Evolvix 0.3.0 Manual

Release 0.3.0-beta

The Evolvix Team

October 01, 2014

The Evolvix vision is to make accurate modeling easy.

If you are new to Evolvix, read in our Quick-start instructions for the impatient about how to get started with building your own simulation models with Evolvix.

We have a way to go before we can make accurate modeling really easy; thus we can only provide you with a prototype today.

While we work on implementing the next round of new features, please let us know, what you think would make your modeling more accurate and easier:

- General ideas: http://evolvix.org/contact/general-feedback
- Bug reports: http://evolvix.org/contact/bug-report
- Other feedback: http://evolvix.org/contact/

We'd like to hear about what you need to work productively in Evolvix.
We'd like to hear especially about what problems you have with Evolvix.

Your input counts. We appreciate your help and patience while we work through the many issues and inconveniences that come with prototypes. We believe it will be worth the wait and the effort.

The version of Evolvix installed in this folder is:

```
| Evolvix 0.3.0-beta
| Release date 2014-09-30
|
| Brought to you by
|
| The Evolvix Team
| Laboratory of Genetics
| Wisconsin Institute for Discovery
| University of Wisconsin-Madison
| + Laurence Loewe, Creator of Evolvix, Project leader
| + Seth Keel, Lead Developer, Systems Integration
| + Kurt Ehlert, Core Developer, Diverse Algorithms and Stochastic Simulator
| + Iratxo Flores-Lorca, Core Developer, Deterministic Simulator, TimeSeries, R-Plots
| + Tanner Engbretson, Core Developer, Graphical User Interface for browser
| + Jacob Goldfinger, Documentation writing and generation setup, usability testing
| + Kate Scheuer, Syntax testing, documentation, usability testing.
```

More details and updates can be found here:

Background http://evolvix.org/about
Tutorials http://evolvix.org/tutorial
Downloads http://evolvix.org/download
Contacts http://evolvix.org/contact/

CHAPTER

ONE

GETTING STARTED - BROWSER

1.1 Install Evolvix in your Browser

You can download Evolvix here[1]. It is available for Windows 7+ and MacOS 10.8+. You do not have to actually install Evolvix, you just need to run it within its folder. Here is how:

1. Unzip the downloaded zip file.
2. You can move the Evolvix folder, but that isn't necessary.
3. Open the Evolvix folder, and double-click on "start_evolvix_[platform]".
4. Two things will appear: a console window and a browser window.

Evolvix is now running, and the rest of the guide details what you can do.

Possible issues:

Browser shows "webpage not available" or a similar message Solution: Refresh the browser window. Check that the console did not display any errors.

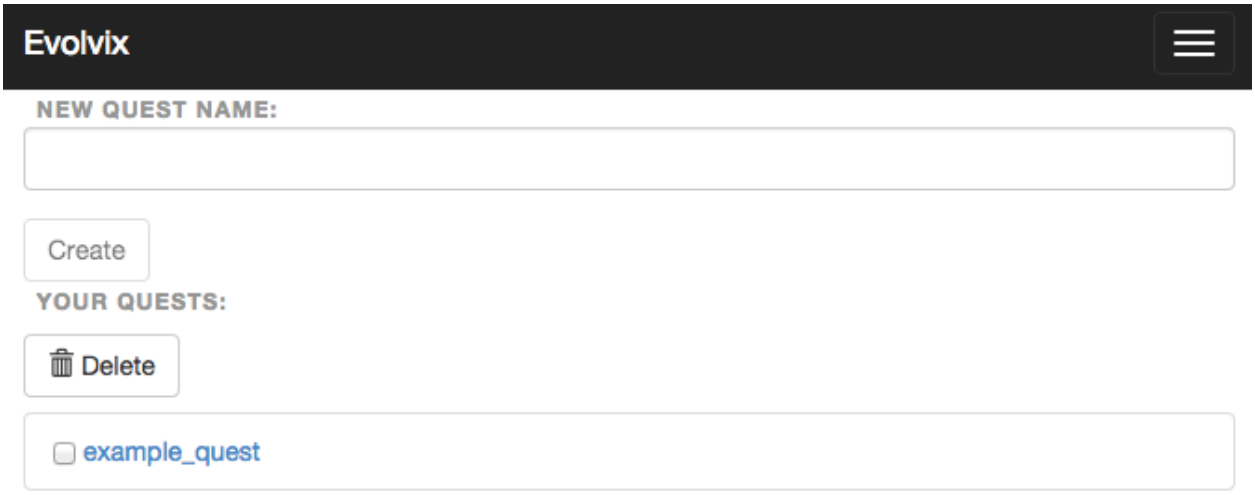
Browser might display a message recommending the latest version of your browser. Solution: Update your browser.

1.2 Introduction

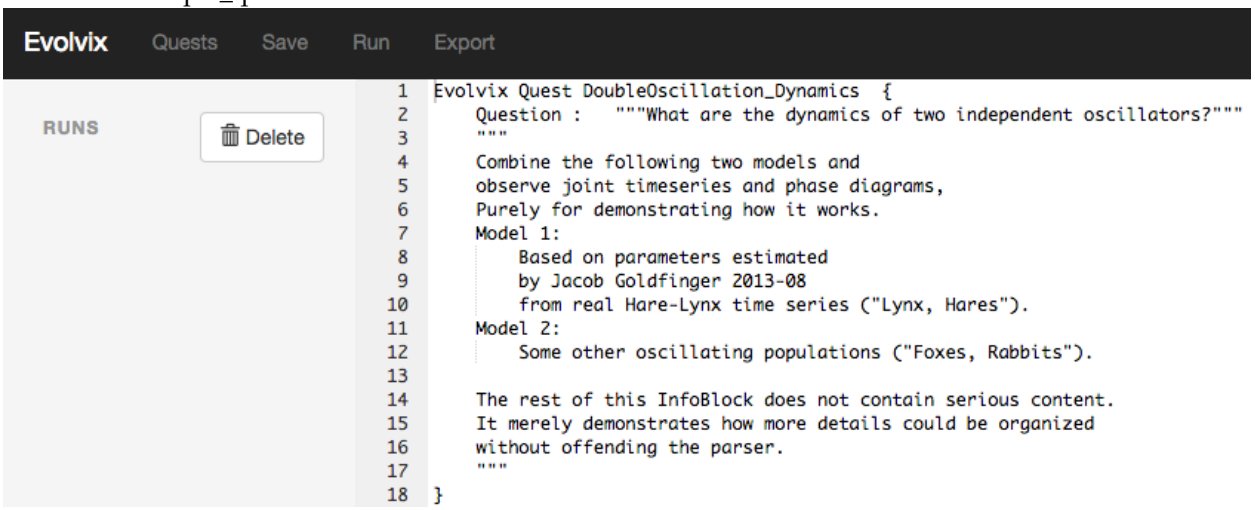
This page shows you how to run an example "quest" in Evolvix. Quests are essentially your independent projects. They contain one or more models and all of the simulation results for those models.

Let's run the "example_quest" that comes with Evolvix. Once you have Evolvix running, your browser should be showing a page similar to this:

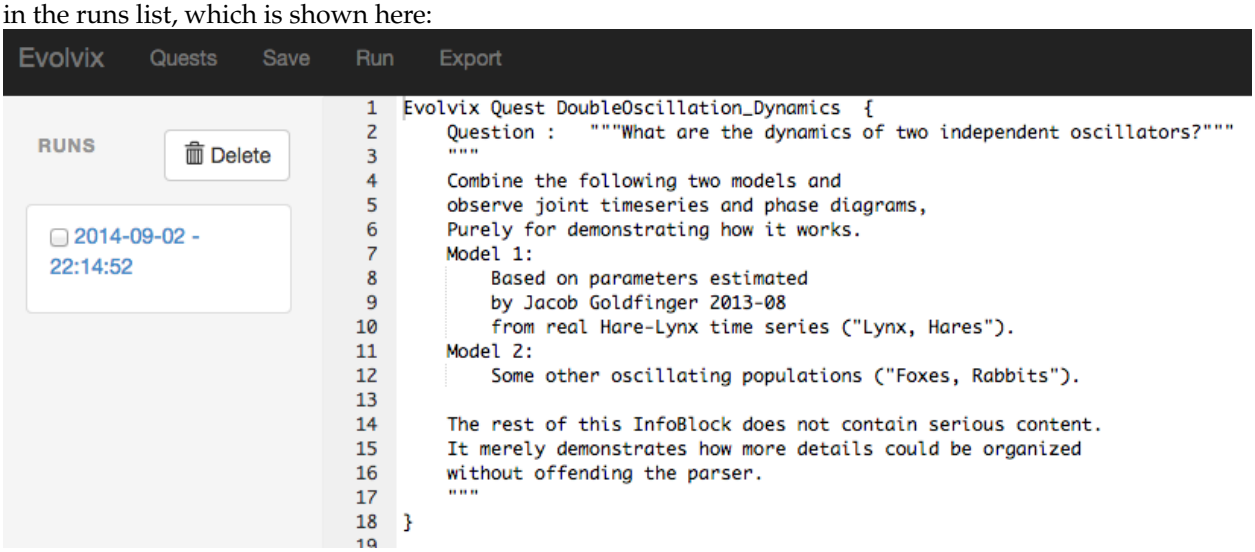
[1] http://evolvix.org/download



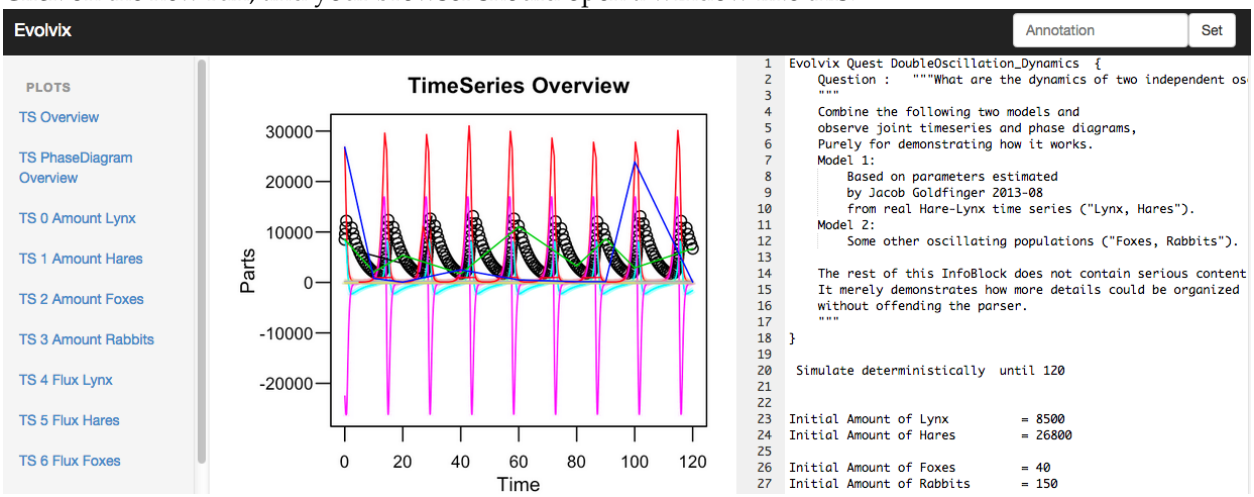
Click on "example_quest". You should now see this:



The text on the right describes a model, which Evolvix uses to run simulations. At the top of the screen are a couple of different buttons. To run the example quest, just click on "Run". Soon after, a run will appear in the runs list, which is shown here:



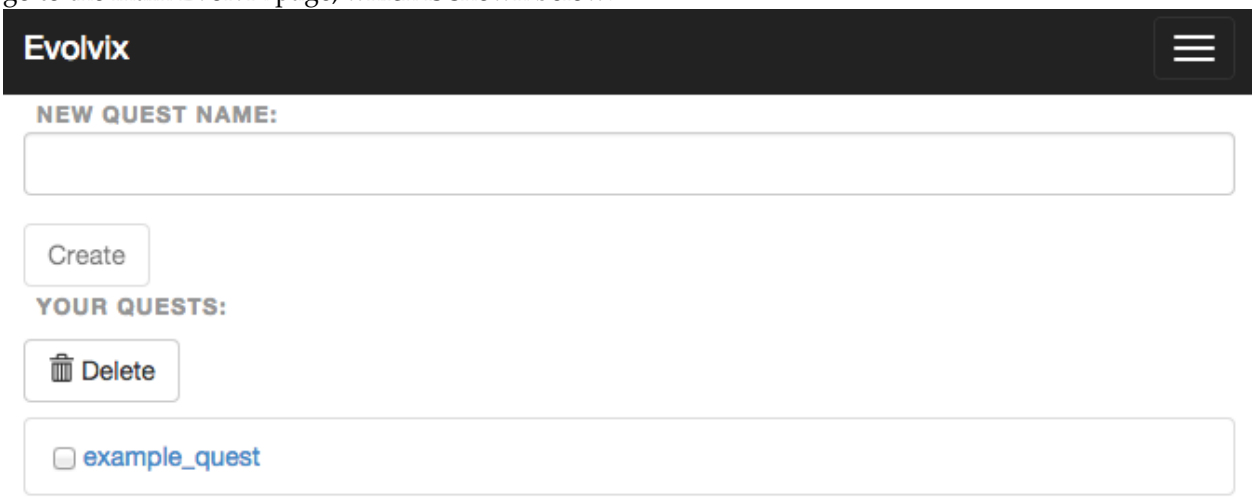
Click on the new run, and your browser should open a window like this:



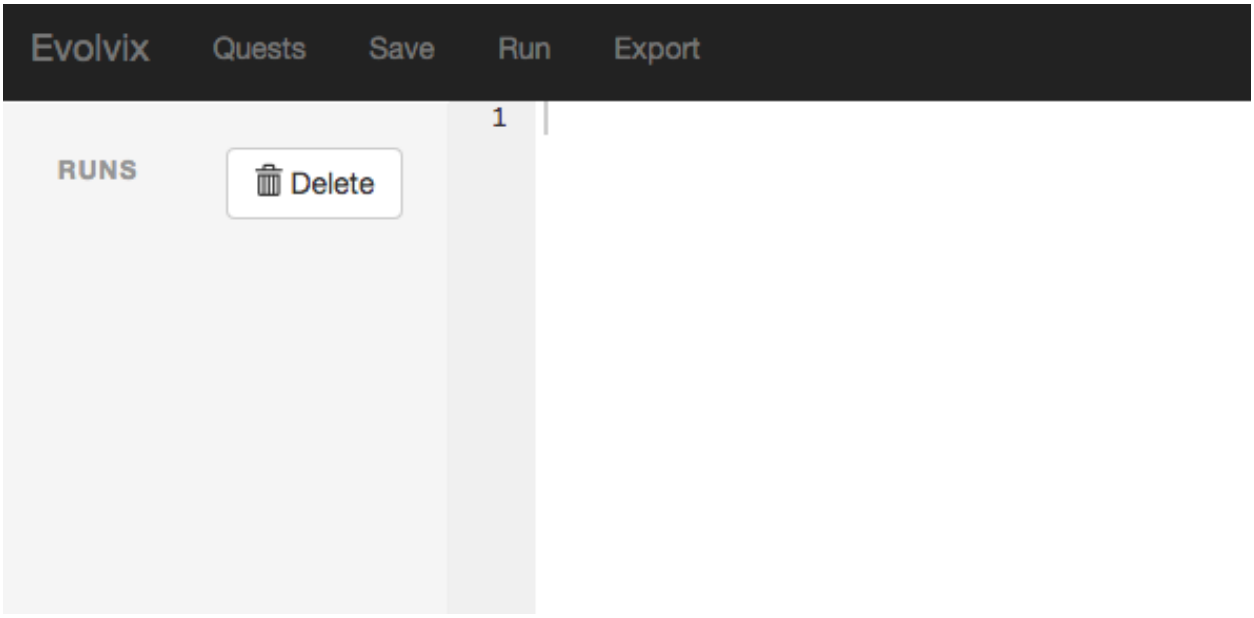
This run window is used to access all of the plots and a snapshot of your quest. will cover the details in another section. The following sections will cover everything in more detail.

1.3 Create and Edit a Quest

Quests are similar to projects. They contain one or more models and runs for each model. To create a quest, go to the main Evolvix page, which is shown below:



Enter a quest name and click “Create”. Quest names can only contain alphanumeric and underscores. Once you click “Create”, you should see a page like this:

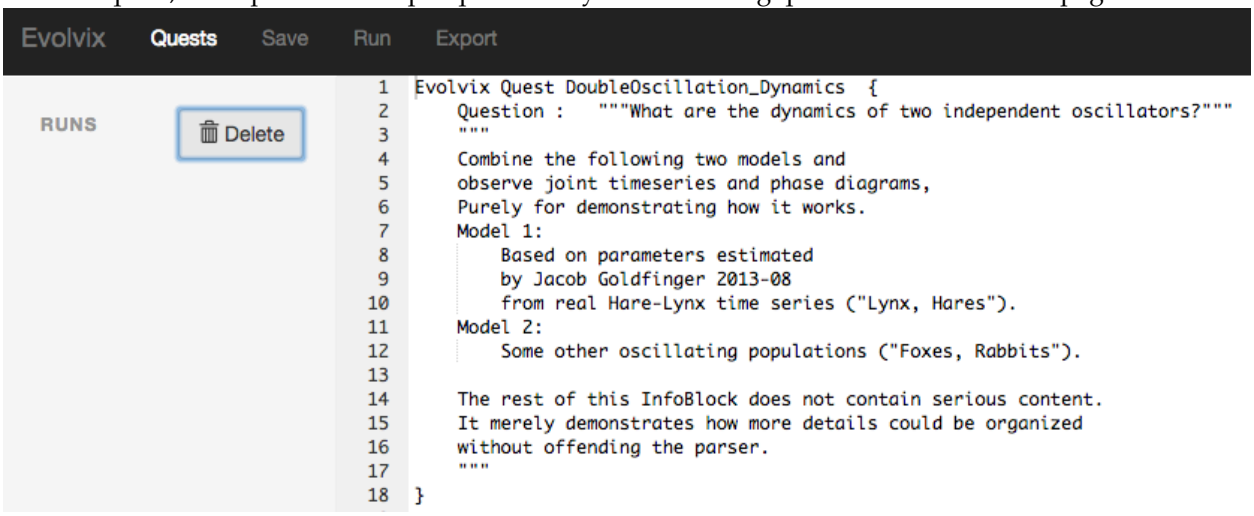


The quest is saved upon creation, so you can come back to it any time. However, the quest is not automatically saved while editing, so be sure to click "Save" often. To edit your quest, click in the text editor half of the page and start typing.

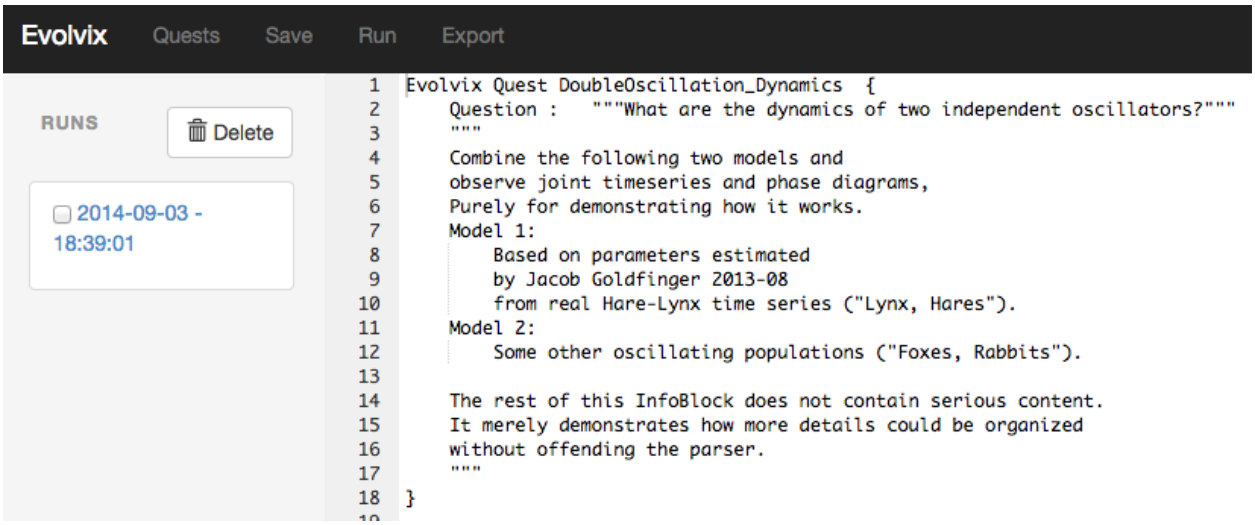
It is also a good idea to backup important quests, which you can do by exporting them and saving them to a secure location. Just click "Export" at the top of the quest editing page. Once you click it, your quest is downloaded by the browser. If you need to restore the quest from a backup, you can import the quest from the main page. Exporting and importing are talked about in more detail in the "Share a Quest" section.

1.4 Run a Quest

To run a quest, first open the example quest or any other working quest. You should see a page like this:



Click "Run", and a link to the run will appear in the runs list on the left side of the browser. You should see something like this:



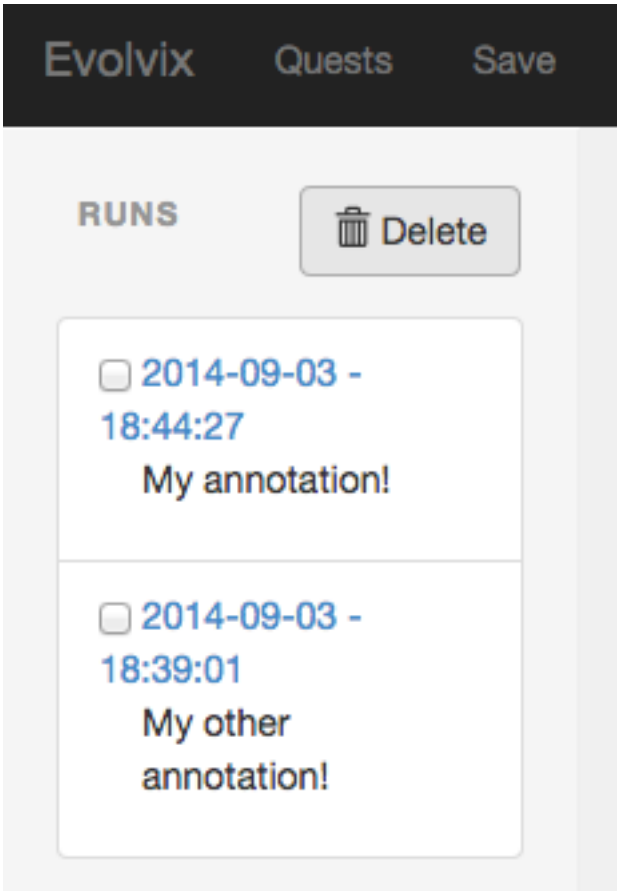
The runs are ordered from the most recent at the top to the oldest at the bottom. Click on the new run, and you should see one of three things:

1. A "quest is still running" message. Once a quest is done running, the page automatically refreshes and shows the results.

2. A successful run with a list of plots and a snapshot of the quest at the time of the run.

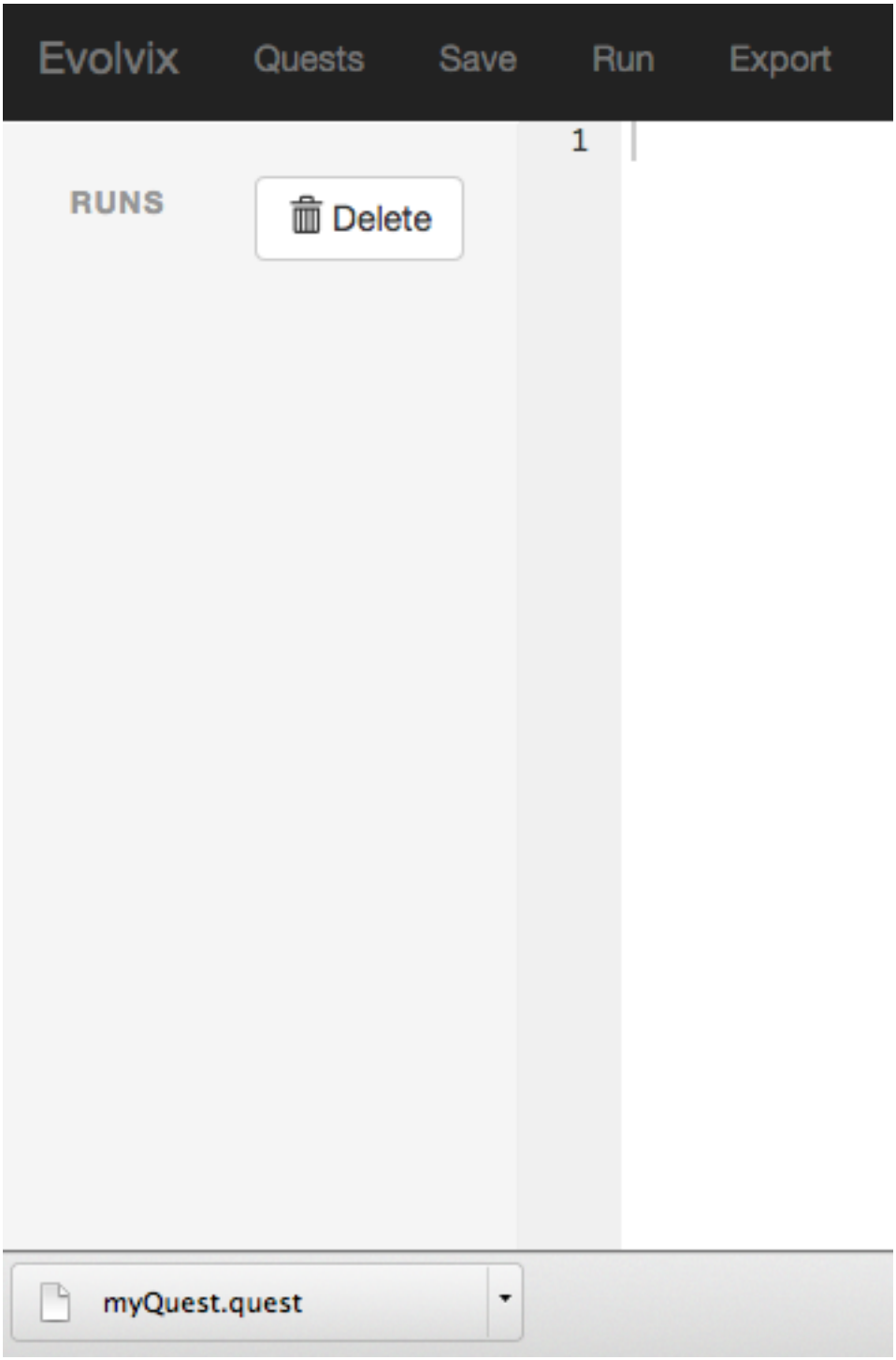
3. An unsuccessful run with an error message generated by Evolvix and a snapshot of the quest at the time of the run.

If need to keep notes about your runs, the annotation feature could be useful. Annotations are set within the run page, and they can be viewed either in the run page or the quest page. Some example annotations are shown here:

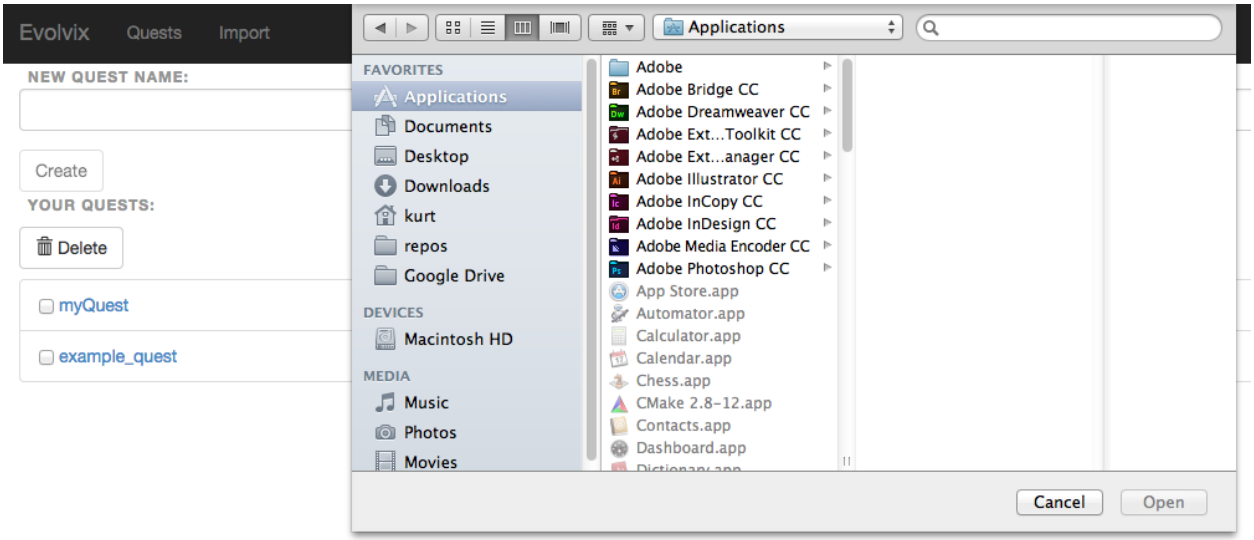


1.5 Share a Quest

There are two main features that you need to use for sharing quests: importing and exporting. If you have a quest that you want to send to a friend, you first need to export it. Open the quest page, and click on "Export" at the top. The browser should download a [quest_name].quest file to your default downloads directory.



Send the .quest file to your friend and have them import it. Importing is done via the main Evolvix page. Click on "Import" at the top and open the .quest file. The quest should appear in the list of quests.


Evolvix
Quests
Import
NEW QUEST NAME:
Create
YOUR QUESTS:
Delete
myQuest
example_quest
Applications
FAVORITES
Applications
Documents
Desktop
Downloads
kurt
repos
Google Drive
DEVICES
Macintosh HD
MEDIA
Music
Photos
Movies
Adobe
Adobe Bridge CC
Adobe Dreamweaver CC
Adobe Ext...Toolkit CC
Adobe Ext...anager CC
Adobe Illustrator CC
Adobe InCopy CC
Adobe InDesign CC
Adobe Media Encoder CC
Adobe Photoshop CC
App Store.app
Automator.app
Calculator.app
Calendar.app
Chess.app
CMake 2.8-12.app
Contacts.app
Dashboard.app
Cancel
Open

CHAPTER

TWO

GETTING STARTED - COMMAND LINE

2.1 Hello and welcome to Evolvix

2.1.1 Quick-start for the impatient

There are three things you need to start using Evolvix on your computer: one you need to know, one you need to decide and one you need to do.

What is a Quest?

A Quest in Evolvix is a collection of input and output files documenting your modeling work as you proceed on your Quest to answer the Quest-ion that motivated you to build a model in the first place. You can think of it as the closest Evolvix has to "programs" or "projects" you may know from other systems, but neither of these analogies fit and modeling is always an adventure. So we decided to define a Quest as the collection of all files that get used and produced while working on a model to answer a question. Evolvix keeps a log of your activities while you work on a Quest.

Which form of Evolvix do you want to use?

1. Work with the Browser based version of Evolvix? Read the documentation here.
    - You gain: Windows, mouse, clicks of buttons.
    - You lose: The power of the command line and more direct access to results.
2. Work with the CommandLine based version of Evolvix? Read the documentation below.
    - You gain: the power of the command line and more direct access to results, access to the R script and other details; great for python scripting.
    - You lose: Windows, mouse, clicks of buttons.

Is Evolvix working as it should?

To test your installation of Evolvix, run the Default Quest example that is provided for both each version of Evolvix.

If you see a plot (or a PDF is produced after you installed R for the CommandLine version), then the installation is working. If you can't see that, go to trouble shooting or let us know.

2.1.2 There is now a BrowserBased GUI version of Evolvix

Instructions for using Evolvix with your web browser

This GUI version of Evolvix was not made for User to look at results directly. Use the command line version if you need that capability for now.

We are in the process of deciding how to next develop this, so please let us know if you have particular preferences.

2.1.3 Quick-start the Command-Line based version of Evolvix

A quick quide on basic use of Evolvix is in section How to get started with Evolvix.

In a new Evolvix installation the file:

```
Default_Quest.Evolvix.txt
```

should be in the main Evolvix folder. It contains a simple demo simulation. To start this, go to the command line, navigate to the main Evolvix folder and type:

```
./Evolvix
```

To call your own Quest-file, add its name on the same line. This will cause Evolvix to parse the default Quest file, run the simulations, write the raw data to the results folder and call the statistical programming language R (if installed) to plot the results.

From time to time updates and new tutorials may be published at

> http://evolvix.org/tutorial

2.1.4 The Rest of the Documentation here is for the CommandLine Version of Evolvix

Not everything has been updated to the latest changes yet (this is an beta version of a prototype...).

We will update the documentation in the following weeks.

2.1.5 How to use the online help

Here are links to all documentation shipped with your installation.

Our goal:

> Give you everything about Evolvix you need to know, where you need to know it.

And that's right in the middle an Evolvix session where you forgot the syntax of some feature you know exists.

Evolvix is being designed to **maximize ease of reading**, as code is read more often than written. That comes with a trade-off: it is a bit harder to write such code, as it needs to use the precise words, so the computer can recognize it. It is hard to precisely memorize all the carefully crafted sentences that define the syntax of Evolvix. Even we as developers sometimes forget the precise wording on options we don't use a lot.

However, that is not a real problem, as we all usually have a computer with a browser around when coding. In principle we can:

- easily look up documentation that is readily available and
- copy and paste example code from there, as
- adjusting example code is very easy after all.

We aim to develop this Evolvix online help system to provide all details needed for making this work. So if this is your personal Evolvix installation, best make sure you always have the main Evolvix documentation close by in your browser:

Bookmark a link to the Table of Content of the Evolvix Documentation

Do it now! On that page there is also a search field, where you can enter any term and start a local search of all documentation files directly.

After packaging this release of Evolvix, additional information may have become available on http://evolvix.org

- more example models
- hot updates on quirks or known bugs of this release, while it is maintained
- new general tricks on how one can use Evolvix
- new releases of Evolvix

2.1.6 Starting the Evolvix online help from the commandline

When you are working in the main Evolvix folder at the command line and need to look something up, but don't have bookmarks for the Evolvix documentation, you can always find it here:

MacOSX command-line:

```
type "open help.html"
(or only "open help" or less and hit the "tab" key to auto-complete)
hit "enter" to start your default browser
```

Windows command-line:

```
type "start help.html"
(or only "start help" or less and hit the "tab" key to auto-complete)
hit "enter" to start your default browser
```

Linux command-line:

```
type "xdg-open help.html"
(or only "xdg-open help" or less and hit the "tab" key to auto-complete)
hit "enter" to start your default browser
```

We aim to find the best possible way for making Evolvix help as accessible as possible.
So if you would like to see a particular improvement, please do let us know.

2.1.7 This software is an early prototype

Like all very early stage software you will find that this code will have all sorts of rough edges and issues associated with it that we did not yet have the time to work out. Please bear with us; we will do what we can to make this more user friendly.

We have tested enough core functionality to believe that it useful at least for some use cases.

If you find any problems with this code, please report them through this form:

http://evolvix.org/contact/bug-report

What we know about the state of our code is listed in the section "State of the Code". This includes bugs we fixed in this release and bugs we know about but could not yet fix, etc.

Thank you for your patience and for your help.

2.2 Operating System specific notes on Evolvix

Below is a list of supported operating systems.

If porting Evolvix to an operating system that is not listed here would be hugely helpful to you and others, please let us know. We will then see what we might be able to do.

We are currently focusing more on future developments and technologies to help us set up the best system we can develop. This means we have less resources for porting our system to older platforms. While we try to develop as platform independent as we can, there are limits to what the libraries and tools that we rely on will allow us to do.

Currently Evolvix requires a 64bit CPU.

2.2.1 Windows

This version is currently available in 64bit-release compiled form for

- Windows 7

We have not tested other Windows versions. If you want to become a beta-tester for Windows-releases of Evolvix, please do let us know.

Evolvix on Windows is now a full release version, so no more need to worry about installing Microsoft Visual Studio as with earlier releases of Evolvix.

2.2.2 MacOSX: Compatibility notice

This version is currently available in 64bit-release compiled form for

- Mac OSX 10.9 (“Mavericks”)
- Mac OSX 10.8 (“Mountain Lion”)

Please use the correspondingly compiled version.

Due to a dependency in the C++ compiler tools that we need, it is currently not possible to offer Evolvix for older MacOSX systems, so we do not support at the moment:

- Mac OSX 10.7 (“Lion”)
- Mac OSX 10.6 (“Snow Leopard”)

2.2.3 Linux: Compatibility notice

This version is currently available in 64bit-release compiled form for

- Ubuntu 14.04 LTS (“Trusty Tahr”)

We can get you a RedHat version, but have not tested other distributions. We are interested in making Evolvix compatible with other Linux distributions in the future.

2.3 How to get started with Evolvix

This document will cover:

- Installing Evolvix
- Running Evolvix
- Navigating Evolvix Results

2.3.1 Installing Evolvix

Installing Evolvix is really easy! All you need to do is:

1. Download the Evolvix.zip package.
2. Unzip the package. This will create a directory named *Evolvix*, which contains everything needed for Evolvix.

Evolvix does not access anything outside of this folder, so put in there all the Quest files you want Evolvix to access.

2.3.2 Installing a good source code Editor

If you never edited what professionals call "source code", then spare yourself the pain and do not use Word or some other text editor that allows formatting etc. There are many free editors that are good enough for starting. Some worth trying (in addition to LightTable):

- MacOSX: TextWrangler from http://www.barebones.com/products/textwrangler/
- Windows: NotePad++ from http://notepad-plus-plus.org
- Linux: You probably know an appropriate editor; other good ones are:
- Try Sublime (runs anywhere): http://www.sublimetext.com/3
- LightTable runs anywhere and is "downloaded from the future": http://www.lighttable.com

2.3.3 Installing R for plotting

To have Evolvix automatically produce plots whenever a QuestRun has finished, you need a recent R install on your computer. Download it from http://www.r-project.org/

If a Quest completes computation, but plotting either crashes or takes too long (e.g. too much data), then you may want to make sure:

- You have the latest version of R
- You didn't send too much data to R for plotting, eg. by demanding too much precision in TimeSeries.
- You didn't adjust the axes for your plot to demand non-existent logs of negative values.

Our plotting code works with the latest R intall (3.1.1), but not with the one before (3.1.0) whereas older ones were OK mostly. Of couse you can write your own R scripts as all the raw time series data is accessible in the files of a QuestRun.

2.3.4 Running Evolvix Efficiently

The following steps explain how to run Evolvix without wasting much time on navigation and other irrelevant complexities:

1. Open a command line interface (CLI)

2. Change the working directory to *Evolvix Home Folder*
3. Shortcut to *Evolvix Home Folder*
4. Check for a Quest file
5. Run Evolvix

We know, it would be easier to just press a button for this. We are working on it.

1. Open a command line interface (CLI)

The first step is to open a CLI application on your computer. These are called by different names depending on the operating system you are using: "Terminal" on OS X, "shell" on linux, and "command prompt" on Windows. Regardless of the name, they all do the same thing. A CLI allows for text-based communication between the user and the computer.

- On Mac OS X, the application *Terminal* is included in the Applications/Utilities folder.
- On Windows, the command prompt can be opened by typing "*cmd*" (without quotes) into the Start Menu search box and pressing enter. You can also open it from Start Menu->Accessories->Command Prompt.
- On many Linux distribution this is called "Terminal" and you can search for it.
- On other Unix like systems, there are many shells, and we will assume if you are using them that you know how to access the CLI or open your favorite shell. (If not, google it)

2. Change the working directory to *Evolvix Home Folder*

After opening a CLI, we need to make sure we're working within the *Evolvix Home Folder*. This directory was created when we installed Evolvix and contains everything needed for Evolvix to run inside. It is very important and is referred to throughout the documentation by names such as

- Evolvix Home Folder
- Evolvix directory
- Evolvix home directory
- Evolvix home
- Evolvix folder
- Evolvix main folder
- Evolvix main directory

If you know how to use the command line, that is all you need to know. For all others, here is how that works:

To change to this directory, we type a 'change of directory' command into the terminal:

MacOSX, Linux and other Unix systems:

```
cd "file path"/Evolvix
```

Windows:

```
cd "file path"\Evolvix
```

Where *cd* is the "change of directory command" and *"file path"* is the file path of your *Evolvix Home Folder*. Depending on where you installed Evolvix, this path can be pretty long.

- *OS X Hint* A useful shortcut is to drag the *Evolvix Home Folder* directly into the terminal window after typing "cd ". Make sure to type the 'space' after cd. In *Terminal*, this will fill in the file path for you.
- *Windows hint* You can right-click the *Evolvix Home Folder* that you unzipped and select properties. The Location identifies the file path to the *Evolvix Home Folder*
- Linux: Let us know your favorite hint and we will put it here.

3. Shortcut to *Evolvix Home Folder*

If you find yourself traveling to the *Evolvix Home Folder* a lot, you may appreciate the following shortcuts:

MacOSX:

The program "Go2Shell" can be installed on your Mac and then a link to it can be added to the tool-bar that you see in all Finder windows. Download it for free from:

https://itunes.apple.com/us/app/go2shell/id445770608

Windows:

As described by http://www.howtogeek.com/howto/windows-vista/stupid-geek-tricks-open-a-command-prompt-from-the-desktop-right-click-menu/ the following steps are quick on Windows:

```
1. Using the Windows GUI, go to the *Evolvix Home Folder*
   with the Evolvix binary file and have it as an open window.

2. With the *Evolvix Home Folder* window in the foreground:
   Press shift and bring up the context menu on your mouse.

3. Select the entry saying "Open Command Window Here".

4. This will open a command line window which has already changed to the right folder.
```

Linux:

Let us know your favorite hint and we will put it here.

4. Check for a Quest File

For Evolvix to run, we need to have a Quest file in the working directory.

A Quest file contains a description of the Model (Actions, Part) you want to compute and the TimeSeries you want to observe, along with the simulation Task you want performed.

This means that we need a Quest in the *Evolvix Home Folder* that we just cd'd into above.

If you just installed Evolvix, there should be a default demo Quest file named:

```
Default_Quest.Evolvix.txt
```

already there. If you want to run a different Quest file, you need to drag or copy it into the *Evolvix Home Folder*.

5. Run Evolvix

The CLI is working in the *Evolvix Home Folder* and we have a Quest file: now we're ready to run Evolvix! To simulate the model in the default Quest file, we simply type:

MacOSX, Linux and other Unix systems:

```
./Evolvix
```

Windows:

```
Evolvix
```

Typing the command above runs *Default_Quest.Evolvix.txt* within the *Evolvix Home Folder*. If we want to run a different Quest within the *Evolvix Home Folder*, we use the following command on Unix systems:

```
./Evolvix YourQuestFileName.txt
```

Windows:

```
Evolvix  YourQuestFileName.txt
```

Where *YourQuestFileName.txt* is the Quest file that we added to the *Evolvix Home Folder*.

6. Removing Errors

Successful modeling is essentially an exercise in removing errors at different levels:

- Syntax Errors
- Modeling Errors
- Visualization Problems

Visualization Problems.

Unless you are an R expert and are happy to write your own scripts, there is little you can configure out of the box in Evolvix 0.2.0; the exceptions are simple plot axes labels etc, see this file in the Evolvix Home Folder, which should be pretty self-explanatory:

```
Quest.Plot.Configuration.R
```

It is copied along with the Quest Source Code to the Archive that holds the information on a QuestRun.

Modeling Errors.

These are the hardest to catch, as they require a solid understanding of the complex dynamic biological networks which are being modeled. Often small parameter combination changes can have huge effects in terms of population sizes and computing times.

If your simulation does not end as expected, try a smaller simulation, i.e. one where there are smaller Part Amounts.

Also, watch out for parts that are unintended. For example:

```
Rabbit, Rabbits
```

are two Parts that are as different to the computer as Rabbit and Fox. However, chances are, the real difference is just a typo. Future versions of Evolvix will have facilities to catch such problems automatically. But for now it is up to the modeler to ensure that all spellings are correct.

Parser Errors.

If there are any typos or wrong syntax statements in the Quest, the parser will spit out Error Messages.

Do pay attention to these. Sometimes a simulation can run seemingly OK despite wrong input, generating results that could keep many biologists puzzled for a long time if it weren't clear that they are essentially caused by a bug.

Were there any parser errors?

Check this before you puzzle about weird biology. If you see this pattern:

```
========== Parse Start ==========
========== Parse End   ==========
```

With nothing in between as above, the parser was happy and did not report any errors.

An error looks like this:

```
``          ========== Parse Start ==========

*** Oops! The Evolvix Parser had problems understanding your code near the following position:
    Filename    :   /Users/.../Default_Quest.Evolvix.txt
    Linenumber  :   28            Position in line :   13
    Error reason is probably    : syntax error...
        For Experts :
        Error Type:   0   Meaning:   Unexpected token
        Problem may be near:
        [Index: 167 (Start: -377477821-Stop: -377477818) ='Info', type<16> Line: 28 LinePos:12]

             ========== Parse End   ==========
``
```

The currently employed parsing technology sometimes produces a long list of errors that are really only caused by one or a few real errors. We are looking into ways for how to produce better error messages.

Above, the useful information is mostly the Linenumber / Position, and a few strings that indicate what might cause the parser to fail.

7. Output of a successful QuestRun

Evolvix produces the following sample output on the command line (with a bit more whitespace):

```
``Welcome to Evolvix!
  Version 0.2.0 will now process your Quest file.
  Processing quest file: "/Users/..../Evolvix_0.2.0_OS_X_10.9_Release/Default_Quest.Evolvix.txt"
  ========== Parse Start ==========
  ========== Parse End   ==========
  *****************
  Using processed Quest file Default_Quest.Evolvix.txt.20140719_012351.epb.
  Evolvix Worker Version 0.2.0 is starting.
  Initializing the DAE_IDAS_Dense Worker
  Starting the DAE_IDAS_Dense Worker
  Plotting data produced by the worker
  Plotting complete
  *****************
  A copy of the new PDFs and raw time series data is in Results/Most_Recent.
  The new data is archived in Results/Temporary/eData_2014-07-19__01h23m51s
  Run_and_Plot.sh script completed.  Exiting. ``
```

It shows the various phases of a run and can be helpful for determining, where something went wrong.

2.3.5 Navigating Evolvix

Running a Quest will automatically produce and open the file *TS_Overview*, which contains the following (TS stands for TimeSeries):

- A TimeSeries with every Part together

- A TimeSeries for each Part individually

But this is just the overview of the results. The actual results are stored in the *Results* directory, which has three subdirectories:

- Most_Recent
- Temporary
- Saved

1. Most_Recent

The *Most_Recent* directory contains all of the results of the most recent Quest run. The *PDFs* subdirectory contains the *TS_Overview* as well as all plots of individual TimeSeries that were defined within the Quest. The *Quest* subdirectory contains a copy of the Quest file. The *Raw_Time_Series* subdirectory contains .txt files of each of the defined TimeSeries. These .txt files are used to create the defined TimeSeries plots.

2. Temporary

The *Temporary* directory contains a separate subdirectory for all Quest runs. Like *Most_Recent*, each of these subdirectories contain *PDFs*, *Quest*, and *Raw_Time_Series* folders. Each subdirectory has a timestamp in its name to indicate when the Quest ran that created this directory.

3. Saved and deleted Quests

Move any Quest folders that you want to keep to the folder

    Saved_(=moved_here_by_you)

or some equivalent folder of your choice.

This makes it very easy to throw away many uninteresting QuestRuns, something you will be doing a lot when you work with Evolvix. To do this just select all Runs to be deleted in the

    Temporary

folder and delete them by moving them to your Trash. That's it.

Just make sure your important QuestRun Results have been moved to your "Saved" folder before.

When using a git repository, results in the *Temporary* and *Most_Recent* directory will not be included in any commits (see the ".gitignore" file in the Evolvix Home Folder).

We set up the *Saved_(=moved_here_by_you)* directory so that files in it will be included in git commits.

However, you must manually drag desired results from *Temporary* to *Saved*. Running Evolvix will never produce results in *Saved*.

CHAPTER

THREE

TUTORIAL

3.1 Tutorial Start Modeling with Evolvix: A very simple model of predator-prey mass action

by Jacob Goldfinger

Here we will cover:

- A very brief introduction to Evolvix and mathematical modeling
- How to convert a simple biological model into Evolvix code
- How to use Evolvix to obtain information about this system

3.1.1 Quests, question and mathematical models

A Quest is a pursuit made in order to obtain something. Creating an Evolvix Quest allows us to obtain information about the model of a biological system by simulating it with the goal to answer a question.

A mathematical model describes a system in a particular mathematical form that builds on well-defined assumptions and facilitates the computation of various properties of interest. There are various mathematical forms that are frequently used for building models of biological (and other) systems. Some of them are easier to compose than others.

Evolvix provides a form that makes it relatively simple to describe a broad range of models. Once described in Evolvix, a particular model can be converted automatically into various mathematical forms that facilitate particular specific analyses or simulation techniques.

It is important when building and analyzing mathematical models to keep in mind the question that motivated the construction of the model, otherwise we risk creating a model that answers a question that nobody asked. Our question is what we seek to know about a system and if we build a good model, this question will profoundly affect what we include into our model and what we abstract away or ignore. Because of this importance, Evolvix provides a way to make such motivating questions first-class citizens in the Evolvix code that describes models and simulation tasks.

In this first lesson we will use a very simple biological model that consists of Snowshoe Hares (the prey) and Canada Lynx (the predators). We will create an Evolvix Quest to answer the Question of how many hares and lynx can we expect in five years if we start with specific population counts and defined rates of birth and death.

The steps we will follow to create our Quest will be:

1. Define the Quest and Question
2. Define the Initial Amounts of the Parts

3. Define the Actions
4. Define the Simulation

Let's make a Quest!

3.1.2 1. Define the Quest and Question

The first step is to define our Question. This Question motivates our Quest for answers, which we will obtain in the form of simulation results after a defined simulation Tasks have been completed.

In this example, we want to know how many hares and lynx there will be after five years if we start with 27000 hares and 8500 lynx.

Let's begin by defining an **Evolvix Quest** and documenting our **Question**:

```
Evolvix Quest (

        Question: "How many hares and lynx will there be in five years?"

    """
    Details:
    We assume a starting point of
     27000 Snowshoe Hares (have always enough to eat) and
     8500 Canada Lynx (only eat Hares)
    in the study area, which we assume to be closed
    (no migration in our out of that part of Canada).
    We also assume that no other species interact with either and
    that the environment is constant enough and that
    our birth and death rates are correct and don't change over the
    time horizon we are interested in.
    """
)
```

Here we have declared that our Quest is devoted to answering the question specified. A one-line-summary of the Question is provided first (in single-double quotes) and additional details are provided later (in triple-double quotes). Later we will learn that this structure is called an "InfoBlock" and that it can carry a great amount of details.

Now that we have begun our Quest by defining our Question, we are ready to describe those aspects of our system that we think are important for answering our Question.

3.1.3 2. Define the Initial Amounts of the Parts

A model in Evolvix consists of Parts and Actions.

- **Parts** are the basic elements of a system. In our system there are two Parts: Hares and Lynx.
- **Actions** are the events within the system that change the Amounts of our Parts. We will discuss Actions in the next section.

Our Question is about the Amount of Hares and Lynx that will be there in five years. To compute this, we need to define somewhere in our Quest how many Hares and Lynx there are now.

In Evolvix we can do this with an "**Initial Amount**" phrase. It gives

- the Name of a Part and
- an Amount, which is assigned to the named Part at the start of a simulation.

The Name of a Part is an Identifier that unambiguously identifies a given Part in an Evolvix Model. Formally, in Evolvix we write:

```
Initial Amount of PartName = AmountNumber
```

where the *PartName* and *AmountNumber* are substituted appropriately. Let's define the Initial Amount for the Parts in our Quest, which is the number of Hares and Lynx there are right now (year 0):

```
Initial Amount of Hares = 27000
Initial Amount of Lynx  =  8500
```

To state the obvious, these lines describe a system that consists of 27000 Hares and 8500 Lynx initially.

Please note that Hares and Lynx are capitalized. Evolvix is case-sensitive: if we capitalize a Part name anywhere, we must capitalize it always.

Now that we've defined the Initial Amounts, we can move on to describe the events that change these Amounts: the Actions.

3.1.4 3. Define the Actions

The Amounts of Hares and Lynx present in the system are constantly changing. In Evolvix, we use **Actions** to represent these changes. Every Action has a Rate that describes how often these changes occur (more often = higher Rate = higher Frequency = lower waiting time between events). Our system consists of four Actions and we will cover each of them in detail.

To define an Action we need to specify:

- The consumed Part(s) ("Input")
- The produced Part(s) ("Output")
- The Rate at which the Action occurs

All of this is very similar to a the concept of chemical reactions, where:

```
Input_Molecule_A  + B    ----[react to produce]----->   Output_Molecule_C
```

Indeed, Evolvix is using a syntax that mimics this way of writing chemical reactions. The reaction above would be written as the following Action in Evolvix:

```
Action  ( Input_Molecule_A  + B  ------>   Output_Molecule_C )
```

Note that neither of the Actions above had specified a Rate. In Evolvix this would mean that we can automatically assume a standard rate of 1.

If we are working in a big system and want to make it easy to specify which Action we are discussing, then we can also specify a:

- `UserName` like for example: `MyFirst_Action` and a
- `UserIndex` which can just number all actions in the model in a sequence.

This translates into the following syntax example, where we made the standard rate explicit:

```
Action UserIndex UserName (

        Input_Consumed_1 + Input_Consumed_2
        ----------[ Rate = 1 ]------------>
        Output_Produced

)
```

If you add a consumed Part *and* a produced part, then the Part Amount is not changed, but it affects the overall rate of this Action. This can be used to model what catalysts do for an Action and can also be used to give parameters a name. Setting such named parameters to an Amount of 0 will switch off an Action, setting them to an Amount of 1, will make the effects of the parameter disappear.

Now back to Hares and Lynx. Before we code our Actions, it is important to think about the Question we are asking and the assumptions we make. We listed the most important assumptions as part of the details of the Question we specified when defining our Quest.

These assumptions mean that our system consists solely of Snowshoe Hares and Canada Lynx. While other animals are certainly present in the real system, we assume that their presence has no effect on the Hares-Lynx system and can thus be ignored in our model. Another assumption we make is an unlimited supply of food for Hares, so we do not need to model vegetation either.

These assumptions lead us to create four Actions for our Quest which implements what is known as the "Lotka-Volterra model of predator-prey dynamics" (see "Supplementary Info" at the end of this lesson).

In qualitative terms, these Actions are:

1. The Hares population increases due to the Hares breeding
2. The Hares population decreases due to Lynx eating Hares
3. The Lynx population increases due to Lynx breeding (to do so, they must consume enough Hares)
4. The Lynx population decreases due to some Lynx dying of starvation

Let's convert these qualitative descriptions into valid Evolvix Actions by using the Action blueprint presented above. Details on how we obtained the the following rates are discussed at the end of this lesson.

Action 1: The Hares population increases due to the Hares breeding

```
Action 1 Hares_Breeding ( 1 Hares ---[ Rate = 1.5 ]---> 2 Hares )
```

This Action states that on average Hares reproduce into two Hares at an effective rate of 1.5 times per year.

Action 2: The Hares population decreases due to Lynx killing Hares

```
Action 2 Lynx_Eating ( 1 Hares + 1 Lynx ---[ Rate = 0.000225 ]---> 1 Lynx )
```

This Action states that on average at a rate of 0.000225 times a year, one of the Lynx kills one of the Hares.

Action 3: The Lynx population increases due to Lynx eating Hares

```
Action Lynx_Breeding ( 1 Lynx + 1 Hares ---[ Rate = 0.000045 ]---> 2 Lynx )
```

This Action states that on average at a rate of 0.000045 times a year, one of the Lynx consumes one of the Hares to produces an additional Lynx. This may be confusing; note that this Action is very similar to Action 2. This will also be addressed in the supplement.

Action 4: The Lynx population decreases due to some Lynx dying from starvation

```
Action Lynx_Dying ( 1 Lynx ---[ Rate = 0.225]---> 0 Lynx )
```

This Action states that on average at a rate of 0.225 times year, a Lynx will die from starvation.

Now that we have defined all of our Actions, we only have one step remaining: to define our simulation!

3.1.5 4. Define the Simulation

The purpose of a Quest is to answer a Question about a biological system. We asked the Question in step one. We converted our system into a series of Parts and Actions in steps two and three. To complete our Quest, we will **Simulate** our system in a manner that is sufficient to answer our Question.

There are two broad classes of simulations in Evolvix: deterministic and stochastic. Stochastic simulations include randomness while deterministic simulations do not. This means that every time we run the same deterministic simulation of the same system, we will get the same results. For this tutorial, we will use a deterministic simulation.

Our Question asked: "How many Lynx and Hares will there be in five years?". We will Simulate our system deterministically for five years to answer this Question:

```
Simulate deterministically until 5
```

That's it! The Simulation we have defined is sufficient to answer our Question. We have now specified the source code for our Quest and when we run it in Evolvix, it will by default produce a plot that allows us to read off the Amounts for a given time.

We have now completed our first very simple Quest! We have covered enough ground to have a starting point for exploring the many possibilities of how to analyze models using Evolvix.

3.1.6 Summary

We began this lesson with a discussion of Evolvix and mathematical modeling. We then discussed Quests and the procedure to construct one. The four steps to create a Quest are:

1. Define the Quest and Question
2. Define the Initial Amounts of the Parts
3. Define the Actions
4. Define the Simulation

Step one asks what we want to know about our biological system. Steps two and three convert that system into a usable format. Step four simulates the system to get us the information we want to know.

Let's look at the complete **Quest**:

```
Evolvix Quest ( Question: "How many hares and lynx will there be in five years?" )

Initial Amount of Hares = 27000
Initial Amount of Lynx = 8500

**Action** Hares_Breeding ( 1 Hares          ---[ Rate = 1.5      ]---> 2 Hares)
**Action** Lynx_Eating    ( 1 Hares + 1 Lynx ---[ Rate = 0.000225 ]---> 1 Lynx )
**Action** Lynx_Breeding  ( 1 Hares + 1 Lynx ---[ Rate = 0.000045 ]---> 2 Lynx )
**Action** Lynx_Dying     (           1 Lynx ---[ Rate = 0.225    ]---> 0 Lynx )

Simulate deterministically until 5
```

3.1.7 Supplementary Info

We used the Lotka-Volterra equations for predator-prey dynamics to create the Actions in this lesson.

$$\frac{dHares}{dt} = (1.5 * Hares) - (0.000225 * Hares * Lynx)$$
$$\frac{dLynx}{dt} = (0.000045 * Hares * Lynx) - (0.225 * Lynx)$$

Because the Lotka-Volterra model is so old, we were unable to find a specific paper that describes the interaction between the Snowshoe Hare and the Canada Lynx using that model. We were, however, able

to find long-term data that suggest that both Hare [1] and Lynx [2] populations fluctuate in an 8-11 year cycle. It has also been verified that the Lynx population cycle follows the Hare population cycle [3]. We created our own parameters for the Lotka-Volterra equations that created oscillations similar to these 8-11 year experimentally-proven oscillations, using the same population data that these papers used.

The four Actions in the paper are based off the four terms of the two Lotka-Volterra equations.

The first term in the first equation, *1.5 * Hares* , corresponds with the first Action: HaresBreeding. On average over the entire Hare population, each Hare produces another Hare at a rate of 1.5.

You may have been confused by the similarity between the second and third Actions; LynxEating and LynxBreeding, which correspond to the second term of the first equation, *-0.000225 * Hares * Lynx* , and the first term of the second equation *0.000045 * Hares * Lynx* , respectively. Whenever a Hare and a Lynx meet, the Lynx kills the Hare. Both of these Actions describe this interaction. However, Lynx eat Hares to use them as a energy source. Once they have consumed enough Hares, they will have gathered enough energy to reproduce. The Action HaresKilled states that a Lynx kills a Hare, but *does not* produce a Lynx from this kill, at a rate of 0.000225. The Action LynxBreeding states that a Lynx kills a hare, and *does* produce another Lynx from this kill, at a rate of 0.000045. To simplify this explanation, look at the ratio of the rates. 0.000225:0.000045 is the same as the ratio 5:1. You can think of these two Actions together as: it takes six Hares for a Lynx to gather enough energy to reproduce.

The second term of the second equation *0.225 * Lynx* corresponds with the fourth Action: LynxDying. When a Lynx fails to consume any Hares, it will die. On average over the entire Lynx population, a Lynx dies at a rate of 0.225.

[1] MacLulich DA (1957) The place of change in population processes. Journal of Wildlife Management 21: 293-299.'

[2] Elton C, Nicholson M (1942) The ten-year cycle in the numbers of the lynx in Canada. Journal of Animal Ecology 11: 215-244.

[3] Tyson R, Haines S, Hodges KE (2010) Modelling the Canada lynx and snowshoe hare population cycle: the role of specialist predators. Theoretical Ecology 3: 97-111.

CHAPTER

FOUR

MINI SYNTAX MANUAL

4.1 Evolvix-0.3.0 Mini-Syntax-Manual

by Laurence Loewe, Jacob Goldfinger and Iratxo Flores-Lorca

Laboratory of Genetics
Wisconsin Institute for Discovery
University of Wisconsin-Madison

This page briefly describes some of the key syntactic constructs of:

Evolvix 0.3.0 beta
Release date 2014-09-30

Some of the syntax given below will only work with this version. Evolvix backwards compatibility will start only with Evolvix Version 1, as we use semantic versioning (http://semver.org).

To enable a simpler and more regular grammar some of the syntax documented here will be replaced in future versions of Evolvix. For more details, please see http://evolvix.org/ and these notes on changes to be expected in the next releases of Evolvix.

The rest of this document will give in a brief overview of various syntax constructs that are used to specify which model Evolvix 0.3.0 should simulate and what data should be recorded during simulations.

4.1.1 Comments all start with ExclamationMarks: !

The following symbols can be used for commenting in Evolvix:

```
!L A one line comment
!l Also a one line comment

!--  A  !--  nestable  --!    multi-line comment that works --!
```

Do not put any of these inside of a TimeSeries statement (else the parser will complain).

These symbols will change in the next release of Evolvix to improve consistency.

4.1.2 Evolvix Quest Question

OneLineSummary of Purpose

Help document why we do what we do.

Description

Model quality stands and falls to a very large degree with the question that is being asked. Thus Evolvix provides a prominent place where to document the

Question or Purpose or Motivation or Hypothesis

that defines what a Quest is all about. This must be present in every Quest.

Usage

A fully specified default expression looks like this:

```
Evolvix Quest (
    InfoType: "OneLineSummary" "Details" AuthorLines  (
        InfoBlockList
    )
)
```

Arguments

InfoType One of four possible InfoType that describe why a Quest is developed:

```
Question
Purpose
Motivation
Hypothesis
```

OneLineSummary A brief (1 Line!) summary that allows quick overviews over many Quests.

Details An arbitrarily long string that describes what the Quests is about and maybe even starts to list intermediate results.

AuthorLines Document the involvement of various people:

```
Reviewed by AuthorZ   on 2014-07-17 version v0r2c1
Modified by AuthorY   on 2014-03-63 version 2.3.4.3.5.4
Created by  AuthorX   on 2014-01-23 version v0r1c0
```

InfoBlockList A list of InfoBlocks with Type, Summary, Details and AuthorLines, as defined above. Can also include an InfoBlockList for arbitrary nesting.

Details

The quotes for *OneLineSummary* and *Details* can be "Single Double Quotes" or """ Triple "Double Quotes" to allow for normal quoting in a longer Details text """.

The OneLineSummary, well, should only be one line.

The DetailsText and all AuthorLines are optional. DetailsText is encapsulated by triple quotes (""") and can be as long as you want. It can contain any combination of characters except for triple quotes, which by definition ends this.

AuthorName must be a valid Identifier, meaning: - only letters A...Z,a...z, digits 0...9, and underscores "_" - must begin with a letter - can not be a keyword

Dates must be given in YYYY-MM-DD format, to avoid the confusion from the various existing permutations that are internationally in use.

Version info can be specified in various ways, including as UserIndex ("2.3.4.3.5.4", see below). If Semantic Versioning is used, then the following format is recommended:

```
v1r2c3 which stands for

v = Version 1 ; A change here breaks the backwards compatibility
                of the meaning of what was used before.
```

```
r = Release 2 ; A change here adds significant new features,
                but does not change existing meaning
                (i.e. the old code still works as intended =
                backwards compatibility is maintained)
c = Change  3 ; Any small change that does not add any
                substantial new feature;

Bugfixes to bring the actual code in accord with the official specification
affect either v or c, depending on gravity.
```

Example

Code:

```
Evolvix Quest (
    Question: "What do foxes and rabbits to after some time?"
    """
    This is the completely optional DetailsText.
    You can write whatever you want here, using any 7bit ASCII
    characters you want, eg. *$*Q#$&*2434 () {}
    Use triple quotes to encapsulate the DetailsText, in
    order to allow for including "Quotes".
    """
    !L AuthorLines
    Reviewed by AuthorZ   on 2014-07-17 version 45.234.2.1
    Modified by AuthorY   on 2014-03-63 version v0r2c0
    Created by  AuthorX   on 2014-01-23 version v0r1c0

    (
        Hypothesis: """Oscillations do not overlap"""
                    "  Details: how different will they be? "
        Created by Researcher_NoSpaces  on 2014-07-17 version v0r2c0
        Evidence: "Hare + Lynx exist"  """ Detailed records """
        Created by Researcher_2  on 2014-07-17 version v0r2c0
        Unknown: "Their Dynamics"  """ Detailed questions """
        Created by Someone  on 2014-07-17 version v0r2c0
    )
)
```

4.1.3 Rules for Names

While we work to expand the options for how to name Parts, Actions and everything else in Evolvix, the parser in Evolvix 0.3.0 supports the following:

- Names MUST start with a letter from the English alphabet 'a...z' or 'A...Z'
- Digits '0...9' and underscores '_' are only allowed after the initial letter.
- No leading or trailing '_'
- No leading digits
- For now the total length of a Name must be more than one character (in Evolvix 0.3.0)
- Part and Actions can have the identical Names; they are recognized as different because they have different Types.
- Different Names for Parts currently automatically create different Parts. While convenient, this creates the danger of creating different Parts by accident. We are working to implement facilities to reduce this problem.

- See below for UserIndexes, which provide an additional way of identification in text and in documentation (but not for use in functional code).

4.1.4 Rules for Numbers

Numbers in Evolvix can of the following Types:

Float "Floating point Numbers" allow the use of fractions and of an exponent to specify for example: 1.2435 or 43352.3243 or 34.2e-24 Please note that the dot '.' is used as decimal point and that a comma in Evolvix ',' is used to separate elements in a list.

Floats *must* start with a digit '0...9' and cannot omit a leading 0, so ".2" would lead to an error, if used instead of "0.2".

Integer are simple sequences of digits "0...9".

UserIndex To number Actions, Parts and other Constructs in Evolvix in addition to giving them Names, they can be given a "UserIndex", which is a manually assigned (for now) number, which can start out as a running number that looks like an Integer. However, as the model code is modified, actions are split, removed, introduced, etc, the running numbers can look increasingly "disordered". They can be renamed by the user at any time and are meant to make it easy to talk about "Action 7.2" if the other Names are rather unwieldy. Format:

- Integer (shortest possible)
- Integer.Integer (if an index needs to be inserted)
- Integer.Integer.Integer (for the next level of insertion)
- Integer.Integer.Integer...(keep going for additional levels of insertion)

Example of legal UserIndexes:

4 , 2.3 , 2.4.3.3 . 7.45.234.1.5.245245.5.2

We are working on a system that allows more expressivity for numbers.

4.1.5 Units

For any Model the Units of the Numbers provided matter very much.

Units are essentially statement about the Type of a Number and since Types will be indicated by a leading ':' in future version of Evolvix, Units are indicated by a leading colon as well.

Other than that Units are given in square brackets (as often done in some contexts).

Future version of Evolvix will allow for automated checks of Units to ensure that the model is consistent. To be compatible with such a future expansion, we use "double quotes" to indicate any Unit that should not be checked automatically by Evolvix (to raise an error if inconsistent). Example use:

:["years"]

Note: Units work in most places where you would expect them; but have not been checked for completeness yet. Also, a previous versions of Evolvix omitted the leading ':', which will become important to maintain backwards compatibility.

If you have difficulties with Units, please let us know.

4.1.6 Initial Amount

OneLineSummary of Purpose

Set the Initial Amount for some named Part at the beginning of a Simulation.

Usage

A fully specified default expression looks like this:

```
Initial Amount of *PartName* = Value
```

Arguments

PartName must be a valid Identifier, meaning: - only letters A...Z,a...z, digits 0...9, and underscores "_" - must begin with a letter - can not be a keyword

Value Any number >0, negative Part Amounts make no sense. The stochastic simulator will add and subtract full numbers even if the Amount is not an integer number. May be used to represent a constant Action Rate Model Parameter, if Actions are set up accordingly (i.e. never change the parameter).

Details

This "sentence" as most in Evolvix is case-sensitive.

What is the logic behind this? Simple:

> "Capitalized words are Essential For the Parser", but lower-case words are just for reading convenience. For example in the quoted text, "words", "are", and "the" could be dropped, without any complaints by the Parser, such that "Capitalized Essential For Parser" would have exactly the same effect (but is less readable).

Example

Code:

```
Initial Amount of Rabbits = 500
```

4.1.7 Simulate

OneLineSummary of Purpose

Specify the simulation task.

Usage

A fully specified default expression looks like this:

```
Simulate   *ApproachUsed*  until *Time*  :["Units"]
```

Arguments

***ApproachUsed* must be one of these:** Deterministically Stochastically

Time Any number >0; negative Times are not allowed as they crash the deterministic ODE solver.

Units Optional specification of Units for the Time. For documentation purposes, currently not enforced.

Details

This "sentence" as most in Evolvix is case-sensitive.

Example

Code:

```
Simulate Stochastically     until 100
Simulate Deterministically  until 100
```

4.1.8 Action

OneLineSummary of Purpose

Specify an Action by listing all the Parts that interact in this Action.

Usage

A fully specified default expression with text to be replaced by numbers looks like this:

```
Action UserIndex Name  (

      Stoichiometry PartName
    + Stoichiometry PartName
    + Stoichiometry PartName
        ...
    ---[ Rate = RateValue ]--->
      Stoichiometry PartName
    + Stoichiometry PartName
    + Stoichiometry PartName
        ...
)
```

where "..." stands for "more of the same".

Arguments

optional *UserIndex* is a number with many levels. Example: 1.4.23.64.23.

optional *Name* identifies the Action and denotes what it does without having to list all reactants. *Name* must be a valid Identifier, meaning: - only letters A...Z,a...z, digits 0...9, and underscores "_" - must begin with a letter - can not be a keyword

optional *Stoichiometry* default = 1 if not specified, denotes how many parts are reacting and get consumed or how many are produced.

PartName must be specified to denote the identity of the Part that is linked to this Action. *PartName* must be a valid Identifier, meaning: - only letters A...Z,a...z, digits 0...9, and underscores "_" - must begin with a letter - can not be a keyword

RateValue Any number >0, negative rates make no sense.

It can currently only accept a floating point number and not a parameter name. While we work to changes this, here is a trick to effectively have parameters in Evolvix 0.2.0:

1. Set Rate = 1 to essentially avoid its use.
2. Define the Initial Amount of

   ParameterX = 2.3

   This can be a floating point number but must be positive (is true for Rates and Amounts in MassActionModels.)
3. Add ParameterX to both sides of an Action, transforming:

   ```
   Action (   Ax + Bx              ---[ Rate = 1 ]--->              Cx )
   ```

   into:

```
Action (   Ax + Bx + ParameterX ---[ Rate = 1 ]---> ParameterX + Cx )
```

which does not change ParameterX (subtract + add cancels out when the Action Occurs), but since it is now a reactant that does not change its Amount, it is multiplied along with all other terms to get the propensity.

The stochastic simulator will add and subtract full numbers even if the Amount is not an integer number.

PartName must be a valid Identifier, meaning: - only letters A...Z,a...z, digits 0...9, and underscores "_" - must begin with a letter - can not be a keyword

Details

Careful with typos: If misspelled (eg, Rabbit vs Rabbits) a Part will create what amounts to a new Part that is linked to this action, but otherwise irrelevant for this system.

The length of Arrows shall be 3 dashes or more (any length >=3 is allowed), so they can be used as line separators.

In Evolvix 0.2.0 all Actions and all Simulators strictly follow mass action kinetics. It is not possible to specify other kinetics in this version of Evolvix. Future versions will relax this restriction.

Currently Actions are not reversible.

Example

Code:

```
Action (   Ax + Bx    ---[ Rate = 1 ]--->  Cx  )

Action 23.5 MyExampleAction
(
    A_Very_Long_Chemical_Name_Following_Some_Standard_Or_Encoding_Space
+   B_Shorter_PartName
+   C_PartName_Makes_Sure_This_ActionCertainlyDoesNotFitInALine +

ParameterX
---------------[ Rate = 1 ]----------------------------------------->
ParameterX

+   D_Product
)
```

4.1.9 TimeSeries Overview

OneLineSummary of Purpose

Report all the amounts in time requested at the precision requested to maintain key features of the time series, but without storing too much information. In other words: remove whatever we don't need. However, in difference to "Separately", this option reports all values if any single value gets reported, so no value is ever left without its simultaneous peers.

Usage

The shortest possible phrase for reporting TimeSeries for the Parts P1, P2 is:

```
TimeSeries   (
        Report Separately     the Amount of Part    P1, P2
)
```

The shortest possible phrase for reporting the data for a PhaseDiagram for the Parts P1, P2 is:

```
TimeSeries   (
        Report Simultaneously the Amount of Part    P1, P2
)
```

A fully specified TimeSeries phrase looks like this:

```
TimeSeries UserIndex Name (
        Report Separately     the Amount of Part    ListOf_CommaSeparated_PartNames
!L or:  Report Simultaneously the Amount of Part    ListOf_CommaSeparated_PartNames
        From                                        StartTime           :["TimeUnit"]
        Until                                       StopTime            :["TimeUnit"]
        Ignore values below                         ValueMin            :["ValueUnit"]
        Ignore values above                         ValueMax            :["ValueUnit"]
        Report whenever values change by            DifferenceMeasure   :[  DifferenceType ]
        Report times when crossing values           ListOf_CommaSeparated_Times
        Report values when crossing times           ListOf_CommaSeparated_Values
)
```

Warning: Due to some parser problem the sequence of most of these statements cannot be changed, as discussed more below. We are planning extensions to the TimeSeries syntax, which is likely to change in the near future.

Arguments

ListOf_CommaSeparated_PartNames

- Names must be separated by comma
- No comments allowed here or in any such list.
- Names must contain only valid Evolvix Names (see above)
- these will be reported simultaneously,
- to "Report Simultaneously" in older versions of Evolvix, a list of pairs has to be given, one for each phase diagram requested.
- to "Report Separately" and in newer versions of Evolvix, all Part Names are just listed once: Example: Part_1, Part_2, Part_3 , Foxes, Rabbits

Time*, StartTime, StopTime, *ListOf_CommaSeparated_Times Any time allowed, but ODE solvers only work with positive times. StopTime must be larger than StartTime. Both default to the simulation start and stop time. The TimeUnit will specify more details about the meaning of the times. The List must contain only numbers, each separated by a comma and nothing else else Evolvix 0.3.0 would crash. The default for ListOf_CommaSeparated_Values is an empty list.

Value*, *ValueMin*, *ValueMax*, *ListOf_CommaSeparated_Values Any number allowed, negative Part Amounts make no sense, but negative fluxes do. The ValueUnit will specify more details about the meaning of the Values The List must contain only numbers, each separated by a comma and nothing else else Evolvix 0.3.0 would crash. ValueMin must be smaller than ValueMax, both default to +-1e100 The default for ListOf_CommaSeparated_Values is an empty list.

DifferenceMeasure*, *DifferenceType The DifferenceMeasure(s) provide a quantitative cutoff for the difference between current value and last recorded value. DifferenceMeasure must be >= 0 and defaults to 0.05. DifferenceType can be any of the known methods:

- % Relative Difference = see relative error definition.
- Absolute Difference = same units as observed values.

- Magnitudinal Difference = decadic logarithm of ratio of differences (see Error of Magnitude in Loewe 2007, SWP&E)
- RelAbsMix Difference = mixes relative and absolute differences (Weighted Root Mean Square, see SunDials manual)

and defaults to Magnitudinal.

You can use one of the following lines to specify both:

- Report whenever values change by 5 [% Relative Difference]
- Report whenever values change by 100 [Absolute Difference]
- Report whenever values change by 0.05 [Magnitudinal Difference]
- Report whenever values change by 0.1, 0.2 [RelAbsMix Difference]

Details

The TimeSeries phrase is case-sensitive, as many in Evolvix 0.3.0. Due to parser restrictions you cannot add comments or change the order of the statements (see below).

The times for "Report values when crossing times" must refer to times after or when the simulation starts. Specifying times after the simulation is scheduled to end will not make a simulation run longer; it will only cause TimeSeries to report nothing for these times, as neither values nor times could be observed.

If the user specifies a TimeSeries Query, then she must specify what Parts she wants to observe in ListOf_CommaSeparated_PartNames. If the user does not specify one of the other inputs, Evolvix will use one of the default values in its place.

"Ignore values" may not work for deterministic simulations if the values are greater or equal to 1000.

Example

Code:

```
TimeSeries 1   (
    Report Simultaneously the Amount of Part    Lynx, Hares
    From                                        5
    Until                                       100
    Ignore values below                         200
    Ignore values above                         10000
    Report whenever values change by            100   [ Absolute Difference ]
    Report times when crossing values           200, 9000, 683
    Report values when crossing times           5,22,25,27,30
)

TimeSeries 2   (
    Report Simultaneously the Amount of Part        Hares
    From                    0    :["years"]
    Until                   100  :["years"]
    Report whenever values change by 20 [% Relative Difference]
)
```

It is possible to report Flux:

```
TimeSeries  (
        Report Separately the Flux of Part           Lynx, Hares
)
```

4.1.10 TimeSeries of Amounts Separately

This way of recording TimeSeries saves most time and space but is also less convenient for some tasks, especially those that assume that all TimeSeries have the same length. Here is how to Report these:

Code:

```
TimeSeries UserIndex Name (
    Report Separately the Amount of Part        ListOf_CommaSeparated_PartNames
    From                                        StartTime           :["TimeUnit"]
    Until                                       StopTime            :["TimeUnit"]
    Ignore values below                         ValueMin            :["ValueUnit"]
    Ignore values above                         ValueMax            :["ValueUnit"]
    Report whenever values change by            100                 [ Absolute Difference ]
    Report times when crossing values           ListOf_CommaSeparated_Times
    Report values when crossing times           ListOf_CommaSeparated_Values
)
```

See above for Argument explanations.

"Separately" as an option is switched on by default if no other time series query is specified.

It is possible to report Flux:

```
TimeSeries  (
        Report Separately the Flux of Part          Lynx, Hares
)
```

4.1.11 TimeSeries Regular

The time series observations above are clever in that they guarantee that no big changes will be missed (within the specified accuracy) and that only the data necessary is reported and not more.

However sometimes value readings are needed at regular time intervals.

This can be obtained with the following query:

Code:

```
Report whenever times change by 20 :[ "years" ] !L interval length
```

It is still possible to add requests for additional times or the crossing of special values:

Code:

```
Report values when crossing times           10,90
```

Example

This simple example combines it all.

Code:

```
TimeSeries 3 (
   Report Separately the Amount of Part Lynx
   Report whenever times change by 2       :[ "min" ] !L this is by interval
   Report values when crossing times           10,90
)
```

It is not possible to set Lower or Upper value limits for this way of observing Timeseries.

In this case no other filters are available and trying to use them will not work.

Default Values for regular time steps 1/2000th of the simulation end time.

4.1.12 TimeSeries Parser Problems

Bad Code Examples to help you Debug Evolvix 0.3.0 code

Due to some deficiencies in the parser

1. you cannot swap the order of statement in a TimeSeries query and
2. you cannot add comments inside of any list of Times, Values or PartsNames that are accepted by TimeSeries

The following code will lead to a parser error and will not report what it should, or even crash. We plan to fix this in the next release. Here is what you should NOT do:

Code:

```
 TimeSeries BAD_CODE_EXAMPLE  (
    Report Separately the Amount of Part Lynx
    Report values when crossing times           10,90
    Report whenever times change by 2 :[ "min" ] !L the order is wrong
)


 TimeSeries REALLY_BAD_CODE_EXAMPLE_WILL_CRASH_THE_PARSER (
    Report Separately the Amount of Part Lynx       !L this comment is fatal
    Report values when crossing times      10,90    !L this comment is fatal
    !L This comment is fatal too, the new line does not help.
    !L Essentially, whenever the TimeSeries expects a list of Times, Values or PartsNames,
    !L you cannot add a comment without crashing the parser.
 )
```

CHAPTER

FIVE

RELEASE NOTES

5.1 Operating System specific notes on Evolvix

Below is a list of supported operating systems.

If porting Evolvix to an operating system that is not listed here would be hugely helpful to you and others, please let us know. We will then see what we might be able to do.

We are currently focusing more on future developments and technologies to help us set up the best system we can develop. This means we have less resources for porting our system to older platforms. While we try to develop as platform independent as we can, there are limits to what the libraries and tools that we rely on will allow us to do.

Currently Evolvix requires a 64bit CPU.

5.1.1 Windows

This version is currently available in 64bit-release compiled form for

- Windows 7

We have not tested other Windows versions. If you want to become a beta-tester for Windows-releases of Evolvix, please do let us know.

Evolvix on Windows is now a full release version, so no more need to worry about installing Microsoft Visual Studio as with earlier releases of Evolvix.

5.1.2 MacOSX: Compatibility notice

This version is currently available in 64bit-release compiled form for

- Mac OSX 10.9 (“Mavericks”)
- Mac OSX 10.8 (“Mountain Lion”)

Please use the correspondingly compiled version.

Due to a dependency in the C++ compiler tools that we need, it is currently not possible to offer Evolvix for older MacOSX systems, so we do not support at the moment:

- Mac OSX 10.7 (“Lion”)
- Mac OSX 10.6 (“Snow Leopard”)

5.1.3 Linux: Compatibility notice

This version is currently available in 64bit-release compiled form for

- Ubuntu 14.04 LTS ("Trusty Tahr")

We can get you a RedHat version, but have not tested other distributions. We are interested in making Evolvix compatible with other Linux distributions in the future.

5.2 Troubleshooting

5.2.1 Plotting issues

1. R 3.1.0 version does not produce plots. The update R 3.1.1 does work.

   Observation: You run a quest, it completes fine and the data is there, but the plots do not show up automatically and the PDFs don't open. Instead the command line produces an error message about plot limits after some time.

   Solution: Get the latest version of R. If that does not work, email us.

5.2.2 Parser issues

1. TimeSeries Syntax problems.

   The sequence of statements in the TimeSeries statement is fixed. You can omit statements, but not yet move them around. For now you will need to know the sequence for these conditional filters. Like with other Evolvix code, the best way to write this at the moment is to use some example code that you can copy, paste and modify.

   The parser does not allow comments *inside* of any of the lists that TimeSeries accept (Part Names, values, times). A Quest will currently not compute if adding such comments (or out-commenting).

5.2.3 Simulation issues

1. Unintentional very large simulations are limited to Amounts of 1e20.

   If you specify a model that produces unbounded Amounts of some Part, then it is possible that the simulation will take a very long amount of time and may produce a very large file. This can happen, if you change a parameter in your model so that your model becomes a model with this behavior. A proper solution for this is being developed (allowing you to limit CPU-time etc). Our current short-term fix is in this release to limit the maximal amount of a Part to 1e+20 for all Amounts in the deterministic Simulator (SunDials IDAS).

2. Actions with a very large stoichiometry and large Part amounts break the simulation.

   Reason: the equation used to compute the instantaneous rates / propensities for such functions produces numbers that are too big for the current double precision floating point numbers. This is a hard problem to fix and we are looking into solutions.

   In the mean time, consider breaking down the processes that are modeled by such large stoichiometries into several processes with smaller stoichiometries.

   Maybe you can package many of the small parts into a "chunk" that bundles many of them together and scales this throughout the simulation appropriately. Maybe you can (slowly!) build up these

parts as more elementary parts appear and thus avoid the need to have any action with a huge stoichiometry at any point in time.

5.2.4 Workflow issues

1. Updating the "Most Recent Results" on Windows does work if Folder is open.

   If you have the "Most Recent Results" folder open and then run a simulation then it will not be repopulated with the most recent results. The scripts that move/delete/replace the files are not given the permission to do so, as long as a high level Windows File Explorer view is open.

2. When simulations occur <1 second apart, data is overwritten.

   Reason: the script that manages all files and moves them into the archive uses a timestamp with a 1 sec resolution. A more systematic solution is in the works.

5.2.5 When reporting problems

Please state clearly the specific problem, include the Quest file and quote the following info that specifies which version of Evolvix is installed in this folder:

Evolvix 0.3.0
Release date 2014-09-30

You can report any bugs, issues, problems, questions, ideas, and feature-requests to:

http://evolvix.org/contact/general-feedback

http://evolvix.org/contact/bug-report

5.3 Issues resolved in this release of Evolvix

5.3.1 Issues resolved in this release

1. Resolved: TimeSeries can now use the "Ignore above..." or "Ignore below..." syntax.

   Problem: The TimeSeries "Ignore" clause can lead to early stops. At the moment you cannot yet use the "Ignore above..." or "Ignore below..." clauses in the TimeSeries statement with the deterministic simulator. This combination will lead to an early stop of the whole simulation, whenever any of the parts actually would be crossing that boundary. Internally, this is caused by a function analyzing roots that is not yet implemented properly.

2. Resolved: The parser for TimeSeries can now handle comments inside of the TimeSeries syntax.

   Problem: Do not use comments of any type inside of the TimeSeries statement. A known deficiency in the parser will handle some of these comments in the wrong way. Thus, no convenient commenting out of single lines of the TimeSeries statement.

5.4 Future Changes in Syntax

The following changes to the syntax will be made in future releases. These are not backwards compatible and will therefore require changes to any Evolvix code that uses these features.

The Evolvix long-term stability and backwards compatibility guarantee will start with Version 1.0 and before then any changes are possible, although we try to minimize any actual changes we require.

The list below is an advance notice for your information.

5.4.1 Future Changes After Evolvix 0.3.0

Timeseries

To simplify the syntax of the TimeSeries command and make it more powerful there are some changes we will implement.

5.5 Changes from Past Syntax

The following changes to the syntax from past versions were made. When we date a change as:

```
(Evolvix <0.3.0)
```

we mean that all versions *before* Evolvix Release 0.3.0 have the old syntax and all versions *including* Evolvix Release 0.3.0 and after have the new syntax.

5.5.1 Action Brackets (Evolvix <0.3.0)

Change from "{" and "}" to "(" and ")" to prepare for consistency with a more general upcoming syntax for Evolvix.

Example in Evolvix 0.3.0 and after: Action (a + b —> c)

Example in Evolvix 0.2.0 and before: Action { a + b —> c }

5.5.2 Brackets in General (Evolvix <0.3.0)

All other brackets were also changed from "{" and "}" to "(" and ")" for the same reasons. While this was certainly good for some structures (eg. TimeSeries) it is unclear whether this change will remain for the Evolvix Quest Statement.

Watch this space to see.

5.5.3 Action Arrow Length (Evolvix <0.3.0)

Change from minimally "->" or "-[]->" to requiring 3 dashes minimal to help distinguish it from other future syntax constructs.

Example in Evolvix 0.3.0 and after: Action (a + b —[Rate=1]—> c)

Example in Evolvix 0.2.0 and before: Action { a + b -[Rate=1]-> c }

5.5.4 Comments (Evolvix <0.3.0)

Change

- OneLineComments from "//" to "!L"
- MultiLineComments from "/!!" and "!!/" to "!–" and "–!"
- InfoBlocks are now also part of comments and introduced using !Info

to help make future syntax more consistent.

Example in Evolvix 0.3.0 and after:

```
!l One line out commented Action { a + b -[Rate=1]-> c } !– Multi line out comment –!
```

Example in Evolvix 0.2.0 and before:

```
// One line out commented Action { a + b -[Rate=1]-> c } /!! Multi line out comment !!/
```

5.5.5 Units (Evolvix <0.3.0)

Denoting Units Changed from Example in Evolvix 0.3.0 and after. Reason: Units are Type statements and Types will be consistently designated with ":" in future versions.

Example in Evolvix 0.3.0 and after:

```
:["YourUnit"]
```

Example in Evolvix 0.2.0 and before:

```
["YourUnit"]
```

5.5.6 TimeSeries (Evolvix <0.1.6)

Consolidation: Only use "Report" in TimeSeries

Before Release 0.1.6 of Evolvix, TimeSeries had a mix of "Report" and "Record" key-words in the TimeSeries syntax. If you run old Evolvix code (before release 0.1.6) and you get syntax errors in TimeSeries make sure you change:

"Record" ==> "Report"

The new syntax is easier to remember (more consistent).

5.6 Organization of Evolvix Home Folder

This is for the CommandLine version only.

5.6.1 Description of Files and Folders

When downloaded, the Evolvix installation package is a zipped copy of the standard **Evolvix Home Folder** for the selected operating system and release, only with all Quest Results removed. The Evolvix Home Folder Content Table below describes all the files that should be there.

5.6.2 List of Files that can be changed by the User

All files that merit any type of modification from any user are denoted as such in the Evolvix Home Folder Content Table below, along with the expertise needed to do so.

5.6.3 Evolvix Home Folder Content Relevant for Beginners

File/Folder in Evolvix Home Folder	Description of what it does	Changeable by user?
Evolvix	Binary executable that starts processing the Quest you specify at the command line.	No
help.html	Entry page for all documentation. Open on command-line using `open` (Mac) or `start` (Win) commands.	No
online-documentation-html	All documentation is in this folder: quickstart, tutorials, manuals, known bugs, examples, license info, ...	No, but it is all searchable.
Default_Quest.Evolvix.txt	This file will be called if no other Quest is specified when calling Evolvix. Recommendation: Copy code from here to jumpstart your own Quest writing.	Yes. Or rather: Write your own.
Results/	All results of a Quest are stored here. Quests are timestamped.	Deletable.
Results/ Most_Recent/	Results of the latest Quest run are stored here. Overwritten by next Quest	Deletable.
Results/ Temporary/	Results archive of Quest not yet deleted. Each Quest has its own timestamped folder.	Deletable.
Results/.../PDFs	PDF plots of results from this Quest.	Deletable.
Results/.../Quest	Original Quest that was run and some transformations. Allows reproducing runs using the corresponding "Worker".	Deletable.
Results/.../Raw_Time_Series	Raw data of results from this Quest.	Deletable.
Quest.Plot.Configuration.R	Specify axes and title etc of overview plots produced by R. Self descriptive. This script is executed by R. R will stop, if there are syntax errors in here.	Yes, easy. Just don't mess up the R Syntax.

5.6.4 Evolvix Home Folder Content Relevant for Programmers

File/Folder in Evolvix Home Folder	Description of what it does	Changeable by user?
Scripts/	Any non-compiled scripts needed for Evolvix to execute normally	Usually, No. Experts: Yes.
Scripts/ Plot_Time_Series_Data.R	After computing all raw data, this script is called to automatically produce plots.	Yes for good R programmer
Scripts/ Run_and_Plot.sh	This script is called by the Evolvix Coordinator to organize files on disk, start Evolvix Workers and call R.	Experts only Know Unix Know Evolvix
Workers/ Worker_DAE_IDAS_Dense Worker_SSA_SDM	Compiled executables that do the work. Each type of Task needs its executable. They expect the "QuestName.epb" file in their folder to run. You can specify an ".epb" file and even change the model by giving the Name of the Part you want to change and its new initial value. Call the workers with -h for help on the command-line options for Workers: Worker_SSA_SDM -h Deterministic Simulator Stochastic Simulator	No. But you can easily re-run Quests and even change parameters.

5.6.5 Updates

For more models and information on how to use Evolvix, please refer to:

http://evolvix.org

By the time you read this, updates may have been published there about how to use this release of Evolvix. Example models are also being published there as they become available.

CHAPTER

SIX

LEGAL

6.1 Evolvix End User Licenses

The legal texts below shall make sure that you understand that you are using this software entirely at on your own risk. Under no circumstances can the creators of this software be held accountable for any problems that arise out of your use of this software.

The source code will be made available at a later time.

6.1.1 Evolvix Binary Distribution

Licensing of this software and associated files is governed by the Evolvix Contributors License Agreement as described at http://evolvix.org/about/legal

The license chosen for this file is the BSD-3-Clause (http://opensource.org/licenses/BSD-3-Clause)

Copyright (c) 2013,2014 Laurence Loewe and team, all rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the names of authors nor the names of their organizations nor the names of other contributors to the project may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT HOLDER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

6.1.2 Boost Software License - Version 1.0 - August 17th, 2003

Permission is hereby granted, free of charge, to any person or organization obtaining a copy of the software and accompanying documentation covered by this license (the "Software") to use, reproduce, display, distribute, execute, and transmit the Software, and to prepare derivative works of the Software, and to permit third-parties to whom the Software is furnished to do so, all subject to the following:

The copyright notices in the Software and this entire statement, including the above license grant, this restriction and the following disclaimer, must be included in all copies of the Software, in whole or in part, and all derivative works of the Software, unless such copies or derivative works are solely in the form of machine-executable object code generated by a source language processor.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE, TITLE AND NON-INFRINGEMENT. IN NO EVENT SHALL THE COPYRIGHT HOLDERS OR ANYONE DISTRIBUTING THE SOFTWARE BE LIABLE FOR ANY DAMAGES OR OTHER LIABILITY, WHETHER IN CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

6.1.3 ANTLR 3 License

[The BSD License] Copyright (c) 2010 Terence Parr All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.

Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution. Neither the name of the author nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

6.1.4 muParser license

Fast math parser Library Copyright (C) 2011 Ingo Berg Web: muparser.beltoforion.de e-mail: muparser@beltoforion.de[1]

> Permission is hereby granted, free of charge, to any person obtaining a copy of this software and associated documentation files (the "Software"), to deal in the Software without restriction, including without limitation the rights to use, copy, modify, merge, publish, distribute, sublicense,

[1] muparser@beltoforion.de

and/or sell copies of the Software, and to permit persons to whom the Software is furnished to do so, subject to the following conditions:

The above copyright notice and this permission notice shall be included in all copies or substantial portions of the Software.

THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND, EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE. OR OTHER DEALINGS IN THE SOFTWARE.

6.1.5 Google Protobuf license

Copyright 2008, Google Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.
- Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Code generated by the Protocol Buffer compiler is owned by the owner of the input file used when generating it. This code is not standalone and requires a support library to be linked with it. This support library is itself covered by the above license.

6.1.6 Google gmock license

Copyright 2008, Google Inc. All rights reserved.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

- Redistributions of source code must retain the above copyright notice, this list of conditions and the following disclaimer.
- Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the following disclaimer in the documentation and/or other materials provided with the distribution.

- Neither the name of Google Inc. nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE COPYRIGHT OWNER OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

6.1.7 Sundials license

Copyright (c) 2002, The Regents of the University of California. Produced at the Lawrence Livermore National Laboratory. Written by S.D. Cohen, A.C. Hindmarsh, R. Serban, D. Shumaker, and A.G. Taylor.

UCRL-CODE-155951 (CVODE) UCRL-CODE-155950 (CVODES) UCRL-CODE-155952 (IDA) UCRL-CODE-237203 (IDAS) UCRL-CODE-155953 (KINSOL) All rights reserved.

This file is part of SUNDIALS.

Redistribution and use in source and binary forms, with or without modification, are permitted provided that the following conditions are met:

1. Redistributions of source code must retain the above copyright notice, this list of conditions and the disclaimer below.

2. Redistributions in binary form must reproduce the above copyright notice, this list of conditions and the disclaimer (as noted below) in the documentation and/or other materials provided with the distribution.

3. Neither the name of the UC/LLNL nor the names of its contributors may be used to endorse or promote products derived from this software without specific prior written permission.

THIS SOFTWARE IS PROVIDED BY THE COPYRIGHT HOLDERS AND CONTRIBUTORS "AS IS" AND ANY EXPRESS OR IMPLIED WARRANTIES, INCLUDING, BUT NOT LIMITED TO, THE IMPLIED WARRANTIES OF MERCHANTABILITY AND FITNESS FOR A PARTICULAR PURPOSE ARE DISCLAIMED. IN NO EVENT SHALL THE REGENTS OF THE UNIVERSITY OF CALIFORNIA, THE U.S. DEPARTMENT OF ENERGY OR CONTRIBUTORS BE LIABLE FOR ANY DIRECT, INDIRECT, INCIDENTAL, SPECIAL, EXEMPLARY, OR CONSEQUENTIAL DAMAGES (INCLUDING, BUT NOT LIMITED TO, PROCUREMENT OF SUBSTITUTE GOODS OR SERVICES; LOSS OF USE, DATA, OR PROFITS; OR BUSINESS INTERRUPTION) HOWEVER CAUSED AND ON ANY THEORY OF LIABILITY, WHETHER IN CONTRACT, STRICT LIABILITY, OR TORT (INCLUDING NEGLIGENCE OR OTHERWISE) ARISING IN ANY WAY OUT OF THE USE OF THIS SOFTWARE, EVEN IF ADVISED OF THE POSSIBILITY OF SUCH DAMAGE.

Additional BSD Notice

1. This notice is required to be provided under our contract with the U.S. Department of Energy (DOE). This work was produced at the University of California, Lawrence Livermore National Laboratory under Contract No. W-7405-ENG-48 with the DOE.

2. Neither the United States Government nor the University of California nor any of their employees, makes any warranty, express or implied, or assumes any liability or responsibility for the accuracy, completeness,

or usefulness of any information, apparatus, product, or process disclosed, or represents that its use would not infringe privately-owned rights.

3. Also, reference herein to any specific commercial products, process, or services by trade name, trademark, manufacturer or otherwise does not necessarily constitute or imply its endorsement, recommendation, or favoring by the United States Government or the University of California. The views and opinions of authors expressed herein do not necessarily state or reflect those of the United States Government or the University of California, and shall not be used for advertising or product endorsement purposes.

CHAPTER

SEVEN

SEARCH

- *search*