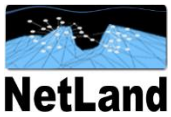


User Manual

NetLand: a Software Tool for Quantitative Modeling and Visualization of Waddington's Epigenetic Landscape

Jing Guo[1,2] and Jie Zheng[1,3,4*]

[1]School of Computer Science and Engineering, Nanyang Technological University, Singapore 639798, Singapore, [2]Bioinformatics Institute, Agency for Science, Technology, and Research (A*STAR), Singapore 138671, Singapore, [3]Genome Institute of Singapore, A*STAR, Singapore 138672, Singapore and [4]Complexity Institute, Nanyang Technological University, Singapore 637723, Singapore

Email: zhengjie@ntu.edu.sg

URL: http://netland-ntu.github.io/NetLand/

Last updated: 7 Dec, 2016

Table of Contents

1. Introduction

1.1. Waddington's epigenetic landscape

Waddington's epigenetic landscape is a visual metaphor for cellular dynamics in embryonic development, proposed by Conrad Hal Waddington (1905 - 1975) in 1957 [1]. In this landscape, the cells are like marbles rolling down a hill along trajectories towards valleys representing stable cell states (e.g. somatic cells). Despite its unifying power and unfading popularity, the original concept of Waddington's epigenetic landscape was only a qualitative metaphor that lacks rigorous interpretation of mechanisms. Nonetheless, it has inspired mathematical ideas in theoretical biology [2, 3]. Recently, quantitative computational models of Waddington's epigenetic landscape have been proposed [4-7]. Typically, these methods use concepts and mathematical models from physics, as analogues for cellular dynamics.

1.2. Limitations of computational methods for landscape modeling

However, practical applications of current modeling methods need address several limitations, such as:

- There is a lack of user-friendly software tools to calculate and visualize the landscape.
- Most of the methods are still restricted to very small networks (e.g. the 2-gene regulatory network with mutual inhibitions [6]).
- It is still difficult to visualize the 3-dimensional landscape for a regulatory network with more than two genes. Although a 52-gene regulatory network has been recently used to plot a potential landscape [7], their method either selects only two marker genes for the coordinates of the state space, or requires the system to have only two attractors.

1.3. Overview of NetLand

To address these issues, we have developed a software tool called "NetLand". Hopefully, NetLand can help make the quantitative models of Waddington's epigenetic landscape widely used by researchers of diverse backgrounds. NetLand is intended for modeling, simulation and visualization of gene regulatory networks (GRNs) and their corresponding quasi-potential landscapes. Users can import models of GRNs from a file in formats such as TSV, SBML, etc., and manually edit the network structure. Then, NetLand will automatically encode differential equations for modeling the kinetics of transcriptional regulations. Model equations and parameters can also be modified by users. As such, a computational model is constructed and can be used to simulate the dynamics of the input networks.

To display the 3D landscape for a network of more than two genes, NetLand allows users to either project the landscape to a state space of two chosen marker genes, or project to a latent space using a dimensionality reduction method called the Gaussian process dynamical model (GPDM) [8, 9]. Therefore, NetLand can provide a global picture of cellular dynamics for a user-specified GRN. Although we designed the NetLand software originally for modeling stem cell fate transitions, it can also be used to study other cellular phenotypes, such as cancer cell death, cellular ageing, etc.

The source code of NetLand was written in Java, and thus it can run under Microsoft Windows, Linux/Unix and Mac OS X. This user manual contains instructions about the installation and basic usage of NetLand. It also includes assessments of running time and memory usage and several case studies of GRN models.

2. Installation

NetLand runs on Microsoft Windows, UNIX/Linux and Mac OS X. We have tested NetLand under 64-bit version of Windows 7 and Windows 10, Linux Fedora 18 and Mac OS X 10.8 Mountain Lion. The Java Platform, Standard Edition Runtime Environment (JRE) of 1.7 or higher, is required to be installed (Java SE is available at **http://java.com/en/download/inc/windows_upgrade_ie.jsp**). The software NetLand can be downloaded for free at **http://netland-ntu.github.io/NetLand/.** Below is a list of external dependencies, followed by instructions for installing NetLand on different operating systems.

2.1. External libraries

A number of external open-source libraries are bundled with the NetLand program. The external library dependencies are listed in Table 1.

Table 1. The list of external libraries bundled with NetLand.

Package	Description	License	Version
GeneNetWeaver (GNW) [10]	NetLand uses the network graphical user interface (GUI) of GNW.	MIT license	3.1
libSDE	libSDE is a Java library for numerical integration of stochastic differential equations (SDEs).	MIT license	1.0.3
JMathPlot	JMathPlot is a light Java library designed to allow easy plotting of 2D and 3D data in a Java panel. It is used to plot the simulated time series.	BSD License	1.0.1
JLaTeXMath	A Java API to render LaTeX project logo. It is used to render mathematical equations in a user-friendly format.	GNU General Public License (GPL)	1.0.3
Dizzy[11]	Dizzy is a chemical kinetics stochastic simulation software package. It is used to perform stochastic simulations using the Gillespie algorithm.	GNU Library or "Lesser" General Public License (LGPL)	1.11.4
BeautyEye	BeautyEye is a Java Swing Look and Feel.	Apache License 2.0	3.7
Jzy3d	Jzy3d is an open source Java library for drawing 3D scientific data.	The BSD 3-Clause License	0.9.1
JOGL 2	JOGL 2 is designed to provide hardware-supported 3D graphics to applications written in Java. It is required by Jzy3d.	The New BSD 2-Clause License	2.1.5
GlueGen	GlueGen is a tool which automatically generates the Java and JNI code necessary to call C libraries.	The New BSD 2-Clause License	2.1.5
GPDM[9]	Gaussian process code in C++ including some implementations of GP-LVM and IVM		0.001

2.2. Dependencies for "GPDM" program

The Gaussian Process Dynamical Model (GPDM) comprises a nonlinear mapping between a latent space and the data space. It is employed to reduce the simulated high-dimensional data to a 2D latent space in NetLand. To run the "GPDM" program, the libraries of BLAS, LAPACK and GFORTRAN are required.

- **Windows OS:** The DLL (Dynamic Link Library) files are provided in the "GPDM/win" folder.
- **Mac OS X**: Please check if *libblas.dylib*, *liblapack.dylib* and *libgfortran.dylib* are already installed in the operating system. The default installation path is /usr/lib/. Otherwise users should install the libraries themselves.
- **Unix/Linux:** Please check if *libblas.so*, *liblapack.so* and *libgfortran.so* are already installed in the operating system. The default installation path is /usr/lib/. Otherwise users should install the libraries themselves.

Note: The actual installation directory of the three libraries might not be the default. To check if they are installed, use a terminal and run: "*locate <library name>"*.

To install the dependencies, the BLAS, LAPACK and GFORTRAN can be downloaded from

- BLAS: http://www.netlib.org/blas/
- LAPACK: http://www.netlib.org/lapack/
- GFORTRAN: https://gcc.gnu.org/wiki/GFortranBinaries

Then follow the installation instructions in the package. For Linux, users can use command line tools for package management, e.g. *yum*, *apt-get*, to install these packages.

2.3. Steps of installation

1. Download the compressed folder, named *NetLand.zip*, from GitHub: **http://netland-ntu.github.io/NetLand/**.
2. Unzip it to your desired location.

   The NetLand package should include the following files and folders:
   a) **"runNetLand.sh" (Linux/Unix or Mac OS X) and "runNetLand.bat" (Windows):** Shell scripts for launching NetLand on the different operating systems.
   b) **"main.jar":** A jar file that contains aggregate Java class files of NetLand which should be in the same directory with the shell script.
   c) **"lib"**: A folder that contains the required libraries to launch NetLand.
   d) **"GPDM"**: A folder that contains the executable files required for running the "GPDM" program to do dimensionality reduction in NetLand.
   e) Two folders, namely **"toy models"** and **"saved results"** (not necessary for launching the software), which contain files of toy network models and pre-computed results respectively.
3. Make sure you have the execute permissions to the executable files in "GPDM" folder under Linux/Unix or Mac OS X. To check the permission of a file, type the following command at a terminal: "*ls –l <filename>*". Use "*chmod 705 <filename>*" to obtain the execute permission.

If everything is installed correctly, you are ready to run a simulation (see next section). **Note that the scripts ("runNetLand.sh" and "runNetLand.bat") must be in the same directory with the "GPDM" folder and the "lib" folder.**

3. Launching NetLand

To launch NetLand:

- **Under Windows, double click the "runNetLand.bat".**
- **Under Linux, run the command "./runNetLand.sh" in a terminal.**
- **Under Mac OS X, run the command "./runNetLand.sh" in a konsole.**

For Windows users, please check if "Java" has been set in your system path. If not, you can either add it to the system path or add the full path of "java.exe" to the script ("runNetLand.bat"). To check and update the system path under Windows, you may follow the steps below:

1. From the **desktop**, right click the **Computer** icon.
2. Choose **Properties** from the context menu.
3. Click the **Advanced system settings** link.
4. Click **Environment Variables**. In the section **System Variables**, find the **PATH** environment variable and select it. Click **Edit** and check if the path of Java is in the "Variable value". Generally, the full path of Java program looks like "C:\Program Files\Java\jre<version>\bin".
5. If Java is not in the system path, then please add the path of Java to the **PATH** environment variable in the **Edit System Variable** (or **New System Variable**) window. Click **OK**. Close all remaining windows by clicking **OK**.

Another way to run the software without changing of the system path is to update the shell script ("runNetLand.bat") by specifying the full path of Java. This script can be edited using text editors, e.g. WordPad, Notepad. For example:

```
"C:\Program Files\Java\jre1.8.0_31\bin\java.exe" -classpath "lib\*;main.jar"
WindowGUI.NetLand
```

For users of Mac OS X and Linux/UNIX, the script should be run in a terminal or konsole using the command "*bash runNetLand.sh*" or "*./runNetLand.sh*". Make sure you have the execute permissions on the script. To check the permission, the command is "*ls –l <filename>*". Use "*chmod 705 <filename>*" to obtain the execute permission.

Memory requirement

NetLand runs on JVM (Java virtual machine) using the default settings of memories. To manually change the RAM, users can add the command line arguments "*-Xms<number>m –Xmx<number>m*" to the shell scripts ("runNetLand.sh" and "runNetLand.bat") using text editors. *"-Xms"* stands for the minimum heap size, and *"-Xmx"* denotes the maximum heap size. For example, the following command allows a minimum of 2GB and a maximum of 4GB of memories used by the JVM:

```
java -Xms2048m -Xmx4096m -classpath "lib\*;main.jar " WindowGUI.NetLand
```

The memory requirement depends on the size of network model and values of parameters, e.g. the number of iterations in a simulation. A minimum memory of 20MB is required to launch NetLand. The details of memory consumption can be found in Chapter 7.

4. Model operation

Dynamical analysis using NetLand is based on the computational modeling of GRNs. A computational model typically contains three key components, i.e. molecular species (nodes), biochemical reactions (edges), and parameters. The molecular species are the products of genes, functioning as reactants or modifiers in a reaction. Biochemical reactions describe the regulatory interactions between species. Parameters control the kinetics of the reactions, such as the production and degradation rates. For example, in a reaction of transcriptional regulation, a transcriptional factor is a modifier binding to the DNA sequence, along with other molecules, to produce mRNA. The reaction rate as a parameter stands for the speed of the process.

In NetLand, the change of the concentration of a target gene is simplified into one differential equation (DE) in the form below:

$$\frac{\mathrm{d[Gene1]}}{\mathrm{d}t} = K^{+}\varphi(\mathrm{Gene1, Gene2, Gene3 \dots GeneN}) - K^{-}[\mathrm{Gene1}] + D \cdot \theta(t), \quad (1)$$

where $[Gene1]$ denotes the concentration of the products of a gene named "Gene1". K^{+} is the maximum transcriptional rate of $Gene1$, and K^{-} is the degradation rate. $\varphi(\cdot)$ represents the *cis*-regulation of $Gene1$ by other genes, $\theta(t)$ is the noise term, and D controls the strength of noise. For each target gene, $\varphi(\cdot)$ consists of two parts, i.e. an active component and an inhibitive component, denoting the regulations of activators and inhibitors by Hill equations (Eq. 2).

$$\begin{aligned}\varphi(\mathrm{Gene1, Gene2, Gene3 \dots GeneN}) &= \mathrm{BasalExpression}_{gene1} \\ &+ \sum_{i=1}^{\mathrm{m}_a}(\mathrm{BasalExpression}_{m_{ai}} + a_i * \frac{\mathrm{Genei}^{n_a}}{\mathrm{Genei}^{n_a} + \mathrm{k}_a{}^{n_a}}) \\ &+ \sum_{i=1}^{\mathrm{m}_b}(\mathrm{BasalExpression}_{m_{bi}} + b_i * \frac{\mathrm{k}_i{}^{n_i}}{\mathrm{Genei}^{n_i} + \mathrm{k}_i{}^{n_i}})\end{aligned} \quad (2)$$

In the above equation, $\mathrm{m_a}$ and $\mathrm{m_i}$ denote the numbers of activators and inhibitors respectively. Two coefficients, $\mathrm{a_i}$ and $\mathrm{b_i}$, represent the strength of activators and inhibitors to their target. $\mathrm{k_a}$ and $\mathrm{k_i}$ are activation constant and inhibition constant. $\mathrm{n_a}$ and $\mathrm{n_i}$ are Hill coefficients. $\mathrm{BasalExpression_{gene1}}$ is the basic expression value of Gene1, while $\mathrm{BasalExpression_{m_{ai}}}$ and $\mathrm{BasalExpression}_{m_{bi}}$ denote the basic values of the regulation of a specific regulator. The default values of the above parameters are defined in Table 2.

Note that this formulation may not be applicable to some other types of biochemical reaction networks, e.g. signaling networks, metabolic networks. For the analysis of those networks or models other than GRNs, we recommend users to upload an existing model in SBML format (e.g. extracted from BioModels [12] database or saved from previous model editing tools such as CellDesigner [13]), and then to edit the equations in the loaded model (which will be elaborated in the following subsections).

Table 2. The default values of parameters in Eq. 1 and 2.

Parameter	Default value
K^+	1
K^-	1
D	0.05
$\mathbf{BasalExpression}_{gene1}$	0
$\mathbf{BasalExpression}_{m_{ai}}$	0
$\mathbf{BasalExpression}_{m_{bi}}$	0
a_i	1
b_i	1
n_a	4
n_i	4
k_a	0.5
k_i	0.5

4.1. Loading a model

NetLand can directly load a dynamic model (in SBML format) or a network structure file (in formats of TSV, GML, or DOT) which will be automatically converted into a dynamic model in NetLand. To load an SBML or a network structure file, choose "**Load Network**" from the "**File**" menu (Figure 1).

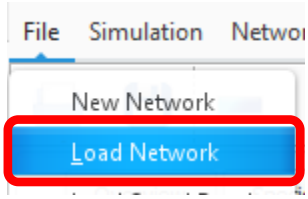


Figure 1. The "Load Network" entry in the "File" menu.

The descriptions of the aforementioned input file formats of network models are listed below.

a) **SBML (Systems Biology Markup Language)** is a representation format, based on XML, for communicating and storing computational models of biological processes. SBML can represent many different classes of biological phenomena, including metabolic networks, cell signaling pathways, regulatory networks, infectious diseases, and many others. For more information of SBML, please refer to http://sbml.org/Main_Page. NetLand can process SBML files that describe transcriptional regulations.

b) **TSV (Tab Separated Value)** is a format of a tab-delimited file used with spreadsheet software. TSV files are used for raw data and can be imported into and exported from spreadsheet software. TSV files are essentially text files, and the raw data can be viewed by text editors, though they are often used when moving raw data between spreadsheets.
Example:

A B +
B A −

c) **GML (Graph Modeling Language)** is a hierarchical ASCII-based file format for describing graphs. It has been also named Graph Meta Language.
Example:

```
graph [
    comment "Two-gene network"
    directed 1
    id 1
    label "Hello, I am a graph"
            node [
        id 1
            label "Gene A"
                    thisIsASampleAttribute 42
            ]
        node [
            id 2
            label "Gene B"
            thisIsASampleAttribute 44
            ]
            edge [
                    source 1
                    target 2
                    label "Edge from Gene A to Gene B"
            ]
 ]
```

d) **DOT** is a plain text graph description language. It is a simple way of describing graphs that both humans and computer programs can use.
Example:

```
graph ethane {
    A -- B [type=s];
}
```

4.2. Saving a model

The dynamic models can be saved in SBML format. The structure of the network can be saved in TSV format. The function of saving models can be found in the "**File**" menu (Figure 2).

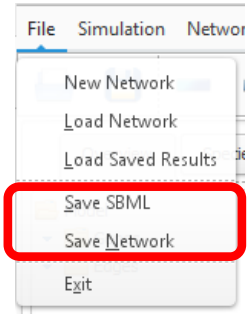


Figure 2. Menu entries to save the model.

4.3. Creating a model

By clicking the "**New Network**" entry in the "**File**" menu, a demo network with a node and a self-loop is created. Then users can modify the network, e.g. adding new nodes/edges or editing regulatory equations.

4.4. Modifying a model

NetLand provides flexible ways to load and modify a network model. In this section, we describe how to use NetLand to modify the network model, e.g. deleting nodes, adding new nodes or edges, editing parameters and reactions.

Block nodes

A node in a gene regulatory network can be blocked (i.e. inhibited or suppressed) to simulate the gene knockout experiment. By clicking the "**Block nodes**" entry in the "**Network**" menu, a dialog box will pop up (Figure 3). Genes in the network are shown on the left. To block a gene, simply select the gene and add it to the right side. If the blocked genes are involved in the reactions of the remaining genes, users will be asked to modify these reactions, i.e. removing the regulation of the blocked genes.

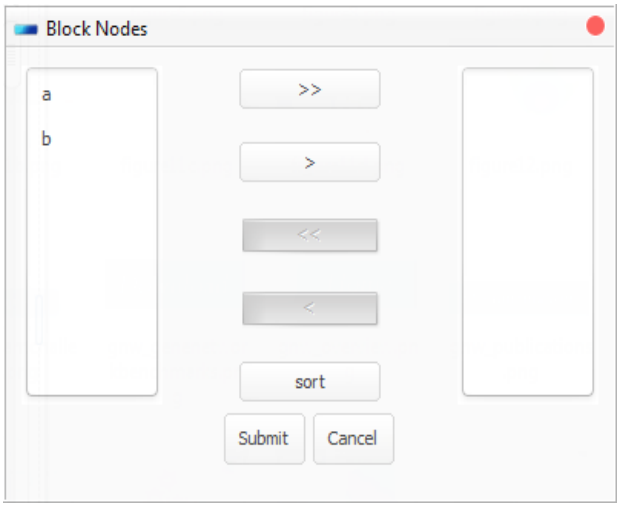


Figure 3. The dialog box for blocking nodes.

Add nodes/edges

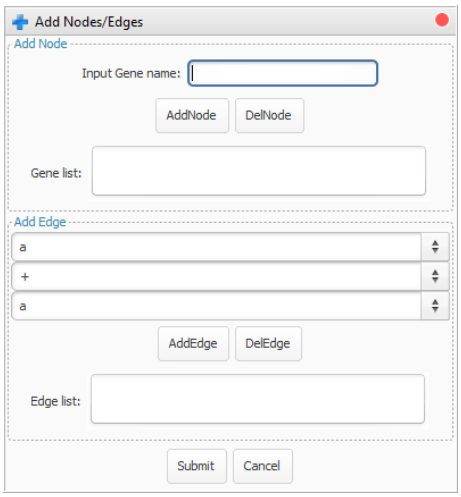
In addition to blocking genes, users can also add genes and edges to build new nodes and regulatory interactions. By clicking the "**Add nodes/edges**" entry in the "**Network**" menu, a dialog box will pop up (Figure 4a). Here are the steps to add a new gene:

1. Input the name of the new node which should be unique.
2. Click the "**AddNode**" button to add it to the "**Gene list**" below.
3. Repeat Step1 and 2 to add more genes.
4. To remove an item from "**Gene list**", select the specific gene and click "**DelNode**".
5. Click "**Submit**" to add the genes in "**Gene list**" to the network.

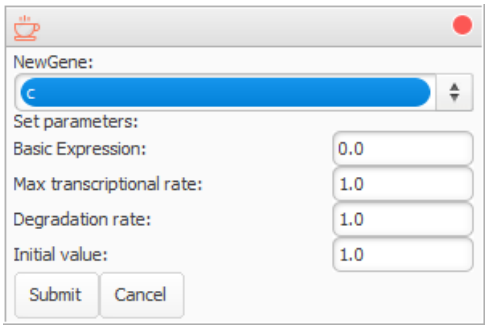
Once a new gene is added, users will be asked to verify a set of parameters including the initial value, degradation rate, *etc*. (Figure 4b). In the generation of new edges, two types of regulations are provided, i.e. activation and inhibition. The upper and bottom combo boxes represent the regulator and its target node respectively. Here are the steps to add a new edge:

1. Select the regulator, the type of regulation and its target from the three combo boxes.
2. Click the "**AddEdge**" button to add the new interaction to the "**Edge list**" below.
3. Repeat Step1 and 2 to add more edges.
4. To remove an item from "**Edge list**", select the specific entry and click "**DelEdge**".
5. Click "**Submit**" to add the edges in "**Edge list**" to the network.

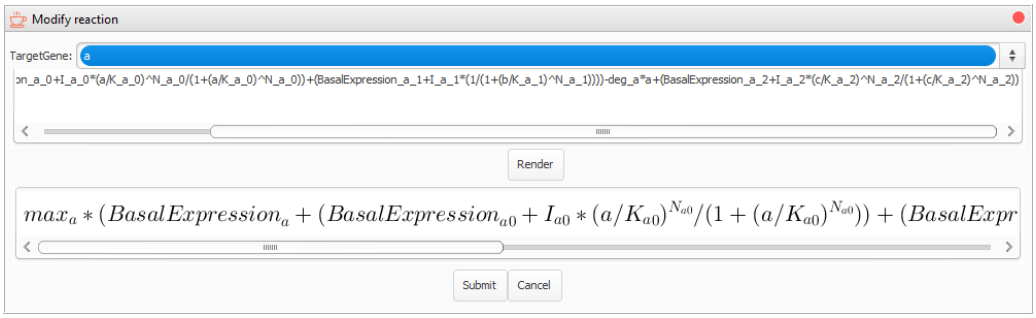
The kinetics of the new regulatory relations (associated with the edges) will be encoded with Hill equations (Eq. 2) and automatically added to the target genes' original formulas (Figure 4c). In this dialog box (Figure 4c), users can also edit the updated reactions.



(a)



(b)

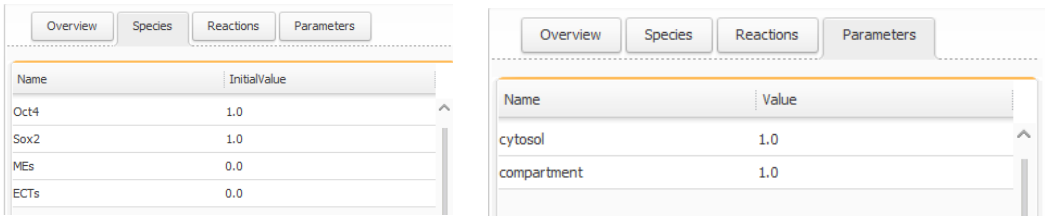


(c)

Figure 4. The dialog box (a) for adding new nodes and edges; (b) for setting the parameters of new genes; (c) for editing the new regulatory relations.

Change parameters

The initial values of the variables for genes' activities (e.g. expression levels) and parameters can be directly modified in the "**Species**" and "**Parameters**" panels respectively (Figure 5). These panels are shown on the left side of the main window of NetLand. User can double click the entry in the column labeled "Value" (for parameters) or "InitialValue" (for species) to change the number.

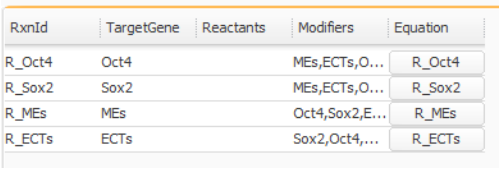


(a) (b)

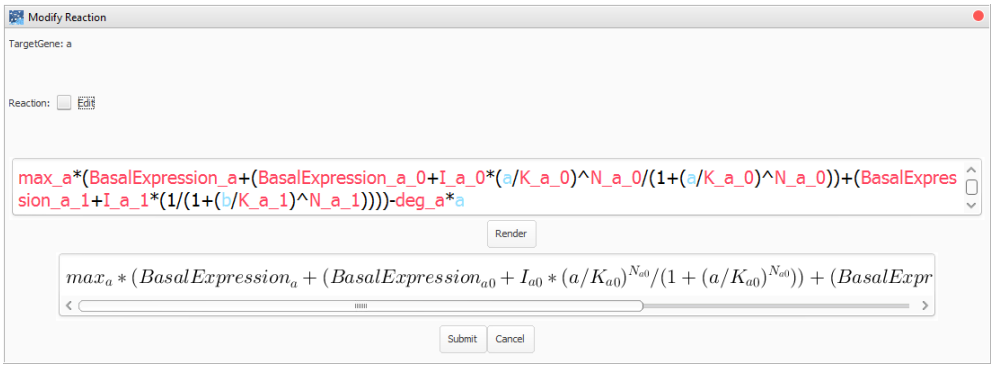
Figure 5. The panels of (a) "**Species**" and (b) "**Parameters**". The initial values for species and parameters in the dynamic model can be modified in the two panels.

Edit reactions

The "**Reactions**" panel lists the basic information of the encoded differential equation (Eq.1) for each gene including a reaction id, the target gene, modifiers and reactants in this equation (Figure 6a). To modify the reaction of a specific gene, users can click the button at the same row in the column labeled "**Equation**". Then a dialog box titled "Modify Reactions" will pop up (Figure 6b). The parameters are colored in red, while molecular species are in blue. The "**Render**" button makes the computer algebra in the upper window display in a human-readable format below.



(a)



(b)

Figure 6. (a) The "**Reactions**" panel and (b) the dialog box for modifying the reactions. The parameters are colored in green, while molecular species are in blue.

5. Running a simulation

NetLand provides a user-friendly interface to facilitate model construction, dynamical simulation and landscape plotting. The chemical interactions in a transcriptional regulatory network can be encoded in ordinary differential equations (ODEs) for deterministic models or stochastic differential equations (SDEs) for stochastic models. Different simulators are integrated to solve different types of models. In this section, we will introduce how to use NetLand to perform dynamical simulation using different simulators. Users can click the "**Trajectory**" entry in the "**Simulation**" menu to open the main window (Figure 7).

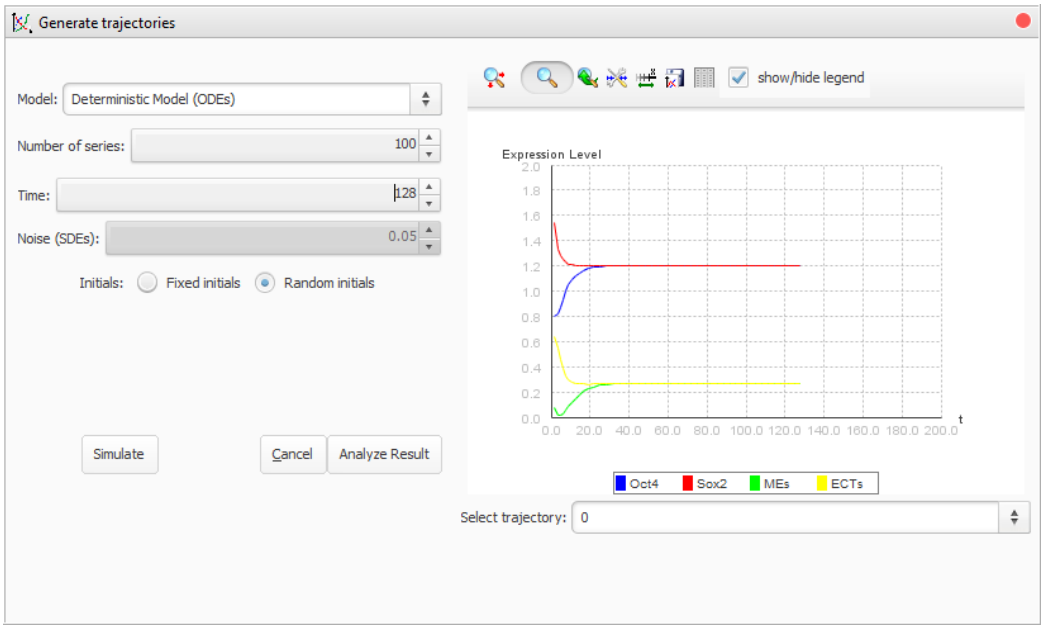


Figure 7. The main window for dynamical simulations.

5.1. Simulators

ODE solver

A deterministic simulator, i.e. the 5(4) Dormand-Prince integrator, implemented in the package of org.apache.commons.math3.ode which provides classes to solve non-stiff ordinary differential equations problems is used to solve ODEs. This simulator is an implicit-explicit ODE solver using variable steps. The same algorithm has been used in Matlab command *ODE45*. The minimal step size is set 0.01. The maximal step size is set 1,000. The allowed absolute and relative scalar errors are both set 0.000001.

SDE solver

In the stochastic model, the changes of species' concentrations are described by Langevin equations which are SDEs in *Itô* scheme. They are numerically solved by the Euler-Maruyama method implemented in the package "*libSDE*" [10].

Stochastic simulation algorithm (SSA)

The Gillespie stochastic algorithm is used to simulate the stochastic processes [11] which is an algorithm for modeling the kinetics of a set of coupled chemical reactions, taking into account stochastic effects from low copy numbers of the chemical species.

In Gillespie's approach, chemical reaction kinetics are modelled as a markov process in which reactions occur at specific instants of time defining intervals that are Poisson-distributed, with a mean reaction time interval that is recomputed after each chemical reaction occurs. For each chemical reaction interval, a specific chemical reaction occurs, randomly selected from the set of all possible reactions with a weight given by the individual reaction rates [14]. NetLand integrates the Java implementation of the Gillespie algorithm in Dizzy [11].

5.2. Usage

1. Select a simulator. The three simulators are listed in the "**Model**" combo box.
2. Run a batch simulation. NetLand supports batch simulations. The number of simulated trajectories can be set in the "**Number of series**".
3. Set the simulation time. The spinner "**Time**" denotes the simulation time for each trajectory with the same time unit as the parameters and initial values of species.
4. Set the initial values of species. The start value for each gene can be either a fixed number or a random value. The fixed initial values are defined in the "**Species**" panel. To change this value, please refer to "Change parameters". For the random initial settings, a set of random values will be generated and assigned to genes after users specify the range (a minimal value and a maximal value).
5. Start and stop a simulation. Press the "**Submit**" button to start a simulation. Press the "**Cancel**" button to stop the simulation.
6. Save the simulation result. Once the simulation is complete, a dialog box will pop up that asks users if they would like to save the result. Not only the simulated time-course data will be saved, the dynamic model (in the format of SBML) will be saved as well.
7. Load the saved result. Click the "**Load saved results**" entry in the "**File**" menu. Then select the "**Load saved trajectory file**" item and choose the file.

5.3. Parsing the output

The simulated trajectories are shown on the right panel in different colors representing different genes (Figure 7). For a batch simulation, the result of each simulation can be retrieved by selecting the index from the combo box. Users can select one or multiple lines in the graph by directly clicking the lines or the labels in the legend table. To view points on the trajectories, select a trajectory first and then hover over it. The coordinates of the selected point will be shown next to it. Right click will cancel the selection. There are 7 facilitating functions on the top of the plot to help users better analyze the result (Figure 8).

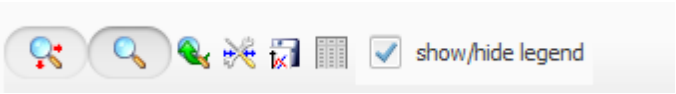
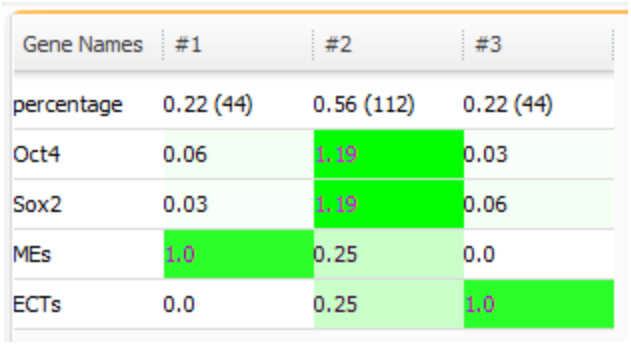


Figure 8. Facilitating functions from left to right: Move the graph with the mouse; Zoom in or Zoom out the graph with the mouse wheel; Reset the view; Edit the scale of the axes; Save the graph to a file in PNG format; Show the simulated data in a table; Show/hide the legend.

The steady state is a situation in which all state variables are constant in spite of ongoing processes that strive to change them. It is an important feature in dynamic simulations. Using NetLand, users can calculate the steady states after simulations by clicking the "**Analyze result**" button. NetLand will first check the convergence of each trajectory and then return the steady states and their proportions (Figure 9).



Gene Names	#1	#2	#3
percentage	0.22 (44)	0.56 (112)	0.22 (44)
Oct4	0.06	1.19	0.03
Sox2	0.03	1.19	0.06
MEs	1.0	0.25	0.0
ECTs	0.0	0.25	1.0

Figure 9. An example of the calculated steady states in which there are three steady states. The second state with Oct4 and Sox2 highly expressed accounts for over half of all the trajectories.

6. Constructing a probabilistic landscape

6.1. Probabilistic landscapes

For a comprehensive understanding of the determination of cellular phenotypes (e.g. cell fate decision in differentiation and reprogramming), the Waddington's epigenetic landscape has been quantified to analyze and visualize the global dynamics of gene regulatory networks [6, 15, 16]. Based on the essential idea of Waddington's epigenetic landscape, changes of phenotypes are represented by transitions overcoming energy barriers between different attractors. Thus the quantified epigenetic landscape is the stage on which the play of cell fate decisions and cell type maturation is choreographed according to physical theories [17].

The quantitative modeling and visualization of Waddington's epigenetic landscape using probabilistic potential are implemented in NetLand. We adopted the self-consistent mean field approximation method in [16] to construct the probabilistic landscape, based on the notion that the probability of gene expression states determines the cell stability [15, 16].

To visualize the landscape of a network with more than 2 genes, NetLand provides two options. The first option is to select two marker genes whose expression levels will be set as the x-axis and y-axis in the landscape. However, the two marker genes may not represent the global dynamics in many cases. To avoid the loss of information, NetLand offers the second option. A probabilistic dimensionality reduction method, Gaussian process dynamical model (GPDM) is employed to project high dimensional gene expression time-series data to a low dimensional latent space with a dynamical model [9, 18, 19]. In this case, the x-axis and y-axis are two constructed components. For both options, the z-axis is the quasi-potential U. Users can click the "**Landscape**" entry in the "**Simulation**" menu to open the main window (Figure 10).

Note: NetLand cannot construct the landscape of a system without stable states. For example, a system with its variables exponentially increasing can never reach a steady state. If NetLand detects an unstable system, a warning message will pop up. Then users may consider modify the model to make it stable. And the current version of NetLand cannot construct the landscape of a system with oscillations.

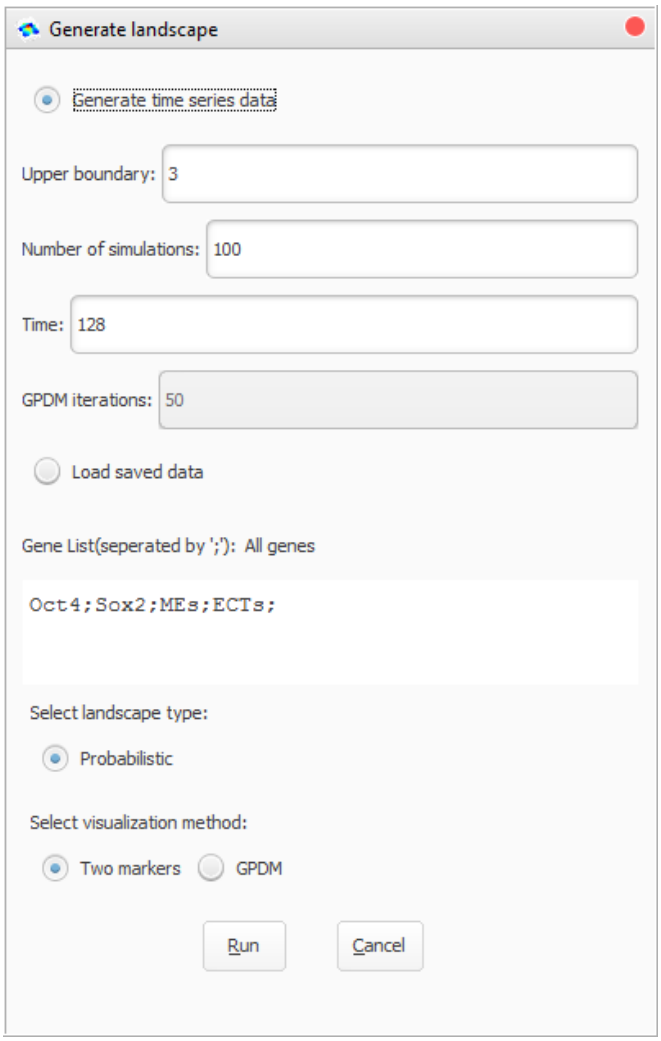


Figure 10. The main window for the construction of landscapes.

6.2. Usage

1. Start the simulation. Choose "**Landscape**" from the "**Simulation**" menu.
2. Simulate multiple trajectories or load pre-computed data. Users only need to generate the time-series data once. The simulated trajectories will be stored in the memory until new data are generated or the software is closed. Before the dynamic simulation, a few parameters need to be settled:
   a) "Upper boundary". It defines the upper boundary of random initial states. Thus the range of the random initial values is defined within [0, UpperBoundary]. This value varies in different models. It is usually set as the maximum expression value across all genes.
   b) "Number of simulations". It denotes the number of trajectories to be generated. In a multi-attractor system, the discovery of attractors depends on whether the simulated data can cover all the steady states. Besides, more simulations are required for a large scale system with a large upper boundary.

c) "Time". It defines the simulation time for each trajectory. For networks with a large number of nodes, normally more iterations and simulation time are required to get their stable states.
d) "GPDM iterations". It is the number of iterations for the training of GPDM.

3. Select the visualization method. Two methods are provided, i.e. mapping to two marker genes and mapping to a 2D latent space using GPDM.
4. Start and stop a simulation. Press the "**Run**" button to start a simulation. Press the "**Cancel**" button to stop the simulation
5. Save the constructed landscape. Once the simulation is complete, a dialog box will pop up that asks users if they would like to save the result. The landscape, the simulated time-course data and the dynamic model (in the format of SBML) will be saved.
6. Load the saved result. Click the "**Load saved results**" entry in the "**File**" menu. Then select the "**Load saved landscape file**" item and choose the file.

6.3. Parsing the output

The constructed landscape will be shown in a pop-up window (Figure 11). It contains three sections, i.e. a canvas with the plotted landscape (left), four control buttons (top-right) and a data grid (bottom-right). NetLand provides an interactive interface for the visualization of the landscape. Figure 12a shows an example of the constructed probabilistic landscapes using the two visualization methods. Users can project the landscape to any two genes selected from the combo boxes below. Double click the cells in the data grid to get the quasi-potential value and coordinates of a point in the landscape (Figure 12c). Users can also plot the contour map or show a mesh of points in the landscape using the control buttons (Figure 12b). The attractors can be retrieved by the "**Show attractors**" function.

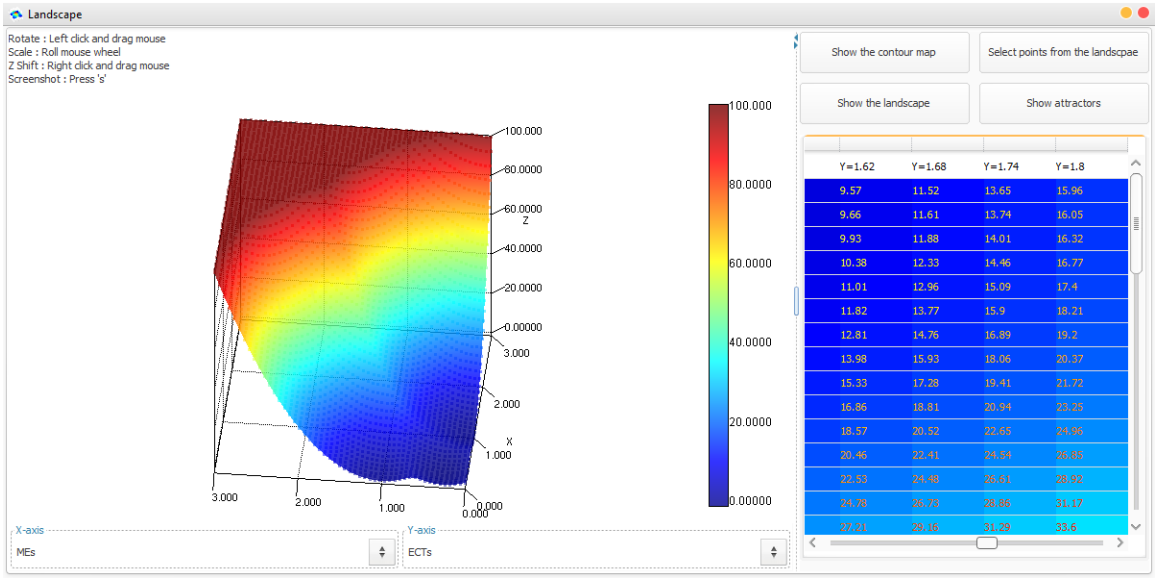
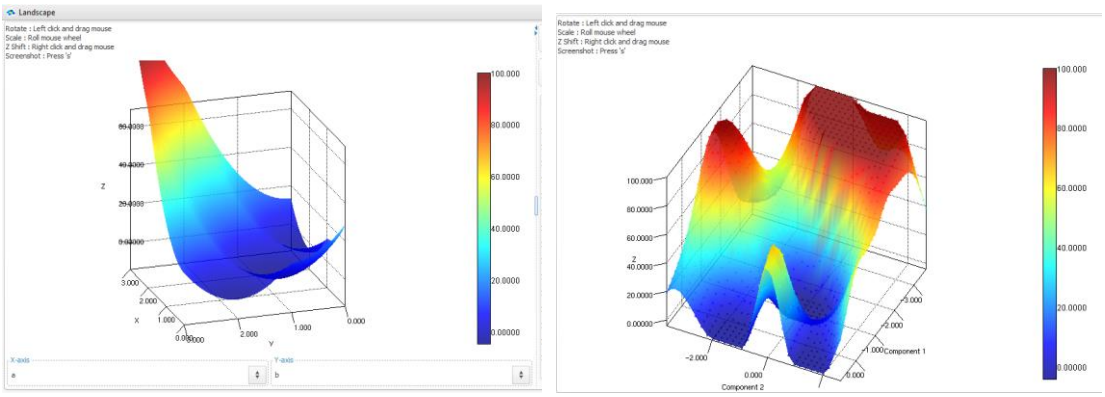
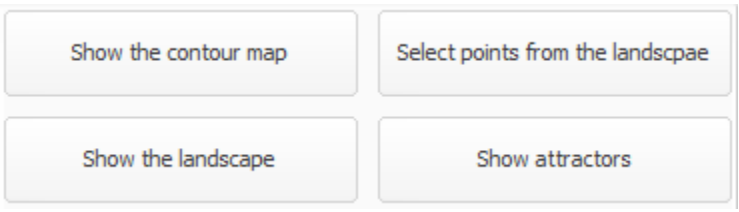


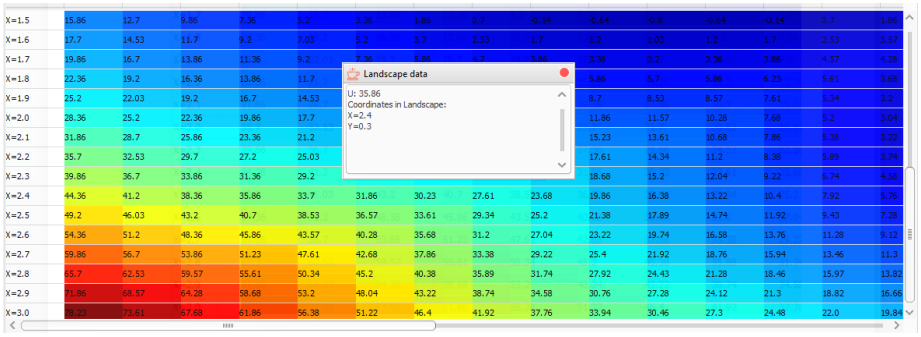
Figure 11. The results window of the constructed landscape.



(a)



(b)



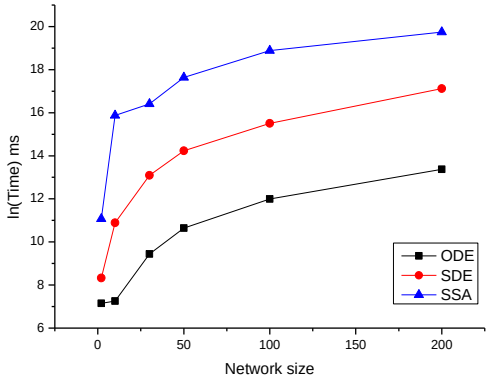
(c)

Figure 12. (a) The constructed landscapes using the two visualization methods, i.e. mapping to two marker genes (left) and mapping to a 2D latent space (right). (b) Control buttons. (c) The table of the grid data in the landscape colored according to the value of quasi-potentials.

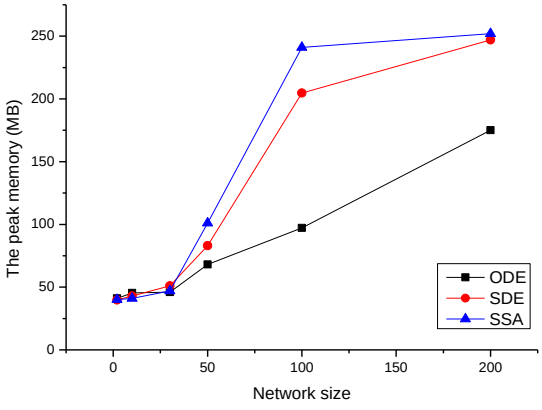
7. Running time assessment and memory usage test

We tested the performance of NetLand on a set of random networks with 2 to 200 genes. These networks were generated with random interactions and automatically transferred into dynamic models by NetLand according to Eq. 1 using default values of parameters (refer to Section 4). The experiments were implemented on a PC of 8-core CPU 3.60 GHz Intel® Xeon® Windows 7 64-bit system. The version of Java is 1.8.0_31 and the maximum heap size of JVM is set 2GB. JProfiler which is a Java profiling tool is used to assess the memory usage.

First, we calculated the running time and memory usage of different simulators implemented in NetLand. Three simulators, i.e. ODE solver, SDE solver and SSA, introduced in Section 5.1 were used to generate simulated time-series data of the random networks. For each network, 100 trajectories were generated with random initial values. The simulation time for each trajectory was set 128. Figure 13 shows the time consumed and the peak value of in-use memory during simulations.



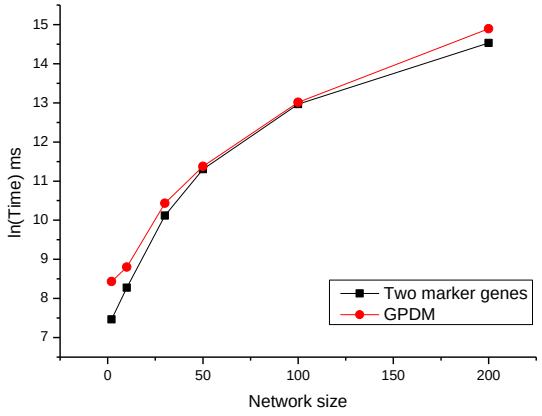
(a)



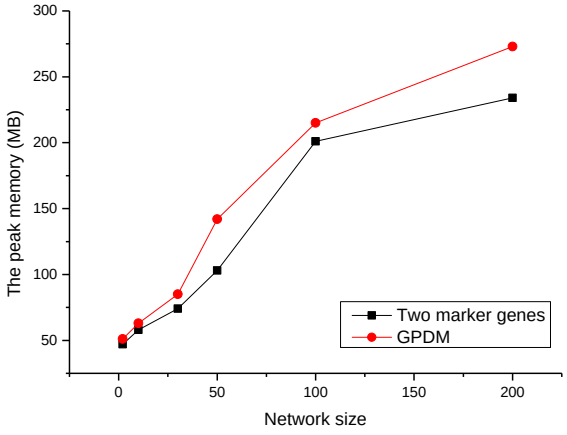
(b)

Figure 13. (a) Running time and (b) the peak value of in-use memory using different simulators.

Then, we constructed the landscapes of each test network using the two visualization methods, i.e. mapping the landscape to two marker genes and mapping the landscape to a 2D latent space by GPDM (for details of the methods, please refer to Section 6.1). The ODE solver is employed to simulate the trajectories. The number of simulated trajectories and the simulation time for each trajectory were set 100 and 200 respectively. The number of iterations for training GPDM was set 50. The execution time and the peak value of in-use memory are shown in Figure 14.



(a)



(b)

Figure 14. (a) Running time and (b) the peak value of in-use memory for the construction of landscapes.

In addition, we tested the three simulators for generating multiple trajectories using different simulation time. The experiments were performed on a computational model, named the "seesaw model" with four genes constructed in [20]. Furthermore, we compared the execution time and the peak value of in-use memory for the construction of landscapes using different settings, e.g.

different number of simulated trajectories, different simulation time. The results are shown in Figure 15-16.

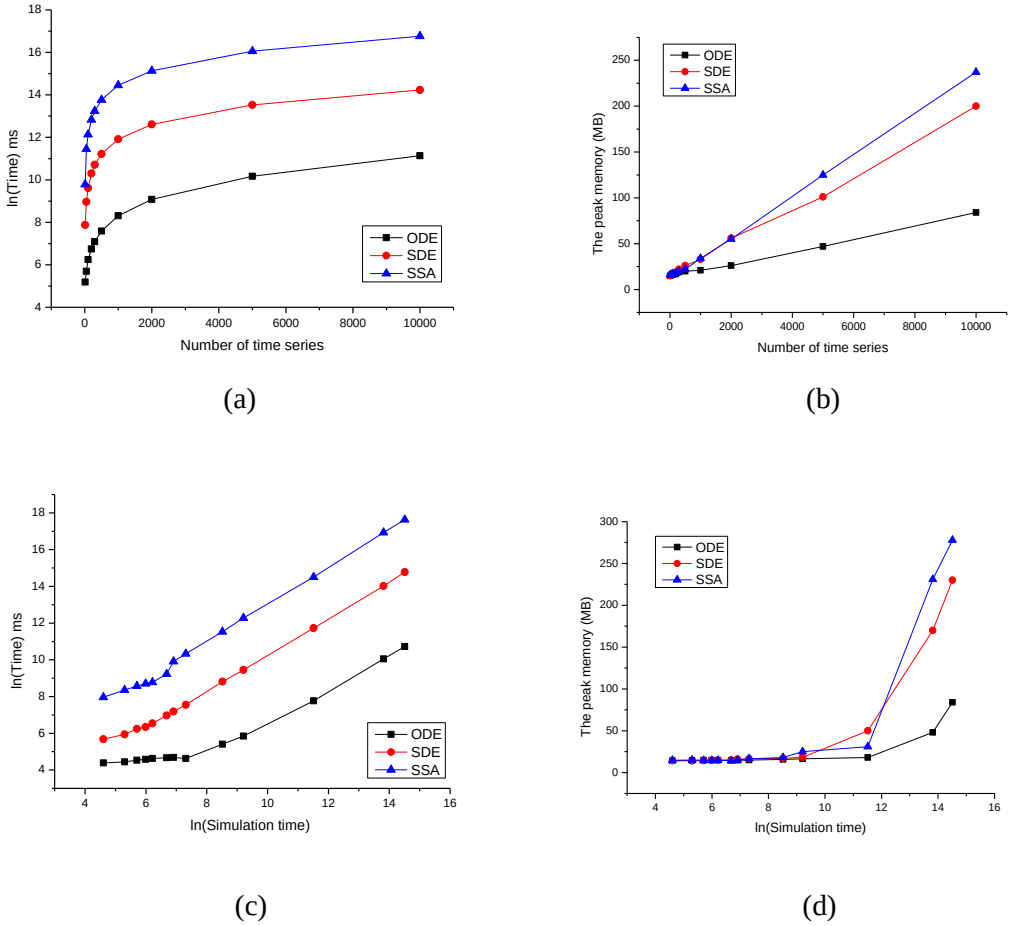
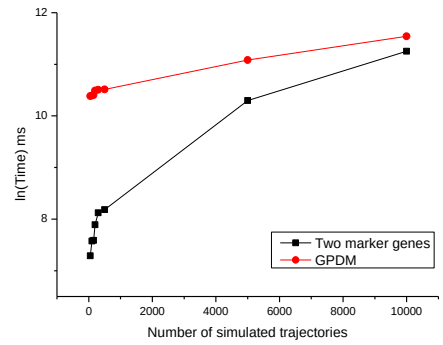
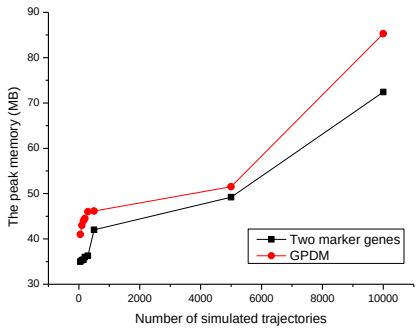


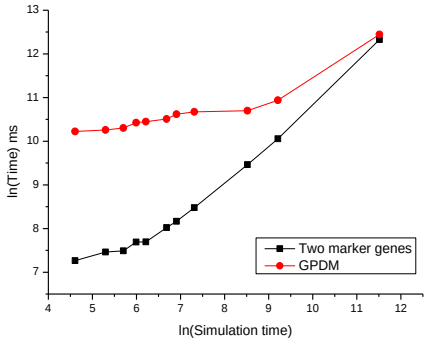
Figure 15. (a) Running time and (b) the peak value of in-use memory for generating different numbers of trajectories. The simulation time was set 128. (c) Running time and (d) the peak value of in-use memory for generating different lengths of trajectories. One trajectory was generated for each experiment.



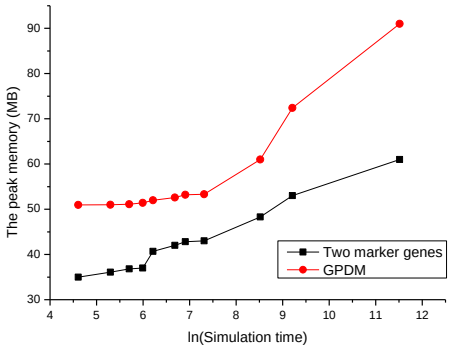
(a)



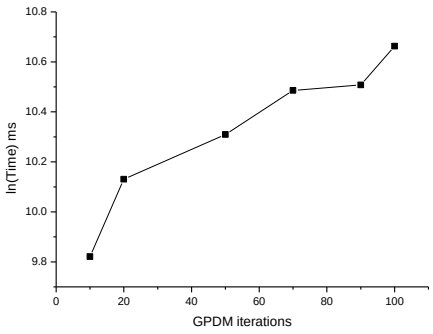
(b)



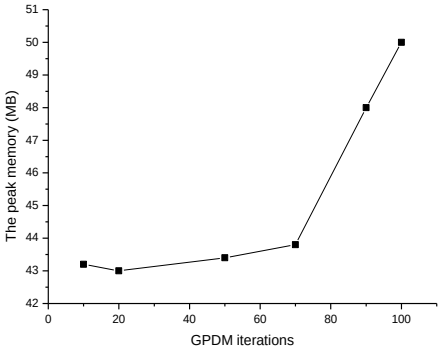
(c)



(d)



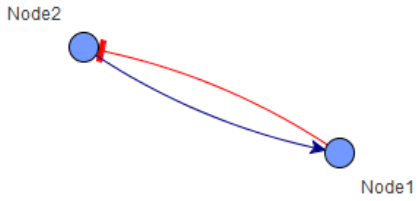
(e)



(f)

Figure 16. (a) Running time and (b) the peak value of in-use memory for constructing landscapes using different numbers of simulated trajectories. The simulation time was set 128. (c) Running time and (d) the peak value of in-use memory for the construction of landscapes using different lengths of trajectories. The number of simulated trajectories was set 100. GPDM was trained for 50 iterations in (a-d). (e) Running time and (f) the peak value of in-use memory using different iterations of GPDM. The simulation time and the number of simulated trajectories were set 128 and 100 respectively.

The network size and the settings of experiments, e.g. the simulation time for each trajectory, the number of simulated trajectories, are the factors that affect the performance of the software. In addition, the internal kinetics in the dynamic model have an impact on the execution time and memory usage as well. For example, dynamic models with the same topology structure but using different sets of parameters could have different performances. Figure 17 shows a two-gene model. We changed the value of the parameter d from 1 to 0.002. The execution time for constructing a landscape increased one fold. Thus the performance of the software may vary in different dynamic models even though they have the same topology structure.



$$\frac{dNode1}{dt} = Node2$$
$$\frac{dNode2}{dt} = 2(1 - Node1) - d \cdot Node2$$

Figure 17. An example model and its mathematical formulas where $d = 1$.

8. Case studies

8.1. Example 1: A two-gene network with mutual inhibitions and self-loops

The two-gene regulatory network with mutual inhibitions and positive autoregulations is published in [6] (Figure 18). The model (2genes.tsv) can be found in the folder **"toy models"**. Here we used NetLand to construct the landscape of this model. The upper boundary, the number of trajectories and simulation time are set 3, 100 and 128 respectively. GPDM was trained for 50 iterations. The probabilistic landscapes mapping to two marker genes and a 2D latent space are shown in Figure 19 (a, b). The system has three attractors listed in Figure 19 (c).

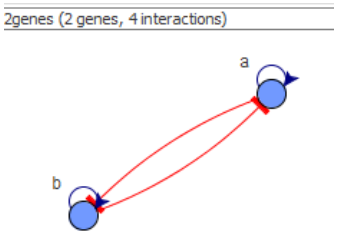
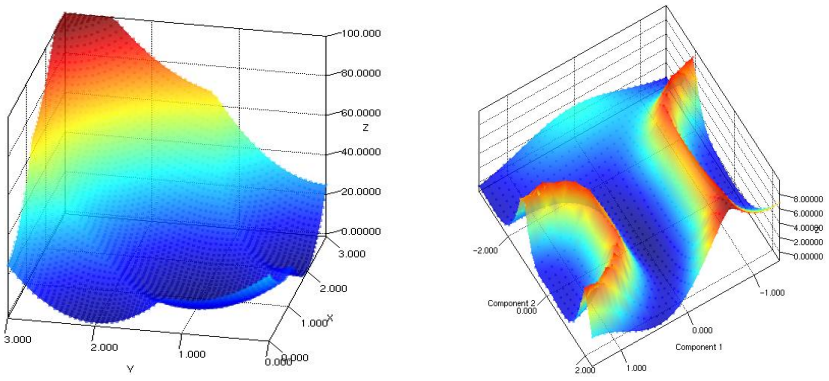


Figure 18. A two-gene network with mutual inhibitions and self-loops published in [6]. There are 2 nodes and 4 edges in the network.



(a) (b)

Gene Names	#1	#2	#3
percentage	0.1 (10)	0.12 (12)	0.78 (78)
a	1.97	0.03	1.03
b	0.05	1.93	1.03

(c)

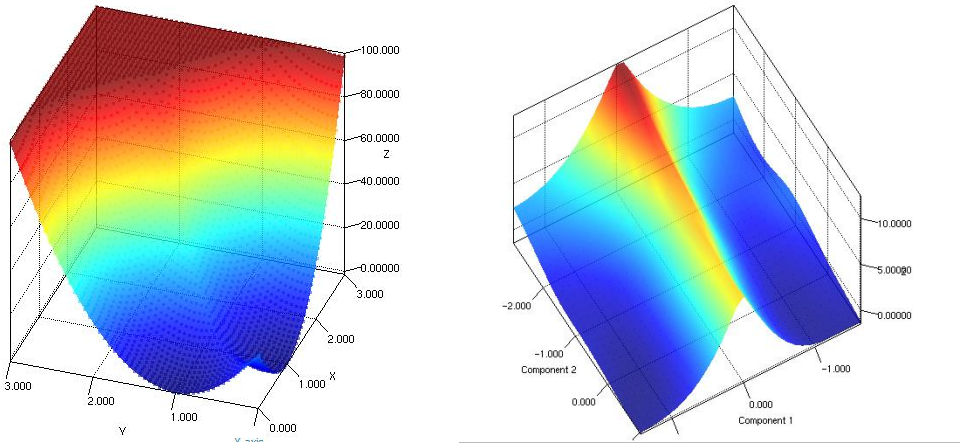
Figure 19. The probabilistic landscapes of the two-gene network when (a) mapping to two marker genes and (b) a 2D latent space. (c) The three attractors. The saved landscapes (landscape_2gene_GPDM and landscape_2gene_Twogenes) are in the folder named **"saved results"**.

Then we removed self-loops from the network by editing the reactions (Figure 20). Instead of three attractors, only two attractors were identified (Figure 21).

```
max_a*(BasalExpression_a+(BasalExpression_a_1+I_a_1*(1/(1+(b/K_a_1)^N_a_1))))-deg_a*a
```

```
max_b*(BasalExpression_b+(BasalExpression_b_1+I_b_1*(1/(1+(a/K_b_1)^N_b_1))))-deg_b*b
```

Figure 20. The modified reactions after removing self-loops.



(a) (b)

Gene Names	#1	#2
percentage	0.45 (45)	0.55 (55)
a	0.98	0.1
b	0.11	1.0

(c)

Figure 21. The probabilistic landscapes of the modified two-gene network when (a) mapping to two marker genes and (b) a 2D latent space. (c) The table of attractors.

8.2. Example 2: The seesaw model

To study the stem cell differentiation and reprogramming process, a computational model, named 'seesaw model' with 4 genes (Figure 22) was constructed in [20]. NetLand was applied to simulate and analyze the model. First, the regulatory relations between genes were extracted into a TSV file. After loading it to NetLand, the differential equations including the parameters were modified according to the paper. The dynamic model (NetLand_seesaw.xml) is saved in the folder **"Toy models"** and the results are saved in two files, named "landscape_seesaw_GPDM" and "landscape_seesaw_Twogenes", in the folder **"Saved results"**. Figure 23 shows landscapes constructed by NetLand.

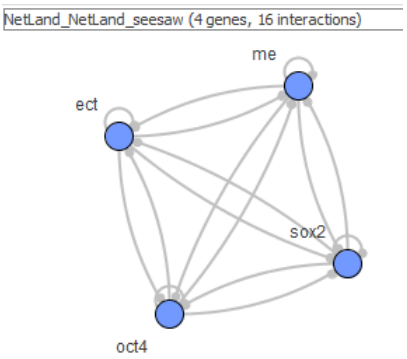
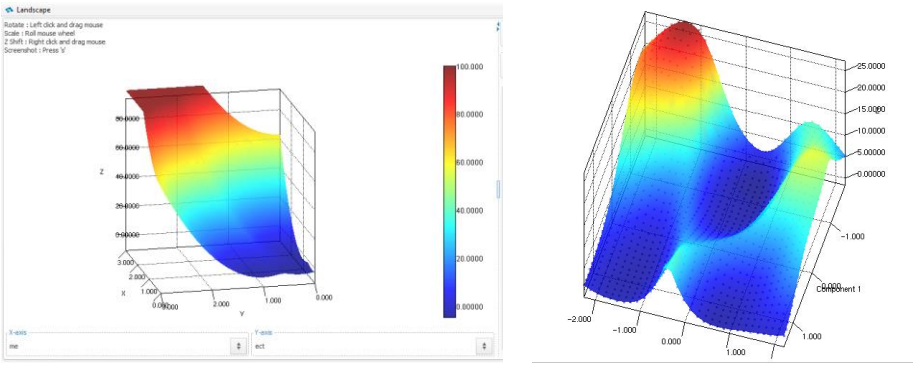


Figure 22. The topology structure of the seesaw model with 4 genes and 16 interactions.



(a) (b)

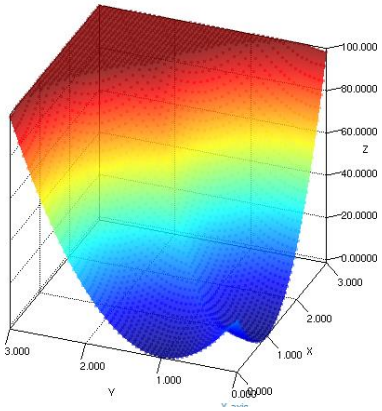
Gene Names	#1	#2	#3
percentage	0.22 (44)	0.56 (112)	0.22 (44)
Oct4	0.06	1.19	0.03
Sox2	0.03	1.19	0.06
MEs	1.0	0.25	0.0
ECTs	0.0	0.25	1.0

(c)

Figure 23. The probabilistic landscapes of the seesaw model when (a) mapping to the nodes, MEs and ECTs and (b) the 2D latent space using GPDM. (c) The table of attractors.

Then we performed a series of parameter sensitivity analysis.

In the first study, we set parameter KM to 0. When the value of KM is set to 0, the cells cannot maintain their pluripotency and start to differentiate. NetLand detected three attractors in this system including two states representing ME (#1) and ECT (#3) states respectively and a rare state (#2) (Figure 24 (b)).



Gene Names	#1	#2	#3
percentage	0.56 (56)	0.02 (2)	0.42 (42)
Oct4	0.12	0.43	0.11
Sox2	0.11	0.43	0.13
ECTs	0.0	0.42	0.98
MEs	0.97	0.35	0.0

(a) (b)

Figure 24. (a) The probabilistic landscapes of the modified network when mapping to MEs and ECTs. (b) The table of attractors.

In the second study, we set parameter KM to 0.4 and simulated the stem cell reprogramming experiment with two cocktails, i.e. using Yamanaka's factors and using lineage specifiers.

Cocktail 1: Using Yamanaka's factors

With the replenishment of Yamanaka's factors (Oct4, Sox2, Klf4, Myc), the somatic cells will be reprogrammed into iPSCs (similar to stem cells) [21]. We tried to repeat the reprogramming experiment using NetLand following the steps below:

Step1. Set the initial state as ME state (Figure 25).

Name	InitialValue
Oct4	0.02
Sox2	0.02
MEs	1.0
ECTs	0.0

Figure 25. The gene expression values of the ME cell states.

Step2. Increase the base production rate of Sox2 and Oct4 (Figure 26).

```
0.2*(0.4+1*0.7^8/(0.7^8+MEs^8)*0.7^8/(0.7^8+ECTs^8)*(0.4+Oct4*Sox2/(0.35+Oct4*Sox2)))-0.2*Oct4
```

Figure 26. The updated reaction of Oct4 with the base production rate increased to 0.4.

Step3. Simulate 10 trajectories using Gillespie algorithm (Figure 27).

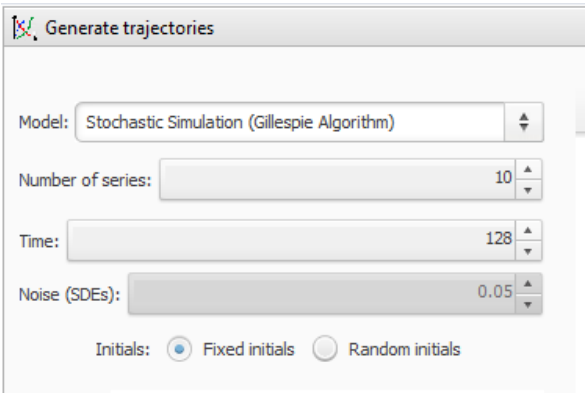


Figure 27. Generation of trajectories.

Step4. Calculate attractors from the simulated data. It shows that all ME cells turned into stem cell state with Oct4 and Sox2 highly expressed (Figure 28).

Gene Names	#1
percentage	1.0 (10)
Oct4	1.38
Sox2	1.4
ECTs	0.51
MEs	0.52

Figure 28. The table of attractor(s).

Cocktail 2: Using lineage specifiers

The somatic cells can be reprogrammed into iPSCs (similar to stem cells) using lineage specifiers [20]. In this reprogramming experiment, the lineage specifiers, MEs and ECTs worked as the substitutions of Oct4 and Sox2. Then we followed the steps stated in the above experiment. We got qualitatively similar result that all ME cells are reprogrammed into stem cells in which Oct4 and Sox2 are highly expressed (data now shown).

8.3. Example 3: A dynamic model from BioModels

A dynamical model of lineage determination based upon a minimal circuit which contains the Oct4/Sox2/Nanog core network as well as other key genes is discussed in [22] (BIOMD0000000209.xml from BioModels). We loaded the model in NetLand (Figure 29) and constructed the landscapes. The upper boundary, the number of trajectories and simulation time are set 8, 100 and 400 respectively. GPDM was trained for 50 iterations. The probabilistic landscapes mapping to two marker genes and a 2D latent space are shown in Figure 30(a)-(b). The system has two attractors listed in Figure 30(c).

The dynamic model can be found in the folder **"Toy models"** and the results are saved in two files, named "landscape_209_GPDM" and "landscape_209_Twogenes", in the folder **"Saved results"**.

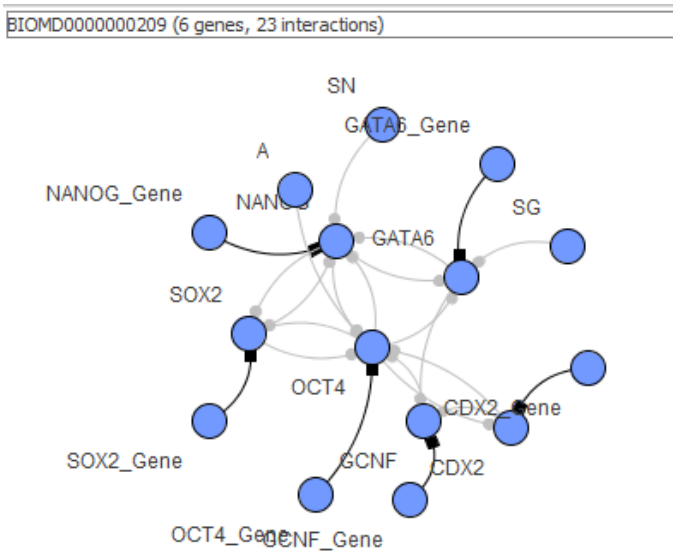
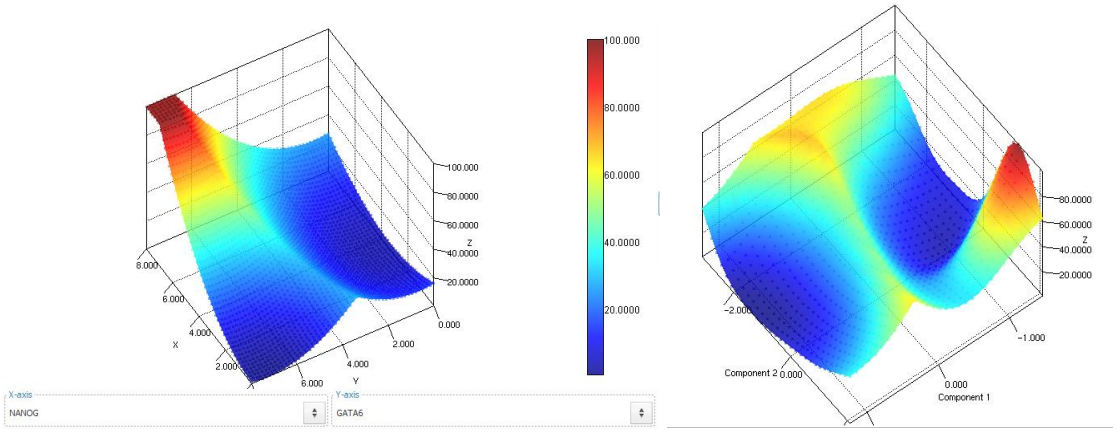


Figure 29. The topology structure of the model with 6 genes and 23 interactions.



(a) (b)

Gene Names	#1	#2
percentage	0.04 (4)	0.96 (96)
OCT4	3.61	2.43
SOX2	7.05	0.03
NANOG	3.33	0.0
CDX2	0.94	1.32
GCNF	1.36	4.56
GATA6	0.65	7.23

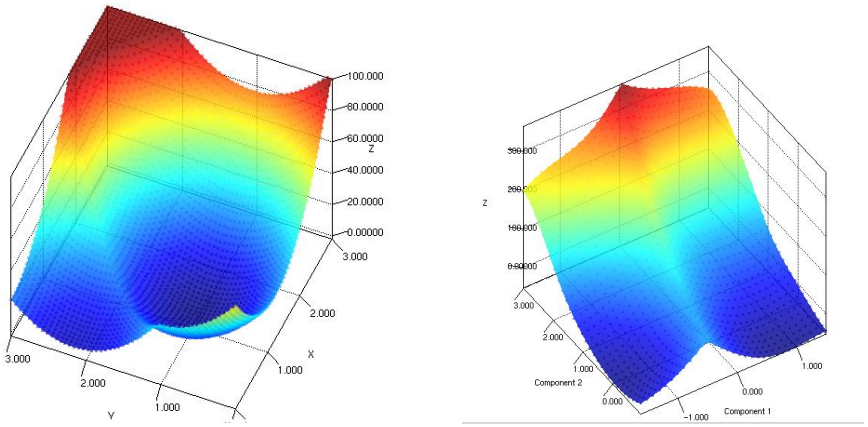
(c)

Figure 30. The probabilistic landscapes of the network when (a) mapping to two marker genes (Nanog and Gata6) and (b) a 2D latent space. (c) The two attractors.

8.4. Example 4: A 52-gene network

A stem cell developmental network with 52 genes was constructed to study stem cell fate decision [7]. It contains marker genes for the pluripotent stem cells and somatic cells. First, the regulatory interactions between genes were extracted into a TSV file. After loading it to NetLand, the differential equations including the parameters were modified according to the paper. We loaded the model in NetLand and constructed the landscapes. The upper boundary, the number of trajectories and simulation time are set 3, 5000 and 128 respectively. GPDM was trained for 50 iterations. The probabilistic landscapes mapping to two marker genes and a 2D latent space are shown in Figure 31 (a, b). The system has two attractors listed in Figure 31 (c).

The dynamic model (NetLand_52gene.xml) is saved in the folder "**Toy models**" and the results are saved in two files, named "landscape_52gene_GPDM" and "landscape_52gene_Twogenes", in the folder "**Saved results**".



(a) (b)

Gene Names	#1	#2
percentage	0.9974 (4987)	0.0026 (13)
CDX2	0.08	1.34
PBX1	0.34	0.0
GATA6	1.03	2.13
E2F4	0.0	0.0
TDGF1	0.9	0.0
FOXA2	0.81	1.16
SOX17	0.24	0.99
OCT4FOXD3	0.65	0.0
FOXA1	0.76	0.79
AFP	0.58	0.64
SUMO1	0.34	0.0
GATA4	0.17	0.99
PIAS1	0.0	0.0
PRDM14	0.9	0.0
LMCD1	0.84	0.5
T	0.64	1.03

Gene Names	#1	#2
CSH1	0.07	0.49
GATA2	1.1	1.98
PIASy	0.0	0.0
GATA3	0.02	0.49
hCGa	0.38	1.03
cAMP	0.0	0.0
SP1	0.82	0.81
hCGb	0.66	1.12
SP3	0.86	0.99
BMP2K	0.08	0.1
BMP2	0.23	0.99
HNF4A	0.33	0.36
bCAT	0.0	0.0
CEBP	0.58	0.59
FOXO1A	0.9	0.0
LEF1	0.07	0.99
LEF1bCat	0.0	0.32

Gene Names	#1	#2
Mad	0.0	0.0
MadMax	0.0	0.0
MAX	0.0	0.0
MYC	0.81	0.57
TCF3	0.0	0.0
MycMax	0.3	0.22
MYCSP1	0.6	0.52
NFYA	0.21	0.0
SALL4	0.56	0.32
ZNF206	0.34	0.0
NANOG	1.24	0.01
OCT4	1.05	0.03
OCT4Sox2	0.55	0.0
SOX2	0.56	0.0
ZIC3	0.56	0.0
KLF4	0.9	0.0
FOXD3	0.9	0.0
ZFP42	0.68	0.0
GDF3	0.21	0.0

(c)

Figure 31. The probabilistic landscapes of the 52-gene network when (a) mapping to two marker genes (Nanog and Gata6) and (b) a 2D latent space. (c) The two attractors.

9. License

NetLand is distributed under the GPL (General Public License). Use of NetLand is subject to the license agreement. See LICENSE in the main NetLand directory.

Any use of NetLand for a **commercial purpose** is subject to and requires a special license. If you intend to use NetLand for a commercial purpose, please contact zhengjie@ntu.edu.sg.

10. Contact information and bug reporting

The authors have carried out extensive testing and debugging of the program, which should be generally stable and functional if the instructions in this manual are followed. It is our hope that, with the GUI, users without much computational sophistication can also apply our algorithm to analyze their network smoothly. However, since the authors are not professional programmers, NetLand does not behave or look like a commercial software. Please send your suggestions, comments, reports of bugs and errors to *Jie Zheng* at zhengjie@ntu.edu.sg. Thank you!

References

1. Waddington, C.H., *The strategy of the genes; a discussion of some aspects of theoretical biology* 1957, London,: Allen & Unwin.

2. Kauffman, S.A., *Metabolic stability and epigenesis in randomly constructed genetic nets.* Journal of theoretical biology, 1969. **22**(3): p. 437-67.

3. Thom, R., *An inventory of Waddingtonian concepts.* Theorical Biology: Epigenetic and Evolutionary Order from Complex Systems, 1989: p. 1-7.

4. Huang, S., *The molecular and mathematical basis of Waddington's epigenetic landscape: a framework for post-Darwinian biology?* BioEssays : news and reviews in molecular, cellular and developmental biology, 2012. **34**(2): p. 149-57.

5. Wang, J., L. Xu, and E. Wang, *Potential landscape and flux framework of nonequilibrium networks: robustness, dissipation, and coherence of biochemical oscillations.* Proc Natl Acad Sci U S A, 2008. **105**(34): p. 12271-6.

6. Bhattacharya, S., Q. Zhang, and M. Andersen, *A deterministic map of Waddington's epigenetic landscape for cell fate specification.* BMC systems biology, 2011. **5**: p. 85.

7. Li, C. and J. Wang, *Quantifying cell fate decisions for differentiation and reprogramming of a human stem cell network: landscape and biological paths.* PLoS Comput Biol, 2013. **9**(8): p. e1003165.

8. Lawrence, N., *Probabilistic Non-linear Principal Component Analysis with Gaussian Process Latent Variable Models.* J. Mach. Learn. Res., 2005. **6**: p. 1783-1816.

9. Wang, J.M., D.J. Fleet, and A. Hertzmann, *Gaussian process dynamical models for human motion.* Pattern Analysis and Machine Intelligence, IEEE Transactions on, 2008. **30**(2): p. 283-298.

10. Schaffter, T., D. Marbach, and D. Floreano, *GeneNetWeaver: in silico benchmark generation and performance profiling of network inference methods.* Bioinformatics, 2011. **27**(16): p. 2263-2270.

11. Ramsey, S., D. Orrell, and H. Bolouri, *Dizzy: stochastic simulation of large-scale genetic regulatory networks.* Journal of bioinformatics and computational biology, 2005. **3**(02): p. 415-436.

12. Chelliah, V., et al., *BioModels: ten-year anniversary.* Nucleic acids research, 2014: p. gku1181.

13. Funahashi, A., et al., *CellDesigner 3.5: a versatile modeling tool for biochemical networks.* Proceedings of the IEEE, 2008. **96**(8): p. 1254-1265.

14. Gillespie, D.T., *A general method for numerically simulating the stochastic time evolution of coupled chemical reactions.* Journal of computational physics, 1976. **22**(4): p. 403-434.

15. Zhou, J.X., et al., *Quasi-potential landscape in complex multi-stable systems.* Journal of The Royal Society Interface, 2012. **9**(77): p. 3539-3553.

16. Li, C. and J. Wang, *Quantifying cell fate decisions for differentiation and reprogramming of a human stem cell network: landscape and biological paths.* PLoS computational biology, 2013. **9**(8): p. e1003165.

17. Huang, S., I. Ernberg, and S. Kauffman. *Cancer attractors: a systems view of tumors from a gene network dynamics and developmental perspective*. in *Seminars in cell & developmental biology*. 2009. Elsevier.

18. Lawrence, N., *Probabilistic non-linear principal component analysis with Gaussian process latent variable models.* The Journal of Machine Learning Research, 2005. **6**: p. 1783-1816.

19. Gopalan, N. and T. IAS, *Gaussian Process Latent Variable Models for Dimensionality Reduction and Time Series Modeling.*

20. Shu, J., et al., *Induction of pluripotency in mouse somatic cells with lineage specifiers.* Cell, 2013. **153**(5): p. 963-975.

21. Takahashi, K. and S. Yamanaka, *Induction of pluripotent stem cells from mouse embryonic and adult fibroblast cultures by defined factors.* cell, 2006. **126**(4): p. 663-676.

22. Chickarmane, V. and C. Peterson, *A computational model for understanding stem cell, trophectoderm and endoderm lineage determination.* PLoS One, 2008. **3**(10): p. e3478.