# General instruction for running the model

Simple instruction would be:

Specify the main path for finding the scenario file and parameter file, click save to initialize the inputs, and click run to run the model step by step. That’s it ^-^

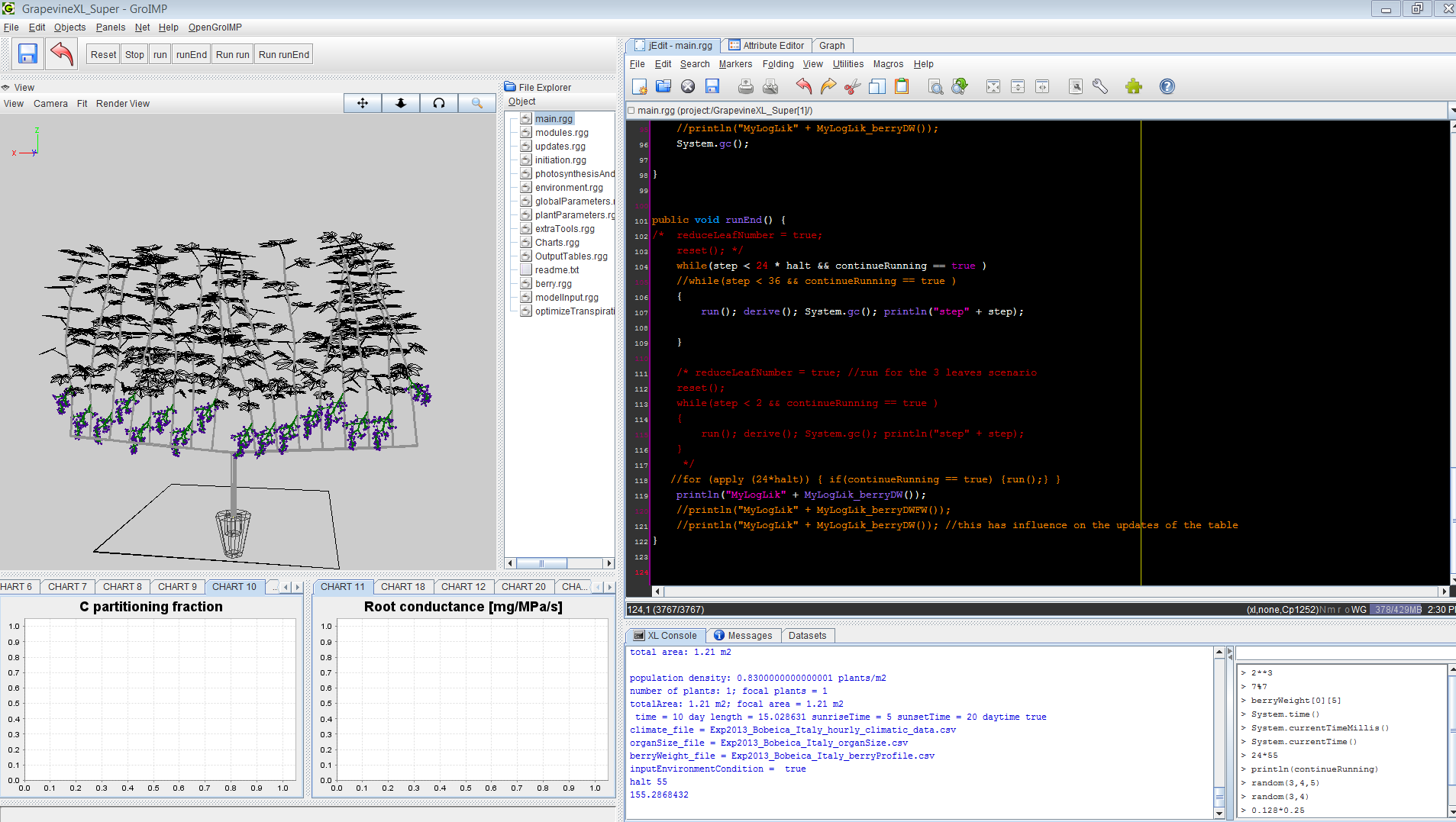
Preparation:

First you need to install GroIMP.exe or the linux version (<https://sourceforge.net/projects/groimp/>) and Java run environment 8.0 (<http://www.oracle.com/technetwork/java/javase/downloads/jre8-downloads-2133155.html>).

The model also use the java common math 3.5 as external library. Put the library in the GroIMP/ext folder and load it: import org.apache.commons.math3.distribution.\*;

After install those two things, you should be able to open the GroIMP and see some examples in the File tab – show examples. You can then open the GrapeXL code through File-open option.

When you open the code, first thing to do is go to the File Explorer, global parameters, then go down to find the definition of MainPath. Then you change the MainPath into the folder where you put the model input and model scenario file.

****

I have written some instructions about how to read external input file, and how to output the model results in below. Right now, you can define the number of cordens (1-2 cordens I did not go further yet), shoot number per corden, and leaf number per shoot in the folder of model scenarios – “model.input.data.Bobeica.2013.txt”. Normally I update these values in the excel, and then use R to generate the txt file. Mode

CORDEN\_NUM=2

SHOOT\_NUM=8

MAX\_PHYTOMER\_NUM=15

MAX\_LEAF\_NUMBER=15

STARTING\_LEAF\_NUMBER=4

**Model structure:**

\* main.rgg

Control the running of the model, sequence of implementing different functions, snapshots, replication of running, data outputs

\* modules.rgg

This file includes all the plant organ definitions and growth functions. A hierarchical structure was used for improving the readability and efficiency of the program. In current model, we use: organ (base of all organs) – growing organ – visible organ. Growing organ includes, leaf, internode, root, Woodstock, berry. Visible organ only include leaf which can do the photosynthesis. Changing the category of a certain organ could affect its functions. Xylem water potential and nitrogen uptake were calculated in the root module; whole-plant water flux and phloem sucrose concentration and all plant level properties were calculated in PlantBase module.

\* berry.rgg

Includes all the functions that controls berry growth

\* updates.rgg

Includes the calculating sequence for plant organs, updating all the rate and state variables

\* initiation.rgg

Initialize the initial running condition of the model and field settings, rows

\* photosynthesisAndTranspiration

Includes all the functions used for calculating photosynthesis and transpiration

\* environment.rgg

Includes the functions for calculating light position, direction and intensity, sunrise and sunset time, temperature and other environment variable

\* globalParameters.rgg

Definition of main path/file structure of the model. Specify which parameter and scenario file to read, and the output path. It also defines the global constants and all other parameters for controlling model running.

\* plantParameters.rgg

Definition of all plant related parameters, and functions for reading those parameters from external file

\* extraTools.rgg

Collection of useful tools, measure stick, read CSV data, leaf shape

\* Charts.rgg

Collections of all plots; user could select different plots. All charts need to be programmed in order to visualize it except the one I already did.

\* OutputTables.rgg

Collections of all output tables

\* readme.txt

Some explanation of the model

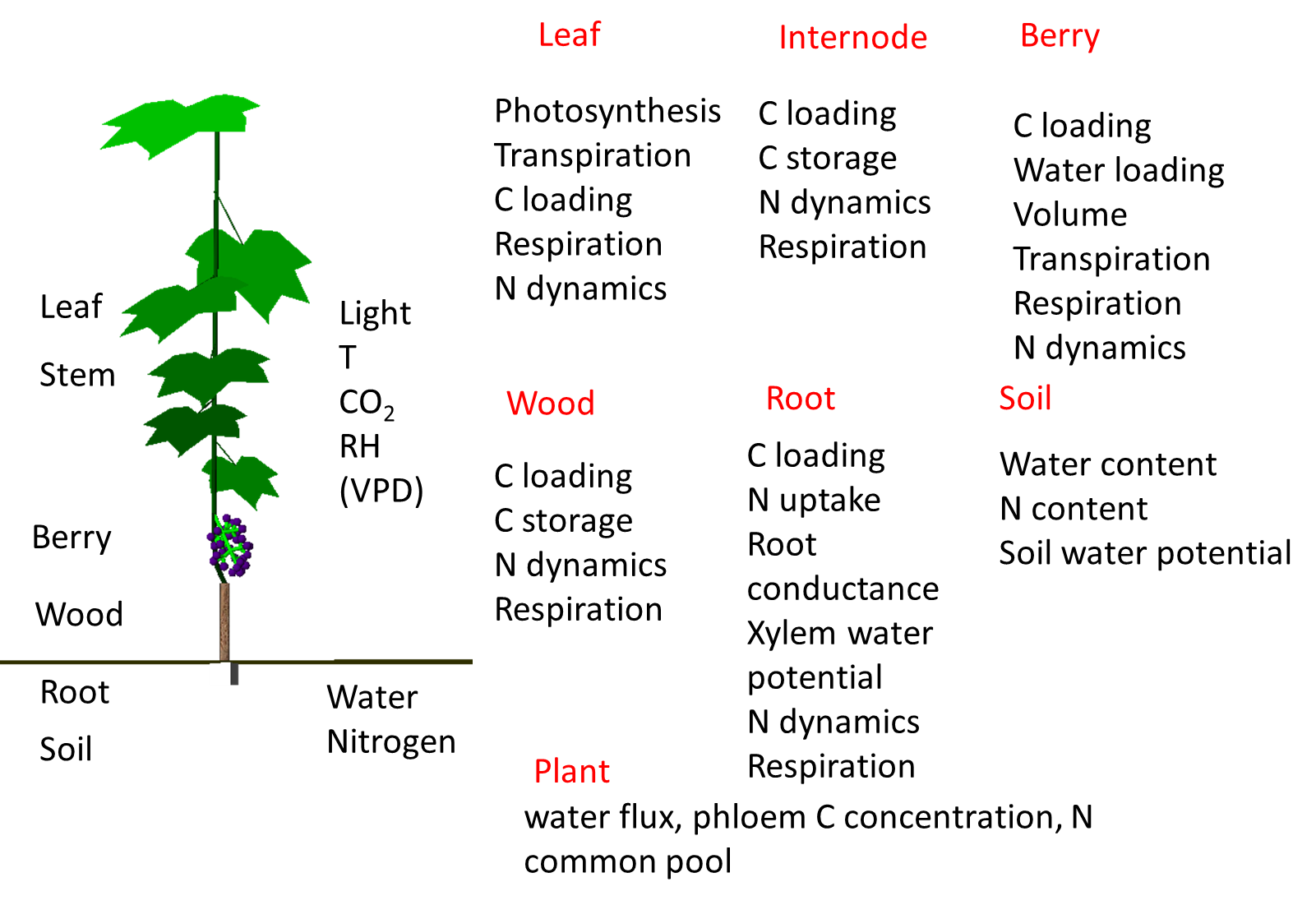
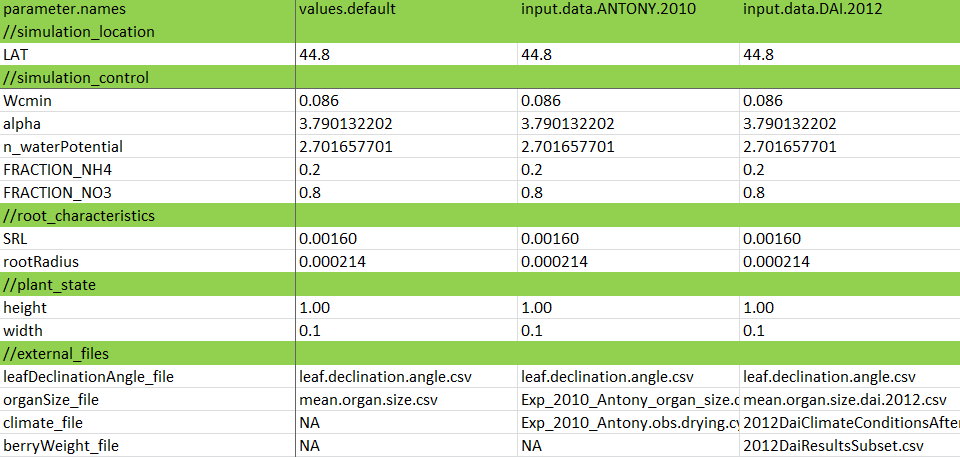
****

Fig. 1 Model components and essential processes of the ecophysiological simulation model of vigne and vine.

**Model input**

1. All the model input files, e.g. climatic conditions, organ size, berry size are stored in the folder of Model\_input
2. The choice about which files to read are specified in the model scenario files, located in the folder of Model\_scenarios.
3. The choices of which scenario and parameter files to read are specified in the globalParameters.rgg.
4. The model scenario files can be created by the R script: ‘model scenario creation.r’, located in the Model\_scenarios\Model scenario file creation\. The R script read the ‘model input data list.xlsx’, and creates the corresponding txt file. The names of file as well as initial conditions for the model are defined in this excel file. The first line defines the name of this scenario, while the names below define the initial conditions and files for read.

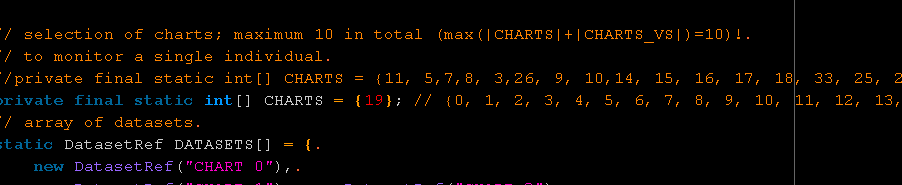


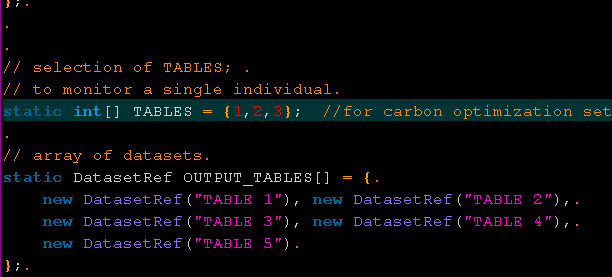
1. Similarly plant parameters are stored in \Model\_scenarios\model plant parameter list git\. The parameter file can also be created by the R script: ‘model scenario creation.r’.

**Model output**

The results of the model are divided into two parts: both can be turned on and off.

1. One part of the result is displayed in charts, which is specified in the Charts.rgg. For simplicity, this part mainly uses for visualization for one plant. The plant number is defined by the variable INDI\_ID\_A which can be changed in the excel of ‘model input data list.xlsx’. A description of each chart is provided in Charts.rgg. The choice of which charts to display can be specified in the array of CHARTS.
2. Charts can be exported to certain location using the following command: **DATASETS[i].**export(new FileWriter(OUTPUT\_LIGHT\_TREATMENT + "Output.chart." + i + ".csv"), ",");
3. The location for exporting (PATH\_OUTPUT) was defined in the globalParameters.rgg.



1. The second part of the result is displayed in OutputTables.rgg. There is lots of variables for output, ~400. Table 0 is for plant-level output, table 1 is for organ-level output, table 2 is for field-level output, table 3 is for optimization of transpiration, and table 4 is for optimization of carbon allocation. One can define whatever he wants to export, either for each leaf or whole plant or whole field. The choice of which output tables to update can be specified in the array of TABLES. 
2. Tables can be exported to certain location using the following command: **OUTPUT\_TABLES**[0].export(new FileWriter(OUTPUT\_CARBON\_OPTIMIZATION + "Carbon.allocation.likelihood.2016.7.8" + ".csv"), ",");