):           run models      7.769536
INFO:93696:runner.run[374]:YggRunner(runner):              at exit      0.044337
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                Total      7.914382
```

In [8]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2.yml'], production_run=True)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v2.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:93696:runner.run[374]:YggRunner(runner):          load drivers      0.010089
INFO:93696:runner.run[374]:YggRunner(runner):         start drivers      0.090419
INFO:93696:runner.run[374]:YggRunner(runner):            run models      7.769536
INFO:93696:runner.run[374]:YggRunner(runner):               at exit      0.044337
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                 Total      7.914382
```

Out[8]:



SPLITTING RPC CALLS

In [9]:
```
tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v2_split.py', number_lines=True)
```

In [9]:
```
tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v2_split.py', number_lines=True)
```

```
file1: models/shoot_v2.py
file2: models/shoot_v2_split.py
================================

...

39:         # If running as part an yggdrasil integration, send the time and
40:         # maximum height of the mesh to the height channel with units
41:         if with_yggdrasil:
42: +           # Send request to the light model
    -           flag, intensity = light_rpc.call(
    ?               -----------                                          ^^^^

43: +           flag = light_rpc.send(
    ?                                                            ^^^^

44:                 [units.add_units(t, 'hrs'),
45:                  units.add_units(max(mesh.vertices[:, 2]), 'm')])
46:             if not flag:
    -               raise Exception("Error calling the light model.")
    ?

47: +               raise Exception("Error sending request to the light model.")
    ?
++++++++++

48: +
49: +           # Calculations that don't rely on the output from the light model
50: +           # can be run here in parallel with the light model calculations
51: +
52: +           # Receive response from the light model
53: +           flag, intensity = light_rpc.recv()
54: +           if not flag:
55: +               raise Exception("Error receiving response from the light model.")

...
```

```
In [9]: tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v2_split.py', number_lines=True)
```

```
file1: models/shoot_v2.py
file2: models/shoot_v2_split.py
================================

...

39:         # If running as part an yggdrasil integration, send the time and
40:         # maximum height of the mesh to the height channel with units
41:         if with_yggdrasil:
42: +           # Send request to the light model
    -           flag, intensity = light_rpc.call(
    ?               -----------                                          ^^^^

43: +           flag = light_rpc.send(
    ?                                                            ^^^^

44:                 [units.add_units(t, 'hrs'),
45:                  units.add_units(max(mesh.vertices[:, 2]), 'm')])
46:             if not flag:
    -               raise Exception("Error calling the light model.")
    ?

47: +               raise Exception("Error sending request to the light model.")
    ?
++++++++++

48: +
49: +           # Calculations that don't rely on the output from the light model
50: +           # can be run here in parallel with the light model calculations
51: +
52: +           # Receive response from the light model
53: +           flag, intensity = light_rpc.recv()
54: +           if not flag:
55: +               raise Exception("Error receiving response from the light model.")

...
```

Calls can be split into send & recv to enhance parallelism

In [10]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2_split.yml'], production_run=True)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

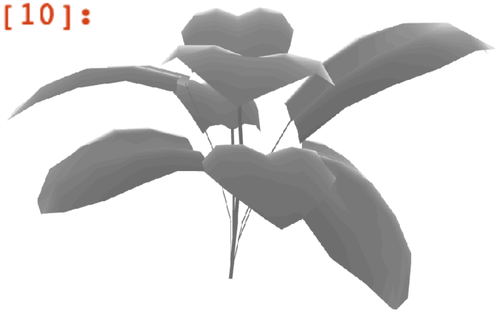
In [10]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2_split.yml'], production_run=True)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v2_split.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:93696:runner.run[374]:YggRunner(runner):         load drivers      0.009048
INFO:93696:runner.run[374]:YggRunner(runner):        start drivers      0.087974
INFO:93696:runner.run[374]:YggRunner(runner):           run models      7.865753
INFO:93696:runner.run[374]:YggRunner(runner):              at exit      0.044422
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                Total      8.007198
```

In [10]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2_split.yml'], production_run=True)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.show()
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v2_split.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:93696:runner.run[374]:YggRunner(runner):         load drivers      0.009048
INFO:93696:runner.run[374]:YggRunner(runner):        start drivers      0.087974
INFO:93696:runner.run[374]:YggRunner(runner):            run models      7.865753
INFO:93696:runner.run[374]:YggRunner(runner):               at exit      0.044422
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                 Total      8.007198
```

Out[10]:



Test your knowledge #4

1. Write a YAML to have the `models/shoot_v2.py` model call both the `models/light_v0.py` model and the `models/weather.py` model.
2. Update the YAMLs and source code for the `models/light_v0.py` and `models/weather.py` models so that the light model acts as a client and calls the weather model as a server and run the integration in the cell below.
3. Write a YAML to have the `models/shoot_v2.py` model call the `models/light_v0.py` model and have the `models/light_v0.py` model call the `models/weather.py` model (nested servers).

Tip: Models that are wrapped functions need to pass the `global_scope=True` keyword to interface functions (e.g. `YggRpcClient`) if they will be called more than once

TEST YOUR KNOWLEDGE (15 MIN)

Test your knowledge #4

1. Write a YAML to have the `models/shoot_v2.py` model call both the `models/light_v0.py` model and the `models/weather.py` model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run

# Part 1: shoot calling light and weather
tools.display_source_diff('models/shoot_v2.py',
                          'solutions/tyk4/models/shoot_calling_light_and_weather.py',
                          number_lines=True)
tools.display_source('solutions/tyk4/yamls/shoot_calling_light_and_weather.yml', number_lines=True)
run(['solutions/tyk4/yamls/shoot_calling_light_and_weather.yml'], production_run=True)

# Part 2: light-calling-weather
tools.display_source_diff('models/light_v0.py',
                          'solutions/tyk4/models/light_calling_weather.py', number_lines=True)
tools.display_source('solutions/tyk4/yamls/light_calling_weather.yml', number_lines=True)
run(['solutions/tyk4/yamls/light_calling_weather.yml'], production_run=True)

# Part 3: shoot calling light calling weather
tools.display_source_diff('models/shoot_v2.py',
                          'solutions/tyk4/models/shoot_calling_light_calling_weather.py',
                          number_lines=True)
tools.display_source('solutions/tyk4/yamls/shoot_calling_light_calling_weather.yml',
                     number_lines=True)
run(['solutions/tyk4/yamls/shoot_calling_light_calling_weather.yml'],
    production_run=True)
```

Test your knowledge #4

1. Write a YAML to have the `models/shoot_v2.py` model call both the `models/light_v0.py` model and the `models/weather.py` model.

```
28:    # If the model is running as part of an yggdrasil integration, import
29:    # the relevant yggdrasil routines and use the interface routine to
30:    # complete the connection defined in the YAML
31:    if with_yggdrasil:
32:        from yggdrasil import units
33:        from yggdrasil.languages.Python.YggInterface import YggRpcClient
34:        light_rpc = YggRpcClient('light_shoot')
35: +      weather_rpc = YggRpcClient('weather_shoot')

...
49: +          flag, temp = weather_rpc.call(intensity)
50: +          if not flag:
51: +              raise Exception("Error calling the weather model.")
52: +
53:            # Compute the scale factor using intensity, stripping units
54:            # of the result to allow use with trimesh
55:            # (pretend this is a biologically complex calculation)
56:            scale = units.get_data(
57: +              (temp / units.add_units(400.0, 'K')) *
58:                units.add_units(mass, 'g') * intensity /
59:                units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
...
```

Test your knowledge #4

1. Write a YAML to have the `models/shoot_v2.py` model call both the `models/light_v0.py` model and the `models/weather.py` model.

```
file: solutions/tyk4/yamls/shoot_calling_light_and_weather.yml
==============================================================
 1: models:
 2:   - name: shoot
 3:     language: python
 4:     args: [../models/shoot_calling_light_and_weather.py, 0.0, 48.0, 6.0]
 5:     client_of: [light, weather]
 6:   - name: light
 7:     language: python
 8:     args: ../models/light_v0.py
 9:     function: light
10:     is_server: true
11:   - name: weather
12:     language: python
13:     args: ../models/weather.py
14:     function: temp
15:     is_server: true
```

Test your knowledge #4

1. Write a YAML to have the `models/shoot_v2.py` model call both the `models/light_v0.py` model and the `models/weather.py` model.

```
INFO:68954:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/ygg_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/ygg_weather.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/shoot_calling_light_and_weather.py 0.0 48.0 6.0

End of input from temp_doy.
End of input from intensity.
INFO:68954:runner.waitModels[553]:YggRunner(runner): weather finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): weather finished exiting.
INFO:68954:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:68954:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:68954:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:68954:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:68954:runner.run[374]:YggRunner(runner):         load drivers      1.313143
INFO:68954:runner.run[374]:YggRunner(runner):        start drivers      0.600414
INFO:68954:runner.run[374]:YggRunner(runner):           run models      24.028359
INFO:68954:runner.run[374]:YggRunner(runner):              at exit      0.132220
INFO:68954:runner.run[376]:YggRunner(runner): ========================================
INFO:68954:runner.run[377]:YggRunner(runner):                Total      26.074137
```

Test your knowledge #4

2. Update the YAMLs and source code for the `models/light_v0.py` and `models/weather.py` models so that the light model acts as a client and calls the weather model as a server and run the integration in the cell below.

```
+    # Check if model is running as a part of an yggdrasil integration
+    with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
+
+    # If the model is running as part of an yggdrasil integration, import
+    # the relevant yggdrasil routines and use the interface routine to
+    # complete the connection defined in the YAML
+    if with_yggdrasil:
+        from yggdrasil.languages.Python.YggInterface import YggRpcClient
+        weather_rpc = YggRpcClient('weather_light', global_scope=True)
+
+        # Call the weather model
+        flag, temp = weather_rpc.call(intensity)
+        if not flag:
+            raise Exception("Failed to call the weather model.")
+
+        # Return both the intensity and temp
+        return intensity, temp
+
+
     return intensity
```

Test your knowledge #4

2. Update the YAMLs and source code for the `models/light_v0.py` and `models/weather.py` models so that the light model acts as a client and calls the weather model as a server and run the integration in the cell below.

```
file: solutions/tyk4/yamls/light_calling_weather.yml
====================================================
models:
  - name: light
    language: python
    args: ../models/light_calling_weather.py
    function: light
    client_of: weather
    inputs:
      name: input
      default_file:
        name: ../input/height.txt
        filetype: table
    outputs:
      name: output
      default_file:
        name: ../output/light_and_weather.txt
        filetype: table
        field_names: [intensity,temperature]
  - name: weather
    language: python
    args: ../models/weather.py
    function: temp
    is_server: true
```

Test your knowledge #4

2. Update the YAMLs and source code for the `models/light_v0.py` and `models/weather.py` models so that the light model acts as a client and calls the weather model as a server and run the integration in the cell below.

```
INFO:68954:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
```

```
/Users/langmm/yggdrasil/yggdrasil/drivers/ModelDriver.py:1815: UserWarning: When wrapping a model function, client co
mms must either be initialized outside the function, pass a 'global_scope' parameter to the comm initialization (e.g.
Python, R, Matlab), or use a 'WITH_GLOBAL_SCOPE' macro (e.g. C, C++, Fortran) around the initialization so that they
are persistent across calls and the call or recv/send methods must be called explicitly (as opposed to the function i
nputs/outputs which will be handled by the wrapper). This model's client comms are:
        ['light:weather_light']
  "\t%s" % client_comms)
```

```
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/ygg_weather.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/ygg_light_calling_weather.py
End of input from temp_doy.
End of input from intensity.
INFO:68954:runner.waitModels[553]:YggRunner(runner): weather finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): weather finished exiting.
INFO:68954:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:68954:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:68954:runner.run[374]:YggRunner(runner):                  init      0.000002
INFO:68954:runner.run[374]:YggRunner(runner):          load drivers      0.058634
INFO:68954:runner.run[374]:YggRunner(runner):         start drivers      0.322480
INFO:68954:runner.run[374]:YggRunner(runner):            run models      13.198760
INFO:68954:runner.run[374]:YggRunner(runner):               at exit      0.099057
INFO:68954:runner.run[376]:YggRunner(runner): ========================================
INFO:68954:runner.run[377]:YggRunner(runner):                 Total      13.678933
```

Test your knowledge #4

3. Write a YAML to have the `models/shoot_v2.py` model call the `models/light_v0.py` model and have the `models/light_v0.py` model call the `models/weather.py` model (nested servers).

```
file1: models/shoot_v2.py
file2: solutions/tyk4/models/shoot_calling_light_calling_weather.py
===================================================================

...

39:         # If running as part an yggdrasil integration, send the time and
40:         # maximum height of the mesh to the height channel with units
41:         if with_yggdrasil:
    -           flag, intensity = light_rpc.call(
    ?                   ^ ^^^^ ^

42: +           flag, light_data = light_rpc.call(
    ?                 + ^^ ^^^ ^

43:                 [units.add_units(t, 'hrs'),
44:                  units.add_units(max(mesh.vertices[:, 2]), 'm')])
45:             if not flag:
46:                 raise Exception("Error calling the light model.")
    -
47: +           intensity, temp = light_data[:]
48: +
49:             # Compute the scale factor using intensity, stripping units
50:             # of the result to allow use with trimesh
51:             # (pretend this is a biologically complex calculation)
52:             scale = units.get_data(
53: +               (temp / units.add_units(400.0, 'K')) *
54:                 units.add_units(mass, 'g') * intensity /
55:                 units.add_units(4.0e10, 'g*erg/(cm**2*s)'))

...
```

Test your knowledge #4

3. Write a YAML to have the `models/shoot_v2.py` model call the `models/light_v0.py` model and have the `models/light_v0.py` model call the `models/weather.py` model (nested servers).

```
file: solutions/tyk4/yamls/shoot_calling_light_calling_weather.yml
====================================================================
models:
  - name: shoot
    language: python
    args: [../models/shoot_calling_light_calling_weather.py, 0.0, 48.0, 6.0]
    client_of: light
  - name: light
    language: python
    args: ../models/light_calling_weather.py
    function: light
    client_of: weather
    is_server: true
  - name: weather
    language: python
    args: ../models/weather.py
    function: temp
    is_server: true
```

Test your knowledge #4

3. Write a YAML to have the `models/shoot_v2.py` model call the `models/light_v0.py` model and have the `models/light_v0.py` model call the `models/weather.py` model (nested servers).

```
/Users/langmm/yggdrasil/yggdrasil/drivers/ModelDriver.py:1815: UserWarning: When wrapping a model function, client co
mms must either be initialized outside the function, pass a 'global_scope' parameter to the comm initialization (e.g.
Python, R, Matlab), or use a 'WITH_GLOBAL_SCOPE' macro (e.g. C, C++, Fortran) around the initialization so that they
are persistent across calls and the call or recv/send methods must be called explicitly (as opposed to the function i
nputs/outputs which will be handled by the wrapper). This model's client comms are:
        ['light:weather_light']
  "\t%s" % client_comms)
```

```
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/ygg_light_calling_weather.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/shoot_calling_light_calling_weather.py 0.0 48.0 6.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk4/models/ygg_weather.py
End of input from temp_doy.
End of input from intensity.
INFO:68954:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:68954:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:68954:runner.waitModels[553]:YggRunner(runner): weather finished running.
INFO:68954:runner.waitModels[559]:YggRunner(runner): weather finished exiting.
INFO:68954:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:68954:runner.run[374]:YggRunner(runner):                 init      0.000006
INFO:68954:runner.run[374]:YggRunner(runner):         load drivers      0.070852
INFO:68954:runner.run[374]:YggRunner(runner):        start drivers      0.559098
INFO:68954:runner.run[374]:YggRunner(runner):           run models      29.969198
INFO:68954:runner.run[374]:YggRunner(runner):              at exit      0.143363
INFO:68954:runner.run[376]:YggRunner(runner): ========================================
INFO:68954:runner.run[377]:YggRunner(runner):                Total      30.742517
```

DUPLICATING MODELS

In [11]:
```
tools.display_source_diff('models/shoot_v2_split.py', 'models/shoot_v2_copies.py', number_lines=True)
```

...

In [11]:
```
tools.display_source_diff('models/shoot_v2_split.py', 'models/shoot_v2_copies.py', number_lines=True)
```

```
file1: models/shoot_v2_split.py
file2: models/shoot_v2_copies.py
================================

...

41:         # If running as part an yggdrasil integration, send the time and
42:         # maximum height of the mesh to the height channel with units
43:         if with_yggdrasil:
    -           # Send request to the light model
44: +           # Send requests to the light model for each mesh vertex
    ?                                 +                    ++++++++++++++++++++

45: +           for v in mesh.vertices[:, 2]:
    -           flag = light_rpc.send(
46: +               flag = light_rpc.send(
    ? ++++

    -               [units.add_units(t, 'hrs'),
47: +                   [units.add_units(t, 'hrs'),
    ? ++++

    -                units.add_units(max(mesh.vertices[:, 2]), 'm')])
48: +                   units.add_units(v, 'm')])
    -           if not flag:
49: +               if not flag:
    ? ++++

...
```

```
In [11]: tools.display_source_diff('models/shoot_v2_split.py', 'models/shoot_v2_copies.py', number_lines=True)
```

```
file1: models/shoot_v2_split.py
file2: models/shoot_v2_copies.py
================================
...
41:         # If running as part an yggdrasil integration, send the time and
42:         # maximum height of the mesh to the height channel with units
43:         if with_yggdrasil:
    -           # Send request to the light model
44: +           # Send requests to the light model for each mesh vertex
    ?                                 +                    ++++++++++++++++++++

45: +           for v in mesh.vertices[:, 2]:
    -           flag = light_rpc.send(
46: +               flag = light_rpc.send(
    ? ++++

    -               [units.add_units(t, 'hrs'),
47: +                   [units.add_units(t, 'hrs'),
    ? ++++

    -                units.add_units(max(mesh.vertices[:, 2]), 'm')])
48: +                   units.add_units(v, 'm')])
    -           if not flag:
49: +               if not flag:
    ? ++++
...
```

Send heights for each vertex to the light model

In [11]:
```
tools.display_source_diff('models/shoot_v2_split.py', 'models/shoot_v2_copies.py', number_lines=True)
```

```
file1: models/shoot_v2_split.py
file2: models/shoot_v2_copies.py
================================

...

56: +         nvert = mesh.vertices.shape[0]
57: +         intensity = np.zeros(nvert, 'f8')
58: +         for iv in range(nvert):
    -         flag, intensity = light_rpc.recv()
59: +             flag, v_intensity = light_rpc.recv()
    ? ++++              ++

    -         if not flag:
60: +             if not flag:
    ? ++++

    -             raise Exception("Error receiving response from the light model.")
61: +                 raise Exception("Error receiving response from the light model.")
    ? ++++

62: +             if not units.has_units(intensity):
63: +                 intensity = units.add_units(intensity,
64: +                                             units.get_units(v_intensity))
65: +             intensity[iv] = v_intensity
66: +         filename_light = os.path.join(_dir, f'../output/light_{i:03d}.pkl')
67: +         with open(filename_light, 'wb') as fd:
68: +             pickle.dump(intensity, fd)

...
```

```
In [11]: tools.display_source_diff('models/shoot_v2_split.py', 'models/shoot_v2_copies.py', number_lines=True)
```

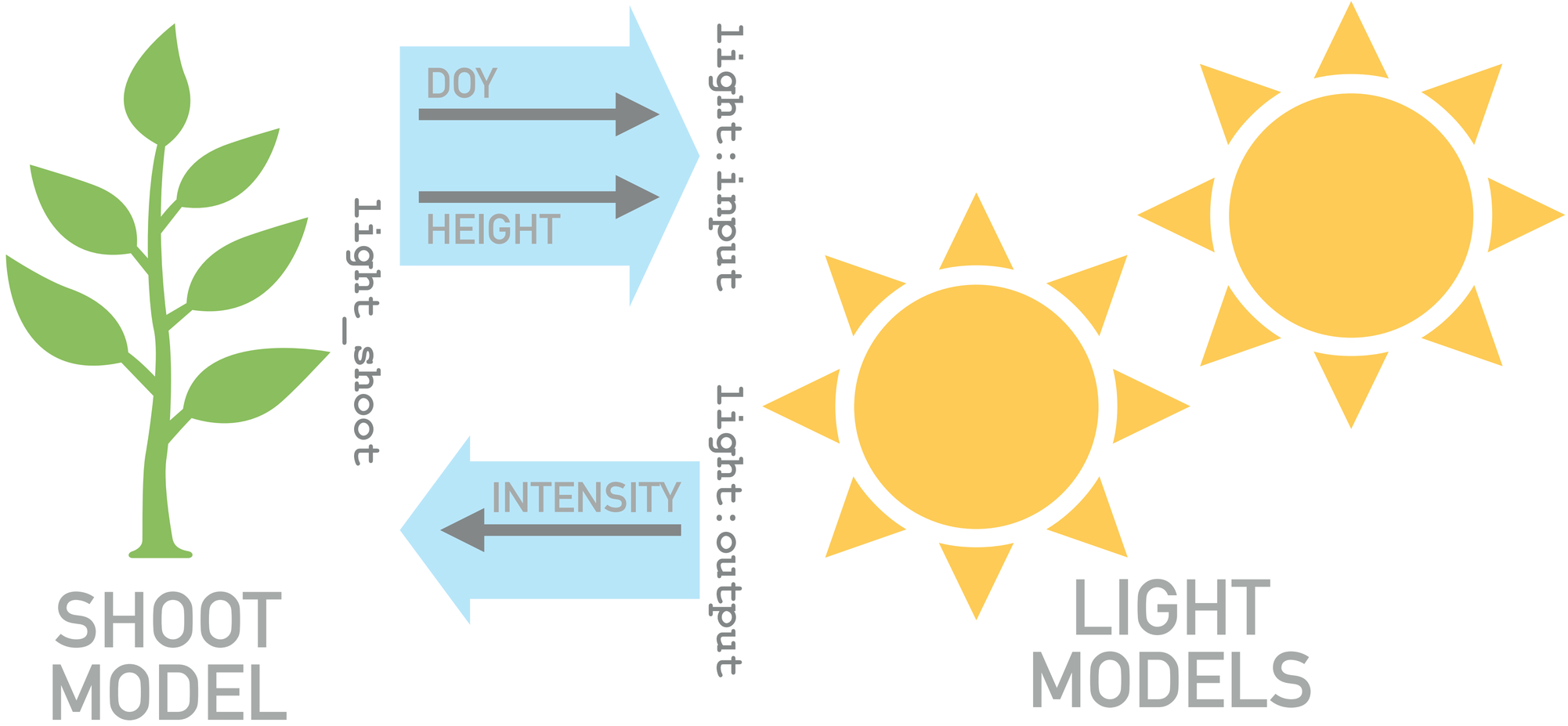
```
file1: models/shoot_v2_split.py
file2: models/shoot_v2_copies.py
================================
...

56: +         nvert = mesh.vertices.shape[0]
57: +         intensity = np.zeros(nvert, 'f8')
58: +         for iv in range(nvert):
    -         flag, intensity = light_rpc.recv()
59: +             flag, v_intensity = light_rpc.recv()
    ? ++++              ++

    -         if not flag:
60: +             if not flag:
    ? ++++

    -             raise Exception("Error receiving response from the light model.")
61: +                 raise Exception("Error receiving response from the light model.")
    ? ++++

62: +             if not units.has_units(intensity):
63: +                 intensity = units.add_units(intensity,
64: +                                             units.get_units(v_intensity))
65: +             intensity[iv] = v_intensity
66: +         filename_light = os.path.join(_dir, f'../output/light_{i:03d}.pkl')
67: +         with open(filename_light, 'wb') as fd:
68: +             pickle.dump(intensity, fd)

...
```

Receive intensities for each vertex

```
In [11]: tools.display_source_diff('models/shoot_v2_split.py', 'models/shoot_v2_copies.py', number_lines=True)
```

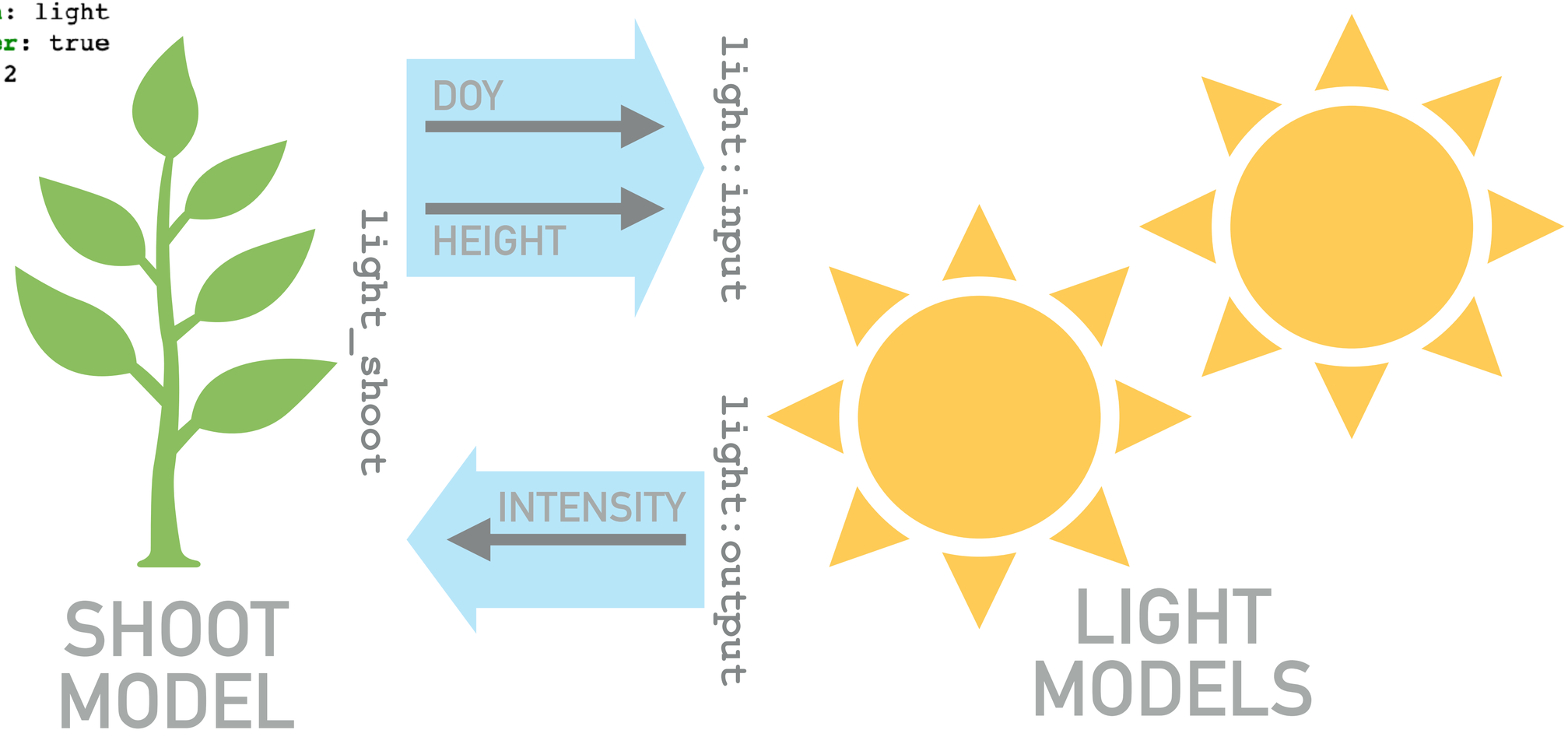
```
file1: models/shoot_v2_split.py
file2: models/shoot_v2_copies.py
================================

...

56: +         nvert = mesh.vertices.shape[0]
57: +         intensity = np.zeros(nvert, 'f8')
58: +         for iv in range(nvert):
    -         flag, intensity = light_rpc.recv()
59: +             flag, v_intensity = light_rpc.recv()
    ? ++++              ++

    -         if not flag:
60: +             if not flag:
    ? ++++

    -             raise Exception("Error receiving response from the light model.")
61: +                 raise Exception("Error receiving response from the light model.")
    ? ++++

62: +             if not units.has_units(intensity):
63: +                 intensity = units.add_units(intensity,
64: +                                             units.get_units(v_intensity))
65: +             intensity[iv] = v_intensity
66: +         filename_light = os.path.join(_dir, f'../output/light_{i:03d}.pkl')
67: +         with open(filename_light, 'wb') as fd:
68: +             pickle.dump(intensity, fd)

...
```

Save intensities for each vertex to a file


SHOOT MODEL
light_shoot
DOY
HEIGHT
light:input
INTENSITY
light:output
LIGHT MODELS

```
In [12]: tools.display_source_diff('yamls/light_v1_python.yml', 'yamls/light_v2_python.yml', number_lines=True)
```

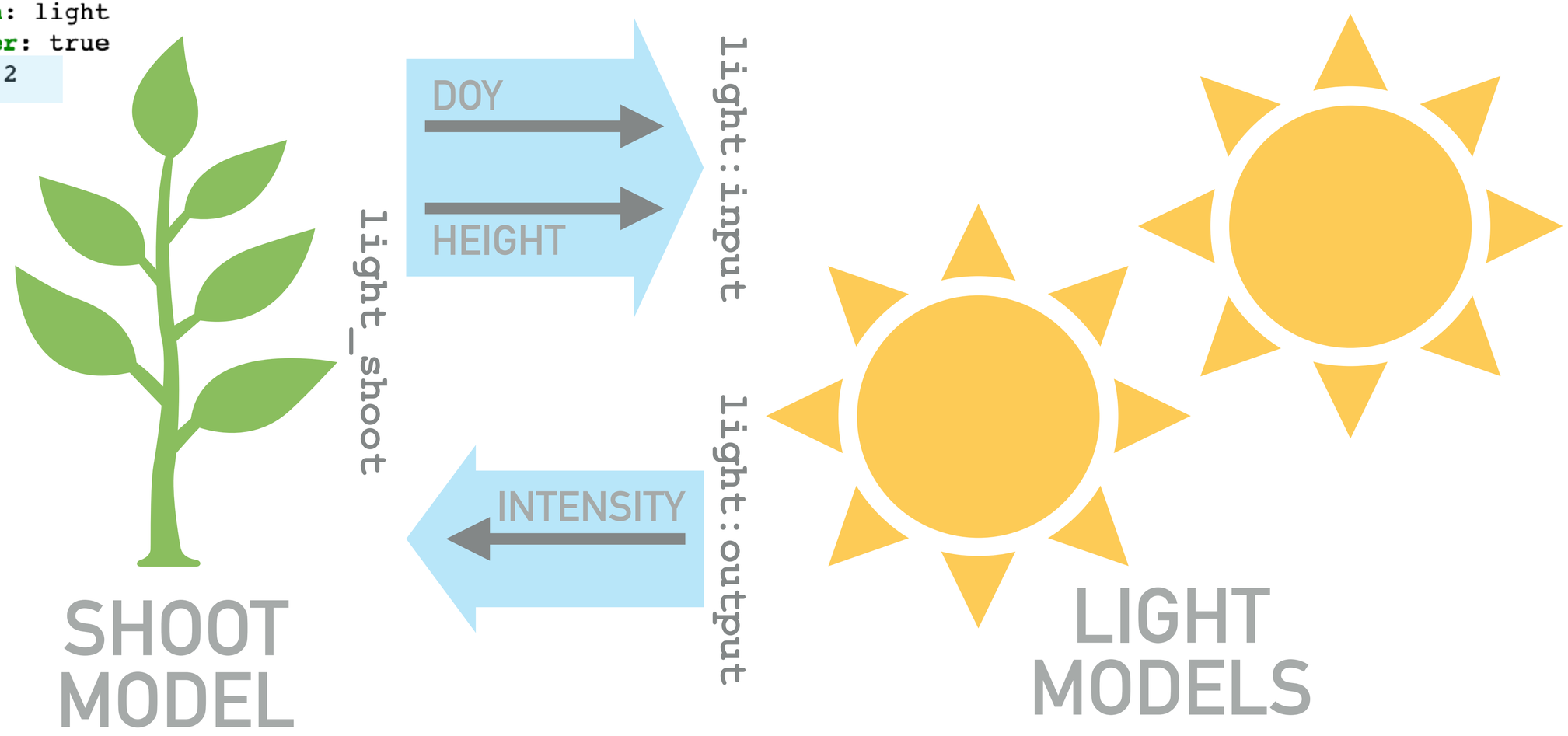
```
file1: yamls/light_v1_python.yml
file2: yamls/light_v2_python.yml
================================
 1:   model:
 2:     name: light
 3:     language: python
 4:     args: ../models/light_v0.py
 5:     function: light
 6:     is_server: true
 7: +   copies: 2
```



```
In [12]: tools.display_source_diff('yamls/light_v1_python.yml', 'yamls/light_v2_python.yml', number_lines=True)
```

```
file1: yamls/light_v1_python.yml
file2: yamls/light_v2_python.yml
================================
 1:    model:
 2:      name: light
 3:      language: python
 4:      args: ../models/light_v0.py
 5:      function: light
 6:      is_server: true
 7: +    copies: 2
```

Add “copies” to indicate multiple instances of a model should be run



In [13]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2_copies.yml'], production_run=True)

# Plot results w/ light intensity mapped to color
import pickle
with open('output/light_008.pkl', 'rb') as fd:
    light = pickle.load(fd)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.visual.vertex_colors = trimesh.visual.interpolate(light/max(light))
mesh.show()
```

Out[13]:

In [13]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2_copies.yml'], production_run=True)

# Plot results w/ light intensity mapped to color
import pickle
with open('output/light_008.pkl', 'rb') as fd:
    light = pickle.load(fd)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.visual.vertex_colors = trimesh.visual.interpolate(light/max(light))
mesh.show()
```

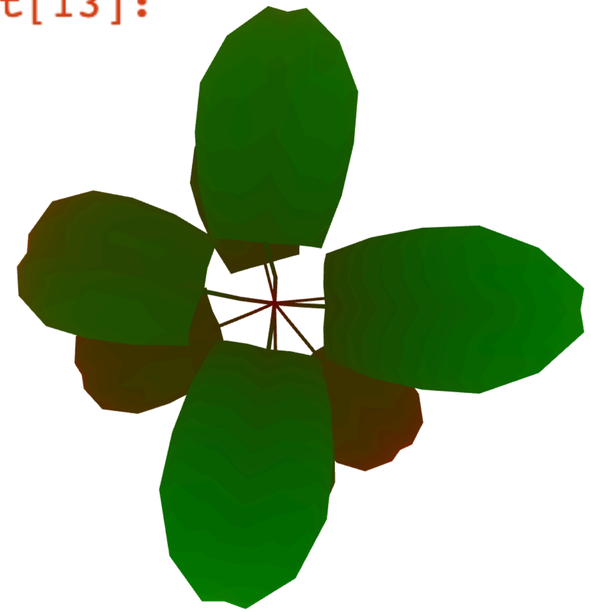
```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v2_copies.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                 init      0.000000
INFO:93696:runner.run[374]:YggRunner(runner):         load drivers      0.009468
INFO:93696:runner.run[374]:YggRunner(runner):        start drivers      0.086156
INFO:93696:runner.run[374]:YggRunner(runner):           run models      107.365030
INFO:93696:runner.run[374]:YggRunner(runner):              at exit      0.028054
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                Total      107.488708
```

Out[13]:

In [13]:
```
run(['yamls/light_v1_python.yml', 'yamls/shoot_v2_copies.yml'], production_run=True)

# Plot results w/ light intensity mapped to color
import pickle
with open('output/light_008.pkl', 'rb') as fd:
    light = pickle.load(fd)
mesh = trimesh.load_mesh('output/mesh_008.obj')
mesh.visual.vertex_colors = trimesh.visual.interpolate(light/max(light))
mesh.show()
```

```
INFO:93696:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v2_copies.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:93696:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:93696:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:93696:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:93696:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:93696:runner.run[374]:YggRunner(runner):                  init      0.000000
INFO:93696:runner.run[374]:YggRunner(runner):          load drivers      0.009468
INFO:93696:runner.run[374]:YggRunner(runner):         start drivers      0.086156
INFO:93696:runner.run[374]:YggRunner(runner):            run models      107.365030
INFO:93696:runner.run[374]:YggRunner(runner):               at exit      0.028054
INFO:93696:runner.run[376]:YggRunner(runner): ========================================
INFO:93696:runner.run[377]:YggRunner(runner):                 Total      107.488708
```

Out[13]:

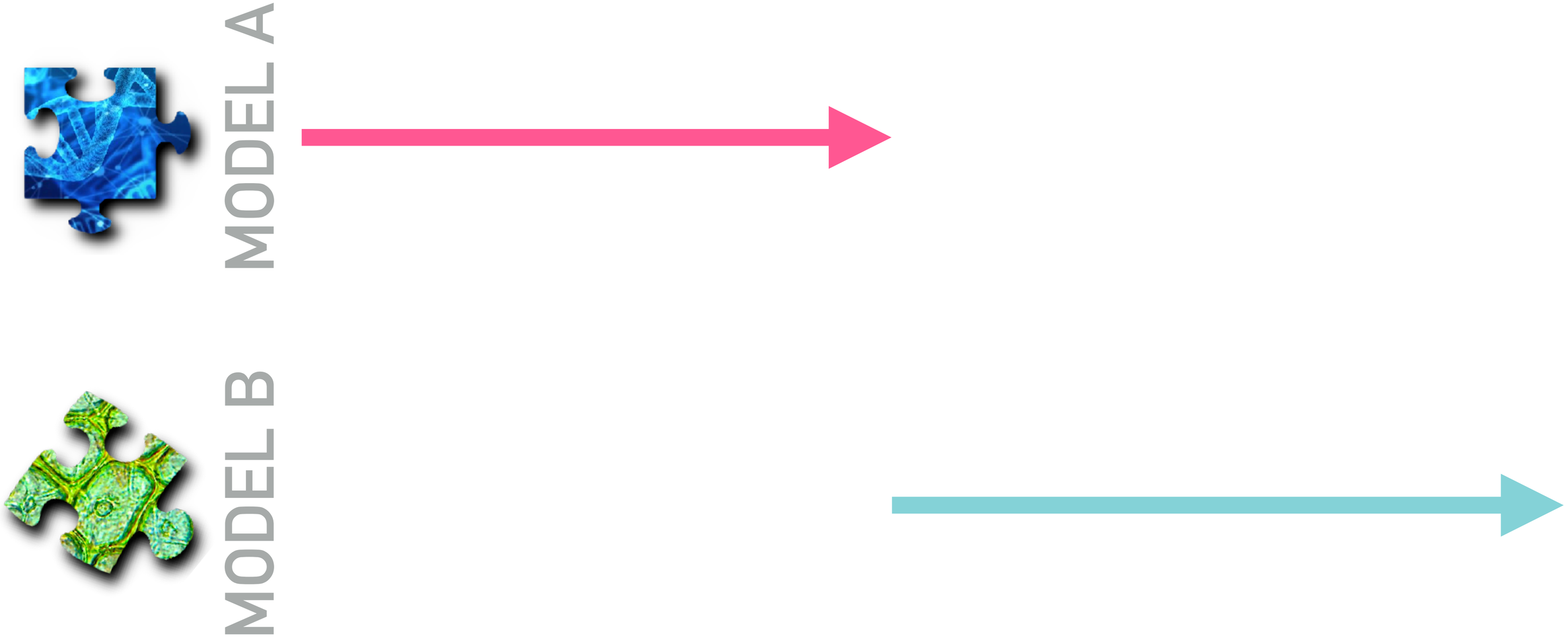


PARALLELISM ON MYBINDER

MyBinder instances only have access to 1 core (models run in serial)

PARALLELISM ON MYBINDER

MyBinder instances only have access to 1 core (models run in serial)



Test your knowledge #5

1. Increase the number of copies of the in the `yamls/shoot_v1.yml` model. What do you think the output file (`output/height.txt`) will look like?
2. Run the version with multiple copies. What does the output file (`output/height.txt`) look like? Why?

Tip: There is a bug(!) When duplicating a wrapped function model (e.g. the light model) that is not a server, the duplicates won't be shut down cleanly. If you run into this (the notebook will hang), you will need to interupt the kernel (`Ctrl+C` on the command line)

TEST YOUR KNOWLEDGE (10 MIN)

Test your knowledge #5

1. Increase the number of copies of the in the `yamls/shoot_v1.yml` model. What do you think the output file (`output/height.txt`) will look like?

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run

# Part 1: add copies to isolated light model
tools.display_source_diff('yamls/shoot_v1.yml',
                          'solutions/tyk5/yamls/shoot_copies.yml',
                          number_lines=True)

# Part 2: run it & check output
run(['solutions/tyk5/yamls/shoot_copies.yml'], production_run=True)
tools.display_source('solutions/tyk5/output/height.txt')
```

```
file1: yamls/shoot_v1.yml
file2: solutions/tyk5/yamls/shoot_copies.yml
==========================================
 1:   model:
 2:     name: shoot
 3:     language: python
 4:     args: [../models/shoot_v1.py, 0.0, 48.0, 6.0]
 5: +   copies: 2
 6:     outputs:
 7:       - name: height
 8:         default_file:
 9:           name: ../output/height.txt
10:           filetype: table
```

Test your knowledge #5

2. Run the version with multiple copies. What does the output file (`output/height.txt`) look like? Why?

```
INFO:72653:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk5/models/shoot_v1.py 0.0 48.0 6.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk5/models/shoot_v1.py 0.0 48.0 6.0
INFO:72653:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:72653:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:72653:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:72653:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:72653:runner.run[374]:YggRunner(runner):         load drivers      0.477997
INFO:72653:runner.run[374]:YggRunner(runner):        start drivers      0.111732
INFO:72653:runner.run[374]:YggRunner(runner):           run models      10.588425
INFO:72653:runner.run[374]:YggRunner(runner):              at exit      0.018711
INFO:72653:runner.run[376]:YggRunner(runner): ========================================
INFO:72653:runner.run[377]:YggRunner(runner):                Total      11.196866
```

Test your knowledge #5

2. Run the version with multiple copies. What does the output file (`output/height.txt`) look like? Why?

```
file: solutions/tyk5/output/height.txt
=====================================
# hr     m
# %g     %g
0        77.2603
6        80.6941
0        77.2603
12       84.4399
6        80.6941
18       88.5415
12       84.4399
24       93.0513
18       88.5415
30       98.0321
24       93.0513
36       103.561
30       98.0321
42       109.73
36       103.561
48       116.656
42       109.73
```

NEW NOTEBOOK!

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

TIME STEP SYNCHRONIZATION

In [2]:
```
tools.display_source('models/roots_v0.py', number_lines=True)
```

...

...

In [2]:
```python
tools.display_source('models/roots_v0.py', number_lines=True)
```

file: models/roots_v0.py
==========

```python
 1: import os
 2: import argparse
 3: import pickle
 4:
 5: _dir = os.path.dirname(os.path.realpath(__file__))
 6:
 7: # Parse command-line arguments
 8: parser = argparse.ArgumentParser("Simulate root growth over time.")
 9: parser.add_argument('tmin', help='Starting time (in days)', type=float)
10: parser.add_argument('tmax', help='Ending time (in days)', type=float)
11: parser.add_argument('tstep', help='Time step (in days)', type=float)
12: args = parser.parse_args()
13: tmin = args.tmin
14: tmax = args.tmax
15: tstep = args.tstep
16:
17: # Set initial conditions
18: mass = 0.0
19: t = tmin
20: times = []
21: masses = []
22:
23: # Continue simulation until time limit is reached
24: while t <= tmax:
25:
26:     # Compute the scale factor
27:     # (pretend this is a biologically complex calculation)
28:     scale = 0.2
29:
30:     # Calculate mass for the time step
31:     # (pretend this is a biologically complex calculation)
32:     mass += t * scale
33:
```

...

...

```python
34:     # Add mass & time to array
35:     times.append(t)
36:     masses.append(mass)
37:
38:     # Advance time step
39:     t += tstep
40:
41: # Write the total mass array to output
42: filename_masses = os.path.join(_dir, '../output/masses.pkl')
43: with open(filename_masses, 'wb') as fd:
44:     pickle.dump({'times': times, 'masses': masses}, fd)
```

In [3]:
```
tools.display_source('yamls/roots_v0.yml', number_lines=True)
run(['yamls/roots_v0.yml'], production_run=True)
```

In [3]:
```
tools.display_source('yamls/roots_v0.yml', number_lines=True)
run(['yamls/roots_v0.yml'], production_run=True)
```

```
file: yamls/roots_v0.yml
========================
 1: model:
 2:   name: roots
 3:   language: python
 4:   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]
```

In [3]:
```
tools.display_source('yamls/roots_v0.yml', number_lines=True)
run(['yamls/roots_v0.yml'], production_run=True)
```

```
file: yamls/roots_v0.yml
========================
 1: model:
 2:   name: roots
 3:   language: python
 4:   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]

INFO:96257:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/roo
ts_v0.py 0.0 2.0 0.5
INFO:96257:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:96257:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
INFO:96257:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:96257:runner.run[374]:YggRunner(runner):                 init      0.000000
INFO:96257:runner.run[374]:YggRunner(runner):         load drivers      0.279398
INFO:96257:runner.run[374]:YggRunner(runner):        start drivers      0.039544
INFO:96257:runner.run[374]:YggRunner(runner):           run models      0.103028
INFO:96257:runner.run[374]:YggRunner(runner):              at exit      0.000614
INFO:96257:runner.run[376]:YggRunner(runner): ========================================
INFO:96257:runner.run[377]:YggRunner(runner):                Total      0.422584
```

In [3]:
```
tools.display_source('yamls/roots_v0.yml', number_lines=True)
run(['yamls/roots_v0.yml'], production_run=True)
```

```
file: yamls/roots_v0.yml
========================
 1: model:
 2:   name: roots
 3:   language: python
 4:   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]

INFO:96257:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/roo
ts_v0.py 0.0 2.0 0.5
INFO:96257:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:96257:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
INFO:96257:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:96257:runner.run[374]:YggRunner(runner):                 init      0.000000
INFO:96257:runner.run[374]:YggRunner(runner):         load drivers      0.279398
INFO:96257:runner.run[374]:YggRunner(runner):        start drivers      0.039544
INFO:96257:runner.run[374]:YggRunner(runner):           run models      0.103028
INFO:96257:runner.run[374]:YggRunner(runner):              at exit      0.000614
INFO:96257:runner.run[376]:YggRunner(runner): ========================================
INFO:96257:runner.run[377]:YggRunner(runner):                Total      0.422584
```

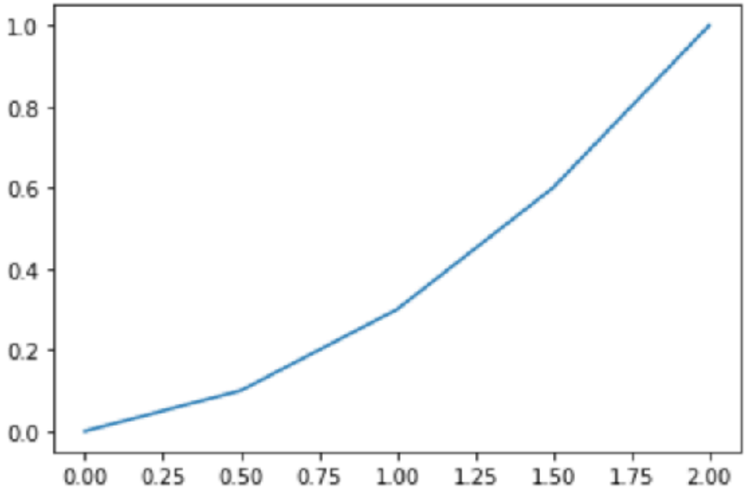
In [4]:
```
import matplotlib.pyplot as plt
filename_masses = 'output/masses.pkl'
with open(filename_masses, 'rb') as fd:
    masses = pickle.load(fd)
plt.plot(masses['times'], masses['masses'])
```

In [3]:
```
tools.display_source('yamls/roots_v0.yml', number_lines=True)
run(['yamls/roots_v0.yml'], production_run=True)
```

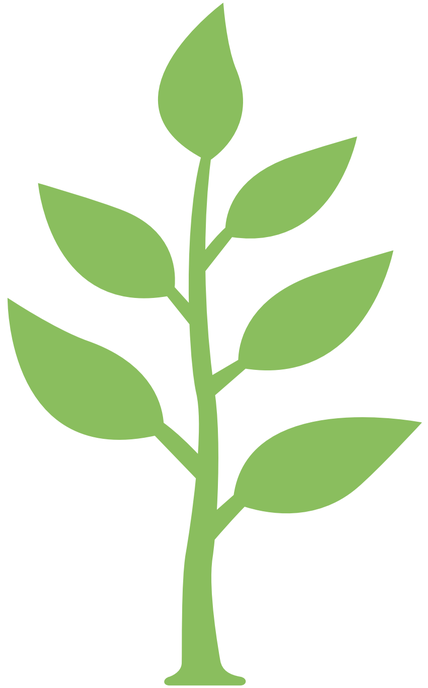
```
file: yamls/roots_v0.yml
========================
 1: model:
 2:   name: roots
 3:   language: python
 4:   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]

INFO:96257:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/roo
ts_v0.py 0.0 2.0 0.5
INFO:96257:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:96257:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
INFO:96257:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:96257:runner.run[374]:YggRunner(runner):                 init      0.000000
INFO:96257:runner.run[374]:YggRunner(runner):         load drivers      0.279398
INFO:96257:runner.run[374]:YggRunner(runner):        start drivers      0.039544
INFO:96257:runner.run[374]:YggRunner(runner):            run models      0.103028
INFO:96257:runner.run[374]:YggRunner(runner):               at exit      0.000614
INFO:96257:runner.run[376]:YggRunner(runner): ======================================
INFO:96257:runner.run[377]:YggRunner(runner):                 Total      0.422584
```

In [4]:
```
import matplotlib.pyplot as plt
filename_masses = 'output/masses.pkl'
with open(filename_masses, 'rb') as fd:
    masses = pickle.load(fd)
plt.plot(masses['times'], masses['masses'])
```

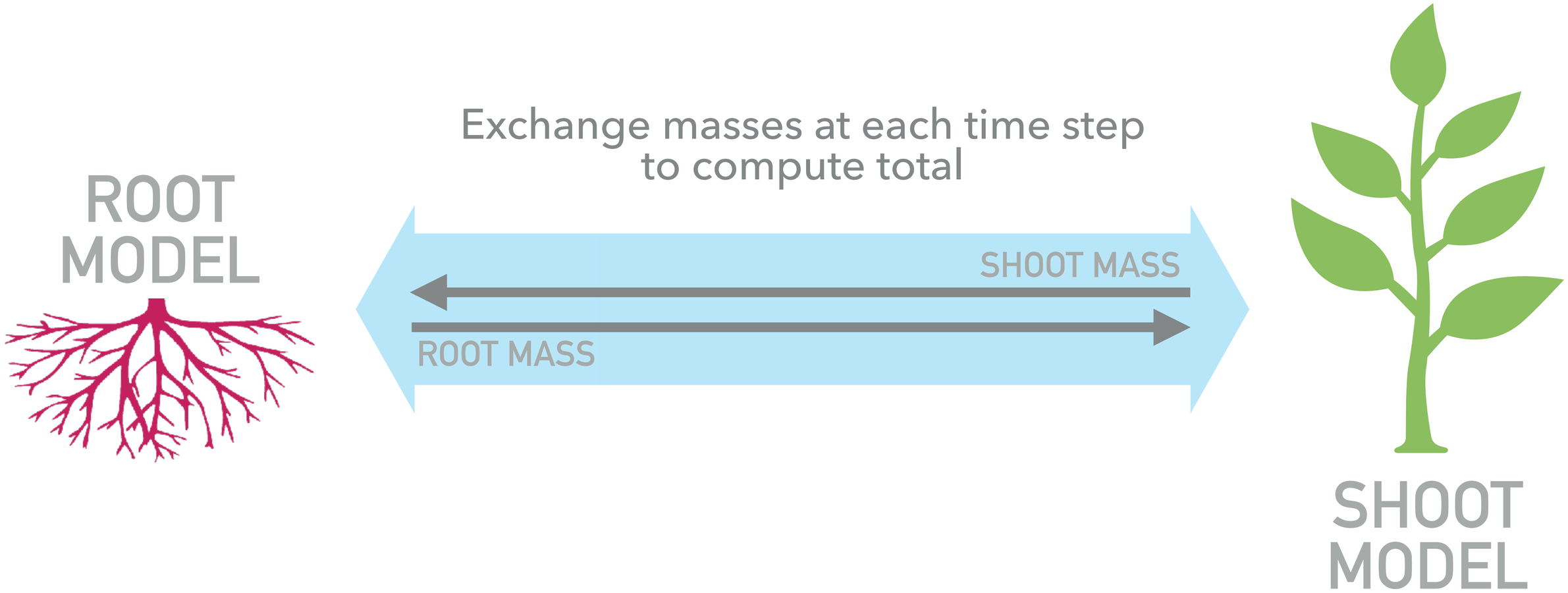
Out[4]:
```
[<matplotlib.lines.Line2D at 0x147647e48>]
```



ROOT MODEL

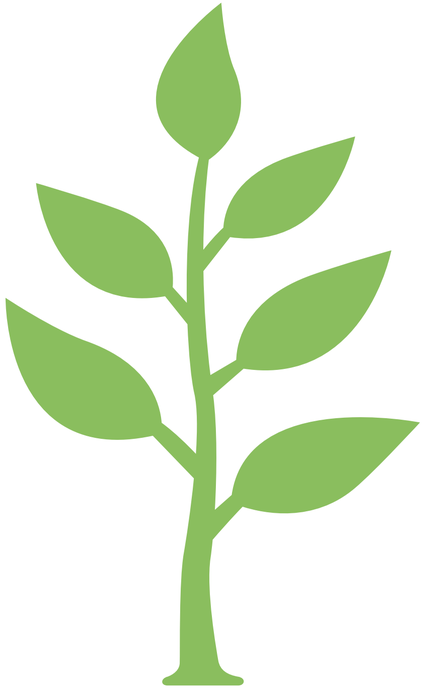


SHOOT MODEL


ROOT MODEL
Exchange masses at each time step to compute total
SHOOT MASS
ROOT MASS
SHOOT MODEL

Exchange masses at each time step to compute total

ROOT MODEL



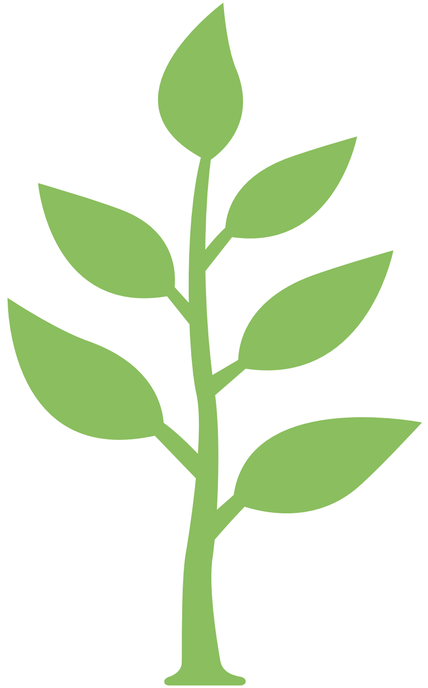
SHOOT MODEL

Models have different time steps so how do they exchange data?

0.5 DAYS

6 HRS

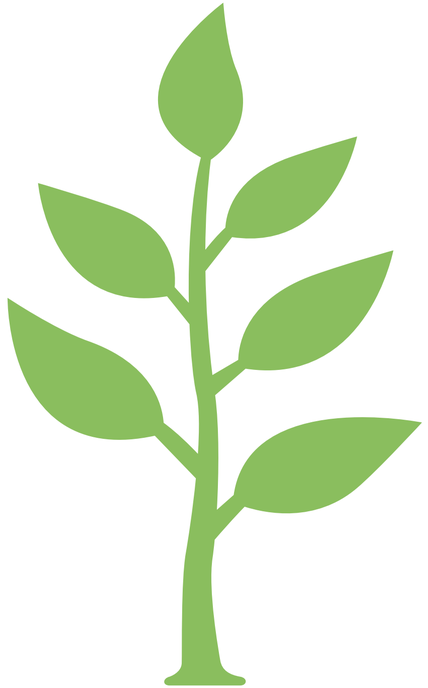
ROOT MODEL



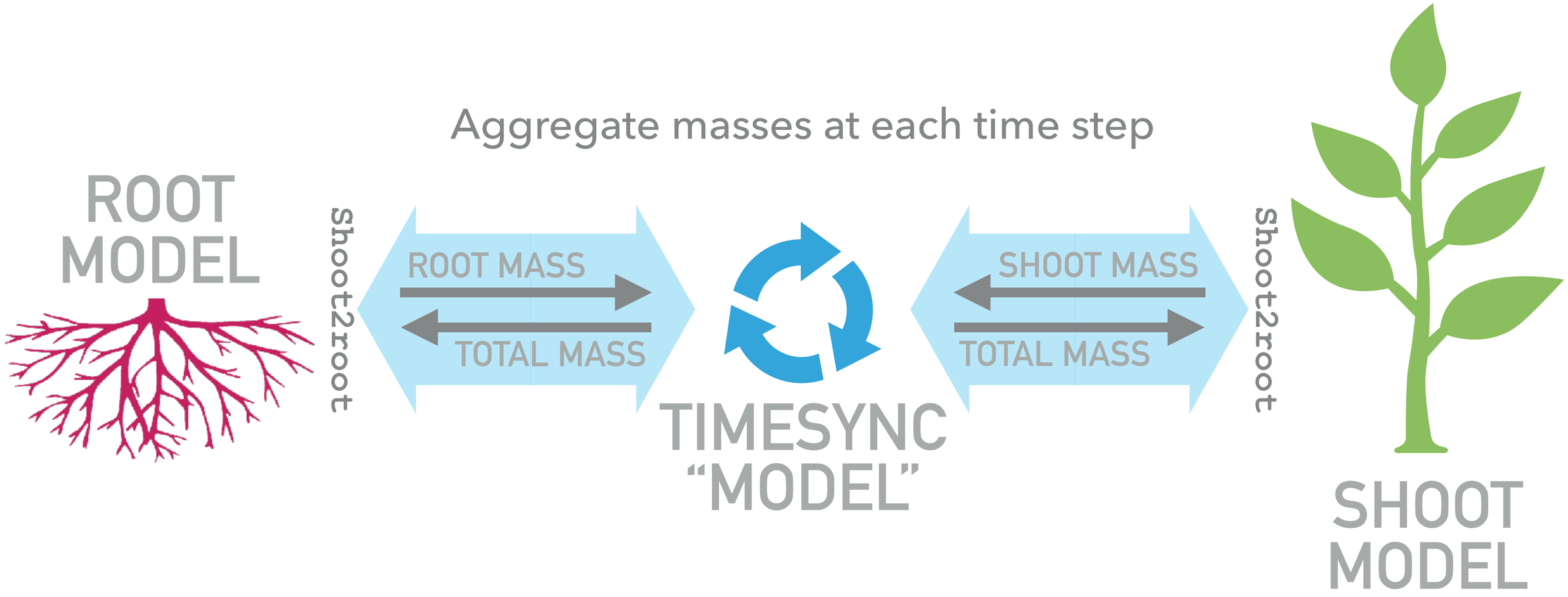
SHOOT MODEL

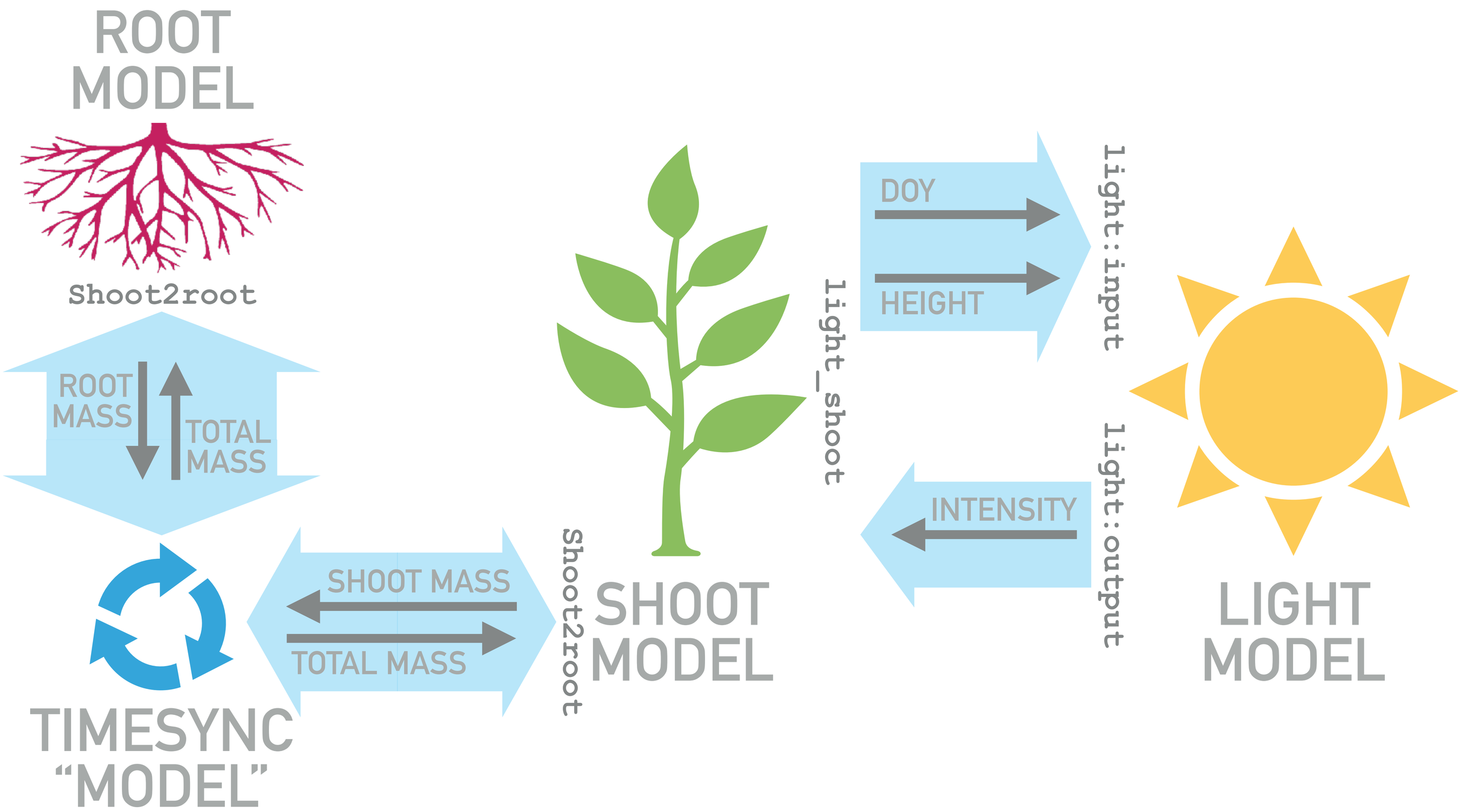
ROOT MODEL

TIMESYNC "MODEL"

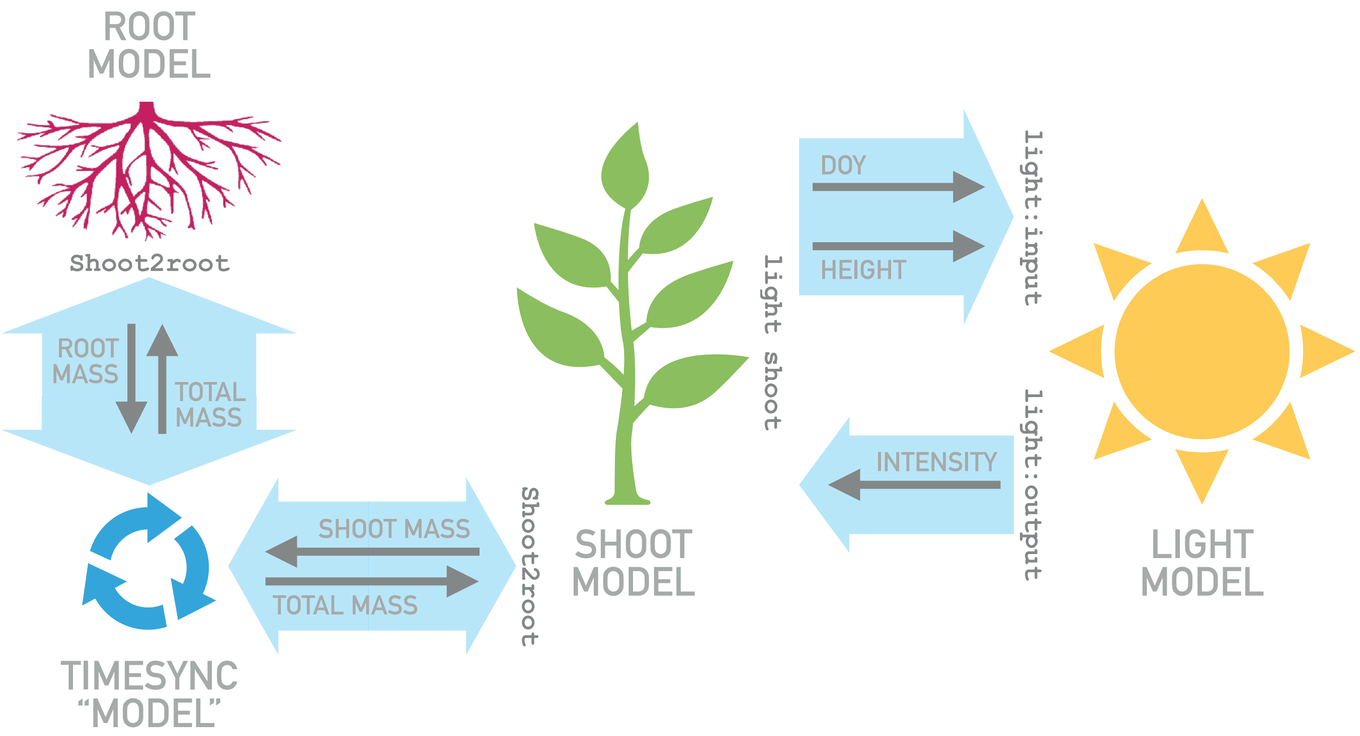


SHOOT MODEL


Aggregate masses at each time step
ROOT MODEL
shoot2root
ROOT MASS
TOTAL MASS
TIMESYNC
"MODEL"
SHOOT MASS
TOTAL MASS
shoot2root
SHOOT MODEL

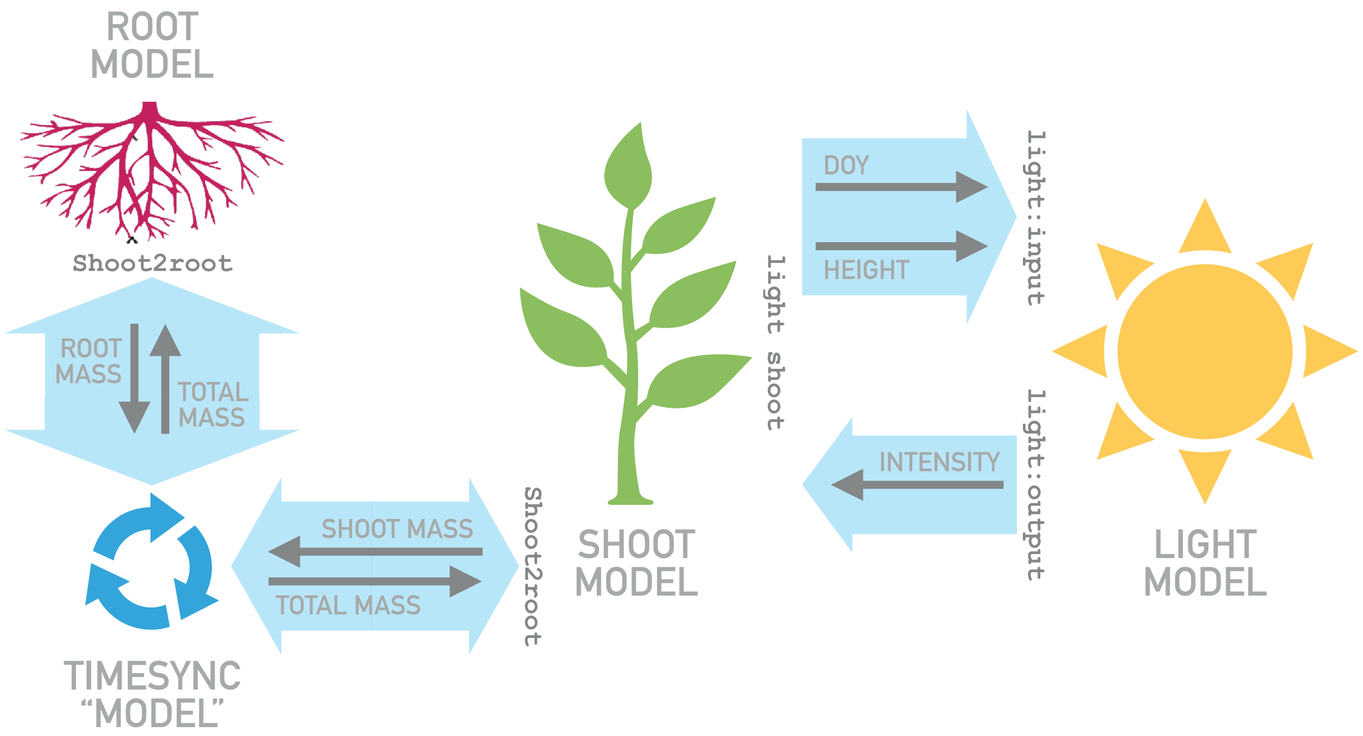

ROOT MODEL
Shoot2root
ROOT MASS
TOTAL MASS
TIMESYNC "MODEL"
SHOOT MASS
TOTAL MASS
Shoot2root
SHOOT MODEL
light_shoot
DOY
HEIGHT
light:input
INTENSITY
light:output
LIGHT MODEL

```
In [5]: tools.display_source_diff('yamls/roots_v0.yml', 'yamls/roots_v1.yml', number_lines=True)
        tools.display_source_diff('yamls/shoot_v2.yml', 'yamls/shoot_v3.yml', number_lines=True)
```



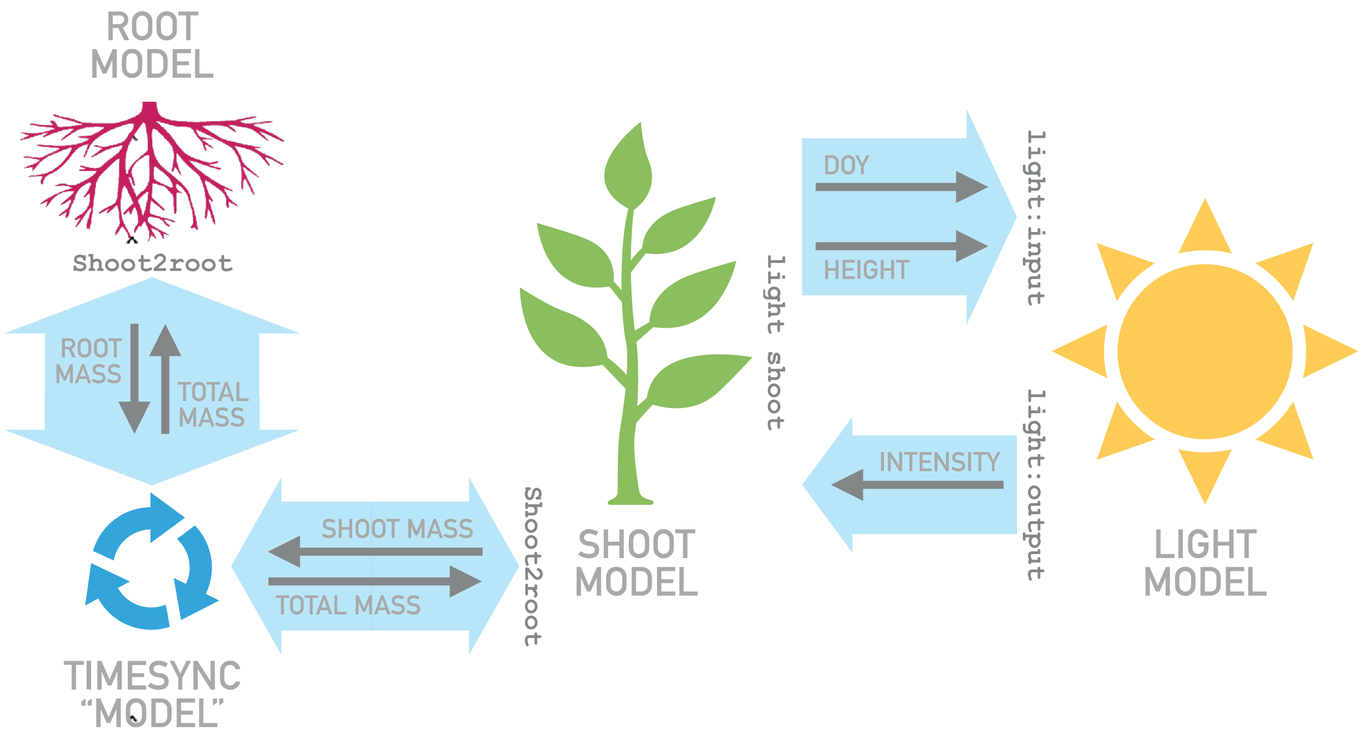
```
In [5]: tools.display_source_diff('yamls/roots_v0.yml', 'yamls/roots_v1.yml', number_lines=True)
        tools.display_source_diff('yamls/shoot_v2.yml', 'yamls/shoot_v3.yml', number_lines=True)
```

```
file1: yamls/roots_v0.yml
file2: yamls/roots_v1.yml
=========================
 1:    model:
 2:      name: roots
 3:      language: python
    -    args: [../models/roots_v0.py, 0.0, 2.0, 0.5]
    ?

 4: +    args: [../models/roots_v1.py, 0.0, 2.0, 0.5]
    ?

 5: +    timesync: shoot2root
```



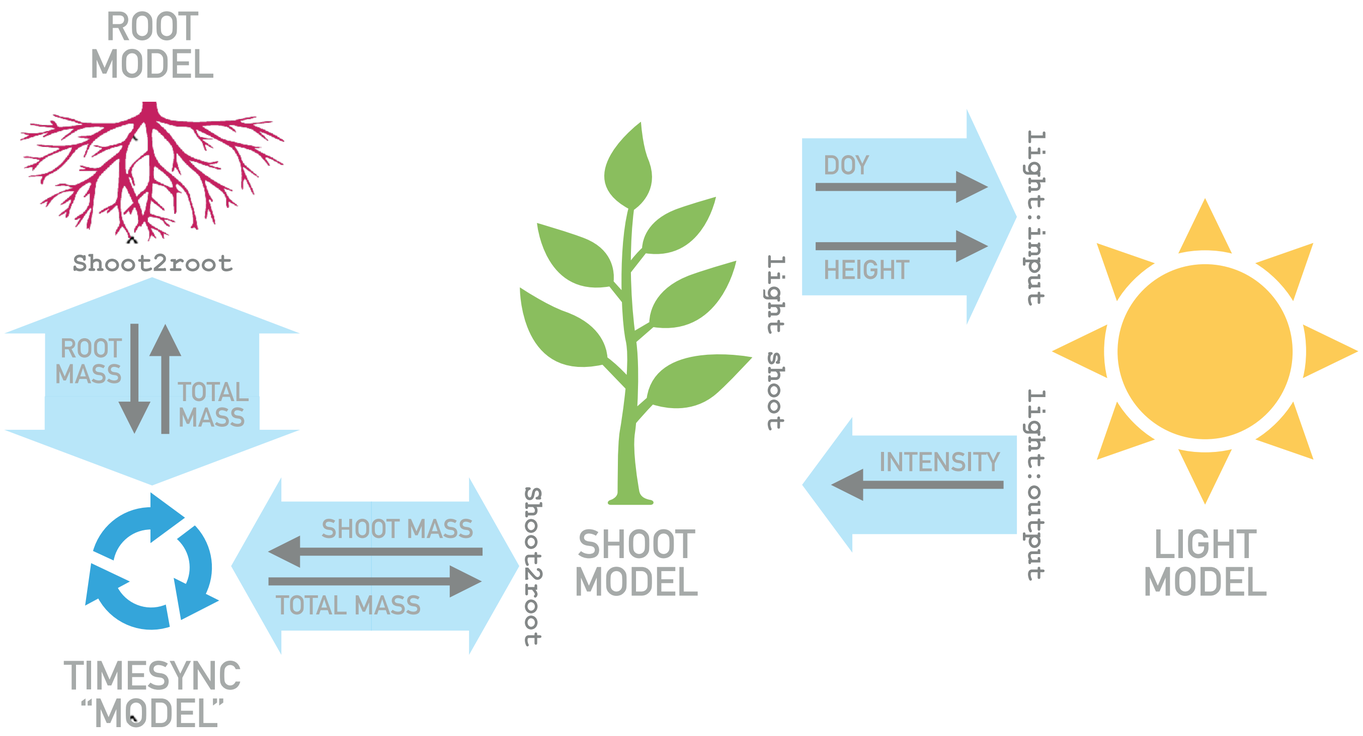
In [5]:
```
tools.display_source_diff('yamls/roots_v0.yml', 'yamls/roots_v1.yml', number_lines=True)
tools.display_source_diff('yamls/shoot_v2.yml', 'yamls/shoot_v3.yml', number_lines=True)
```

```
file1: yamls/roots_v0.yml
file2: yamls/roots_v1.yml
=========================
 1:   model:
 2:     name: roots
 3:     language: python
    -   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]
    ?

 4: +   args: [../models/roots_v1.py, 0.0, 2.0, 0.5]
    ?

 5: +   timesync: shoot2root

file1: yamls/shoot_v2.yml
file2: yamls/shoot_v3.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?

 4: +   args: [../models/shoot_v3.py, 0.0, 48.0, 6.0]
    ?

 5:     client_of: light
 6: +   timesync: shoot2root
```



In [5]:
```
tools.display_source_diff('yamls/roots_v0.yml', 'yamls/roots_v1.yml', number_lines=True)
tools.display_source_diff('yamls/shoot_v2.yml', 'yamls/shoot_v3.yml', number_lines=True)
```

```
file1: yamls/roots_v0.yml
file2: yamls/roots_v1.yml
=========================
 1:   model:
 2:     name: roots
 3:     language: python
    -   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]
    ?

 4: +   args: [../models/roots_v1.py, 0.0, 2.0, 0.5]
    ?

 5: +   timesync: shoot2root

file1: yamls/shoot_v2.yml
file2: yamls/shoot_v3.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?

 4: +   args: [../models/shoot_v3.py, 0.0, 48.0, 6.0]
    ?

 5:     client_of: light
 6: +   timesync: shoot2root
```

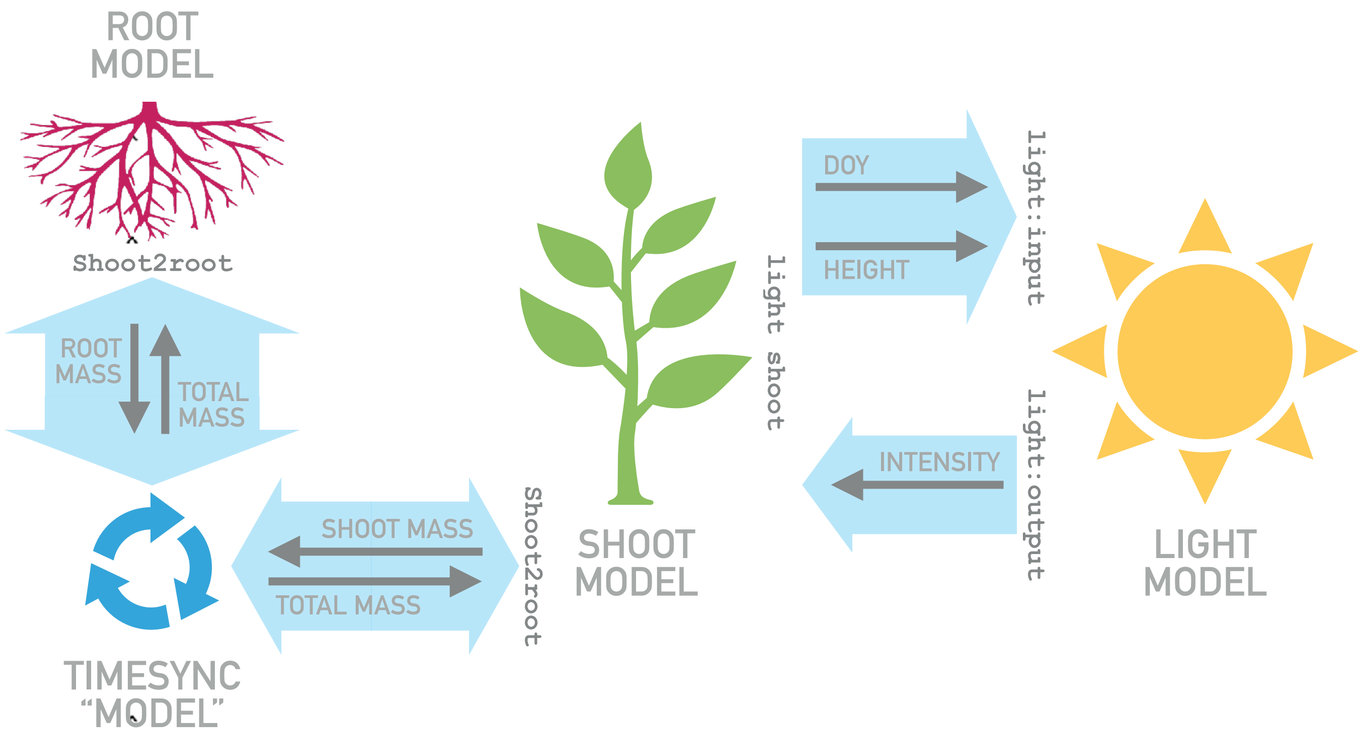


In [6]:
```
tools.display_source('yamls/timesync.yml', number_lines=True)
```

```
file: yamls/timesync.yml
========================
 1: model:
 2:   name: shoot2root
 3:   language: timesync
 4:   aggregation: sum
```

```
In [5]: tools.display_source_diff('yamls/roots_v0.yml', 'yamls/roots_v1.yml', number_lines=True)
        tools.display_source_diff('yamls/shoot_v2.yml', 'yamls/shoot_v3.yml', number_lines=True)
```

```
file1: yamls/roots_v0.yml
file2: yamls/roots_v1.yml
=========================
 1:   model:
 2:     name: roots
 3:     language: python
    -   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]
    ?

 4: +   args: [../models/roots_v1.py, 0.0, 2.0, 0.5]
    ?

 5: +   timesync: shoot2root

file1: yamls/shoot_v2.yml
file2: yamls/shoot_v3.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?

 4: +   args: [../models/shoot_v3.py, 0.0, 48.0, 6.0]
    ?

 5:     client_of: light
 6: +   timesync: shoot2root
```



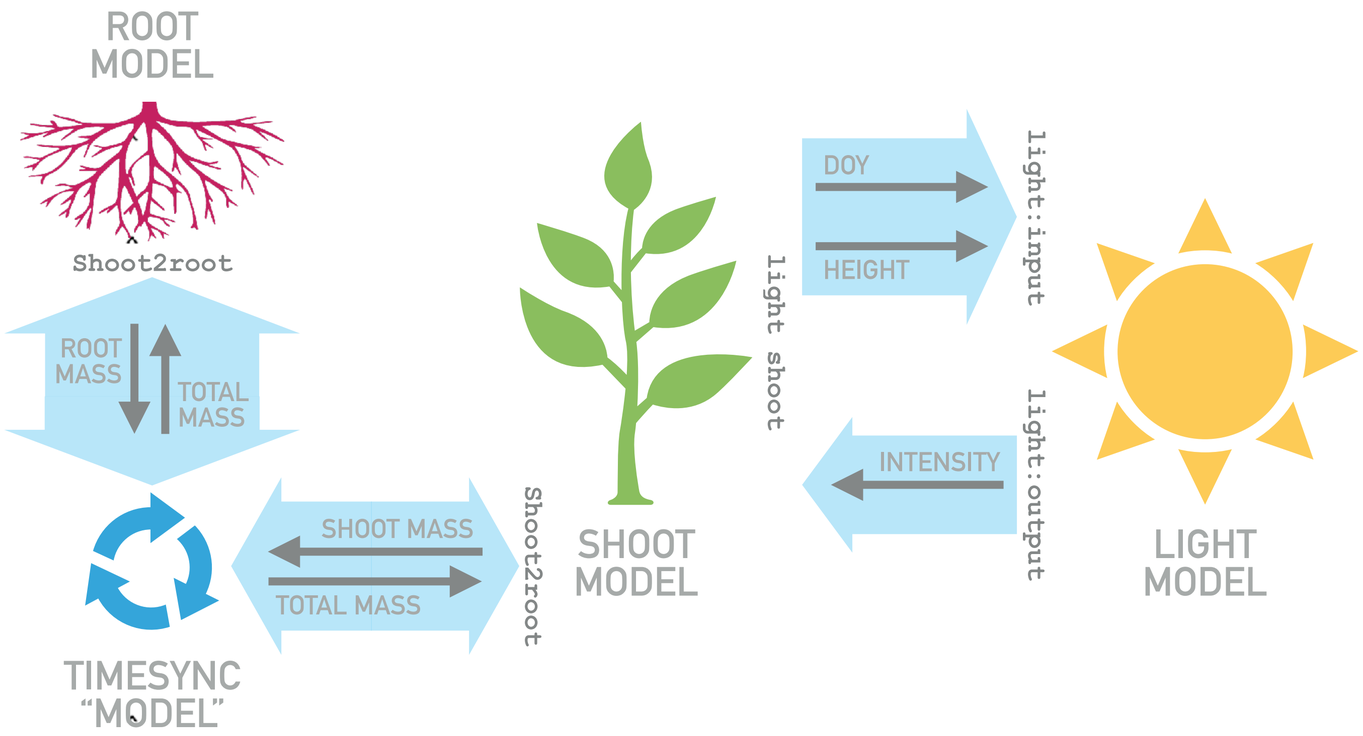

```
In [6]: tools.display_source('yamls/timesync.yml', number_lines=True)
```

```
file: yamls/timesync.yml
========================
 1: model:
 2:   name: shoot2root
 3:   language: timesync
 4:   aggregation: sum
```

Match name of time sync model "`shoot2root`"

```
In [5]: tools.display_source_diff('yamls/roots_v0.yml', 'yamls/roots_v1.yml', number_lines=True)
        tools.display_source_diff('yamls/shoot_v2.yml', 'yamls/shoot_v3.yml', number_lines=True)
```

```
file1: yamls/roots_v0.yml
file2: yamls/roots_v1.yml
=========================
 1:   model:
 2:     name: roots
 3:     language: python
    -   args: [../models/roots_v0.py, 0.0, 2.0, 0.5]
    ?

 4: +   args: [../models/roots_v1.py, 0.0, 2.0, 0.5]
    ?

 5: +   timesync: shoot2root

file1: yamls/shoot_v2.yml
file2: yamls/shoot_v3.yml
=========================
 1:   model:
 2:     name: shoot
 3:     language: python
    -   args: [../models/shoot_v2.py, 0.0, 48.0, 6.0]
    ?

 4: +   args: [../models/shoot_v3.py, 0.0, 48.0, 6.0]
    ?

 5:     client_of: light
 6: +   timesync: shoot2root
```




```
In [6]: tools.display_source('yamls/timesync.yml', number_lines=True)
```

```
file: yamls/timesync.yml
========================
 1: model:
 2:   name: shoot2root
 3:   language: timesync
 4:   aggregation: sum
```

"aggregation" determines how values should be combined

In [7]:
```
tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

In [7]:
```
tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/roots_v0.py
file2: models/roots_v1.py
==========================
23: + # Check if model is running as a part of an yggdrasil integration
24: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
25: +
26: + # If the model is running as part of an yggdrasil integration, import
27: + # the relevant yggdrasil routines and use the interface routine to
28: + # complete the connection defined in the YAML
29: + if with_yggdrasil:
30: +     from yggdrasil import units
31: +     from yggdrasil.languages.Python.YggInterface import YggTimesync
32: +     shoot2root = YggTimesync('shoot2root')
33: +
34:   # Continue simulation until time limit is reached
35:   while t <= tmax:
36:
    -     # Compute the scale factor
37: +     # If running as part an yggdrasil integration, send the time and
38: +     # mass to the timesync channel and then updated the mass based on
39: +     # the returned state
40: +     if with_yggdrasil:
41: +         root_state = {'mass': units.add_units(mass, 'kg')}
42: +         flag, total_state = shoot2root.call(units.add_units(t, 'days'),
43: +                                             root_state)
44: +         if not flag:
45: +             raise Exception("Error performing time-step synchronization "
46: +                             "with shoot model.")
47: +
48: +         # Compute the scale factor using total mass, stripping units
49: +         # of the result to allow use with original code
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++
51: +         scale = units.get_data(
52: +             units.convert_to(
53: +                 units.add_units(0.05, 'days-1') * total_state['mass'],
54: +                 'kg/day'))
55: +
56: +     else:
57: +         # Compute the scale factor
58: +         # (pretend this is a biologically complex calculation)
    -     scale = 0.2
59: +         scale = 0.2
    ? ++++
```

```
In [7]: tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
        tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/roots_v0.py
file2: models/roots_v1.py
==========================
23: + # Check if model is running as a part of an yggdrasil integration
24: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
25: +
26: + # If the model is running as part of an yggdrasil integration, import
27: + # the relevant yggdrasil routines and use the interface routine to
28: + # complete the connection defined in the YAML
29: + if with_yggdrasil:
30: +     from yggdrasil import units
31: +     from yggdrasil.languages.Python.YggInterface import YggTimesync
32: +     shoot2root = YggTimesync('shoot2root')
33: +
34:   # Continue simulation until time limit is reached
35:   while t <= tmax:
36:
    -     # Compute the scale factor
37: +     # If running as part an yggdrasil integration, send the time and
38: +     # mass to the timesync channel and then updated the mass based on
39: +     # the returned state
40: +     if with_yggdrasil:
41: +         root_state = {'mass': units.add_units(mass, 'kg')}
42: +         flag, total_state = shoot2root.call(units.add_units(t, 'days'),
43: +                                             root_state)
44: +         if not flag:
45: +             raise Exception("Error performing time-step synchronization "
46: +                             "with shoot model.")
47: +
48: +         # Compute the scale factor using total mass, stripping units
49: +         # of the result to allow use with original code
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++
51: +         scale = units.get_data(
52: +             units.convert_to(
53: +                 units.add_units(0.05, 'days-1') * total_state['mass'],
54: +                 'kg/day'))
55: +
56: +     else:
57: +         # Compute the scale factor
58: +         # (pretend this is a biologically complex calculation)
    -     scale = 0.2
59: +         scale = 0.2
    ? ++++
```

Import yggdrasil functions and connect to the time sync channel listed in the YAML.

```
In [7]: tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
        tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/roots_v0.py
file2: models/roots_v1.py
==========================
23: + # Check if model is running as a part of an yggdrasil integration
24: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
25: +
26: + # If the model is running as part of an yggdrasil integration, import
27: + # the relevant yggdrasil routines and use the interface routine to
28: + # complete the connection defined in the YAML
29: + if with_yggdrasil:
30: +     from yggdrasil import units
31: +     from yggdrasil.languages.Python.YggInterface import YggTimesync
32: +     shoot2root = YggTimesync('shoot2root')
33: +
34:   # Continue simulation until time limit is reached
35:   while t <= tmax:
36:
    -     # Compute the scale factor
37: +     # If running as part an yggdrasil integration, send the time and
38: +     # mass to the timesync channel and then updated the mass based on
39: +     # the returned state
40: +     if with_yggdrasil:
41: +         root_state = {'mass': units.add_units(mass, 'kg')}
42: +         flag, total_state = shoot2root.call(units.add_units(t, 'days'),
43: +                                             root_state)
44: +         if not flag:
45: +             raise Exception("Error performing time-step synchronization "
46: +                             "with shoot model.")
47: +
48: +         # Compute the scale factor using total mass, stripping units
49: +         # of the result to allow use with original code
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++
51: +         scale = units.get_data(
52: +             units.convert_to(
53: +                 units.add_units(0.05, 'days-1') * total_state['mass'],
54: +                 'kg/day'))
55: +
56: +     else:
57: +         # Compute the scale factor
58: +         # (pretend this is a biologically complex calculation)
    -     scale = 0.2
59: +         scale = 0.2
    ? ++++
```

Send root state to the time sync model and receive the total state back

```
In [7]: tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
        tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/roots_v0.py
file2: models/roots_v1.py
==========================
23: + # Check if model is running as a part of an yggdrasil integration
24: + with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
25: +
26: + # If the model is running as part of an yggdrasil integration, import
27: + # the relevant yggdrasil routines and use the interface routine to
28: + # complete the connection defined in the YAML
29: + if with_yggdrasil:
30: +     from yggdrasil import units
31: +     from yggdrasil.languages.Python.YggInterface import YggTimesync
32: +     shoot2root = YggTimesync('shoot2root')
33: +
34:   # Continue simulation until time limit is reached
35:   while t <= tmax:
36:
    -     # Compute the scale factor
37: +     # If running as part an yggdrasil integration, send the time and
38: +     # mass to the timesync channel and then updated the mass based on
39: +     # the returned state
40: +     if with_yggdrasil:
41: +         root_state = {'mass': units.add_units(mass, 'kg')}
42: +         flag, total_state = shoot2root.call(units.add_units(t, 'days'),
43: +                                             root_state)
44: +         if not flag:
45: +             raise Exception("Error performing time-step synchronization "
46: +                             "with shoot model.")
47: +
48: +         # Compute the scale factor using total mass, stripping units
49: +         # of the result to allow use with original code
    -     # (pretend this is a biologically complex calculation)
50: +         # (pretend this is a biologically complex calculation)
    ? ++++
51: +         scale = units.get_data(
52: +             units.convert_to(
53: +                 units.add_units(0.05, 'days-1') * total_state['mass'],
54: +                 'kg/day'))
55: +
56: +     else:
57: +         # Compute the scale factor
58: +         # (pretend this is a biologically complex calculation)
    -     scale = 0.2
59: +         scale = 0.2
    ? ++++
```

Compute the scale using the total state when run with yggdrasil

In [7]:
```
tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/shoot_v2.py
file2: models/shoot_v3.py
==========================
...

31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggRpcClient
33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient, YggTimesync
    ?
++++++++++++

34:       light_rpc = YggRpcClient('light_shoot')
35: +     shoot2root = YggTimesync('shoot2root')
36:
37:   # Continue simulation until time limit is reached
38:   while t <= tmax:
39:
40:       # If running as part an yggdrasil integration, send the time and
    -     # maximum height of the mesh to the height channel with units
41: +     # maximum height of the mesh to the height channel with units and
    ?                                                                   ++++

42: +     # send the current mass and tiem to the timesync channel
43:       if with_yggdrasil:
44: +         shoot_state = {'mass': units.add_units(mass, 'g')}
45: +         flag, total_state = shoot2root.call(units.add_units(t, 'hrs'),
46: +                                             shoot_state)
47: +         if not flag:
48: +             raise Exception("Error performing time-step synchronization "
49: +                             "with root model.")
50: +

...

60:           scale = units.get_data(
    -             units.add_units(mass, 'g') * intensity /
    ?             ^^^  ^ ------------------------------                         ^

61: +                 total_state['mass'] * intensity /
    ?             ^^^^ +++++ ^^^^                             ^^^^

    -             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
62: +                 units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
    ? ++++


...
```

```
In [7]: tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
        tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/shoot_v2.py
file2: models/shoot_v3.py
==========================
...

31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggRpcClient
33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient, YggTimesync
    ?
++++++++++++

34:       light_rpc = YggRpcClient('light_shoot')
35: +     shoot2root = YggTimesync('shoot2root')
36:
37:   # Continue simulation until time limit is reached
38:   while t <= tmax:
39:
40:       # If running as part an yggdrasil integration, send the time and
    -     # maximum height of the mesh to the height channel with units
41: +     # maximum height of the mesh to the height channel with units and
    ?                                                                  ++++

42: +     # send the current mass and tiem to the timesync channel
43:       if with_yggdrasil:
44: +         shoot_state = {'mass': units.add_units(mass, 'g')}
45: +         flag, total_state = shoot2root.call(units.add_units(t, 'hrs'),
46: +                                             shoot_state)
47: +         if not flag:
48: +             raise Exception("Error performing time-step synchronization "
49: +                             "with root model.")
50: +

...

60:           scale = units.get_data(
    -             units.add_units(mass, 'g') * intensity /
    ?             ^^^   ^ ------------------------------                 ^

61: +                 total_state['mass'] * intensity /
    ?             ^^^^ +++++ ^^^^                           ^^^^

    -             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
62: +                 units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
    ? ++++

...
```

Import the time sync interface & connect to the "`shoot2root`" channel

```
In [7]: tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
        tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/shoot_v2.py
file2: models/shoot_v3.py
==========================
...

31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggRpcClient
33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient, YggTimesync
    ?
++++++++++++

34:       light_rpc = YggRpcClient('light_shoot')
35: +     shoot2root = YggTimesync('shoot2root')
36:
37:   # Continue simulation until time limit is reached
38:   while t <= tmax:
39:
40:       # If running as part an yggdrasil integration, send the time and
    -     # maximum height of the mesh to the height channel with units
41: +     # maximum height of the mesh to the height channel with units and
    ?                                                                  ++++

42: +     # send the current mass and tiem to the timesync channel
43:       if with_yggdrasil:
44: +         shoot_state = {'mass': units.add_units(mass, 'g')}
45: +         flag, total_state = shoot2root.call(units.add_units(t, 'hrs'),
46: +                                             shoot_state)
47: +         if not flag:
48: +             raise Exception("Error performing time-step synchronization "
49: +                             "with root model.")
50: +

...

60:           scale = units.get_data(
    -             units.add_units(mass, 'g') * intensity /
    ?             ^^^  ^ ------------------------------                 ^

61: +                 total_state['mass'] * intensity /
    ?             ^^^^ +++++ ^^^^                           ^^^^

    -             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
62: +                 units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
    ? ++++

...
```

Send the shoot state to the time sync model & receive back the total state

```
In [7]: tools.display_source_diff('models/roots_v0.py', 'models/roots_v1.py', number_lines=True)
        tools.display_source_diff('models/shoot_v2.py', 'models/shoot_v3.py', number_lines=True)
```

```
file1: models/shoot_v2.py
file2: models/shoot_v3.py
==========================
...

31:   if with_yggdrasil:
32:       from yggdrasil import units
    -     from yggdrasil.languages.Python.YggInterface import YggRpcClient
33: +     from yggdrasil.languages.Python.YggInterface import YggRpcClient, YggTimesync
    ?
++++++++++++

34:       light_rpc = YggRpcClient('light_shoot')
35: +     shoot2root = YggTimesync('shoot2root')
36:
37:   # Continue simulation until time limit is reached
38:   while t <= tmax:
39:
40:       # If running as part an yggdrasil integration, send the time and
    -     # maximum height of the mesh to the height channel with units
41: +     # maximum height of the mesh to the height channel with units and
    ?                                                                  ++++

42: +     # send the current mass and tiem to the timesync channel
43:       if with_yggdrasil:
44: +         shoot_state = {'mass': units.add_units(mass, 'g')}
45: +         flag, total_state = shoot2root.call(units.add_units(t, 'hrs'),
46: +                                             shoot_state)
47: +         if not flag:
48: +             raise Exception("Error performing time-step synchronization "
49: +                             "with root model.")
50: +

...

60:           scale = units.get_data(
    -             units.add_units(mass, 'g') * intensity /
    ?             ^^^   ^ ------------------------------                 ^

61: +                 total_state['mass'] * intensity /
    ?             ^^^^ +++++ ^^^^                           ^^^^

    -             units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
62: +                 units.add_units(4.0e10, 'g*erg/(cm**2*s)'))
    ? ++++

...
```

Change the scale calculation to use the total state

In [8]:
```
run(['yamls/shoot_v3.yml', 'yamls/roots_v1.yml', 'yamls/light_v1_python.yml', 'yamls/timesync.yml'], production_run=Tru
```

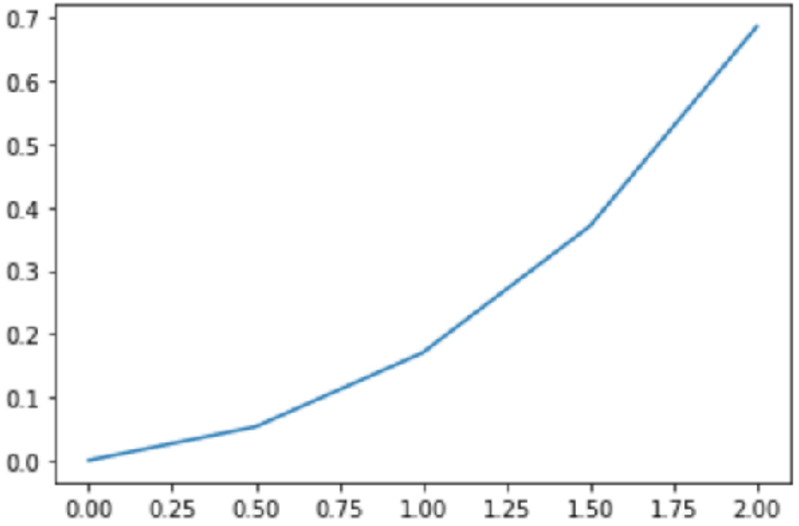
In [8]:
```
run(['yamls/shoot_v3.yml', 'yamls/roots_v1.yml', 'yamls/light_v1_python.yml', 'yamls/timesync.yml'], production_run=Tru
```

```
INFO:96257:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v3.py 0.0 48.0 6.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/roo
ts_v1.py 0.0 2.0 0.5
End of input from temp_doy.
INFO:96257:runner.waitModels[553]:YggRunner(runner): shoot finished running.
No more messages from model process.
INFO:96257:DSLModelDriver.after_loop[131]:TimeSyncModelDriver(shoot2root): returncode = 0
INFO:96257:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:96257:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:96257:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:96257:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:96257:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
INFO:96257:runner.waitModels[553]:YggRunner(runner): shoot2root finished running.
INFO:96257:runner.waitModels[559]:YggRunner(runner): shoot2root finished exiting.
INFO:96257:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:96257:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:96257:runner.run[374]:YggRunner(runner):         load drivers      0.051068
INFO:96257:runner.run[374]:YggRunner(runner):        start drivers      0.303378
INFO:96257:runner.run[374]:YggRunner(runner):           run models      19.234870
INFO:96257:runner.run[374]:YggRunner(runner):              at exit      0.117888
INFO:96257:runner.run[376]:YggRunner(runner): ========================================
INFO:96257:runner.run[377]:YggRunner(runner):                Total      19.707205
```

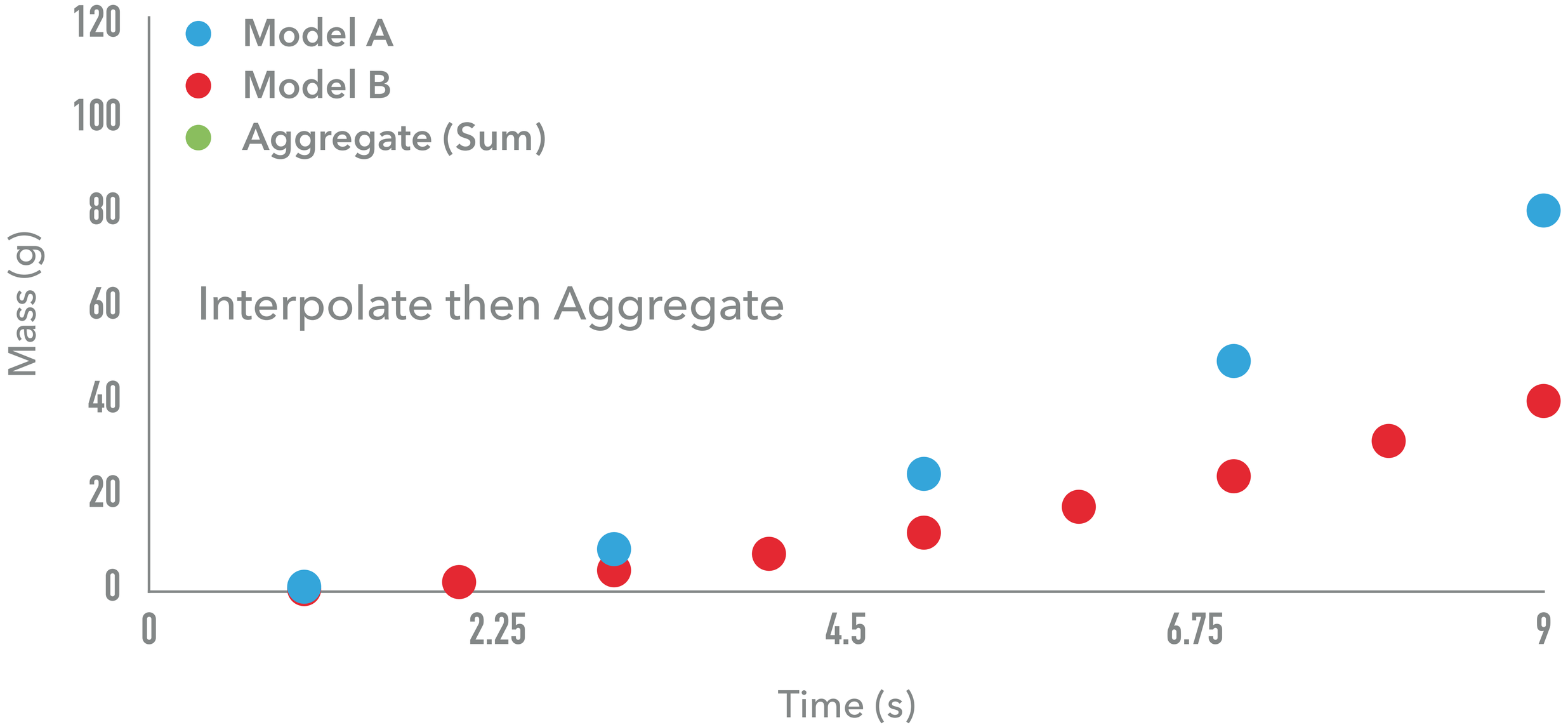
```
In [8]: run(['yamls/shoot_v3.yml', 'yamls/roots_v1.yml', 'yamls/light_v1_python.yml', 'yamls/timesync.yml'], production_run=Tru
```

```
INFO:96257:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-MacBook-Air.
local in namespace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/sho
ot_v3.py 0.0 48.0 6.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/roo
ts_v1.py 0.0 2.0 0.5
End of input from temp_doy.
INFO:96257:runner.waitModels[553]:YggRunner(runner): sho
No more messages from model process.
INFO:96257:DSLModelDriver.after_loop[131]:TimeSyncModelD
INFO:96257:runner.waitModels[559]:YggRunner(runner): sho
INFO:96257:runner.waitModels[553]:YggRunner(runner): lig
INFO:96257:runner.waitModels[559]:YggRunner(runner): lig
INFO:96257:runner.waitModels[553]:YggRunner(runner): roo
INFO:96257:runner.waitModels[559]:YggRunner(runner): roo
INFO:96257:runner.waitModels[553]:YggRunner(runner): sho
INFO:96257:runner.waitModels[559]:YggRunner(runner): sho
INFO:96257:runner.waitModels[573]:YggRunner(runner): All
INFO:96257:runner.run[374]:YggRunner(runner):
INFO:96257:runner.run[374]:YggRunner(runner):          lo
INFO:96257:runner.run[374]:YggRunner(runner):         sta
INFO:96257:runner.run[374]:YggRunner(runner):
INFO:96257:runner.run[374]:YggRunner(runner):
INFO:96257:runner.run[376]:YggRunner(runner): ==========
INFO:96257:runner.run[377]:YggRunner(runner):
```

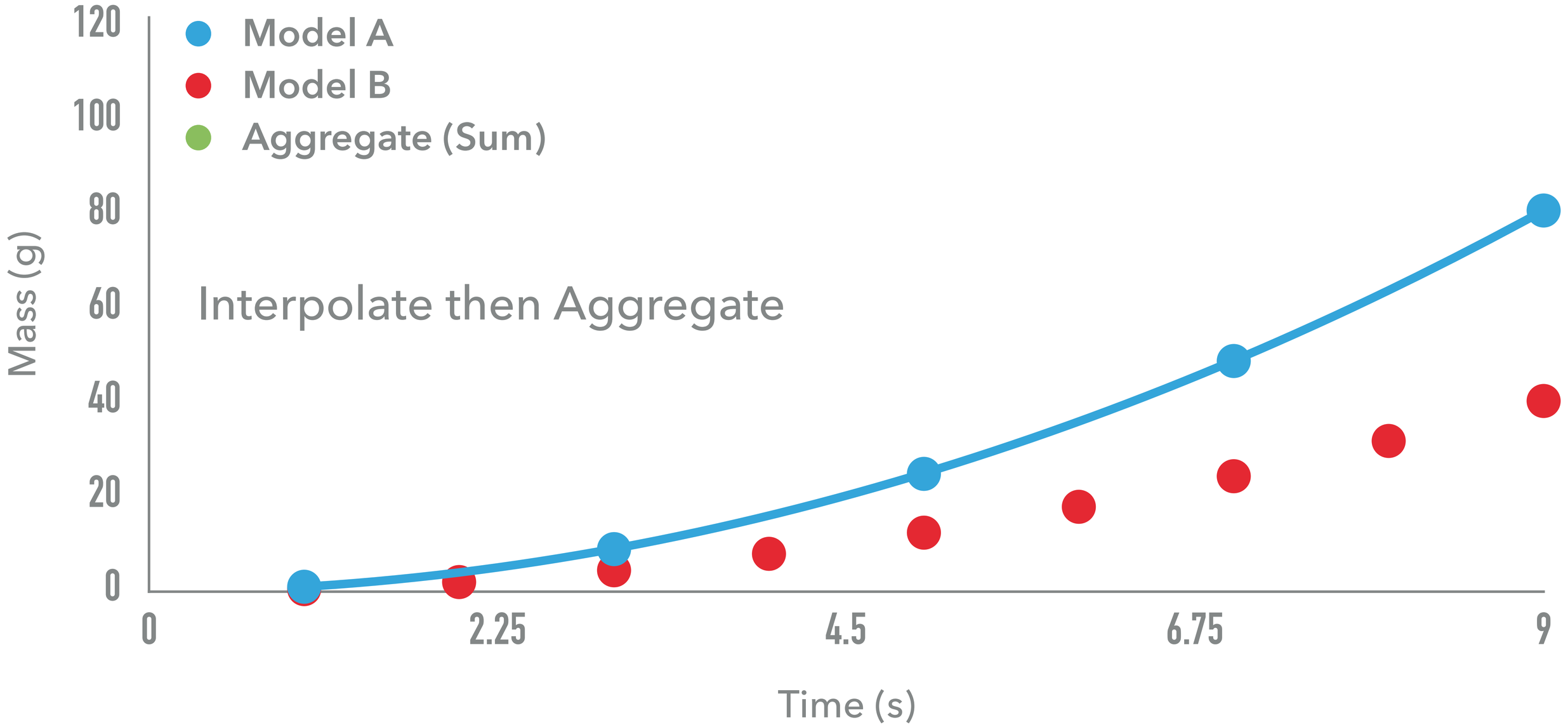
```
In [9]: import matplotlib.pyplot as plt
        filename_masses = 'output/masses.pkl'
        with open(filename_masses, 'rb') as fd:
            masses = pickle.load(fd)
        plt.plot(masses['times'], masses['masses'])

Out[9]: [<matplotlib.lines.Line2D at 0x147d40048>]
```



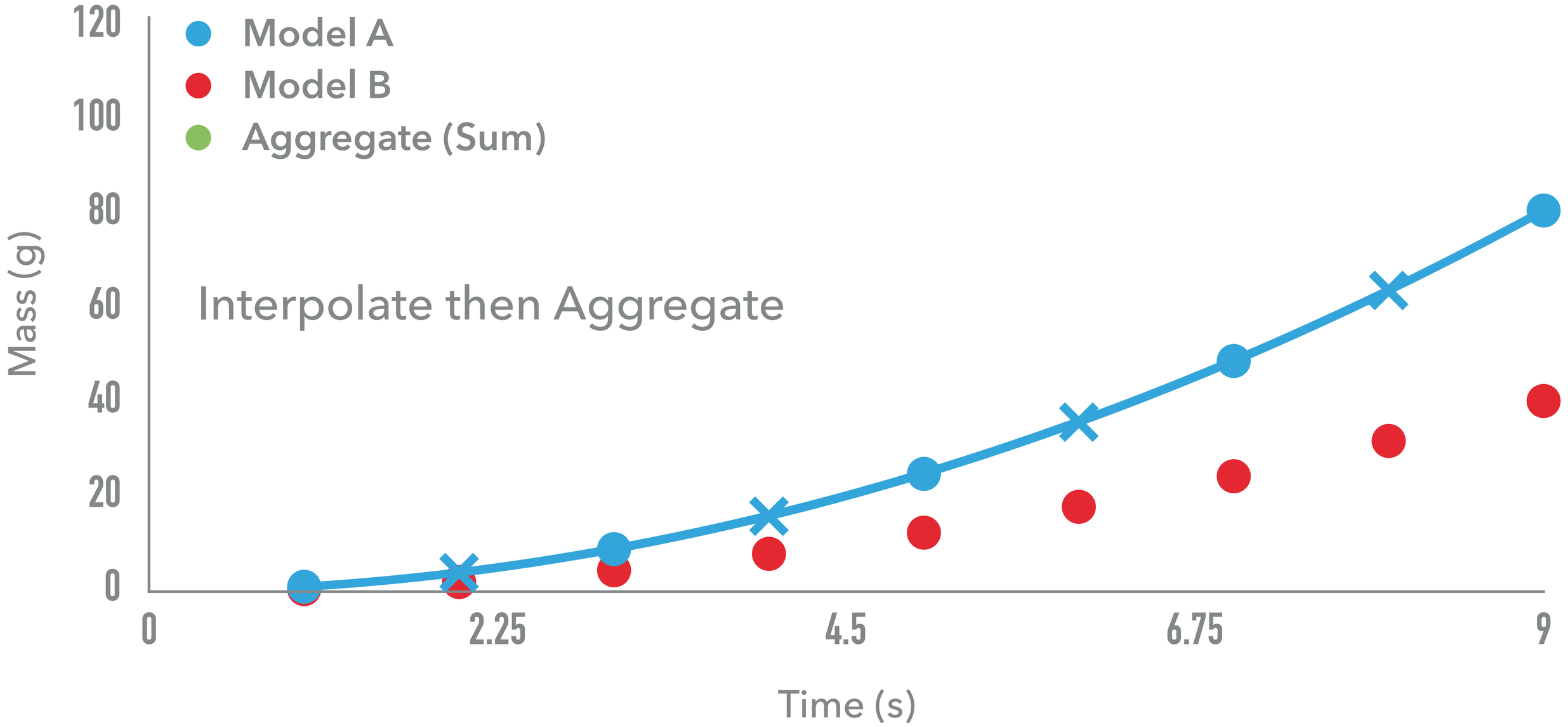
AGGREGATION OF TIME STEPS



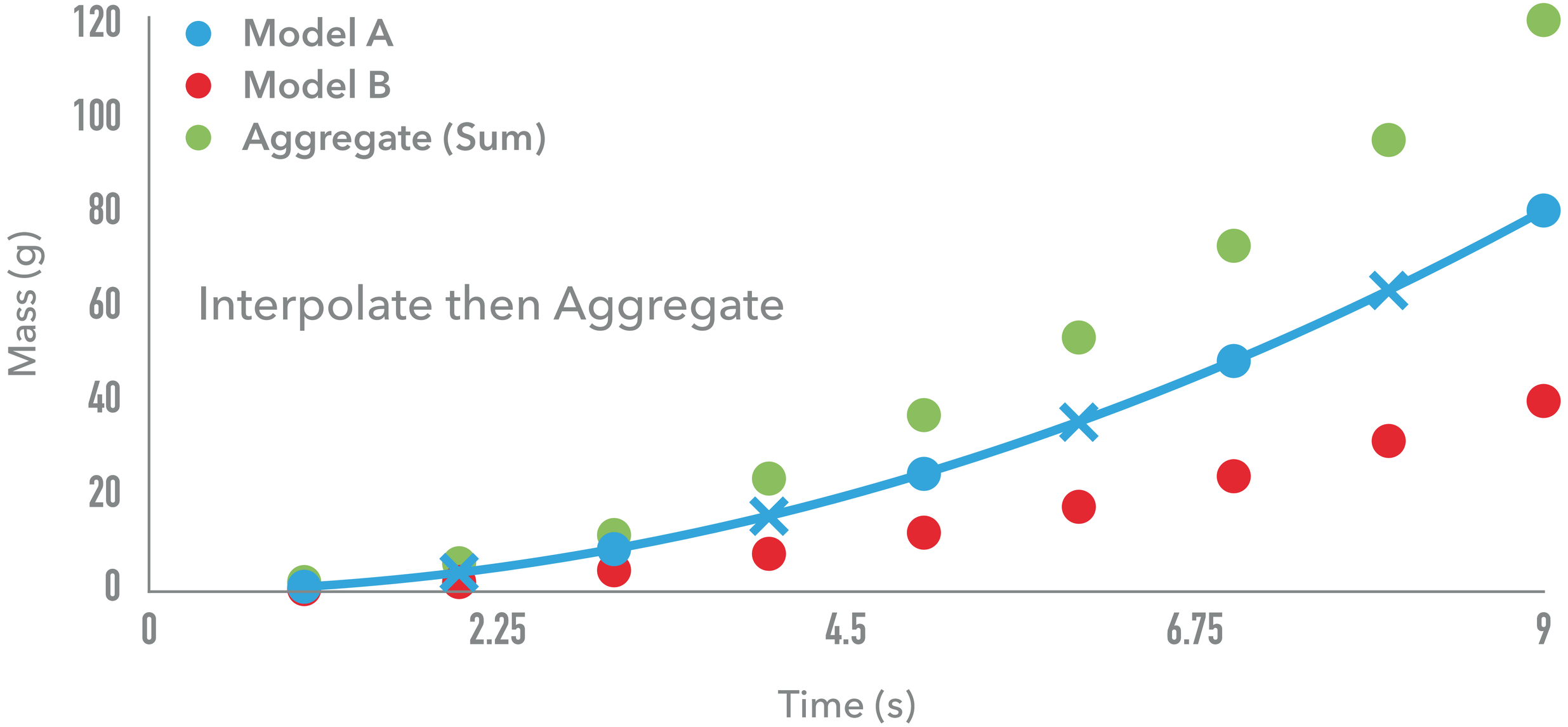
AGGREGATION OF TIME STEPS



AGGREGATION OF TIME STEPS



AGGREGATION OF TIME STEPS



Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.
2. Run the `models/microbe.py` model in a timesync integration with the root model and the shoot model.
3. Plot the masses for the root, shoot, and microbe models as a function of time.

TEST YOUR KNOWLEDGE (15 MIN)

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run as run

# Part 1: integrate microbe w/ root model
tools.display_source_diff('models/microbe.py',
                          'solutions/tyk6/models/microbe.py', number_lines=True)
tools.display_source('solutions/tyk6/yamls/microbe.yml', number_lines=True)
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml'],
    production_run=True)

# Part 2: integrate microbe w/ root & shoot model
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml',
     'solutions/tyk6/yamls/shoot_v3.yml', 'solutions/tyk6/yamls/light_v1_python.yml'],
    production_run=True)
```

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run as run

# Part 1: integrate microbe w/ root model
tools.display_source_diff('models/microbe.py',
                          'solutions/tyk6/models/microbe.py', number_lines=True)
tools.display_source('solutions/tyk6/yamls/microbe.yml', number_lines=True)
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml'],
    production_run=True)

# Part 2: integrate microbe w/ root & shoot model
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml',
     'solutions/tyk6/yamls/shoot_v3.yml', 'solutions/tyk6/yamls/light_v1_python.yml'],
    production_run=True)
```

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run as run

# Part 1: integrate microbe w/ root model
tools.display_source_diff('models/microbe.py',
                          'solutions/tyk6/models/microbe.py', number_lines=True)
tools.display_source('solutions/tyk6/yamls/microbe.yml', number_lines=True)
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml'],
    production_run=True)

# Part 2: integrate microbe w/ root & shoot model
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml',
     'solutions/tyk6/yamls/shoot_v3.yml', 'solutions/tyk6/yamls/light_v1_python.yml'],
    production_run=True)
```

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

```
...
+ # Check if model is running as a part of an yggdrasil integration
+ with_yggdrasil = os.environ.get('YGG_SUBPROCESS', False)
+
+ # If the model is running as part of an yggdrasil integration, import
+ # the relevant yggdrasil routines and use the interface routine to
+ # complete the connection defined in the YAML
+ if with_yggdrasil:
+     from yggdrasil import units
+     from yggdrasil.languages.Python.YggInterface import YggTimesync
+     shoot2microbe = YggTimesync('shoot2root')
+
  # Continue simulation until time limit is reached
  while t <= tmax:

-     # Compute the scale factor
+     # If running as part an yggdrasil integration, send the time and
+     # mass to the timesync channel and then updated the mass based on
+     # the returned state
+     if with_yggdrasil:
+         microbe_state = {'mass': units.add_units(mass, 'g')}
+         flag, total_state = shoot2microbe.call(units.add_units(t, 'days'),
+                                                microbe_state)
+         if not flag:
+             raise Exception("Error performing time-step synchronization "
+                             "with shoot model.")
...
```

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

```
...

51: +         scale = units.get_data(
52: +             units.convert_to(
53: +                 units.add_units(0.005, 'min-1') * total_state['mass'],
54: +                 'g/min'))
55: +
56: +     else:
57: +         # Compute the scale factor
58: +         # (pretend this is a biologically complex calculation)
    -     scale = 0.0007
59: +         scale = 0.0007
    ? ++++


...
```

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

```
file: solutions/tyk6/yamls/microbe.yml
=====================================
 1: model:
 2:   name: microbe
 3:   language: python
 4:   args: [../models/microbe.py, 0.0, 2880.0, 60.0]
 5:   timesync: shoot2root
```

Test your knowledge #6

1. Add the interface calls to the `models/microbe.py` model and write a YAML for it to allow the model to run in a timesync integration with the root model.

```
INFO:73601:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk6/models/roots_v1.py 0.0 2.0 0.5
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk6/models/microbe.py 0.0 2880.0 60.0
INFO:73601:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
No more messages from model process.
INFO:73601:DSLModelDriver.after_loop[131]:TimeSyncModelDriver(shoot2root): returncode = 0
INFO:73601:runner.waitModels[553]:YggRunner(runner): shoot2root finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): shoot2root finished exiting.
INFO:73601:runner.waitModels[553]:YggRunner(runner): microbe finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): microbe finished exiting.
INFO:73601:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:73601:runner.run[374]:YggRunner(runner):                 init      0.000013
INFO:73601:runner.run[374]:YggRunner(runner):         load drivers      0.496946
INFO:73601:runner.run[374]:YggRunner(runner):        start drivers      0.333115
INFO:73601:runner.run[374]:YggRunner(runner):           run models      48.798267
INFO:73601:runner.run[374]:YggRunner(runner):              at exit      0.062231
INFO:73601:runner.run[376]:YggRunner(runner): ========================================
INFO:73601:runner.run[377]:YggRunner(runner):                Total      49.690572
```

Test your knowledge #6

2. Run the `models/microbe.py` model in a timesync integration with the root model and the shoot model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run as run

# Part 1: integrate microbe w/ root model
tools.display_source_diff('models/microbe.py',
                          'solutions/tyk6/models/microbe.py', number_lines=True)
tools.display_source('solutions/tyk6/yamls/microbe.yml', number_lines=True)
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml'],
    production_run=True)

# Part 2: integrate microbe w/ root & shoot model
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml',
     'solutions/tyk6/yamls/shoot_v3.yml', 'solutions/tyk6/yamls/light_v1_python.yml'],
    production_run=True)
```

Test your knowledge #6

2. Run the `models/microbe.py` model in a timesync integration with the root model and the shoot model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run as run

# Part 1: integrate microbe w/ root model
tools.display_source_diff('models/microbe.py',
                          'solutions/tyk6/models/microbe.py', number_lines=True)
tools.display_source('solutions/tyk6/yamls/microbe.yml', number_lines=True)
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml'],
    production_run=True)

# Part 2: integrate microbe w/ root & shoot model
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml',
     'solutions/tyk6/yamls/shoot_v3.yml', 'solutions/tyk6/yamls/light_v1_python.yml'],
    production_run=True)
```

Don’t need to write anything, use the same YAMLs

Test your knowledge #6

2. Run the `models/microbe.py` model in a timesync integration with the root model and the shoot model.

In [1]:
```
from yggdrasil import tools
from yggdrasil.runner import run as run

# Part 1: integrate microbe w/ root model
tools.display_source_diff('models/microbe.py',
                          'solutions/tyk6/models/microbe.py', number_lines=True)
tools.display_source('solutions/tyk6/yamls/microbe.yml', number_lines=True)
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml'],
    production_run=True)

# Part 2: integrate microbe w/ root & shoot model
run(['solutions/tyk6/yamls/roots_v1.yml', 'solutions/tyk6/yamls/microbe.yml', 'solutions/tyk6/yamls/timesync.yml',
     'solutions/tyk6/yamls/shoot_v3.yml', 'solutions/tyk6/yamls/light_v1_python.yml'],
    production_run=True)
```

Test your knowledge #6

2. Run the `models/microbe.py` model in a timesync integration with the root model and the shoot model.

```
INFO:73601:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in names
pace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk6/models/roots_v1.py 0.0 2.0 0.5
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk6/models/microbe.py 0.0 2880.0 60.0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk6/models/ygg_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk6/models/shoot_v3.py 0.0 48.0 6.0
End of input from temp_doy.
INFO:73601:runner.waitModels[553]:YggRunner(runner): shoot finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:73601:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
INFO:73601:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): light finished exiting.
No more messages from model process.
INFO:73601:DSLModelDriver.after_loop[131]:TimeSyncModelDriver(shoot2root): returncode = 0
INFO:73601:runner.waitModels[553]:YggRunner(runner): shoot2root finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): shoot2root finished exiting.
INFO:73601:runner.waitModels[553]:YggRunner(runner): microbe finished running.
INFO:73601:runner.waitModels[559]:YggRunner(runner): microbe finished exiting.
INFO:73601:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:73601:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:73601:runner.run[374]:YggRunner(runner):         load drivers      0.067016
INFO:73601:runner.run[374]:YggRunner(runner):        start drivers      0.744601
INFO:73601:runner.run[374]:YggRunner(runner):           run models      80.924944
INFO:73601:runner.run[374]:YggRunner(runner):              at exit      0.156641
INFO:73601:runner.run[376]:YggRunner(runner): ========================================
INFO:73601:runner.run[377]:YggRunner(runner):                Total      81.893203
```

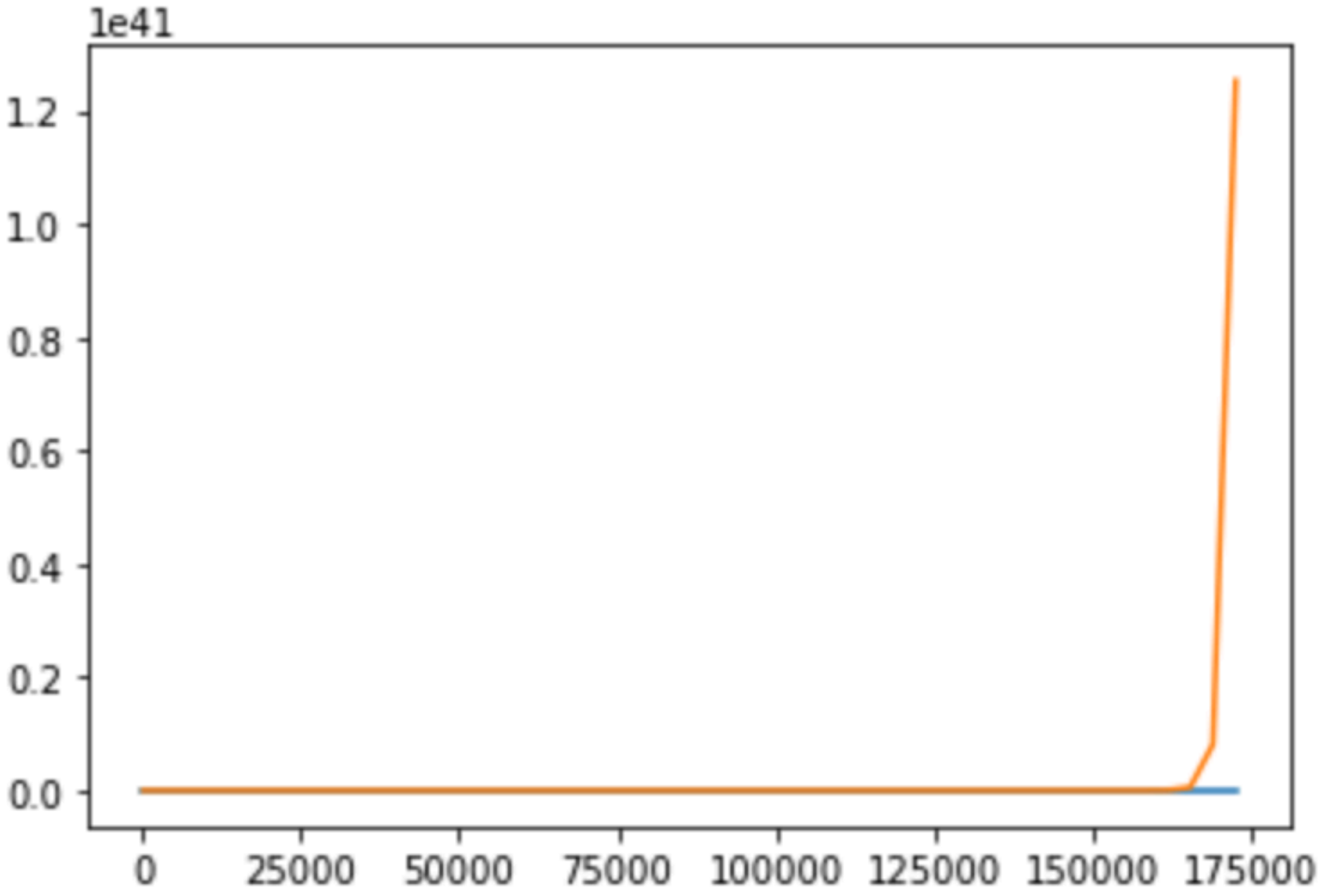
Test your knowledge #6

3. Plot the masses for the root, shoot, and microbe models as a function of time.

In [2]:
```
import pickle
import numpy as np
import matplotlib.pyplot as plt
from yggdrasil import units

# Part 3: plot
filename_root_masses = 'solutions/tyk6/output/masses.pkl'
with open(filename_root_masses, 'rb') as fd:
    root_masses = pickle.load(fd)
    root_masses['times'] = units.add_units(
        np.array(root_masses['times']), 'days')
    root_masses['masses'] = units.add_units(
        np.array(root_masses['masses']), 'kg')
    root_masses['times'].convert_to_mks()
    root_masses['masses'].convert_to_mks()
filename_microbe_masses = 'solutions/tyk6/output/microbe_masses.pkl'
with open(filename_microbe_masses, 'rb') as fd:
    microbe_masses = pickle.load(fd)
    microbe_masses['times'] = units.add_units(
        np.array(microbe_masses['times']), 'min')
    microbe_masses['masses'] = units.add_units(
        np.array(microbe_masses['masses']), 'g')
    microbe_masses['times'].convert_to_mks()
    microbe_masses['masses'].convert_to_mks()
plt.plot(root_masses['times'], root_masses['masses'])
plt.plot(microbe_masses['times'], microbe_masses['masses'])
plt.show()
```

Test your knowledge #6

3. Plot the masses for the root, shoot, and microbe models as a function of time.



NEW NOTEBOOK!

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

jupyter

Visit repo | Copy Binder link | Quit

Files | Running | Clusters | Nbextensions

Select items to perform actions on them.

Upload | New

0 /

Name	Last Modified	File size
images	33 minutes ago	
input	33 minutes ago	
meshes	33 minutes ago	
models	33 minutes ago	
yamls	33 minutes ago	
00-intro.ipynb	33 minutes ago	457 kB
01-connections.ipynb	33 minutes ago	470 kB
02-timesync.ipynb	33 minutes ago	298 kB
03-misc.ipynb	33 minutes ago	3.56 kB

IMPORTING MODELS AS PYTHON FUNCTIONS

In [1]:
```
from yggdrasil import tools
tools.display_source('models/light_v0.f90', number_lines=True)
tools.display_source('yamls/light_v0_fortran.yml', number_lines=True)
```

In [1]:
```python
from yggdrasil import tools
tools.display_source('models/light_v0.f90', number_lines=True)
tools.display_source('yamls/light_v0_fortran.yml', number_lines=True)
```

```
file: models/light_v0.f90
=========================
 1: !-----------------------------------------------------------------------
 2: !> @brief Compute the intensity of light.
 3: !
 4: !> @param[in] doy: Day of year.
 5: !> @param[in] height: Distance from ground in cm.
 6: !
 7: !> @return intensity: Intensity of light in ergs cm^-2 s^-1.
 8: !-----------------------------------------------------------------------
 9: function light(doy, height) result(intensity)
10:   real(kind=8) :: doy
11:   real(kind=8) :: height
12:   real(kind=8) :: intensity
13:   real, parameter :: Pi = 3.1415927
14:
15:   ! Define parameters that are static across a run
16:   real, parameter :: amplitude = 80.0
17:   real, parameter :: doy_offset = 0.0
18:
19:   ! Calculate intensity
20:   intensity = amplitude * height * (1.0 + SIN(2.0 * Pi * (doy - doy_offset) / 365))
21: end function light
```

In [1]:
```python
from yggdrasil import tools
tools.display_source('models/light_v0.f90', number_lines=True)
tools.display_source('yamls/light_v0_fortran.yml', number_lines=True)
```

file: models/light_v0.f90
==========

```fortran
!-----------------------------------------------------------------------
!> @brief Compute the intensity of light.
!
!> @param[in] doy: Day of year.
!> @param[in] height: Distance from ground in cm.
!
!> @return intensity: Intensity of light in ergs cm^-2 s^-1.
!-----------------------------------------------------------------------
function light(doy, height) result(intensity)
  real(kind=8) :: doy
  real(kind=8) :: height
  real(kind=8) :: intensity
  real, parameter :: Pi = 3.1415927

  ! Define parameters that are static across a run
  real, parameter :: amplitude = 80.0
  real, parameter :: doy_offset = 0.0

  ! Calculate intensity
  intensity = amplitude * height * (1.0 + SIN(2.0 * Pi * (doy - doy_offset) / 365))
end function light
```

file: yamls/light_v0_fortran.yml
==========

```yaml
model:
  name: light
  language: fortran
  args: ../models/light_v0.f90
  function: light
  inputs:
    - name: input
      vars: [doy, height]
      datatype:
        type: array
        items:
          - type: float
            units: day
          - type: float
            units: cm
  output:
    - name: output
      datatype:
        type: float
        units: ergs/(cm**2*s)
```

```
In [2]: from yggdrasil import import_as_function
        light = import_as_function('yamls/light_v0_fortran.yml')
```

Model can be called from Python regardless of the language

```
In [2]: from yggdrasil import import_as_function
        light = import_as_function('yamls/light_v0_fortran.yml')
```

```
INFO:97475:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg_light_v0_f90_gfortranx_gfortranx.out
INFO:97475:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:97475:runner.run[374]:YggRunner(runner):          load drivers      0.828631
INFO:97475:runner.run[374]:YggRunner(runner):         start drivers      0.061332
INFO:97475:runner.run[376]:YggRunner(runner): ========================================
INFO:97475:runner.run[377]:YggRunner(runner):                 Total      0.889964
```

Model can be called from Python regardless of the language

```
In [2]: from yggdrasil import import_as_function
        light = import_as_function('yamls/light_v0_fortran.yml')
```

```
INFO:97475:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg_light_v0_f90_gfortranx_gfortranx.out
INFO:97475:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:97475:runner.run[374]:YggRunner(runner):          load drivers      0.828631
INFO:97475:runner.run[374]:YggRunner(runner):         start drivers      0.061332
INFO:97475:runner.run[376]:YggRunner(runner): ========================================
INFO:97475:runner.run[377]:YggRunner(runner):                 Total      0.889964
```

```
In [3]: light.model_info()
```

Model can be called from Python regardless of the language

In [2]:
```
from yggdrasil import import_as_function
light = import_as_function('yamls/light_v0_fortran.yml')
```

```
INFO:97475:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg_light_v0_f90_gfortranx_gfortranx.out
INFO:97475:runner.run[374]:YggRunner(runner):                    init      0.000001
INFO:97475:runner.run[374]:YggRunner(runner):            load drivers      0.828631
INFO:97475:runner.run[374]:YggRunner(runner):           start drivers      0.061332
INFO:97475:runner.run[376]:YggRunner(runner): ========================================
INFO:97475:runner.run[377]:YggRunner(runner):                   Total      0.889964
```

In [3]:
```
light.model_info()
```

```
Models: light
Inputs:
        light:input_to_light:input (vars=['doy', 'height'])
Outputs:
        light:output (vars=['intensity'])
```

Model can be called from Python regardless of the language

In [4]:
```
print(light(100.0, 100.0))
print(light(1.0, 2.9))
print(light(2.0, 3.0))
```

```
In [4]: print(light(100.0, 100.0))
        print(light(1.0, 2.9))
        print(light(2.0, 3.0))
```

```
{'intensity': unyt_quantity(15909.42066435, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(235.99349874, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(248.26118702, 'erg/(cm**2*s)')}
```

Model units are assumed if not provided

In [4]:
```
print(light(100.0, 100.0))
print(light(1.0, 2.9))
print(light(2.0, 3.0))
```

```
{'intensity': unyt_quantity(15909.42066435, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(235.99349874, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(248.26118702, 'erg/(cm**2*s)')}
```

Model units are assumed if not provided

In [5]:
```
from yggdrasil import units
print(light(units.add_units(24.0, 'hrs'), units.add_units(2.9, 'cm')))
print(light(units.add_units(1.0, 'days'), units.add_units(0.029, 'm')))
```

```
In [4]: print(light(100.0, 100.0))
        print(light(1.0, 2.9))
        print(light(2.0, 3.0))
```

```
{'intensity': unyt_quantity(15909.42066435, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(235.99349874, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(248.26118702, 'erg/(cm**2*s)')}
```

Model units are assumed if not provided

```
In [5]: from yggdrasil import units
        print(light(units.add_units(24.0, 'hrs'), units.add_units(2.9, 'cm')))
        print(light(units.add_units(1.0, 'days'), units.add_units(0.029, 'm')))
```

```
{'intensity': unyt_quantity(235.99349874, 'erg/(cm**2*s)')}
{'intensity': unyt_quantity(235.99349874, 'erg/(cm**2*s)')}
```

If units are provided, yggdrasil can handle transformations to/from the model's units

Test your knowledge #7

1. Try importing the `models/weather.py` model in the cell below. Depending on how you wrote it, you may have to modify the YAML so that the model is alone in a file (i.e. no other models or connections).
2. Try importing the `models/co2.py` model in the cell below.

Tip: If a model receives or send multiple variables from/to a channel, those variables will need to be explicitly named to be passed separately to/from the imported function

TEST YOUR KNOWLEDGE (10 MIN)

Test your knowledge #7

1. Try importing the `models/weather.py` model in the cell below. Depending on how you wrote it, you may have to modify the YAML so that the model is alone in a file (i.e. no other models or connections).

In [1]:
```
from yggdrasil import import_as_function
from yggdrasil import tools, units

# Part 1: import the weather model
x = import_as_function('solutions/tyk7/yamls/weather_alone.yml')
x.model_info()
print(x(123.0))
x.stop()

# Part 2: import the co2 model
x = import_as_function('solutions/tyk7/yamls/co2.yml')
x.model_info()
print(x(units.add_units(24.0, 'hrs'), units.add_units(2.9, 'cm')))
x.stop()
```

Test your knowledge #7

1. Try importing the `models/weather.py` model in the cell below. Depending on how you wrote it, you may have to modify the YAML so that the model is alone in a file (i.e. no other models or connections).

```
INFO:76065:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk7/models/ygg_weather.py
INFO:76065:runner.run[374]:YggRunner(runner):                  init      0.000001
INFO:76065:runner.run[374]:YggRunner(runner):          load drivers      0.050186
INFO:76065:runner.run[374]:YggRunner(runner):         start drivers      0.118609
INFO:76065:runner.run[376]:YggRunner(runner): ========================================
INFO:76065:runner.run[377]:YggRunner(runner):                 Total      0.168796
Models: weather
Inputs:
        weather:input_to_weather:input (vars=['intensity'])
Outputs:
        weather:output (vars=['T'])

{'T': unyt_quantity(51.09377384, 'K')}
End of input from intensity.
INFO:76065:runner.waitModels[553]:YggRunner(runner): weather finished running.
INFO:76065:runner.waitModels[559]:YggRunner(runner): weather finished exiting.
INFO:76065:runner.waitModels[553]:YggRunner(runner): function_model finished running.
INFO:76065:runner.waitModels[559]:YggRunner(runner): function_model finished exiting.
INFO:76065:runner.waitModels[573]:YggRunner(runner): All models completed
```

Test your knowledge #7

2. Try importing the `models/co2.py` model in the cell below.

```
INFO:76065:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk7/models/co2.py 5.0 23.0 126.0
INFO:76065:runner.run[374]:YggRunner(runner):                 init      0.000001
INFO:76065:runner.run[374]:YggRunner(runner):         load drivers      0.032829
INFO:76065:runner.run[374]:YggRunner(runner):        start drivers      0.181778
INFO:76065:runner.run[376]:YggRunner(runner): ========================================
INFO:76065:runner.run[377]:YggRunner(runner):                Total      0.214608
Models: co2
Inputs:
        co2:height_to_co2:height (vars=['doy', 'height'])
Outputs:
        co2:co2 (vars=['doy', 'co2'])

{'doy': unyt_quantity(24., 'hr'), 'co2': unyt_quantity(0.000949, 'cm**(-3)')}
Concentration 0.0009489989903767828 cm**(-3)
End of height input
INFO:76065:runner.waitModels[553]:YggRunner(runner): co2 finished running.
INFO:76065:runner.waitModels[559]:YggRunner(runner): co2 finished exiting.
INFO:76065:runner.waitModels[553]:YggRunner(runner): function_model finished running.
INFO:76065:runner.waitModels[559]:YggRunner(runner): function_model finished exiting.
INFO:76065:runner.waitModels[573]:YggRunner(runner): All models completed
```

OVERHEAD

```
In [7]: # Calculation using numpy in Python
        import numpy as np
        %timeit np.sin(0.0)
```

Yggdrasil wrapped models can’t outperform the direct call

In [7]:
```
# Calculation using numpy in Python
import numpy as np
%timeit np.sin(0.0)
```

```
2.76 µs ± 274 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)
```

Yggdrasil wrapped models can’t outperform the direct call

In [7]:
```
# Calculation using numpy in Python
import numpy as np
%timeit np.sin(0.0)
```

```
2.76 µs ± 274 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)
```

In [8]:
```
# Calculation calling numpy in Python via yggdrasil
sine_python = import_as_function('yamls/sine_model_python.yml')
%timeit sine_python(0.0)
sine_python.stop()
```

Yggdrasil wrapped models can’t outperform the direct call

In [7]:
```
# Calculation using numpy in Python
import numpy as np
%timeit np.sin(0.0)
```

```
2.76 µs ± 274 ns per loop (mean ± std. dev. of 7 runs, 100000 loops each)
```

In [8]:
```
# Calculation calling numpy in Python via yggdrasil
sine_python = import_as_function('yamls/sine_model_python.yml')
%timeit sine_python(0.0)
sine_python.stop()
```

```
INFO:63595:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_sine_model.py
INFO:63595:runner.run[374]:YggRunner(runner):                    init      0.000002
INFO:63595:runner.run[374]:YggRunner(runner):            load drivers      0.035941
INFO:63595:runner.run[374]:YggRunner(runner):           start drivers      0.092159
INFO:63595:runner.run[376]:YggRunner(runner): ========================================
INFO:63595:runner.run[377]:YggRunner(runner):                   Total      0.128102
186 ms ± 43.1 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
End of input from x.
INFO:63595:runner.waitModels[553]:YggRunner(runner): sine_model finished running.
INFO:63595:runner.waitModels[559]:YggRunner(runner): sine_model finished exiting.
INFO:63595:runner.waitModels[553]:YggRunner(runner): function_model finished running.
INFO:63595:runner.waitModels[559]:YggRunner(runner): function_model finished exiting.
INFO:63595:runner.waitModels[573]:YggRunner(runner): All models completed
```

Yggdrasil wrapped models can’t outperform the direct call

```
In [9]: # Calculation calling the fortran SIN function via yggdrasil
        sine_fortran = import_as_function('yamls/sine_model_fortran.yml')
        %timeit sine_fortran(0.0)
        sine_fortran.stop()
```

Method	Time (ms)
Direct Python	0.00276
Python via Yggdrasil	186
Fortran via Yggdrasil	337

Yggdrasil wrapped models can’t outperform the direct call

```
In [9]: # Calculation calling the fortran SIN function via yggdrasil
        sine_fortran = import_as_function('yamls/sine_model_fortran.yml')
        %timeit sine_fortran(0.0)
        sine_fortran.stop()
```

Method	Time (ms)
Direct Python	0.00276
Python via Yggdrasil	186
Fortran via Yggdrasil	337

Extreme example (highly optimized function, no parallelism, etc.), but useful demonstration.

Yggdrasil wrapped models can't outperform the direct call

```
In [9]: # Calculation calling the fortran SIN function via yggdrasil
        sine_fortran = import_as_function('yamls/sine_model_fortran.yml')
        %timeit sine_fortran(0.0)
        sine_fortran.stop()
```

```
INFO:63595:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system None in namespace yg
gdrasil with rank 0
/Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg_sine_model_f90_gfortranx_gfortranx.out
INFO:63595:runner.run[374]:YggRunner(runner):                 init      0.000015
INFO:63595:runner.run[374]:YggRunner(runner):         load drivers      1.385137
INFO:63595:runner.run[374]:YggRunner(runner):        start drivers      0.118329
INFO:63595:runner.run[376]:YggRunner(runner): ========================================
INFO:63595:runner.run[377]:YggRunner(runner):                Total      1.503481
337 ms ± 32.7 ms per loop (mean ± std. dev. of 7 runs, 1 loop each)
End of input from yggarg(x).
INFO:63595:runner.waitModels[553]:YggRunner(runner): sine_model finished running.
INFO:63595:runner.waitModels[559]:YggRunner(runner): sine_model finished exiting.
INFO:63595:runner.waitModels[553]:YggRunner(runner): function_model finished running.
INFO:63595:runner.waitModels[559]:YggRunner(runner): function_model finished exiting.
INFO:63595:runner.waitModels[573]:YggRunner(runner): All models completed
```

Method	Time (ms)
Direct Python	0.00276
Python via Yggdrasil	186
Fortran via Yggdrasil	337

Extreme example (highly optimized function, no parallelism, etc.), but useful demonstration.

Yggdrasil wrapped models can't outperform the direct call

COMPILATION VIA BUILD TOOLS

Yggdrasil can inject compilation flags to existing builds that use build tools like make & cmake

```
In [10]: tools.display_source('yamls/roots_v1_c.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_make.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_cmake.yml', number_lines=True)
```

```
file: yamls/roots_v1_c.yml
==========================
 1: model:
 2:   name: roots
 3:   language: c
 4:   args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 5:   timesync: shoot2root
```

Yggdrasil can inject compilation flags to existing builds that use build tools like make & cmake

```
In [10]: tools.display_source('yamls/roots_v1_c.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_make.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_cmake.yml', number_lines=True)
```

```
file: yamls/roots_v1_c.yml
==========================
 1: model:
 2:   name: roots
 3:   language: c
 4:   args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 5:   timesync: shoot2root
```

Yggdrasil can inject compilation flags to existing builds that use build tools like make & cmake

```
In [10]: tools.display_source('yamls/roots_v1_c.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_make.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_cmake.yml', number_lines=True)
```

```
file: yamls/roots_v1_c.yml
==========================
 1: model:
 2:   name: roots
 3:   language: c
 4:   args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 5:   timesync: shoot2root


file1: yamls/roots_v1_c.yml
file2: yamls/roots_v1_make.yml
==============================
 1:   model:
 2:     name: roots
    -   language: c
    ?                                ^

 3: +   language: make
    ?                                ^^^^

 4: +   target: roots_v1
 5:     args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 6:     timesync: shoot2root
```

Yggdrasil can inject compilation flags to existing builds that use build tools like make & cmake

```
In [10]: tools.display_source('yamls/roots_v1_c.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_make.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_cmake.yml', number_lines=True)
```

```
file: yamls/roots_v1_c.yml
==========================
 1: model:
 2:   name: roots
 3:   language: c
 4:   args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 5:   timesync: shoot2root


file1: yamls/roots_v1_c.yml
file2: yamls/roots_v1_make.yml
==============================
 1:   model:
 2:     name: roots
    -   language: c
    ?                          ^

 3: +   language: make
    ?                          ^^^^

 4: +   target: roots_v1
 5:     args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 6:     timesync: shoot2root
```

Yggdrasil can inject compilation flags to existing builds that use build tools like make & cmake

```
In [10]: tools.display_source('yamls/roots_v1_c.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_make.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_cmake.yml', number_lines=True)
```

```
file: yamls/roots_v1_c.yml
==========================
 1: model:
 2:   name: roots
 3:   language: c
 4:   args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 5:   timesync: shoot2root
```

```
file1: yamls/roots_v1_c.yml
file2: yamls/roots_v1_make.yml
==============================
 1:   model:
 2:     name: roots
    -   language: c
    ?                          ^

 3: +   language: make
    ?                          ^^^^

 4: +   target: roots_v1
 5:     args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 6:     timesync: shoot2root
```

```
file1: yamls/roots_v1_c.yml
file2: yamls/roots_v1_cmake.yml
===============================
 1:   model:
 2:     name: roots
    -   language: c
 3: +   language: cmake
    ?                          ++++

 4: +   target: roots_v1
 5:     args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 6:     timesync: shoot2root
```

Yggdrasil can inject compilation flags to existing builds that use build tools like make & cmake

```
In [10]: tools.display_source('yamls/roots_v1_c.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_make.yml', number_lines=True)
         tools.display_source_diff('yamls/roots_v1_c.yml', 'yamls/roots_v1_cmake.yml', number_lines=True)
```

```
file: yamls/roots_v1_c.yml
==========================
 1: model:
 2:   name: roots
 3:   language: c
 4:   args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 5:   timesync: shoot2root
```

```
file1: yamls/roots_v1_c.yml
file2: yamls/roots_v1_make.yml
==============================
 1:   model:
 2:     name: roots
    -   language: c
    ?                          ^

 3: +   language: make
    ?                          ^^^^

 4: +   target: roots_v1
 5:     args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 6:     timesync: shoot2root
```

```
file1: yamls/roots_v1_c.yml
file2: yamls/roots_v1_cmake.yml
===============================
 1:   model:
 2:     name: roots
    -   language: c
 3: +   language: cmake
    ?                          ++++

 4: +   target: roots_v1
 5:     args: [../models/roots_v1.c, 0.0, 2.0, 0.5]
 6:     timesync: shoot2root
```

TRANSFORMING & FILTERING I/O

TRANSFORMING I/O

```
In [11]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_statement.yml', number_lines=True)
```

TRANSFORMING I/O

```
In [11]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_statement.yml', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_transform_statement.yml
==================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       transform:
 9: +         statement: "%x%*2"
    -       name: ../output/light_v0.txt
    ?                                                 ^^

10: +       name: ../output/light_transform_statement.txt
    ?                                                 ^^^^^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

TRANSFORMING I/O

```
In [11]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_statement.yml', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_transform_statement.yml
================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       transform:
 9: +         statement: "%x%*2"
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_transform_statement.txt
    ?                                                  ^^^^^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

%x% stands in for the values passing through the connection

TRANSFORMING I/O

In [12]:
```
from yggdrasil.runner import run
run(['yamls/light_v0_python.yml', 'yamls/connections_transform_statement.yml'], production_run=True)
tools.display_source_diff('output/light_v0.txt', 'output/light_transform_statement.txt', number_lines=True)
```

TRANSFORMING I/O

In [12]:
```
from yggdrasil.runner import run
run(['yamls/light_v0_python.yml', 'yamls/connections_transform_statement.yml'], production_run=True)
tools.display_source_diff('output/light_v0.txt', 'output/light_transform_statement.txt', number_lines=True)
```

```
INFO:4261:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in namesp
ace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
End of input from temp_doy.
INFO:4261:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:4261:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:4261:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:4261:runner.run[374]:YggRunner(runner):                  init       0.000003
INFO:4261:runner.run[374]:YggRunner(runner):          load drivers       0.059429
INFO:4261:runner.run[374]:YggRunner(runner):         start drivers       0.276533
INFO:4261:runner.run[374]:YggRunner(runner):            run models       14.971059
INFO:4261:runner.run[374]:YggRunner(runner):               at exit       0.039609
INFO:4261:runner.run[376]:YggRunner(runner): ========================================
INFO:4261:runner.run[377]:YggRunner(runner):                 Total       15.346633
```

TRANSFORMING I/O

In [12]:
```
from yggdrasil.runner import run
run(['yamls/light_v0_python.yml', 'yamls/connections_transform_statement.yml'], production_run=True)
tools.display_source_diff('output/light_v0.txt', 'output/light_transform_statement.txt', number_lines=True)
```

```
file1: output/light_v0.txt
file2: output/light_transform_statement.txt
==========================================
 1:   # intensity
 2:   # erg/(cm**2*s)
 3:   # %g
 4:   0
    - 40.6885
    - 82.7537
    - 168.259
    - 342.017
    - 434.386
    - 617.737
    - 896.166
 5: + 81.3771
 6: + 165.507
 7: + 336.518
 8: + 684.034
 9: + 868.772
10: + 1235.47
11: + 1792.33
```

TRANSFORMING I/O

```
In [12]: from yggdrasil.runner import run
         run(['yamls/light_v0_python.yml', 'yamls/connections_transform_statement.yml'], production_run=True)
         tools.display_source_diff('output/light_v0.txt', 'output/light_transform_statement.txt', number_lines=True)
```

```
file1: output/light_v0.txt
file2: output/light_transform_statement.txt
==========================================
 1:   # intensity
 2:   # erg/(cm**2*s)
 3:   # %g
 4:   0
    - 40.6885
    - 82.7537
    - 168.259
    - 342.017
    - 434.386
    - 617.737
    - 896.166
 5: + 81.3771
 6: + 165.507
 7: + 336.518
 8: + 684.034
 9: + 868.772
10: + 1235.47
11: + 1792.33
```

Values doubled

TRANSFORMING I/O

```
In [13]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_function.yml', number_lines=True)
         tools.display_source('models/light_transform.py', number_lines=True)
```

TRANSFORMING I/O

```
In [13]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_function.yml', number_lines=True)
         tools.display_source('models/light_transform.py', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_transform_function.yml
================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       transform:
 9: +         function: ../models/light_transform.py:double_light
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_transform_function.txt
    ?                                                  ^^^^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

TRANSFORMING I/O

```
In [13]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_function.yml', number_lines=True)
         tools.display_source('models/light_transform.py', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_transform_function.yml
================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       transform:
 9: +         function: ../models/light_transform.py:double_light
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_transform_function.txt
    ?                                                  ^^^^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

Can also use function

TRANSFORMING I/O

```
In [13]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_transform_function.yml', number_lines=True)
         tools.display_source('models/light_transform.py', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_transform_function.yml
================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       transform:
 9: +         function: ../models/light_transform.py:double_light
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_transform_function.txt
    ?                                                  ^^^^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]

file: models/light_transform.py
================================
 1: def double_light(intensity):
 2:     r"""Double the intensity."""
 3:     return 2.0 * intensity
```

Can also use function

FILTERING I/O

```
In [14]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_statement.yml', number_lines=True)
```

FILTERING I/O

```
In [14]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_statement.yml', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_filter_statement.yml
==============================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       filter:
 9: +         statement: "%x% < 400"
    -       name: ../output/light_v0.txt
    ?                                                ^^

10: +       name: ../output/light_filter_statement.txt
    ?                                                ^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

FILTERING I/O

```
In [14]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_statement.yml', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_filter_statement.yml
==============================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       filter:
 9: +         statement: "%x% < 400"
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_filter_statement.txt
    ?                                                  ^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

%x% stands in for the values passing through the connection

FILTERING I/O

```
In [15]: run(['yamls/light_v0_python.yml', 'yamls/connections_filter_statement.yml'], production_run=True)
         tools.display_source_diff('output/light_v0.txt', 'output/light_filter_statement.txt', number_lines=True)
```

FILTERING I/O

```
In [15]: run(['yamls/light_v0_python.yml', 'yamls/connections_filter_statement.yml'], production_run=True)
         tools.display_source_diff('output/light_v0.txt', 'output/light_filter_statement.txt', number_lines=True)
```

```
INFO:7795:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in namesp
ace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/models/ygg
_light_v0.py
End of input from temp_doy.
INFO:7795:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:7795:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:7795:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:7795:runner.run[374]:YggRunner(runner):                  init       0.000001
INFO:7795:runner.run[374]:YggRunner(runner):          load drivers       0.045665
INFO:7795:runner.run[374]:YggRunner(runner):         start drivers       0.279095
INFO:7795:runner.run[374]:YggRunner(runner):            run models       12.120064
INFO:7795:runner.run[374]:YggRunner(runner):               at exit       0.033430
INFO:7795:runner.run[376]:YggRunner(runner): ========================================
INFO:7795:runner.run[377]:YggRunner(runner):                 Total       12.478255
```

FILTERING I/O

In [15]:
```
run(['yamls/light_v0_python.yml', 'yamls/connections_filter_statement.yml'], production_run=True)
tools.display_source_diff('output/light_v0.txt', 'output/light_filter_statement.txt', number_lines=True)
```

```
file1: output/light_v0.txt
file2: output/light_filter_statement.txt
========================================
 1:   # intensity
 2:   # erg/(cm**2*s)
 3:   # %g
 4:   0
 5:   40.6885
 6:   82.7537
 7:   168.259
 8:   342.017
    - 434.386
    - 617.737
    - 896.166
```

FILTERING I/O

```
In [15]: run(['yamls/light_v0_python.yml', 'yamls/connections_filter_statement.yml'], production_run=True)
         tools.display_source_diff('output/light_v0.txt', 'output/light_filter_statement.txt', number_lines=True)
```

```
file1: output/light_v0.txt
file2: output/light_filter_statement.txt
========================================
 1:   # intensity
 2:   # erg/(cm**2*s)
 3:   # %g
 4:   0
 5:   40.6885
 6:   82.7537
 7:   168.259
 8:   342.017
    - 434.386
    - 617.737
    - 896.166
```

Values >400 filtered out

FILTERING I/O

```
In [16]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_function.yml', number_lines=True)
         tools.display_source('models/light_filter.py', number_lines=True)
```

FILTERING I/O

```
In [16]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_function.yml', number_lines=True)
         tools.display_source('models/light_filter.py', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_filter_function.yml
==============================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       filter:
 9: +         function: ../models/light_filter.py:filter_light
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_filter_function.txt
    ?                                                  ^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

FILTERING I/O

```
In [16]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_function.yml', number_lines=True)
         tools.display_source('models/light_filter.py', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_filter_function.yml
===========================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       filter:
 9: +         function: ../models/light_filter.py:filter_light
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_filter_function.txt
    ?                                                  ^^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]
```

Can also use function

FILTERING I/O

```
In [16]: tools.display_source_diff('yamls/connections_v0.yml', 'yamls/connections_filter_function.yml', number_lines=True)
         tools.display_source('models/light_filter.py', number_lines=True)
```

```
file1: yamls/connections_v0.yml
file2: yamls/connections_filter_function.yml
===========================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
 7:       output:
 8: +       filter:
 9: +         function: ../models/light_filter.py:filter_light
    -       name: ../output/light_v0.txt
    ?                                                  ^^

10: +       name: ../output/light_filter_function.txt
    ?                                                  ^^^^^^^^^^^^^^^

11:         filetype: table
12:         field_names: [intensity]

file: models/light_filter.py
============================
 1: def filter_light(intensity):
 2:     return itensity < 400
```

Can also use function

Test your knowledge #8

1. Write a YAML that outputs intensities `<400` to one file and `>=400` to a different file.
2. Create a new version of the light model that calculates light using different parameters or a different algorithm and write a YAML that directs heights `>2` to the original model and `<=2` to the new model, but both models output to the same file.
3. Add a transformation to the output from the new model.

TEST YOUR KNOWLEDGE (10 MIN)

Test your knowledge #8

1. Write a YAML that outputs intensities `<400` to one file and `>=400` to a different file.

In [1]:
```
from yggdrasil.runner import run
from yggdrasil import tools

# Part 1: Sort intensities
tools.display_source_diff('yamls/connections_v0.yml',
                          'solutions/tyk8/yamls/connections_sort_intensity.yml',
                          number_lines=True)
run(['solutions/tyk8/yamls/light_v0_python.yml', 'solutions/tyk8/yamls/connections_sort_intensity.yml'],
    production_run=True)
tools.display_source('solutions/tyk8/output/light_filter_lt400.txt', number_lines=True)
tools.display_source('solutions/tyk8/output/light_filter_ge400.txt', number_lines=True)

# Part 2: New light model
tools.display_source_diff('solutions/tyk8/models/light_v0.py', 'solutions/tyk8/models/light_new.py',
                          number_lines=True)
tools.display_source_diff('solutions/tyk8/yamls/light_v0_python.yml',
                          'solutions/tyk8/yamls/light_new_python.yml', number_lines=True)
tools.display_source_diff('yamls/connections_v0.yml',
                          'solutions/tyk8/yamls/connections_sort_height.yml',
                          number_lines=True)
run(['solutions/tyk8/yamls/light_v0_python.yml', 'solutions/tyk8/yamls/light_new_python.yml',
     'solutions/tyk8/yamls/connections_sort_height.yml'],
    production_run=True)
tools.display_source('solutions/tyk8/output/light_multi_model.txt', number_lines=True)

# Part 3: Transformation
tools.display_source_diff('solutions/tyk8/yamls/connections_sort_height.yml',
                          'solutions/tyk8/yamls/connections_sort_height_transform.yml',
                          number_lines=True)
run(['solutions/tyk8/yamls/light_v0_python.yml', 'solutions/tyk8/yamls/light_new_python.yml',
     'solutions/tyk8/yamls/connections_sort_height_transform.yml'],
    production_run=True)
tools.display_source('solutions/tyk8/output/light_multi_model_transform.txt', number_lines=True)
```

Test your knowledge #8

1. Write a YAML that outputs intensities `<400` to one file and `>=400` to a different file.

```
file1: yamls/connections_v0.yml
file2: solutions/tyk8/yamls/connections_sort_intensity.yml
===========================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       output: light:input
 6:     - input: light:output
    -     output:
 7: +     outputs:
    ?                         +

 8: +       - filter:
 9: +           statement: "%x% < 400"
    -       name: ../output/light_v0.txt
    ?                                                 ^

10: +         name: ../output/light_filter_lt400.txt
    ? ++                                                ^^^^^^^^^^^

    -       filetype: table
11: +         filetype: table
    ? ++

    -       field_names: [intensity]
12: +         field_names: [intensity]
    ? ++

13: +       - filter:
14: +           statement: "%x% >= 400"
15: +         name: ../output/light_filter_ge400.txt
16: +         filetype: table
17: +         field_names: [intensity]
```

Test your knowledge #8

1. Write a YAML that outputs intensities `<400` to one file and `>=400` to a different file.

```
INFO:9698:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in namesp
ace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk8/models/ygg_light_v0.py
End of input from temp_doy.
INFO:9698:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:9698:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:9698:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:9698:runner.run[374]:YggRunner(runner):                  init       0.000002
INFO:9698:runner.run[374]:YggRunner(runner):          load drivers       0.535908
INFO:9698:runner.run[374]:YggRunner(runner):         start drivers       0.143507
INFO:9698:runner.run[374]:YggRunner(runner):             run models       11.749588
INFO:9698:runner.run[374]:YggRunner(runner):                at exit       0.062938
INFO:9698:runner.run[376]:YggRunner(runner): ========================================
INFO:9698:runner.run[377]:YggRunner(runner):                  Total       12.491943
```

Test your knowledge #8

1. Write a YAML that outputs intensities `<400` to one file and `>=400` to a different file.

```
file: solutions/tyk8/output/light_filter_lt400.txt
==================================================
 1: # intensity
 2: # erg/(cm**2*s)
 3: # %g
 4: 0
 5: 40.6885
 6: 82.7537
 7: 168.259
 8: 342.017

file: solutions/tyk8/output/light_filter_ge400.txt
==================================================
 1: # intensity
 2: # erg/(cm**2*s)
 3: # %g
 4: 434.386
 5: 617.737
 6: 896.166
```

Test your knowledge #8

2. Create a new version of the light model that calculates light using different parameters or a different algorithm and write a YAML that directs heights `>2` to the original model and `<=2` to the new model, but both models output to the same file.

```
file1: solutions/tyk8/models/light_v0.py
file2: solutions/tyk8/models/light_new.py
=========================================

 ...

20:         # Calculate intensity
21:         intensity = (
22:             amplitude * height *
    -           (1.0 + np.sin(2.0 * np.pi * (doy - doy_offset) /
    ?                                                            --

23: +           (1.0 + np.cos(2.0 * np.pi * (doy - doy_offset) /
    ?                                                          ++

24:                         units.add_units(365.0, 'days'))))
25:
26:         return intensity
```

Test your knowledge #8

2. Create a new version of the light model that calculates light using different parameters or a different algorithm and write a YAML that directs heights `>2` to the original model and `<=2` to the new model, but both models output to the same file.

```
file1: solutions/tyk8/yamls/light_v0_python.yml
file2: solutions/tyk8/yamls/light_new_python.yml
===================================================
 1:   model:
    -   name: light
 2: +   name: light_new
    ?                                   ++++

 3:     language: python
    -   args: ../models/light_v0.py
    ?                                           ^^

 4: +   args: ../models/light_new.py
    ?                                           ^^^

 5:     function: light
```

Test your knowledge #8

2. Create a new version of the light model that calculates light using different parameters or a different algorithm and write a YAML that directs heights `>2` to the original model and `<=2` to the new model, but both models output to the same file.

```
file1: yamls/connections_v0.yml
file2: solutions/tyk8/yamls/connections_sort_height.yml
=========================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5: +     outputs:
    -     output: light:input
    ?                  ^^^^^^

 6: +       - name: light:input
    ?    ++++              ^^^^

 7: +         filter:
 8: +           statement: "%x%[0] > 2"
 9: +       - name: light_new:input
10: +         filter:
11: +           statement: "%x%[0] <= 2"
    -   - input: light:output
    ?                                -------------

12: +   - inputs:
    ?                       +

13: +       - light:output
14: +       - light_new:output
15:       output:
    -       name: ../output/light_v0.txt
    ?                                                  ^^

16: +       name: ../output/light_multi_model.txt
    ?                                                  ^^^^^^^^^^^

17:         filetype: table
18:         field_names: [intensity]
```

Test your knowledge #8

2. Create a new version of the light model that calculates light using different parameters or a different algorithm and write a YAML that directs heights `>2` to the original model and `<=2` to the new model, but both models output to the same file.

```
INFO:9698:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in namesp
ace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk8/models/ygg_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk8/models/ygg_light_new.py
End of input from temp_doy.
End of input from temp_doy.
INFO:9698:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:9698:runner.waitModels[559]:YggRunner(runner): light finished exiting.

INFO:9698:runner.waitModels[553]:YggRunner(runner): light_new finished running.
INFO:9698:runner.waitModels[559]:YggRunner(runner): light_new finished exiting.
INFO:9698:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:9698:runner.run[374]:YggRunner(runner):                 init       0.000001
INFO:9698:runner.run[374]:YggRunner(runner):         load drivers       0.062529
INFO:9698:runner.run[374]:YggRunner(runner):        start drivers       0.483236
INFO:9698:runner.run[374]:YggRunner(runner):           run models       18.671072
INFO:9698:runner.run[374]:YggRunner(runner):              at exit       0.022364
INFO:9698:runner.run[376]:YggRunner(runner): ========================================
INFO:9698:runner.run[377]:YggRunner(runner):                Total       19.239202
```

Test your knowledge #8

2. Create a new version of the light model that calculates light using different parameters or a different algorithm and write a YAML that directs heights `>2` to the original model and `<=2` to the new model, but both models output to the same file.

```
file: solutions/tyk8/output/light_multi_model.txt
==================================================
 1: # intensity
 2: # erg/(cm**2*s)
 3: # %g
 4: 168.259
 5: 0
 6: 342.017
 7: 79.9941
 8: 434.386
 9: 159.953
10: 617.737
11: 896.166
```

Test your knowledge #8

3. Add a transformation to the output from the new model.

```
file1: solutions/tyk8/yamls/connections_sort_height.yml
file2: solutions/tyk8/yamls/connections_sort_height_transform.yml
==================================================================
 1:   connections:
 2:     - input:
 3:         name: ../input/light_v0.txt
 4:         filetype: table
 5:       outputs:
 6:         - name: light:input
 7:           filter:
 8:             statement: "%x%[0] > 2"
 9:         - name: light_new:input
10:           filter:
11:             statement: "%x%[0] <= 2"
12:     - inputs:
13:         - light:output
    -       - light_new:output
14: +       - name: light_new:output
15: +         transform:
16: +           statement: "2*%x%"
17:       output:
    -       name: ../output/light_multi_model.txt
18: +       name: ../output/light_multi_model_transform.txt
    ?                                                          ++++++++++

19:         filetype: table
20:         field_names: [intensity]
```

Test your knowledge #8

3. Add a transformation to the output from the new model.

```
INFO:9698:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system Meagans-Air in namesp
ace yggdrasil with rank 0
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk8/models/ygg_light_v0.py
/Users/langmm/miniconda3/envs/conda36/bin/python /Users/langmm/yggdrasil/yggdrasil/demos/CiS2021-hackathon/solutions/
tyk8/models/ygg_light_new.py
End of input from temp_doy.
End of input from temp_doy.
INFO:9698:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:9698:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:9698:runner.waitModels[553]:YggRunner(runner): light_new finished running.
INFO:9698:runner.waitModels[559]:YggRunner(runner): light_new finished exiting.
INFO:9698:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:9698:runner.run[374]:YggRunner(runner):                 init       0.000001
INFO:9698:runner.run[374]:YggRunner(runner):         load drivers       0.086844
INFO:9698:runner.run[374]:YggRunner(runner):        start drivers       0.486678
INFO:9698:runner.run[374]:YggRunner(runner):           run models       11.687946
INFO:9698:runner.run[374]:YggRunner(runner):              at exit       0.030099
INFO:9698:runner.run[376]:YggRunner(runner): ========================================
INFO:9698:runner.run[377]:YggRunner(runner):                Total       12.291568
```

Test your knowledge #8

3. Add a transformation to the output from the new model.

```
file: solutions/tyk8/output/light_multi_model_transform.txt
============================================================
 1: # intensity
 2: # erg/(cm**2*s)
 3: # %g
 4: 168.259
 5: 342.017
 6: 0
 7: 434.386
 8: 159.988
 9: 617.737
10: 319.905
11: 896.166
```

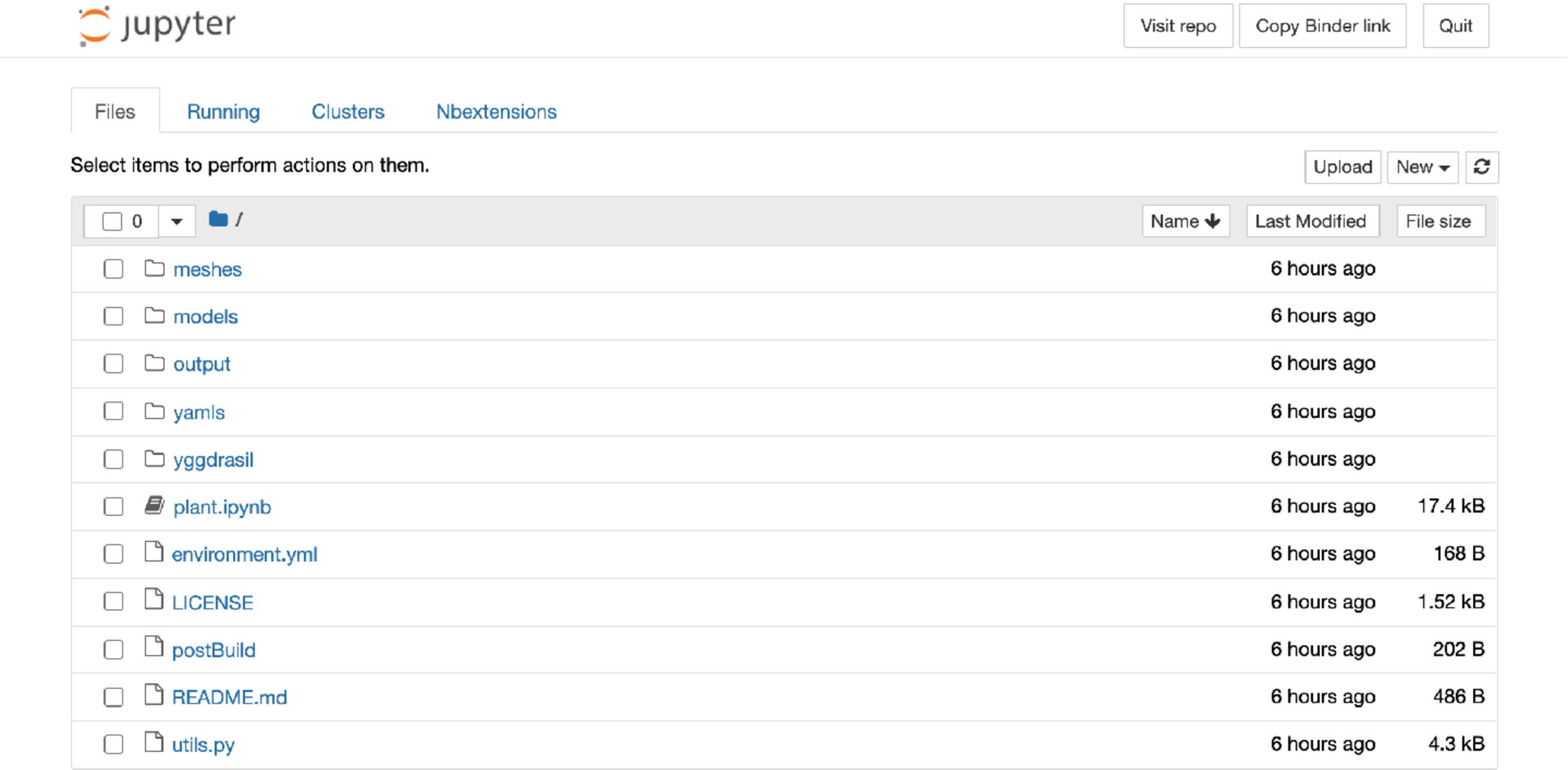
COMMAND LINE INTERFACE (CLI)

COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python

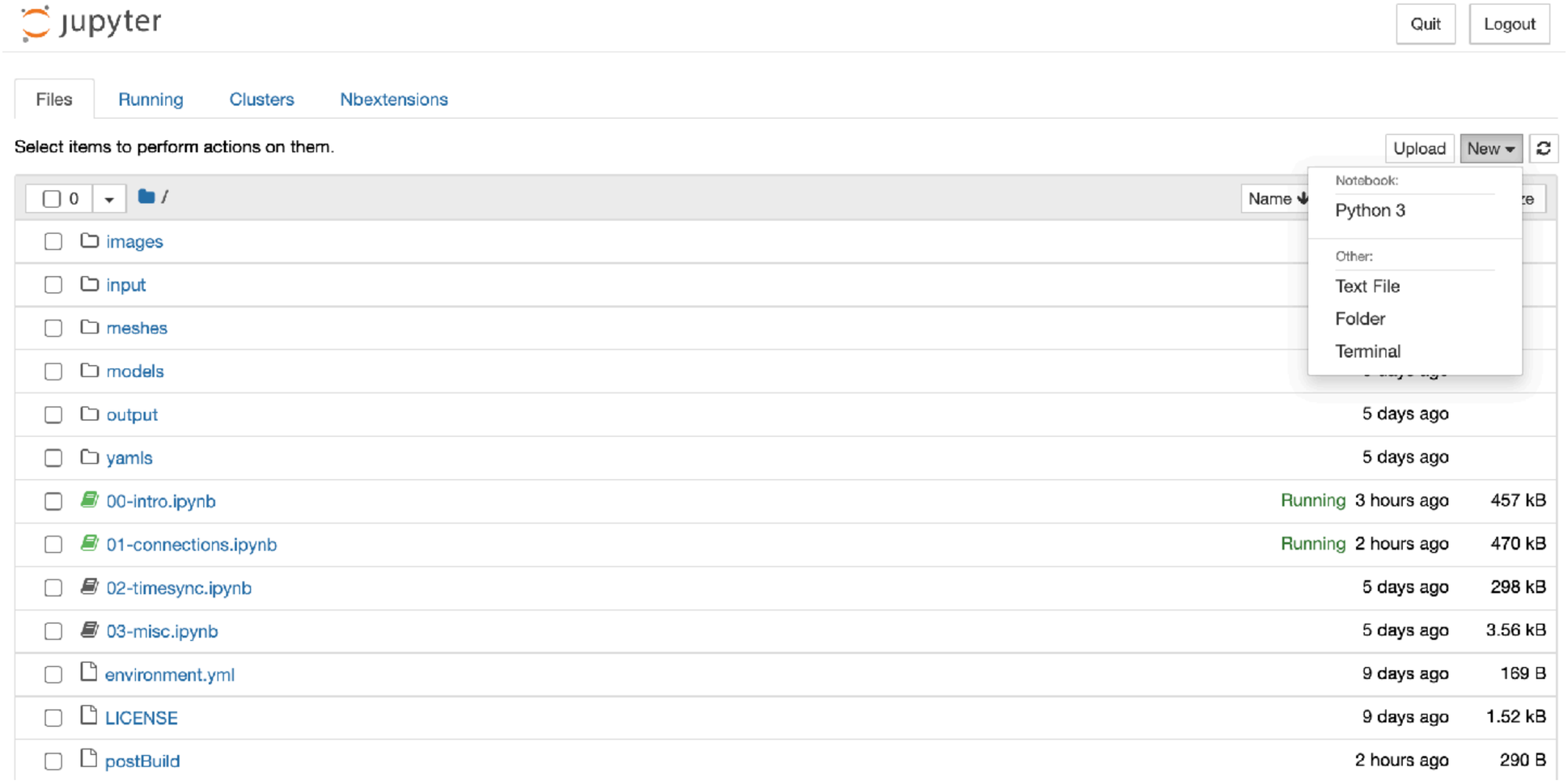
COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python



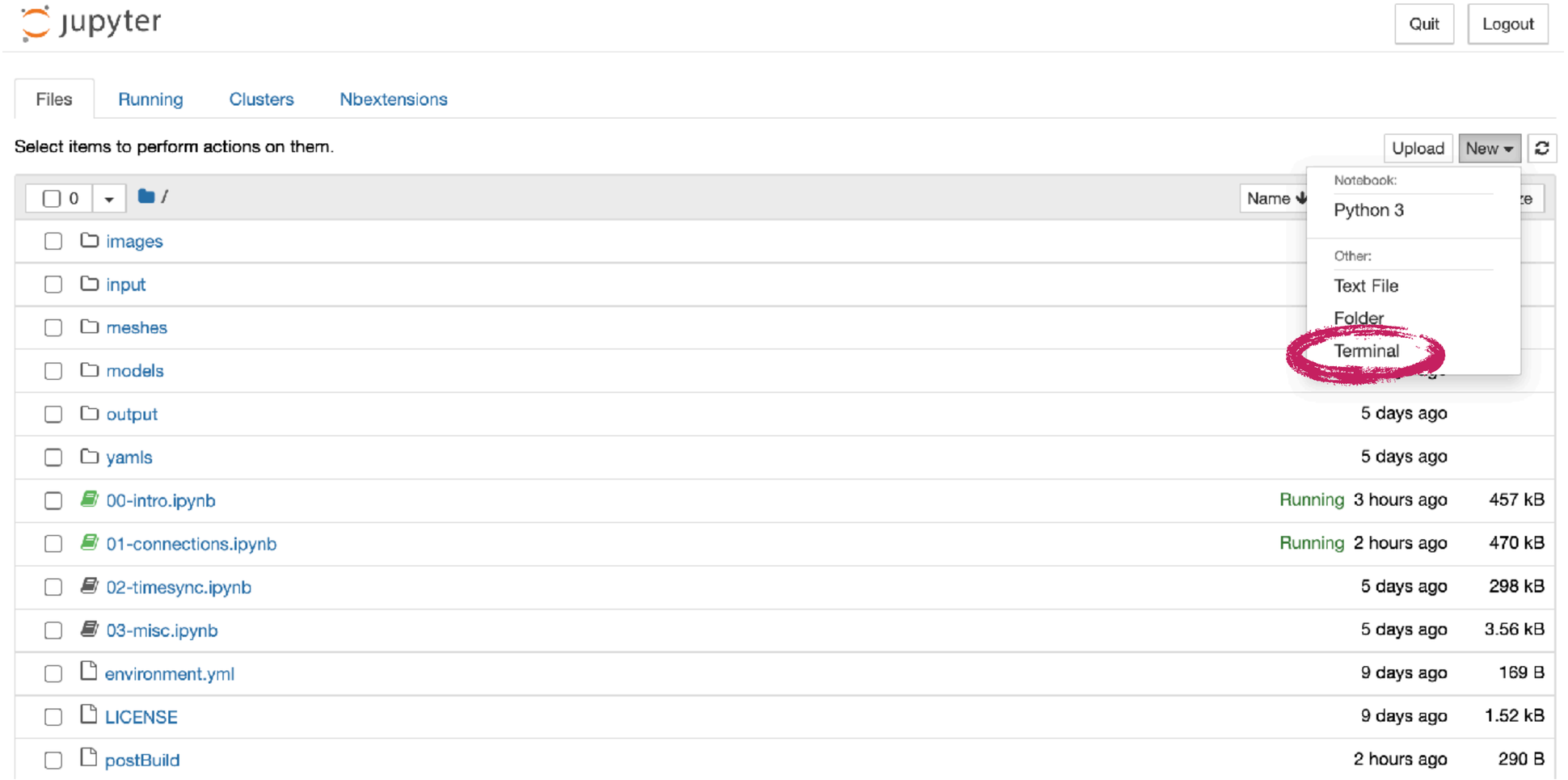
COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python



COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python

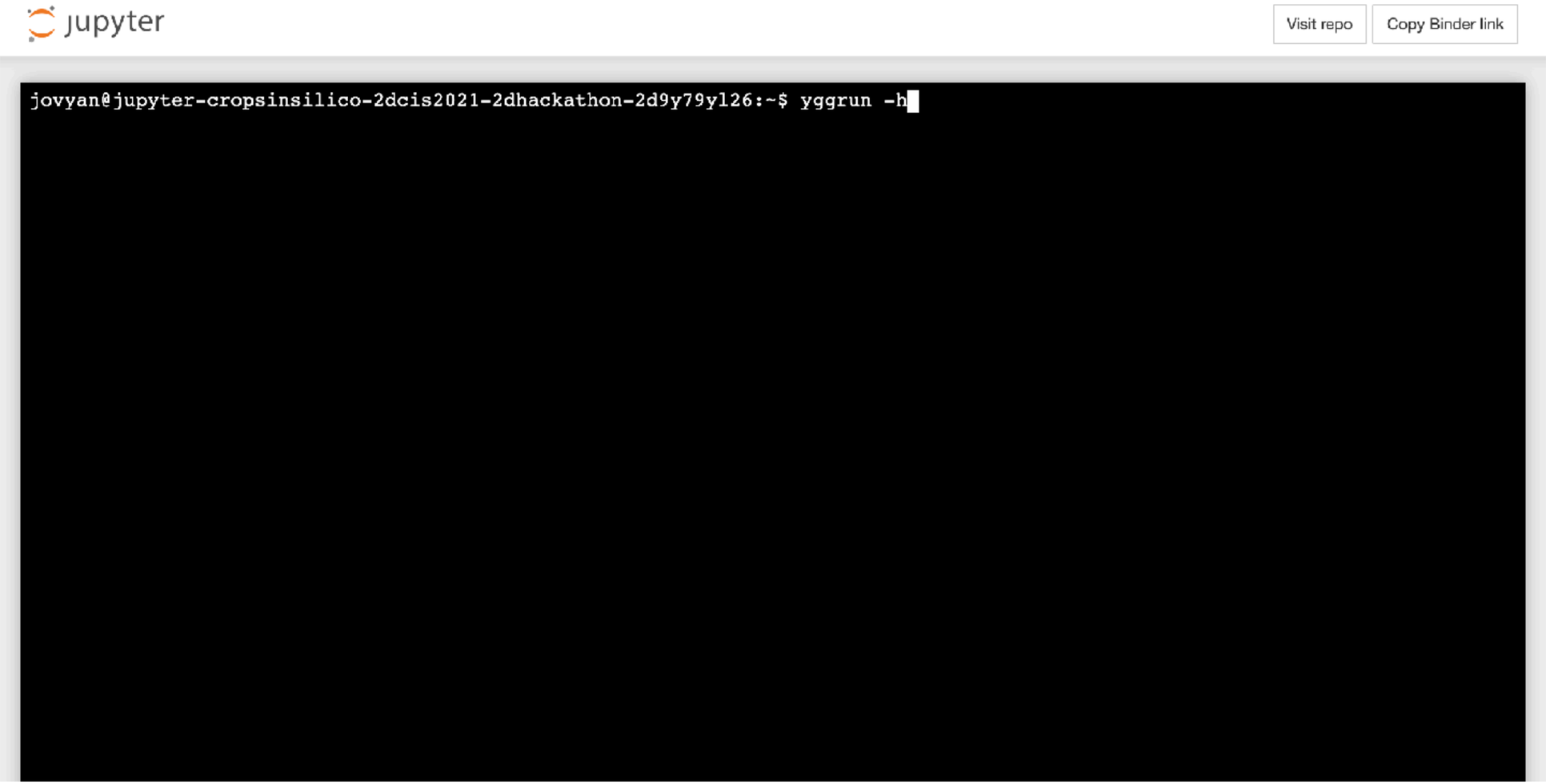


COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python

COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python



COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python

jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2d9y79yl26:~$ yggrun -h
usage: Run an integration. [-h] [--loglevel LOGLEVEL] [--rmq-loglevel RMQ_LOGLEVEL] [--client-loglevel CLIENT_LOGLEVEL]
                           [--validate-components] [--validate-messages {False,True,First}] [--namespace NAMESPACE]
                           [--host HOST] [--vhost VHOST] [--user USER] [--password PASSWORD] [--cluster CLUSTER]
                           [--default-comm DEFAULT_COMM] [--production-run] [--debug]
                           yamlfile [yamlfile ...]

positional arguments:
  yamlfile              One or more yaml specification files.

optional arguments:
  -h, --help            show this help message and exit
  --loglevel LOGLEVEL   Logging level for yggdrasil operations.
  --rmq-loglevel RMQ_LOGLEVEL, --rmqloglevel RMQ_LOGLEVEL
                        Logging level for RabbitMQ operations.
  --client-loglevel CLIENT_LOGLEVEL, --clientloglevel CLIENT_LOGLEVEL
                        Logging level for yggdrasil operations on model processes.
  --validate-components, --validatecomponents
                        Validate components on creation using their JSON schema (Decreases performance).
  --validate-messages {False,True,First}, --validatemessages {False,True,First}
                        Which messages should be validated during communication. 'True': all messages (decreases
                        performance), 'False': no messages, or 'First': only the first message a comm sends/receives.
  --namespace NAMESPACE
                        RabbitMQ namespace.
  --host HOST           RabbitMQ host address.
  --vhost VHOST         RabbitMQ virtual host address.
  --user USER           RabbitMQ username.
  --password PASSWORD   RabbitMQ password.
  --cluster CLUSTER     Cluster that should be used.
  --default-comm DEFAULT_COMM, --defaultcomm DEFAULT_COMM
                        Comm type that should be used by default.
```

COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python

jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2d9y79yl26:~$ yggrun yamls/shoot_v3.yml yamls/roots_v1.yml yamls/light_v
1_python.yml yamls/timesync.yml --production-run
```

COMMAND LINE INTERFACE (CLI)

Run yggdrasil integration from the command line w/o opening Python

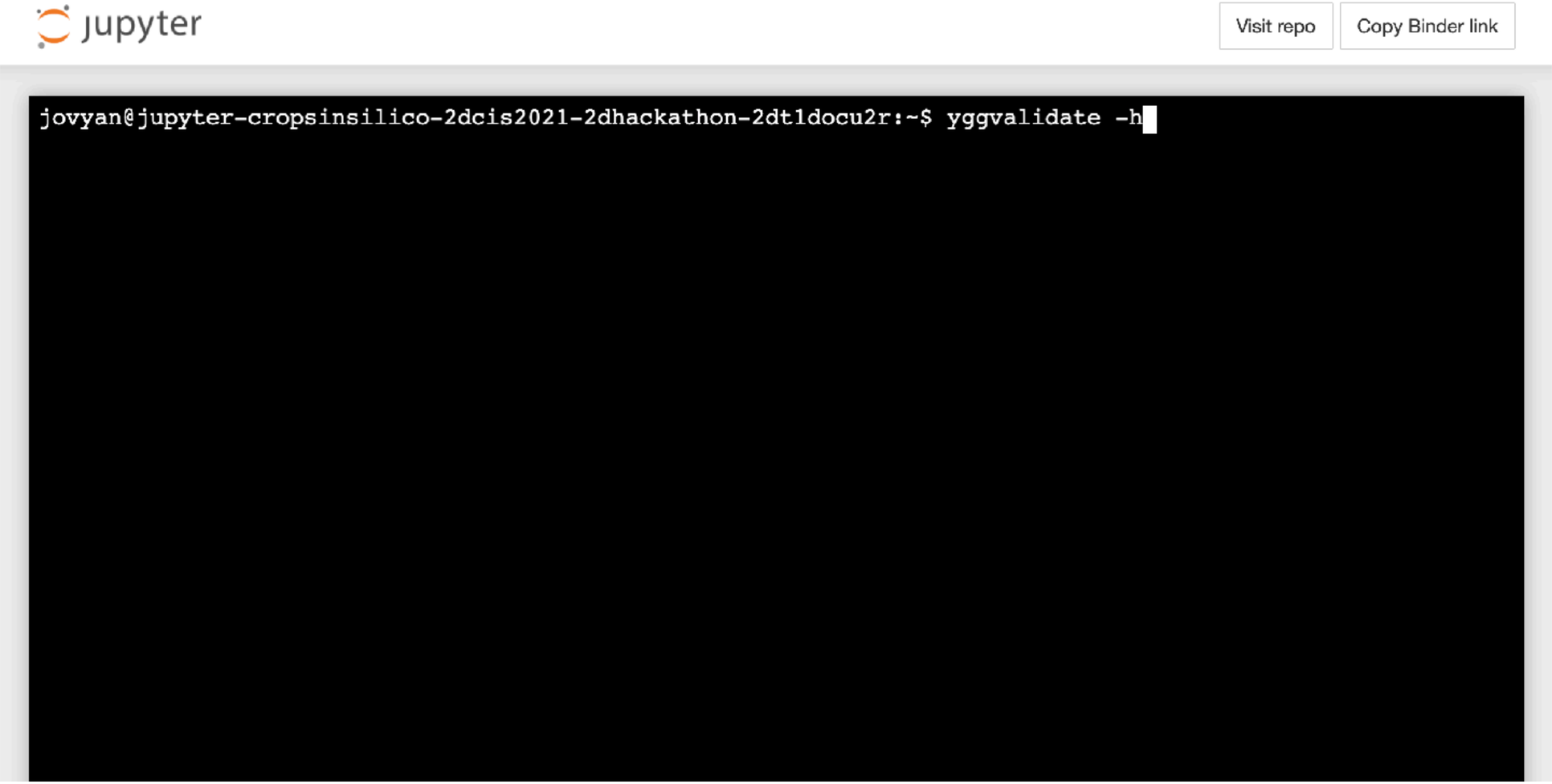
jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2d9y79yl26:~$ yggrun yamls/shoot_v3.yml yamls/roots_v1.yml yamls/light_v
1_python.yml yamls/timesync.yml --production-run
INFO:1423:runner.startDrivers[499]:YggRunner(runner): Starting I/O drivers and models on system jupyter-cropsinsilico-2dcis
2021-2dhackathon-2d9y79yl26 in namespace yggdrasil with rank 0
/srv/conda/envs/notebook/bin/python /home/jovyan/models/ygg_light_v0.py
/srv/conda/envs/notebook/bin/python /home/jovyan/models/shoot_v3.py 0.0 48.0 6.0
/srv/conda/envs/notebook/bin/python /home/jovyan/models/roots_v1.py 0.0 2.0 0.5
End of input from temp_doy.
timesync server: End of input.
INFO:1423:runner.waitModels[553]:YggRunner(runner): shoot finished running.
No more messages from model process.
INFO:1423:DSLModelDriver.after_loop[131]:TimeSyncModelDriver(shoot2root): returncode = 0
INFO:1423:runner.waitModels[559]:YggRunner(runner): shoot finished exiting.
INFO:1423:runner.waitModels[553]:YggRunner(runner): light finished running.
INFO:1423:runner.waitModels[559]:YggRunner(runner): light finished exiting.
INFO:1423:runner.waitModels[553]:YggRunner(runner): roots finished running.
INFO:1423:runner.waitModels[559]:YggRunner(runner): roots finished exiting.
INFO:1423:runner.waitModels[553]:YggRunner(runner): shoot2root finished running.
INFO:1423:runner.waitModels[559]:YggRunner(runner): shoot2root finished exiting.
INFO:1423:runner.waitModels[573]:YggRunner(runner): All models completed
INFO:1423:runner.run[374]:YggRunner(runner):                 init       0.000001
INFO:1423:runner.run[374]:YggRunner(runner):         load drivers       0.164377
INFO:1423:runner.run[374]:YggRunner(runner):        start drivers       0.372987
INFO:1423:runner.run[374]:YggRunner(runner):           run models       23.552745
INFO:1423:runner.run[374]:YggRunner(runner):              at exit       0.054842
INFO:1423:runner.run[376]:YggRunner(runner): ========================================
INFO:1423:runner.run[377]:YggRunner(runner):                Total       24.144952

jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2d9y79yl26:~$
```

COMMAND LINE INTERFACE (CLI)

Validate sets of YAML files for integration



COMMAND LINE INTERFACE (CLI)

Validate sets of YAML files for integration

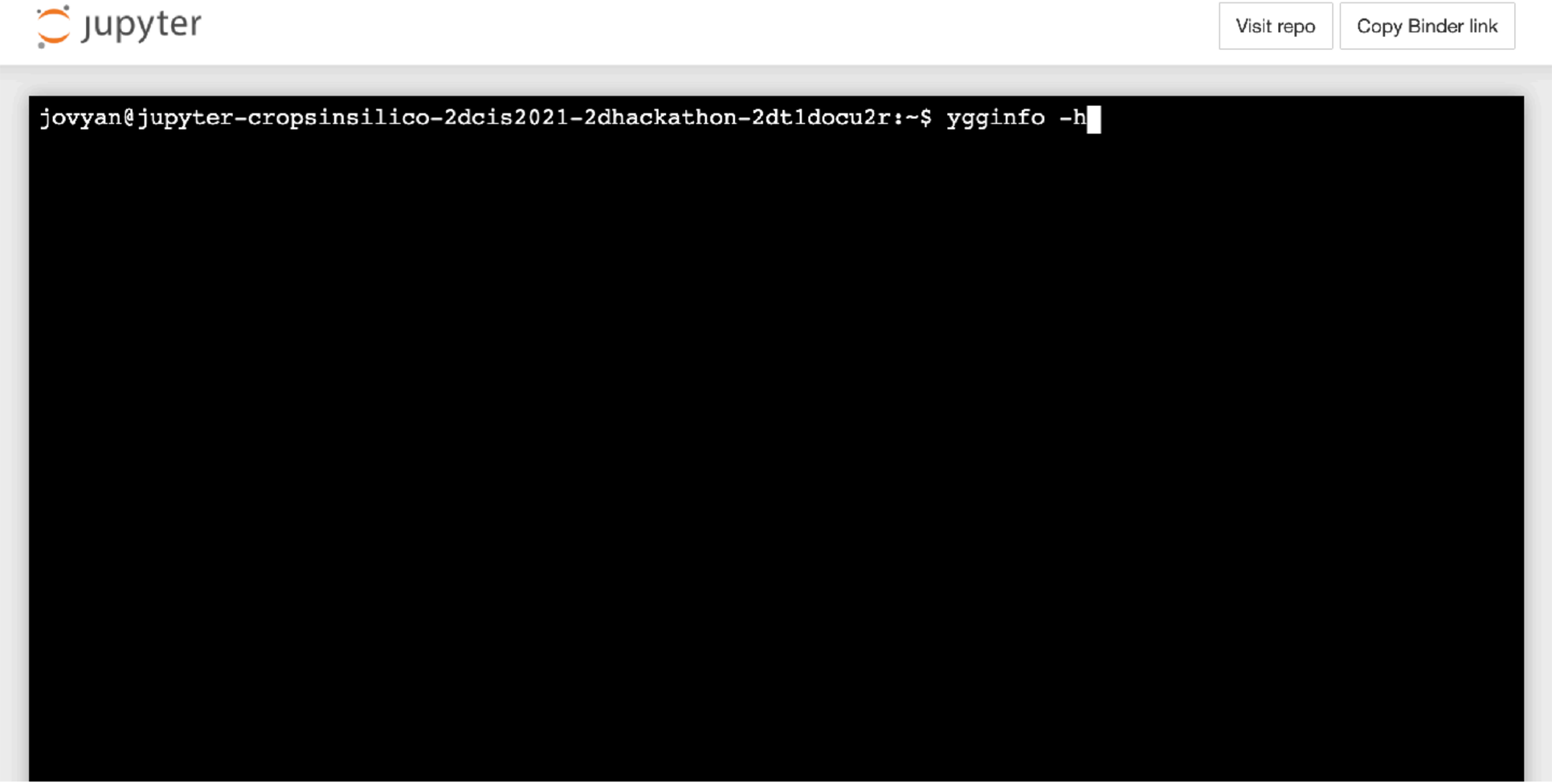
jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2dt1docu2r:~$ yggvalidate -h
usage: Validate a set of YAML specification files for an integration. [-h] yamlfile [yamlfile ...]

positional arguments:
  yamlfile    One or more YAML specification files.

optional arguments:
  -h, --help  show this help message and exit
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2dt1docu2r:~$
```

COMMAND LINE INTERFACE (CLI)

Get information about the yggdrasil installation



COMMAND LINE INTERFACE (CLI)

Get information about the yggdrasil installation

jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2dt1docu2r:~$ ygginfo -h
usage: Display information about the current yggdrasil installation. [-h] [--no-languages]
                                                                     [--no-comms] [--verbose]
                                                                     {compiler,linker,archiver} ...

optional arguments:
  -h, --help            show this help message and exit
  --no-languages        Don't print information about individual languages.
  --no-comms            Don't print information about individual comms.
  --verbose, -v         Increase the verbosity of the printed information.

tool:
  Compilation tool types to get info about.

  {compiler,linker,archiver}
    compiler            Get information about a compiler.
    linker              Get information about a linker.
    archiver            Get information about a archiver.
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2dt1docu2r:~$
```

jupyter

Visit repo | Copy Binder link

```
INFO:14976:command_line.func[667]:yggdrasil info:
Location                                /Users/langmm/yggdrasil/yggdrasil
Version                                 v1.6.3+3.gc31f41f5b.dirty
Languages                               matlab, c, cmake, cpp, make, osr, python, function, fortran, timesync, executable
, R
Communication Mechanisms                zmq, ipc
Default Comm Mechanism                  zmq
Config File                             /Users/langmm/miniconda3/envs/conda36/.yggdrasil.cfg
Installed Languages:
    R:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/Rscript
        Version                         R scripting front-end version 3.6.3 (2020-02-29)
    C:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/clang
        Version                         clang version 11.1.0
    CMAKE:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/cmake
        Version                         cmake version 3.20.1
    CPP:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/x86_64-apple-darwin13.4.0-clang++
        Version                         clang version 11.1.0
    EXECUTABLE:
        Version                         Darwin-18.6.0-x86_64-i386-64bit
    FORTRAN:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/x86_64-apple-darwin13.4.0-gfortran
        Version                         GNU Fortran (GCC) 9.3.0
    FUNCTION:
        Location                        None
        Version                         0
    MAKE:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/make
        Version                         GNU Make 4.3
    MATLAB:
        Location                        /Applications/MATLAB_R2019a.app/bin/matlab
        Version                         R2019a
    OSR:
        Location                        /Users/langmm/OpenSimRoot/OpenSimRoot/StaticBuild/OpenSimRootYgg
        Version                         Darwin-18.6.0-x86_64-i386-64bit
    PYTHON:
        Location                        /Users/langmm/miniconda3/envs/conda36/bin/python
        Version                         Python 3.6.13
    TIMESYNC:
        Location                        None
        Version                         None
```

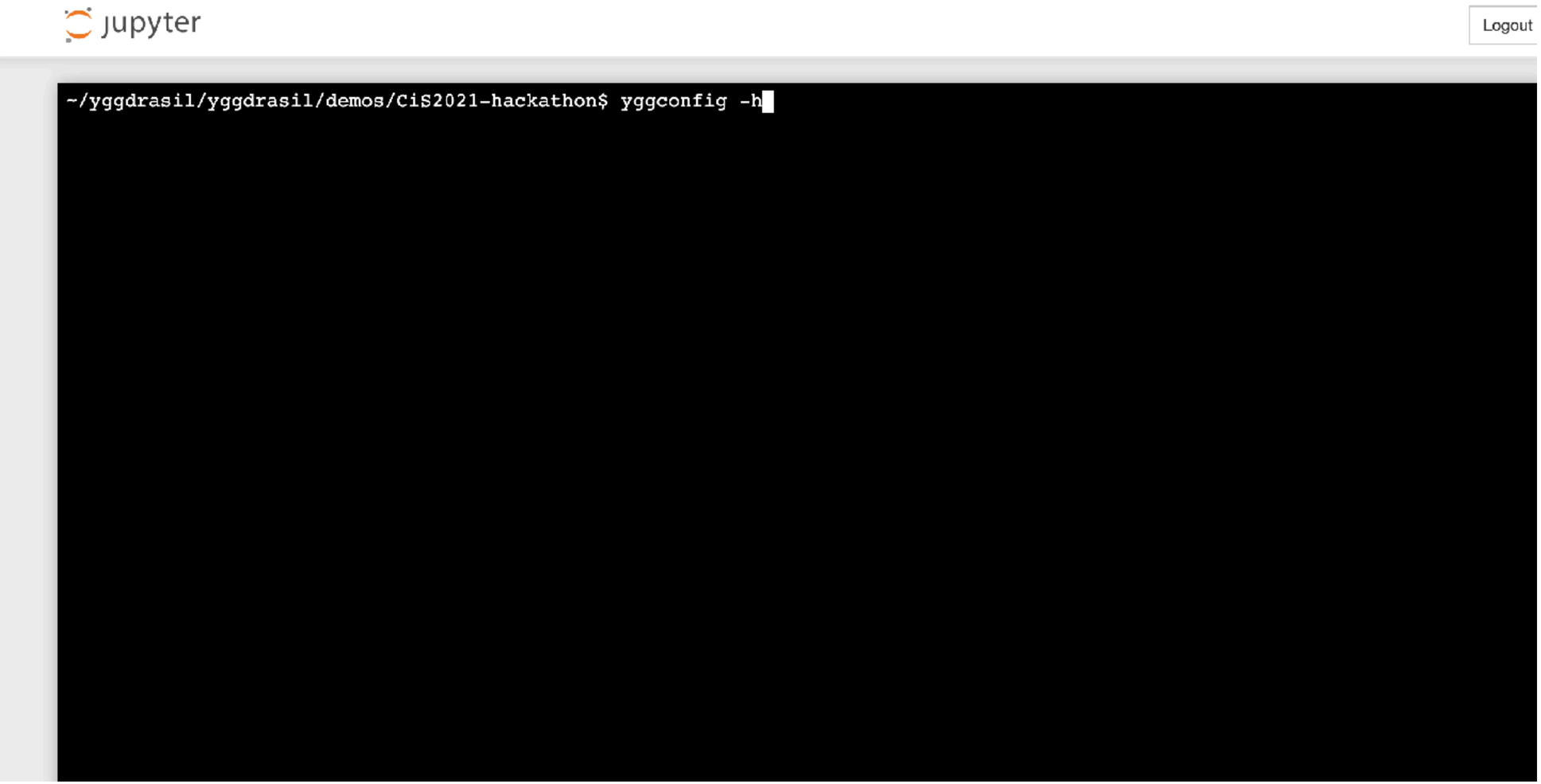
jupyter

Visit repo | Copy Binder link

```
Languages Not Installed:
    SBML:
        Language Installed              True
        Base Languages Installed        True
        Dependencies Installed          False
        Dependencies Not Installed      ['roadrunner']
        Interface Installed             True
        Comm Installed                  True
        Configured                      True
        Disabled                        False
    LPY:
        Language Installed              True
        Base Languages Installed        True
        Dependencies Installed          False
        Dependencies Not Installed      ['openalea.lpy']
        Interface Installed             True
        Comm Installed                  True
        Configured                      True
        Disabled                        False
Comms Available for All Languages:
    IPC
    ZMQ
Comms Available for Some/No Languages:
    VALUE:
        Available for                   ['PYTHON']
        Not Available for               ['C', 'CMAKE', 'CPP', 'FORTRAN', 'MAKE', 'MATLAB', 'R', 'TIMESYNC']
    RMQ_ASYNC:
        Available for                   []
        Not Available for               ['C', 'CMAKE', 'CPP', 'FORTRAN', 'MAKE', 'MATLAB', 'PYTHON', 'R', 'TIMESYNC']
    BUFFER:
        Available for                   ['PYTHON']
        Not Available for               ['C', 'CMAKE', 'CPP', 'FORTRAN', 'MAKE', 'MATLAB', 'R', 'TIMESYNC']
    RMQ:
        Available for                   []
        Not Available for               ['C', 'CMAKE', 'CPP', 'FORTRAN', 'MAKE', 'MATLAB', 'PYTHON', 'R', 'TIMESYNC']
```

COMMAND LINE INTERFACE (CLI)

Change yggdrasil settings



COMMAND LINE INTERFACE (CLI)

Change yggdrasil settings

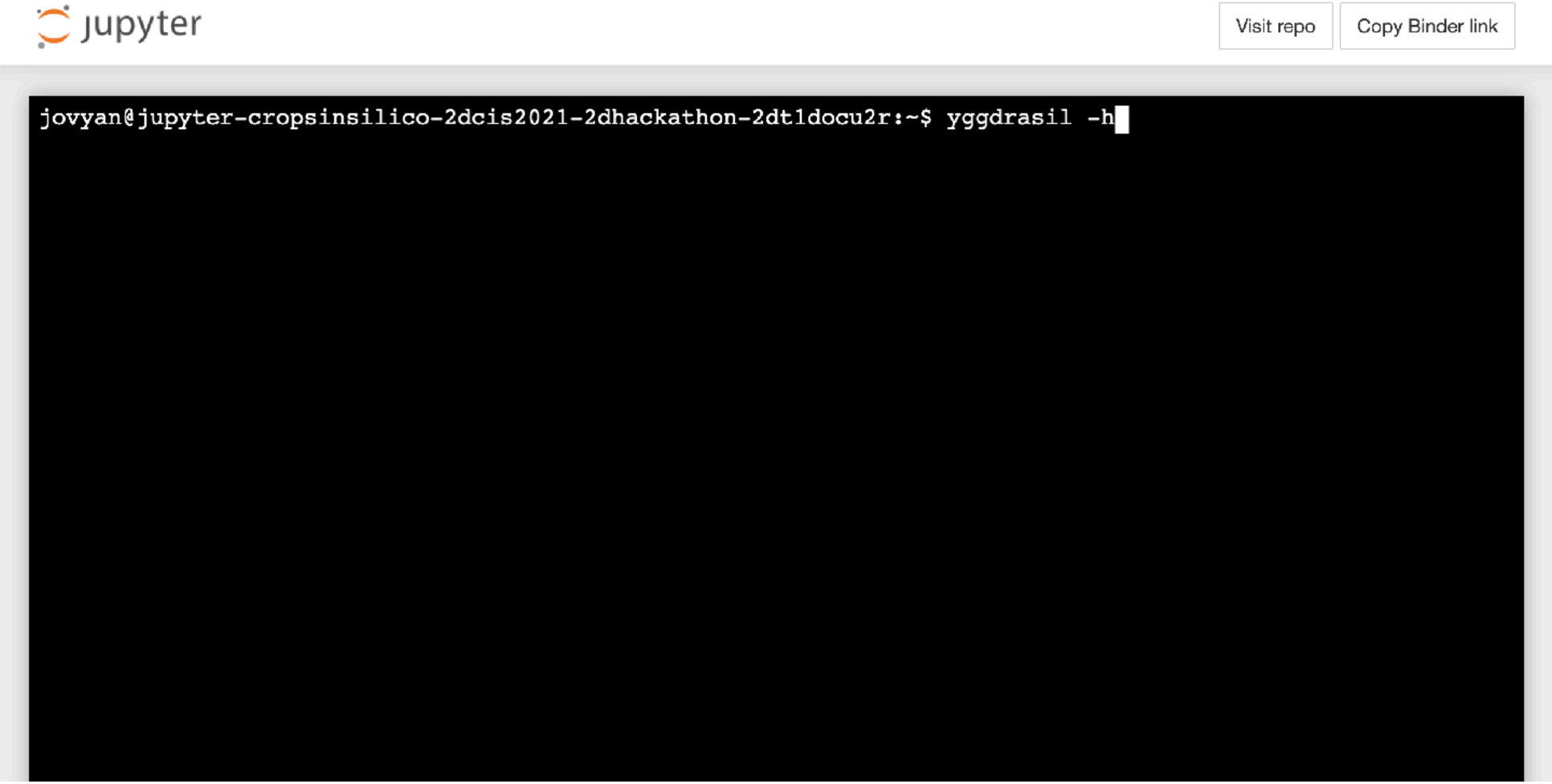
jupyter

Logout

```
~/yggdrasil/yggdrasil/demos/CiS2021-hackathon$ yggconfig -h
usage: Update the user config file. [-h] [--languages LANGUAGES_FLAG [LANGUAGES_FLAG ...]] [--show-file] [--remove-file]
                                    [--overwrite]
                                    [--disable-languages {c,c++,cpp,cxx,fortran,R,r,matlab,python,cmake,make,lpy,osr,sbml,e
xecutable,function,timesync} [{c,c++,cpp,cxx,fortran,R,r,matlab,python,cmake,make,lpy,osr,sbml,executable,function,timesync
} ...]]
                                    [--enable-languages {c,c++,cpp,cxx,fortran,R,r,matlab,python,cmake,make,lpy,osr,sbml,ex
ecutable,function,timesync} [{c,c++,cpp,cxx,fortran,R,r,matlab,python,cmake,make,lpy,osr,sbml,executable,function,timesync}
 ...]]
                                    [--quiet] [--allow-multiple-omp] [--dont-allow-multiple-omp]
                                    [--c-compiler C_COMPILER] [--c++-compiler C++_COMPILER]
                                    [--fortran-compiler FORTRAN_COMPILER] [--c-linker C_LINKER] [--c++-linker C++_LINKER]
                                    [--fortran-linker FORTRAN_LINKER] [--c-archiver C_ARCHIVER]
                                    [--c++-archiver C++_ARCHIVER] [--fortran-archiver FORTRAN_ARCHIVER]
                                    [--macos-sdkroot MACOS_SDKROOT] [--disable-matlab-engine-for-python]
                                    [--enable-matlab-engine-for-python] [--hide-matlab-libiomp]
                                    [--restore-matlab-libiomp] [--osr-repository-path REPOSITORY]
                                    [languages [languages ...]]

positional arguments:
  languages             One or more languages that should be configured.

optional arguments:
  -h, --help            show this help message and exit
  --languages LANGUAGES_FLAG [LANGUAGES_FLAG ...]
                        One or more languages that should be configured.
  --show-file           Print the path to the config file without updating it.
  --remove-file         Remove the existing config file and return.
  --overwrite           Overwrite the existing file.
  --disable-languages {c,c++,cpp,cxx,fortran,R,r,matlab,python,cmake,make,lpy,osr,sbml,executable,function,timesync} [{c,c+
+,cpp,cxx,fortran,R,r,matlab,python,cmake,make,lpy,osr,sbml,executable,function,timesync} ...]
```

COMMAND LINE INTERFACE (CLI)

Other utilities



COMMAND LINE INTERFACE (CLI)

Other utilities

jupyter

Visit repo | Copy Binder link

```
jovyan@jupyter-cropsinsilico-2dcis2021-2dhackathon-2dt1docu2r:~$ yggdrasil -h
usage: Command line interface for the yggdrasil package. [-h] [--version]
                                                         {run,info,validate,compile,compile-deps,clean,
install,config,metaschema,schema,model-form-schema,dev-update,test,timing,gha}
                                                         ...

optional arguments:
  -h, --help            show this help message and exit
  --version             show program's version number and exit

subcommands:
  {run,info,validate,compile,compile-deps,clean,install,config,metaschema,schema,model-form-schema,dev-
update,test,timing,gha}
    run                 Run an integration.
    info                Display information about the current yggdrasil installation.
    validate            Validate a set of YAML specification files for an integration.
    compile             Compile a program from source files for use in an yggdrasil integration.
    compile-deps        Compile yggdrasil dependency libraries. Existing libraries are first deleted.
    clean               Remove dependency libraries compiled by yggdrasil.
    install
    config              Update the user config file.
    metaschema          Regenerate the yggdrasil metaschema.
    schema              Regenerate the yggdrasil schema.
    model-form-schema   Save/print the JSON schema for generating the model specification form.
```

Test your knowledge #9

1. Try running some of the other integrations via the CLI. What does the error look like if a YAML is missing?
2. Try running an integration with the `--debug` flag. What information does each log line include?

TEST YOUR KNOWLEDGE (10 MIN)

ADDITIONAL EXAMPLES

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessary. We will be using Github Issues to track problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder icon (or the link below).

  It may take a few moments to initialize. If you encounter an error, open an issue and try with another browser. If you still cannot launch the binder, you can install the materials on your machine by following the instructions at one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

40+ EXAMPLES IN SUPPORTED LANGUAGES

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessary. We will be using Github Issues to track problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder icon (or the link below).

  It may take a few moments to initialize. If you encounter an error, open an issue and try with another browser. If you still cannot launch the binder, you can install the materials on your machine by following the instructions at one of the links below

  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

40+ EXAMPLES IN SUPPORTED LANGUAGES

40+ EXAMPLES IN SUPPORTED LANGUAGES

README.md

Requirements

- Browser (tested on Google Chrome, Safari, Firefox)
- Github Account

Preparing for the hackathon

- Check that you can sign-in to Github, creating an account as necessa
  problems encountered during the hackathon.
- Try launching a mybinder instance by clicking on this launch binder
  It may take a few moments to initialize. If you encounter an error, oper
  you still cannot launch the binder, you can install the materials on you
  one of the links below
  - Local install (via conda)
  - Docker container

https://mybinder.org/v2/gh/cropsinsilico/CiS2021-hackathon/HEAD

Useful links

- Hackathon Repository
- Hackathon Documentation
- yggdrasil Repository
- yggdrasil Documentation
- Additional Examples
- Debugging Tips & Documented Errors

yggdrasil
v1.6.3

Search docs

CONTENTS:

Overview
Installation
Getting started
Formatted I/O
Server/Client I/O
Autowrapping Model Functions
Notes on Autowrapping C/C++ Model Functions
Conditional I/O
Transformed I/O
Timestep Synchronization
YAML Files
Configuration Files
Units
C-Style Format Strings
Debugging
OpenMP Threading in Models
Examples
Advanced
Development

» Examples

View page source

Examples

Name	Description
SaM	One model that receives input from two files and sends output to one file in the temporary directory. The YAML uses the deprecated driver-based method to represent connections.
ascii_io	A single model that receives data from three input files and sends received messages to three output files. The first file is an unstructure ASCII text file, the second file is a table delimited ASCII table that is read row by row, and the third file is a table identical to the second that is read all at once.
backwards	A single model using the deprecated "Cis" versions of the interface.
conditional_io	Three models, A, B1, & B2, that conditionally pass messages. Model A receives input from a tab delimited table and sends output to both models B1 & B2. The outputs to models B1 & B2 only succeed if the data satisfies conditions described in the YAML. Output from both models B1 & B2 are sent to a tab delimited table.
fakeplant	Four models that can be run in isolation or as an integration. Each model approximates a simplified model of a process governing plant growth. When run in isolation each model receives input from a file and sends output to a file. In the larger integration, the canopy model receives input from three files (initial structure, time steps, and some growth parameters. For each time step received, the canopy model also receives a growth rate from the growth model, computes the new structure. and send the structure to the light model. The light model receives the ambient light level from a file, calculates the intensity for each element of the structure, and sends the output to the photosynthesis model. The photosynthesis model receives temperature and CO2 from files, calculates the photosynthesis rate, and sends the result to the growth model. The growth model calculates the growth rate and sends the output to the canopy model.
formatted_io1	Two models, A & B, that send/receive ASCII data (strings). Model A receives input from a file and then sends it's output to model B. Model B receives input from model A and sends it's output to a file.
formatted_io2	Two models, A & B, that send/receive rows of table data. Model A receives input from a file and then sends it's output to model B. Model B receives input from

INTEGRATION TIPS

1. DETERMINE THE COMMUNICATION PATTERN YOU WANT
2. WRITE THE MODEL YAML WITHOUT CONNECTIONS
3. WRITE A YAML WITH CONNECTIONS TO FILES (OR DUMMY MODELS) FOR TESTING
4. WRITE A YAML WITH CONNECTIONS BETWEEN MODELS

Materials: https://github.com/cropsinsilico/CiS2021-hackathon

QUESTIONS? MODELS?

Github: https://github.com/cropsinsilico/yggdrasil

Docs: https://cropsinsilico.github.io/yggdrasil/

Paper: https://doi.org/10.1093/insilicoplants/diz001

Project Website: http://cropsinsilico.org/