StabilityChecker

Miriam Leon

August 10, 2015

Contents

1 Introduction **2**

1.1 Installation 3

1.2 Linux 3

2 Using the package **4**

2.1 The working directory 4

2.2 The input file 5

2.2.1 Required arguments 5

2.2.2 Output 7

2.2.3 Running StabilityChecker 8

3 Examples **9**

3.1 The simple genetic toggle switch using ODEs 9

3.1.1 Background 9

3.1.2 Setting up 9

3.1.3 Running StabilityChecker 9

3.1.4 Results 9

Introduction

1.1 Installation

StabilityChecker has been developed to work on a Linux operating system, with GPU support.

The following dependencies are essential for the successful implementation of the package:

- numpy
- logging
- cuda-sim
- libsbml
- R
- ggplot2

```
$ cd StabilityChecker
$ python setup.py install
```

1.2 Linux

On linux this will copy the module `stabilitychecker` into the `lib/pythonXX/site-packages` directory, where `XX` corresponds to the python version that was used. An `exe=<dir>` variable should also be added to the `run.sh` script pointing the script to the right package installation. Alternatively the path to the module can be added to the top of the `run.sh` file without the need for installation.

```
$ export PATH=<dir>:$PATH
```

Using the package

2.1 The working directory

The working directory must contain the following files. Each one is described in detail in the section following.

input_file.xml The user input file

run.sh The shell script that will initialise the scripts

plot_posterior.R The R code to plot the results

model file This can take two formats:

- **model.cu** Cuda file of the model that is required to have this name
- **SBML model** This can have any name, as long as this is provided in the `input_file.xml`. `Cuda-sim` will create the `model.cu` file using this SBML model.

2.2 The input file

The `input.xml` file contains all the necessary information that is not contained in the model itself. This file must follow the specific format showed in the 'Examples' folder. No whitespaces are to be included between the tags.

2.2.1 Required arguments

epsilons The acceptable distance from the desired value.

- **epsilon_t** Total variance in the data
  - **start** The initial cut-off value for the distance from the desired values.
  - **end** The final cut-off value for the distance from the desired values.
- **epsilon_vcl** Within-cluster variance
  - **start** The initial cut-off value for the distance from the desired values.
  - **end** The final cut-off value for the distance from the desired values.
- **eps_cl** Number of clusters
  - **start** The initial cut-off value for the distance from the desired values.
  - **end** The final cut-off value for the distance from the desired values.

Desired final values The desired values the algorithm will converge to.

- **number_of_clusters** The number of clusters in the data,describing its stability.
- **total_variance** The total variance in the data.
- **cluster_variance** The within-cluster variance in the data.

particles The number of accepted parameter sets required to proceed to the next iteration.

number_to_sample Number of parameter sets to sample from for iteration. This needs to be greater or equal to the particles.

initial_conditions_samples Number of initial condition sets to sample for each parameter set at each iteration.

alpha A value that describes the step size for the incremental reduction of the epsilon at each iteration. This should be set to a small value (a = 0.1) to speed up the algorithm.

times The time points for the simulation are given as a whitespace delimited list.

species_numb_to_fit Which two species numbers to be fit by the algorithm. The should be whitespace delimited.

stoch_determ The type of simulation to be used. The two options are stochastic or deterministic.

model_file Whether the model provided is in SBML or cuda format. The two options are sbml and cuda.

sbml_name The name of the SBML file that will be provided. Note that if a cuda file is provided, that must be named model.cu.

parameters The parameters included in the model.

- **item** One item corresponds to one parameter. The number and order of the items must strictly correspond to those of the parameters in the model provided.
  - **name** The name of the parameter. Note that the parameters are read by order and not name from the model. The name provided here is used in the plotting module.

distribution From what distribution to sample values from. The two options are constant and uniform.

start The lower limit of the distribution to sample from.

end The upper limit of the distribution to sample from. Note that if the distribution is constant this value is not read.

initial_conditions The species included in the model.

item One item corresponds to one species. The number and order of the items must strictly correspond to those of the species in the model provided.

name The name of the species. Note that the species are read by order and not name from the model.

distribution From what distribution to sample values from. The two options are constant and uniform. Strictly two species must be set to uniform and the rest constant.

start The lower limit of the distribution to sample from.

end The upper limit of the distribution to sample from. Note that if the distribution is constant this value is not read.

2.2.2 Output

The outputs from running StabilityChecker are saved in the `results_txt_files` folder. This folder contains two files and one folder per population. The two files are the following:

- `Parameter_values_final` Each line contains the values of the parameters in the order set in the `input_file.xml` file and each line corresponds to an accepted particle in the final accepted population.
- `Parameter_weights_final` Contains the weights of each particle (parameter set) in the final accepted population.

Each population folder contains the following files:

- `data_PopulationN.txt` Each line contains the values of the parameters in the order set in the `input_file.xml` file and each line corresponds to an accepted particle.
- `data_WeightsN.txt` Contains the weights of each particle (parameter set).
- One `set_resultXX` per parameter set. Each line contains the steady state value of each species, in order specified in the `input_file.xml` file, species_numb_to_fit. Each line corresponds to one initial condition set.

The plot of the posterior is saved in the `posterior.pdf` file in the working directory.

2.2.3 Running StabilityChecker

To run StabilityChecker, and once the `input_file.xml` file is completed, the `run.sh` file must be executed. The progress of the algorithm can be followed in the `my_abc_scan.log` file. Once StabilityChecker is finished the posterior can be seen in the `posterior.pdf` file. The phase plots of the populations can be visualised by plotting the contents of the `set_resultXX` file for each parameter set.

Examples

3.1 The simple genetic toggle switch using ODEs

3.1.1 Background

3.1.2 Setting up

Model

This example does not use an SBML model but instead provides StabilityChecker with the cuda file directly. This is the `model.cu` file.

Input file

run file

3.1.3 Running StabilityChecker

The working directory must contain the following:

- The `model.cu` file
- The completed `input.xml` file
- The customized `run.sh` file
- The `plot_posterior.R` file

3.1.4 Results

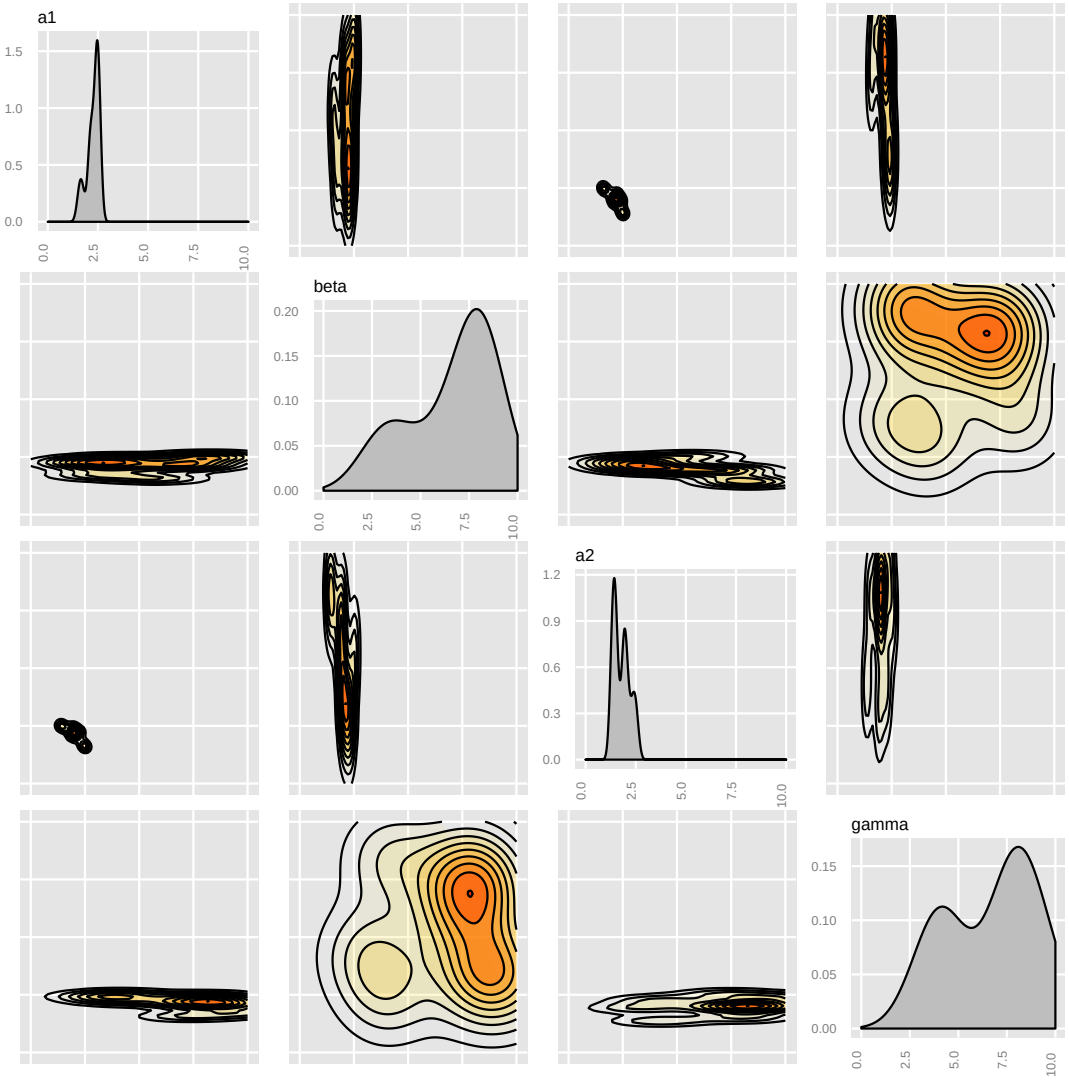

Figure 3.1: bla bla