# Thoughts on simulation project management

Andrew Davison
UNIC, CNRS

FACETS CodeJam #2
Gif sur Yvette, 5th-8th May 2008

# Outline

# Reproducible research

- "I thought I used the same parameters but I'm getting different results"
- I can't remember which version of the code I used to generate figure 6
- Ted Carnevale wants to put the code for that model I published 3 years ago into ModelDB but he can't reproduce the figures
- Why did I do that?

# Drowning in data

```
$ tree bfstdp_data | tail -1
33 directories, 7018 files
```

- photo of lab notebook?
- photo of big stack of printouts? laid out on floor?
- physically take a file of printouts from my thesis in a honking big binder

# Outline

# Solutions?

## Stage 1

- Version control by filename, parameter values in filenames
- Lab notebook with printouts stuck in

## Stage 2

- Parameters, model-definition code, model control code in separate files
- Excel spreadsheet to record parameters used, reasons for doing each simulation, summary of results of each simulations

## Stage 3

- Eureka! Version control (versioning of entire tree, not just individual files)
- Keep using spreadsheet, now record svn revision for each simulation

## But still...

- A lot of manual work, easy to forget to check-in changes

# Solutions?

## Stage 1

- Version control by filename, parameter values in filenames
- Lab notebook with printouts stuck in

## Stage 2

- Parameters, model-definition code, model control code in separate files
- Excel spreadsheet to record parameters used, reasons for doing each simulation, summary of results of each simulations

## Stage 3

- Eureka! Version control (versioning of entire tree, not just individual files)
- Keep using spreadsheet, now record svn revision for each simulation

## But still...

- A lot of manual work, easy to forget to check-in changes

# Solutions?

Stage 1

- Version control by filename, parameter values in filenames
- Lab notebook with printouts stuck in

Stage 2

- Parameters, model-definition code, model control code in separate files
- Excel spreadsheet to record parameters used, reasons for doing each simulation, summary of results of each simulations

Stage 3

- Eureka! Version control (versioning of entire tree, not just individual files)
- Keep using spreadsheet, now record svn revision for each simulation

But still...

- A lot of manual work, easy to forget to check-in changes

# Solutions?

## Stage 1

- Version control by filename, parameter values in filenames
- Lab notebook with printouts stuck in

## Stage 2

- Parameters, model-definition code, model control code in separate files
- Excel spreadsheet to record parameters used, reasons for doing each simulation, summary of results of each simulations

## Stage 3

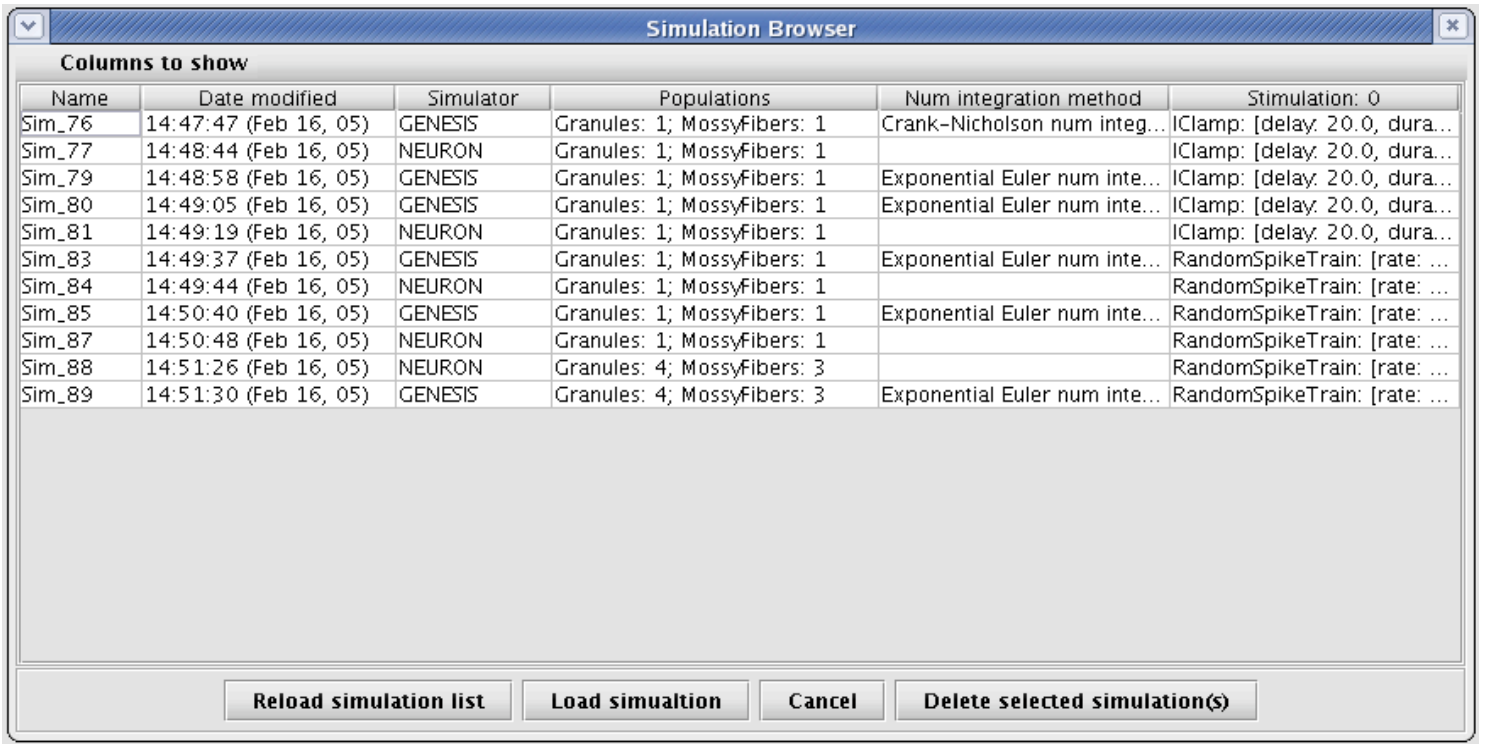
- Eureka! Version control (versioning of entire tree, not just individual files)
- Keep using spreadsheet, now record svn revision for each simulation

## But still...

- A lot of manual work, easy to forget to check-in changes

# Existing tools

- Project management tools in NEURON: RCS control of simulation projects in a single directory using hoc commands `ivdialog`, `prjnrncmp`, `prjnrninit`, `prjnrnci`, `prjnrnco`, `prjnrnpr`
- NeuroConstruct Simulation Browser




| Name | Date modified | Simulator | Populations | Num integration method | Stimulation: 0 |
|---|---|---|---|---|---|
| Sim_76 | 14:47:47 (Feb 16, 05) | GENESIS | Granules: 1; MossyFibers: 1 | Crank-Nicholson num integ... | IClamp: [delay: 20.0, dura... |
| Sim_77 | 14:48:44 (Feb 16, 05) | NEURON | Granules: 1; MossyFibers: 1 | | IClamp: [delay: 20.0, dura... |
| Sim_79 | 14:48:58 (Feb 16, 05) | GENESIS | Granules: 1; MossyFibers: 1 | Exponential Euler num inte... | IClamp: [delay: 20.0, dura... |
| Sim_80 | 14:49:05 (Feb 16, 05) | GENESIS | Granules: 1; MossyFibers: 1 | Exponential Euler num inte... | IClamp: [delay: 20.0, dura... |
| Sim_81 | 14:49:19 (Feb 16, 05) | NEURON | Granules: 1; MossyFibers: 1 | | IClamp: [delay: 20.0, dura... |
| Sim_83 | 14:49:37 (Feb 16, 05) | GENESIS | Granules: 1; MossyFibers: 1 | Exponential Euler num inte... | RandomSpikeTrain: [rate: ... |
| Sim_84 | 14:49:44 (Feb 16, 05) | NEURON | Granules: 1; MossyFibers: 1 | | RandomSpikeTrain: [rate: ... |
| Sim_85 | 14:50:40 (Feb 16, 05) | GENESIS | Granules: 1; MossyFibers: 1 | Exponential Euler num inte... | RandomSpikeTrain: [rate: ... |
| Sim_87 | 14:50:48 (Feb 16, 05) | NEURON | Granules: 1; MossyFibers: 1 | | RandomSpikeTrain: [rate: ... |
| Sim_88 | 14:51:26 (Feb 16, 05) | NEURON | Granules: 4; MossyFibers: 3 | | RandomSpikeTrain: [rate: ... |
| Sim_89 | 14:51:30 (Feb 16, 05) | GENESIS | Granules: 4; MossyFibers: 3 | Exponential Euler num inte... | RandomSpikeTrain: [rate: ... |

# Outline

# Automating record-keeping

Core functionality:

- Make it easy to record code versions, parameter sets, datafiles. Automate as much as possible, prompt me for the rest
- Make it easy to review the history of the project
- Make it **very** easy to repeat a previous simulation **and** check the results haven't changed
- Make it easy to run distributed simulations
- Make it easy to run batch simulations (e.g. repeat n times with different random seeds, systematic stepping through $n$-D parameter space)
- Support any command-line driven simulator/arbitrary executable

# Automating record keeping

Desirable, but non-core functionality:

- Help me manage output datafiles, easily preview file contents, visualise as graphs, archive, compare between simulations
- Analysis workflow management...
- More difficult - interactive sessions, GUI sessions

# Outline

# Sumatra

a command-line tool for simulation management/record-keeping

- Written in Python (big surprise)
- Supports any simulator that allows simulations to be run from the command-line*, although offers extra support for NEURON (e.g. finds the executables automatically, will ensure .mod files are recompiled if the code has changed)*
- Requirements:
  - pysvn
  - sqlite
  - django
- Still alpha software, but I use it and anyone is welcome to try it (GSL licence?).

# smt help

```
$ smt help
Usage: smt <subcommand> [options] [args]
Simulation management tool, version 0.1

Available subcommands:
  run
  batch
  setup
  info
  list
  comment
  repeat
  package
  delete
  runserver
  debug
```

# smt setup

```
smt setup [options] NAME REPOS MAINFILE

  NAME is the project name.

  REPOS is the URL of a Subversion repository with the path
        of the project.

  MAINFILE is the name of the simulator script that would be
           supplied on the command line if running the
           simulator normally, e.g. init.hoc.

Options:
  -d [--datapath] PATH : set the path to the directory in which
                         smt will search for datafiles generated
                         by the simulation. Defaults to ./Data
  -s [--simpath] PATH  : set the path to the simulator
                         executable. If this is not set, smt
                         will assume the simulator is NEURON,
                         and will search for the executables.
```

# smt setup

```
$ smt setup Test1 https://svn.example.com/repos/myproject smttest.hoc
Creating table simulation_database_booleanparameter
Creating table simulation_database_simrecord
Creating table simulation_database_floatparameter
Creating table simulation_database_stringparameter
Creating table simulation_database_integerparameter
Creating table simulation_database_listparameter
Creating table simulation_database_tag
Creating table simulation_database_parametergroup
Installing index for simulation_database.BooleanParameter model
Installing index for simulation_database.FloatParameter model
Installing index for simulation_database.StringParameter model
Installing index for simulation_database.IntegerParameter model
Installing index for simulation_database.ListParameter model
Installing index for simulation_database.ParameterGroup model
Simulation project successfully set up
```

# smt info

```
$ smt info
Name:         Test1
Repository:   https://svn.example.com/repos/myproject
Data root:    Data
Main file:    smttest.hoc
Simulator:    /usr/local/nrn6.1/i686/bin/nrniv
```

# smt run

```
 $ smt run smttest1.param
Label:       smttest1.param
Time stamp:  20080502-155932
Subversion: No version number provided. Using working copy (revision 136)
Writing simulation parameters to  smttest1.param_20080502-155932.param
Command:     i686/special smttest1.param_20080502-155932.param smttest.hoc
loading membrane mechanisms from /home/andrew/tmp/smt_test/i686/.libs/
libnrnmech.so
>>> Created cell
>>> Inserted mechanisms
>>> Inserted electrode
>>> Set parameters
        1
>>> Running...
        1
        0
Archiving data to file /home/andrew/tmp/smt_test/
smttest1.param_20080502-155932.tar.gz
Data [] ['smttest1.param_20080502-155759.log',
'smttest1.param_20080502-155911.log', 'smttest1.param_20080502-155932.log']
Deleting ['Data/smttest1.param_20080502-155932.log']
```

# smt run

```
$ smt run smttest1.param i_stim=100.0
```

```
$ smt run --label=Figure3 --reason='Test for CodeJam' smttest1.param
```

```
$ smt run smttest1.param
Label:       smttest1.param
Time stamp:  20080502-161150
There are local changes to the simulation code.
Do you want to commit them (y/n)? [default='y']:
Please enter a log message: Fixed bug

$ smt run --version=136 smttest1.param
Label:       smttest1.param
Time stamp:  20080502-161508
Subversion: Version requested is not the same as the working copy.
            Checked out code version 136
```

# smt comment

```
$ smt comment 'Wow! Nature here we come!'

$ smt comment Figure3_20080502-160909 'Veni, vidi, vici'
```

# smt list

```
$ smt list
smttest1.param_20080502-155932
smttest1.param_20080502-160650
Figure3_20080502-160909
smttest1.param_20080502-161150
smttest1.param_20080502-161508

$ smt list Figure3
Figure3_20080502-160909
```

# smt list

```
$ smt list --mode=long Figure3
------------------------------------------------------------
Id           : Figure3_20080502-160909
Reason       : Test for CodeJam
Label        : Figure3
Time_Taken   : 0.0612869262695
Code_Version : 136
Sim_Version  : {'date': '2007-11-24', 'version': '6.1.1',
                'revision': '1894'}
Outcome      : Veni, vidi, vici
Timestamp    : 2008-05-02 16:09:09.795710
```

# smt delete

```
$ smt delete smttest1.param_20080502-155932
1 record deleted
```

# smt batch


DON'T WORRY, IT'S GONNA BE COOL... HOMER'S ON THE JOB, MAN!
UNDER CONSTRUCTION

# smt runserver

## List of simulations

| Delete | ID | Label | Time Taken | Code Version | Sim Version | Reason | Outcome | Timestamp | Tags | CM | celsius | dt | i_stim | infile | tstop | v_init |
|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|---|
| ☐ | smttest1.param_20080502-155932 | smttest1.param | 0.0609200000763 | 136 | {'date': '2007-11-24', 'version': '6.1.1', 'revision': '1894'} | | | 2008-05-02 15:59:32.117383 | | 1.0 | 5 | 0.1 | 70.0 | test.dat | 30 | -67 |
| ☐ | smttest1.param_20080502-160650 | smttest1.param | 0.0615830421448 | 136 | {'date': '2007-11-24', 'version': '6.1.1', 'revision': '1894'} | | | 2008-05-02 16:06:50.298532 | | 1.0 | 5 | 0.1 | 100.0 | test.dat | 30 | -67 |
| ☐ | Figure3_20080502-160909 | Figure3 | 0.0612869262695 | 136 | {'date': '2007-11-24', 'version': '6.1.1', 'revision': '1894'} | Test for CodeJam | Veni, vidi, vici | 2008-05-02 16:09:09.795710 | | 1.0 | 5 | 0.1 | 70.0 | test.dat | 30 | -67 |
| ☐ | smttest.param_20080502-161150 | smttest.param | 0.0608308315277 | 137 | {'date': '2007-11-24', 'version': '6.1.1', 'revision': '1894'} | | | 2008-05-02 16:11:50.174672 | | 1.0 | 5 | 0.1 | 70.0 | test.dat | 30 | -67 |
| ☐ | smttest1.param_20080502-161508 | smttest1.param | 0.0614829063416 | 136 | {'date': '2007-11-24', 'version': '6.1.1', 'revision': '1894'} | | Wow! Nature here we come! | 2008-05-02 16:15:08.928594 | | 1.0 | 5 | 0.1 | 70.0 | test.dat | 30 | -67 |

# Browser interface

# Outline

# Limitations of the current version

- Subversion only
- No GUI. Browser interface is read-only (would be nice to be able to lauch simulations via web interface as well)
- No support for multi-user, distributed projects
- MPI support could be better
- No support for post-simulation data analysis
- Built with my own preferred workflow in mind - I have no idea if other people work in the same or a similar way

# Proposed redesign

- A more modular, loosely-coupled structure...
- ...to give flexibility and support many different workflows
- Support multiple interfaces (command-line, GUI, web)
- Support different version control tools (Subversion, Bazaar, ...)
- Plug-in based analysis workflow
- Support multi-user, distributed projects