What's new with

PyNN? Sumatra?

Andrew Davison

Unité de Neuroscience, Information et Complexité (UNIC)
CNRS, Gif sur Yvette, France

FACETS CodeJam #4, Marseille, 24th June 2010

What the **** is

PyNN?
Sumatra?

PyNN

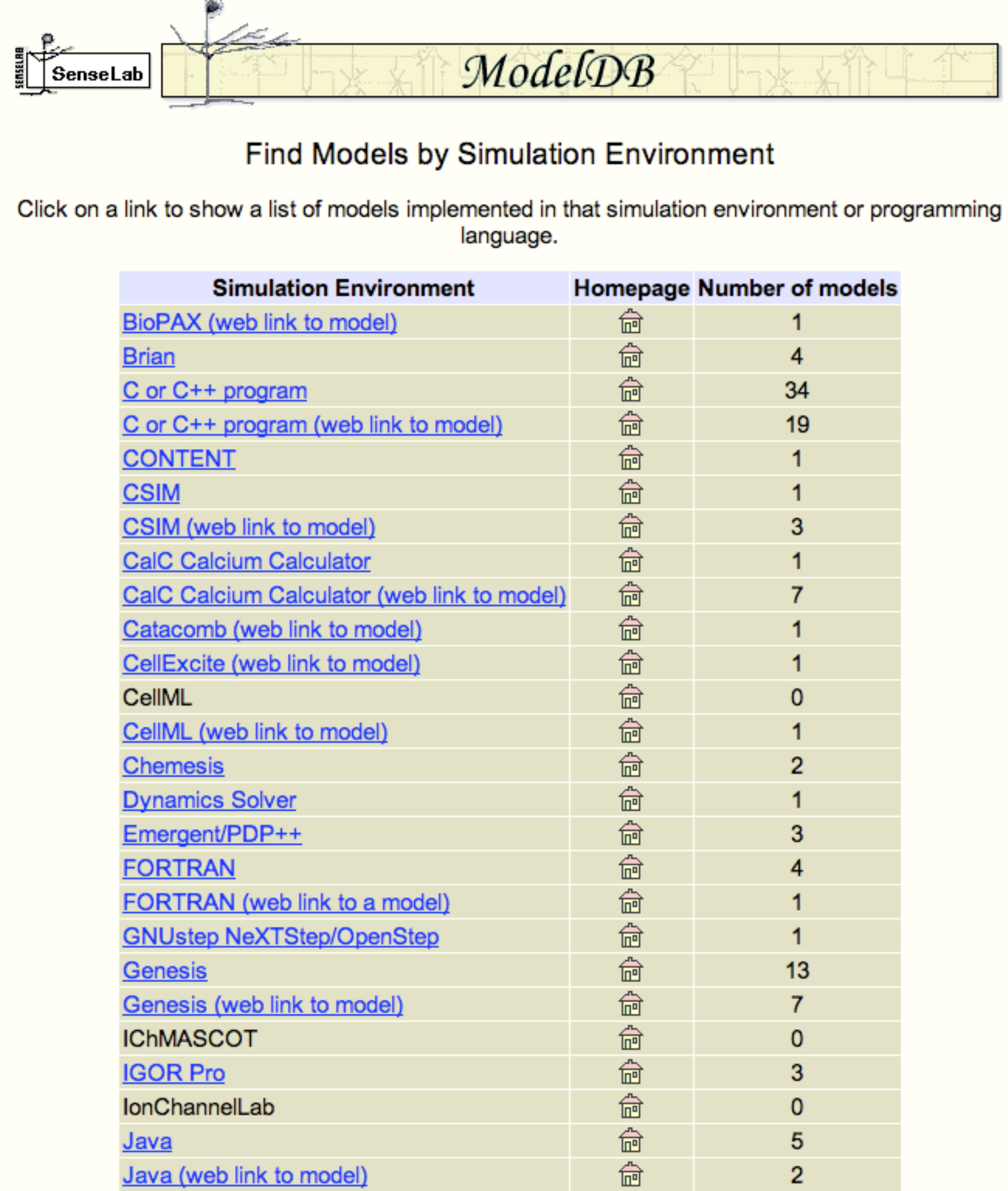
Simulator diversity: problem and opportunity

Cons

- Considerable difficulty in translating models from one simulator to another...
- ...or even in understanding someone else's code.
- This:
  - impedes communication between investigators,
  - makes it harder to reproduce other people's work,
  - makes it harder to build on other people's work.

Pros

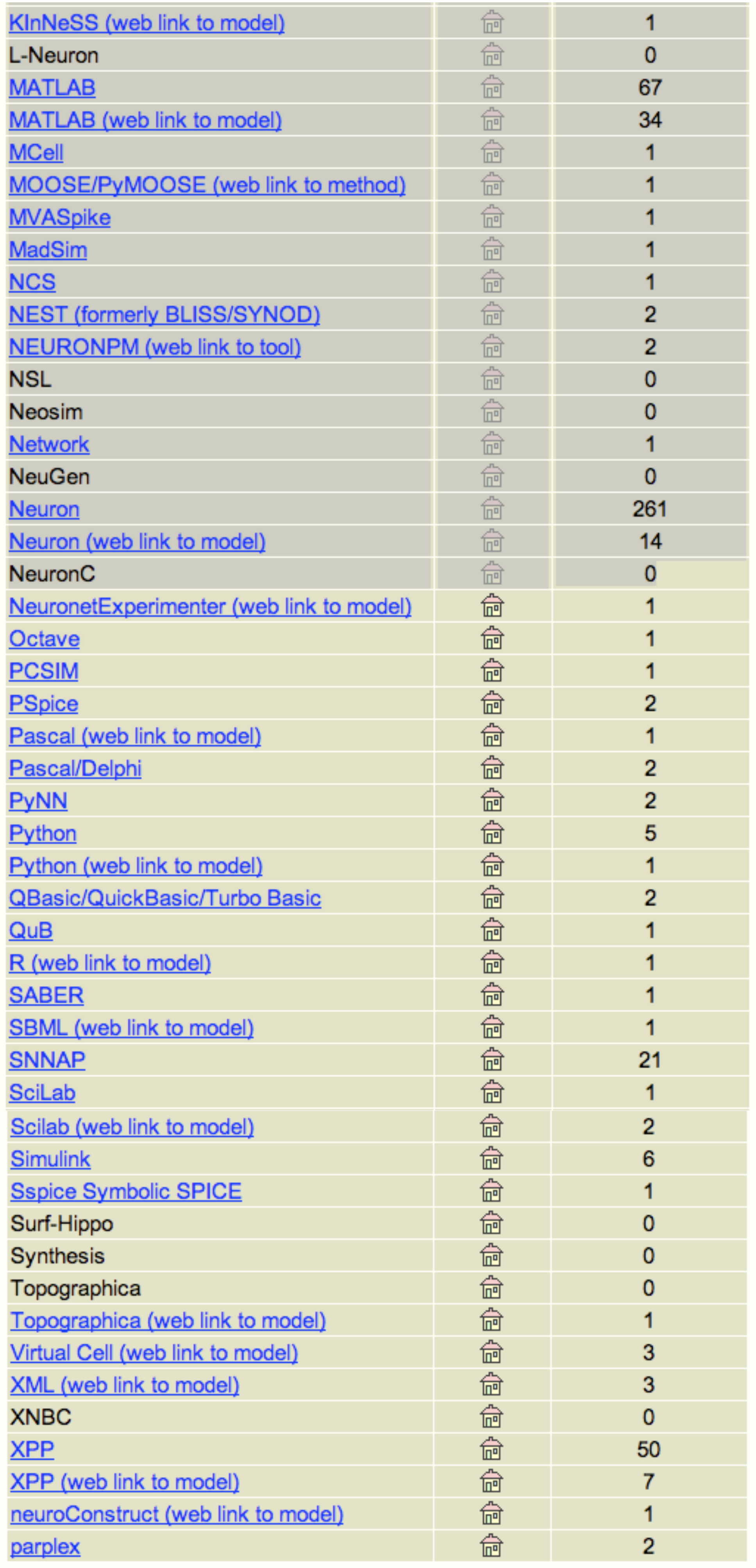
- Each simulator has a different balance between efficiency, flexibility, scalability and user-friendliness → can choose the most appropriate for a given problem.
- Any given simulator is likely to have bugs and hidden assumptions, which will be revealed by cross-checking results between different simulators → greater confidence in correctness of results.



SenseLab ModelDB

Find Models by Simulation Environment

Click on a link to show a list of models implemented in that simulation environment or programming language.



Simulation Environment	Homepage	Number of models
BioPAX (web link to model)	🏠	1
Brian	🏠	4
C or C++ program	🏠	34
C or C++ program (web link to model)	🏠	19
CONTENT	🏠	1
CSIM	🏠	1
CSIM (web link to model)	🏠	3
CalC Calcium Calculator	🏠	1
CalC Calcium Calculator (web link to model)	🏠	7
Catacomb (web link to model)	🏠	1
CellExcite (web link to model)	🏠	1
CellML	🏠	0
CellML (web link to model)	🏠	1
Chemesis	🏠	2
Dynamics Solver	🏠	1
Emergent/PDP++	🏠	3
FORTRAN	🏠	4
FORTRAN (web link to a model)	🏠	1
GNUstep NeXTStep/OpenStep	🏠	1
Genesis	🏠	13
Genesis (web link to model)	🏠	7
IChMASCOT	🏠	0
IGOR Pro	🏠	3
IonChannelLab	🏠	0
Java	🏠	5
Java (web link to model)	🏠	2
KInNeSS (web link to model)	🏠	1
L-Neuron	🏠	0
MATLAB	🏠	67
MATLAB (web link to model)	🏠	34
MCell	🏠	1
MOOSE/PyMOOSE (web link to method)	🏠	1
MVASpike	🏠	1
MadSim	🏠	1
NCS	🏠	1
NEST (formerly BLISS/SYNOD)	🏠	2
NEURONPM (web link to tool)	🏠	2
NSL	🏠	0
Neosim	🏠	0
Network	🏠	1
NeuGen	🏠	0
Neuron	🏠	261
Neuron (web link to model)	🏠	14
NeuronC	🏠	0
NeuronetExperimenter (web link to model)	🏠	1
Octave	🏠	1
PCSIM	🏠	1
PSpice	🏠	2
Pascal (web link to model)	🏠	1
Pascal/Delphi	🏠	2
PyNN	🏠	2
Python	🏠	5
Python (web link to model)	🏠	1
QBasic/QuickBasic/Turbo Basic	🏠	2
QuB	🏠	1
R (web link to model)	🏠	1
SABER	🏠	1
SBML (web link to model)	🏠	1
SNNAP	🏠	21
SciLab	🏠	1
Scilab (web link to model)	🏠	2
Simulink	🏠	6
Sspice Symbolic SPICE	🏠	1
Surf-Hippo	🏠	0
Synthesis	🏠	0
Topographica	🏠	0
Topographica (web link to model)	🏠	1
Virtual Cell (web link to model)	🏠	3
XML (web link to model)	🏠	3
XNBC	🏠	0
XPP	🏠	50
XPP (web link to model)	🏠	7
neuroConstruct (web link to model)	🏠	1
parplex	🏠	2

PyNN

a common API for spiking network simulators

- Goal: write the code for a simulation once, run it on any supported simulator or hardware device *without modification*.
    - ✦ keep the advantages of having multiple simulators or hardware devices
    - ✦ but remove the translation barrier*.

Cawan Cake by Nono Fara http://www.flickr.com/photos/n-o-n-o/3280580620/

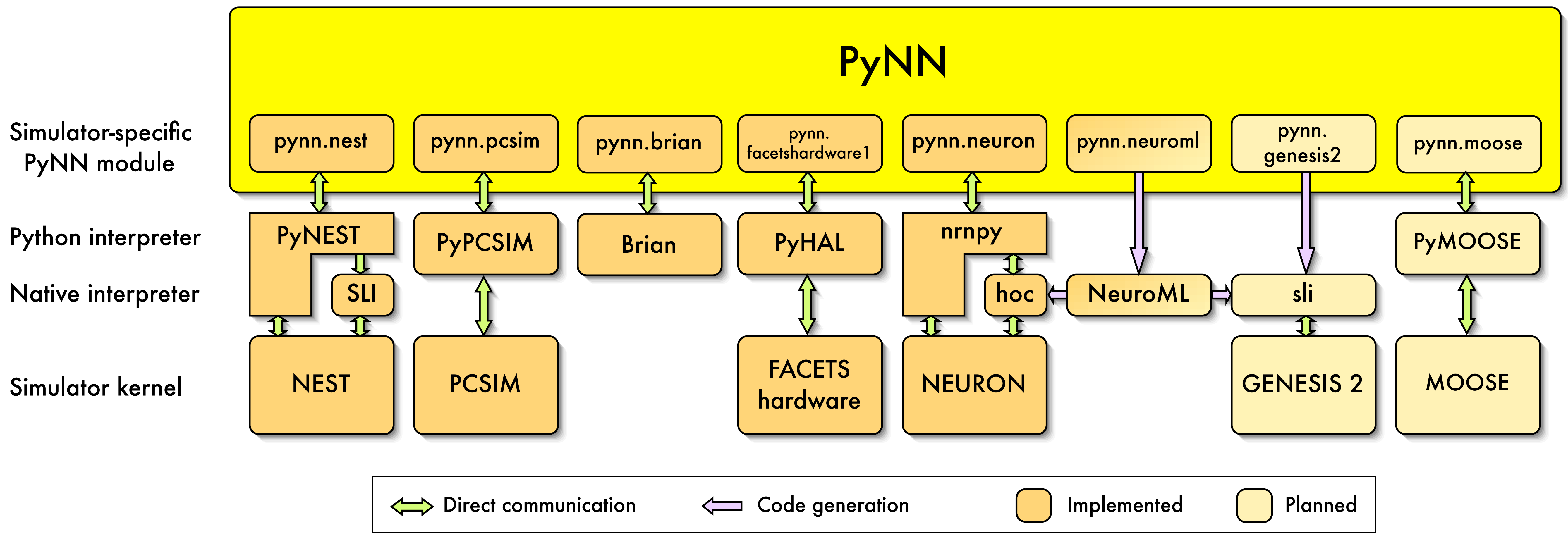
PyNN

a common API for spiking network simulators

- Goal: write the code for a simulation once, run it on any supported simulator or hardware device *without modification*.
  - ✦ keep the advantages of having multiple simulators or hardware devices
  - ✦ but remove the translation barrier*.

*aka: having your cake and eating it
nono farahshila | http://www.nonowrites.com

Cawan Cake by Nono Fara http://www.flickr.com/photos/n-o-n-o/3280580620/


PyNN
Simulator-specific PyNN module
Python interpreter
Native interpreter
Simulator kernel
pynn.nest
pynn.pcsim
pynn.brian
pynn.facetshardware1
pynn.neuron
pynn.neuroml
pynn.genesis2
pynn.moose
PyNEST
SLI
PyPCSIM
Brian
PyHAL
nrnpy
hoc
NeuroML
sli
PyMOOSE
NEST
PCSIM
FACETS hardware
NEURON
GENESIS 2
MOOSE
Direct communication
Code generation
Implemented
Planned

```
sim.setup(timestep=0.1)
cell_parameters = {"tau_m": 12.0, "cm": 0.8, "v_thresh": -50.0,
                   "v_reset": -65.0}
pE = sim.Population((100,100), sim.IF_cond_exp, cell_parameters,
                    label="excitatory neurons")
pI = sim.Population((50,50), sim.IF_cond_exp, cell_parameters,
                    label="inhibitory neurons")
input = sim.Population(100, sim.SpikeSourcePoisson)
rate_distr = random.RandomDistribution("normal", (10.0, 2.0))
input.rset("rate", rate_distr)
background = sim.NoisyCurrentSource(mean=0.1, stdev=0.01)
pE.inject(background)
pI.inject(background)
DDPC = sim.DistanceDependentProbabilityConnector
weight_distr = random.RandomDistribution("uniform", (0.0, 0.1))
connector = DDPC("exp(-d**2/400.0)", weights=weight_distr,
                 delays="0.5+0.01d")
TMM = sim. TsodyksMarkramMechanism
depressing = sim.DynamicSynapse(fast=TMM(U=0.5,tau_rec=800.0))
e2e = sim.Projection(pE, pE, connector, target="excitatory",
                     synapse_dynamics=plasticity)
e2i = sim.Projection(pE, pI, connector, target="excitatory")
i2e = sim.Projection(pI, pE, connector, target="inhibitory")
```

```
import pyNN.neuron as sim
sim.setup(timestep=0.1)
cell_parameters = {”tau_m”: 12.0, ”cm”: 0.8, ”v_thresh”: -50.0,
                   ”v_reset”: -65.0}
pE = sim.Population((100,100), sim.IF_cond_exp, cell_parameters,
                    label=”excitatory neurons”)
pI = sim.Population((50,50), sim.IF_cond_exp, cell_parameters,
                    label=”inhibitory neurons”)
input = sim.Population(100, sim.SpikeSourcePoisson)
rate_distr = random.RandomDistribution(”normal”, (10.0, 2.0))
input.rset(”rate”, rate_distr)
background = sim.NoisyCurrentSource(mean=0.1, stdev=0.01)
pE.inject(background)
pI.inject(background)
DDPC = sim.DistanceDependentProbabilityConnector
weight_distr = random.RandomDistribution(”uniform”, (0.0, 0.1))
connector = DDPC(”exp(-d**2/400.0)”, weights=weight_distr,
                 delays=”0.5+0.01d”)
TMM = sim. TsodyksMarkramMechanism
depressing = sim.DynamicSynapse(fast=TMM(U=0.5,tau_rec=800.0))
e2e = sim.Projection(pE, pE, connector, target=”excitatory”,
                     synapse_dynamics=plasticity)
e2i = sim.Projection(pE, pI, connector, target=”excitatory”)
i2e = sim.Projection(pI, pE, connector, target=”inhibitory”)
```

```
import pyNN.nest as sim
sim.setup(timestep=0.1)
cell_parameters = {"tau_m": 12.0, "cm": 0.8, "v_thresh": -50.0,
                   "v_reset": -65.0}
pE = sim.Population((100,100), sim.IF_cond_exp, cell_parameters,
                    label="excitatory neurons")
pI = sim.Population((50,50), sim.IF_cond_exp, cell_parameters,
                    label="inhibitory neurons")
input = sim.Population(100, sim.SpikeSourcePoisson)
rate_distr = random.RandomDistribution("normal", (10.0, 2.0))
input.rset("rate", rate_distr)
background = sim.NoisyCurrentSource(mean=0.1, stdev=0.01)
pE.inject(background)
pI.inject(background)
DDPC = sim.DistanceDependentProbabilityConnector
weight_distr = random.RandomDistribution("uniform", (0.0, 0.1))
connector = DDPC("exp(-d**2/400.0)", weights=weight_distr,
                 delays="0.5+0.01d")
TMM = sim. TsodyksMarkramMechanism
depressing = sim.DynamicSynapse(fast=TMM(U=0.5,tau_rec=800.0))
e2e = sim.Projection(pE, pE, connector, target="excitatory",
                     synapse_dynamics=plasticity)
e2i = sim.Projection(pE, pI, connector, target="excitatory")
i2e = sim.Projection(pI, pE, connector, target="inhibitory")
```

```
import pyNN.hardware.facets.stage1 as sim
sim.setup(timestep=0.1)
cell_parameters = {"tau_m": 12.0, "cm": 0.8, "v_thresh": -50.0,
                   "v_reset": -65.0}
pE = sim.Population((100,100), sim.IF_cond_exp, cell_parameters,
                    label="excitatory neurons")
pI = sim.Population((50,50), sim.IF_cond_exp, cell_parameters,
                    label="inhibitory neurons")
input = sim.Population(100, sim.SpikeSourcePoisson)
rate_distr = random.RandomDistribution("normal", (10.0, 2.0))
input.rset("rate", rate_distr)
background = sim.NoisyCurrentSource(mean=0.1, stdev=0.01)
pE.inject(background)
pI.inject(background)
DDPC = sim.DistanceDependentProbabilityConnector
weight_distr = random.RandomDistribution("uniform", (0.0, 0.1))
connector = DDPC("exp(-d**2/400.0)", weights=weight_distr,
                 delays="0.5+0.01d")
TMM = sim. TsodyksMarkramMechanism
depressing = sim.DynamicSynapse(fast=TMM(U=0.5,tau_rec=800.0))
e2e = sim.Projection(pE, pE, connector, target="excitatory",
                     synapse_dynamics=plasticity)
e2i = sim.Projection(pE, pI, connector, target="excitatory")
i2e = sim.Projection(pI, pE, connector, target="inhibitory")
```

Since CodeJam #3

0.6

1. **Spikes, membrane potential and synaptic conductances can now be saved to file in various binary formats.** To do this, pass a PyNN File object to Population.print_X(), instead of a filename. There are various types of PyNN File object, defined in the recording.files module, e.g., StandardTextFile, PickleFile, NumpyBinaryFile, HDF5ArrayFile.

2. **Added a reset() function and made the behaviour of setup() consistent across simulators.** reset() sets the simulation time to zero and sets membrane potentials to their initial values, but does not change the network structure. setup() destroys any previously defined network.

3. The possibility of expressing **distance-dependent weights and delays** was extended to the AllToAllConnector and FixedProbabilityConnector classes. To reduce the number of arguments to the constructors, the arguments affecting the spatial topology (periodic boundary conditions, etc.) were moved to a new Space class, so that only a single Space instance need be passed to the Connector constructor.

4. **Assorted speed-up**s

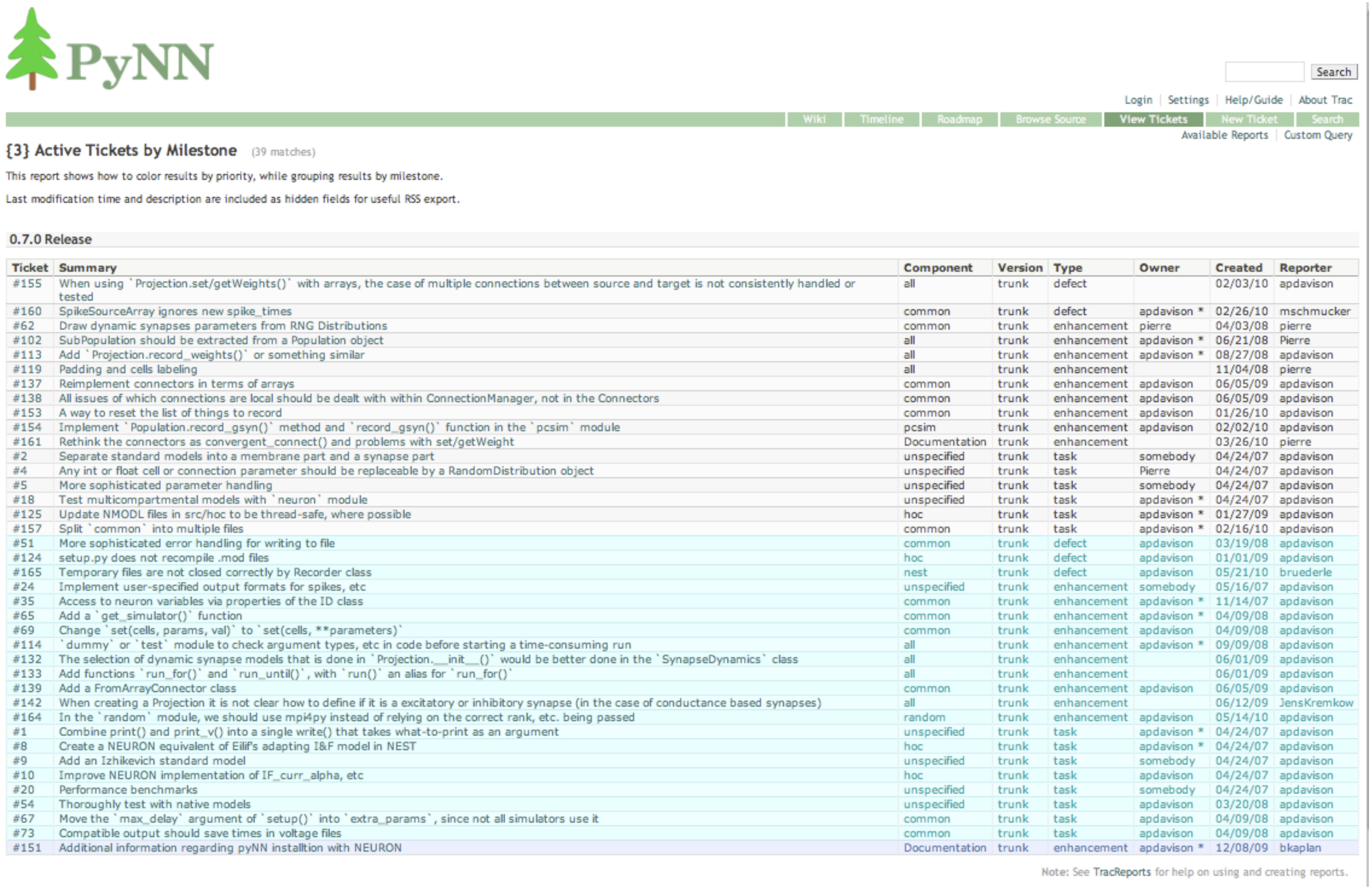
5. **Testing** that results are independent of number of processors added to regression tests.

Since CodeJam #3

trunk

1. internal sub-package reorganisation in preparation for multi-compartmental models
2. connection speed-ups
3. added SmallWorldConnector
4. removed v_init as a parameter, replaced with Population.initialize() method
5. Population structure no longer restricted to a grid
6. began implementing PopulationView (sub-populations) and Assembly (collection of Populations)

http://neuralensemble.org/PyNN


PyNN
Search
Login | Settings | Help/Guide | About Trac
Wiki | Timeline | Roadmap | Browse Source | View Tickets | New Ticket | Search
Available Reports | Custom Query

{3} Active Tickets by Milestone (39 matches)

This report shows how to color results by priority, while grouping results by milestone.

Last modification time and description are included as hidden fields for useful RSS export.

0.7.0 Release

Ticket	Summary	Component	Version	Type	Owner	Created	Reporter
#155	When using `Projection.set/getWeights()` with arrays, the case of multiple connections between source and target is not consistently handled or tested	all	trunk	defect		02/03/10	apdavison
#160	SpikeSourceArray ignores new spike_times	common	trunk	defect	apdavison *	02/26/10	mschmucker
#62	Draw dynamic synapses parameters from RNG Distributions	common	trunk	enhancement	pierre	04/03/08	pierre
#102	SubPopulation should be extracted from a Population object	all	trunk	enhancement	apdavison *	06/21/08	Pierre
#113	Add `Projection.record_weights()` or something similar	all	trunk	enhancement	apdavison *	08/27/08	apdavison
#119	Padding and cells labeling	all	trunk	enhancement		11/04/08	pierre
#137	Reimplement connectors in terms of arrays	common	trunk	enhancement	apdavison	06/05/09	apdavison
#138	All issues of which connections are local should be dealt with within ConnectionManager, not in the Connectors	common	trunk	enhancement	apdavison	06/05/09	apdavison
#153	A way to reset the list of things to record	common	trunk	enhancement	apdavison	01/26/10	apdavison
#154	Implement `Population.record_gsyn()` method and `record_gsyn()` function in the `pcsim` module	pcsim	trunk	enhancement	apdavison	02/02/10	apdavison
#161	Rethink the connectors as convergent_connect() and problems with set/getWeight	Documentation	trunk	enhancement		03/26/10	pierre
#2	Separate standard models into a membrane part and a synapse part	unspecified	trunk	task	somebody	04/24/07	apdavison
#4	Any int or float cell or connection parameter should be replaceable by a RandomDistribution object	unspecified	trunk	task	Pierre	04/24/07	apdavison
#5	More sophisticated parameter handling	unspecified	trunk	task	somebody	04/24/07	apdavison
#18	Test multicompartmental models with `neuron` module	unspecified	trunk	task	apdavison *	04/24/07	apdavison
#125	Update NMODL files in src/hoc to be thread-safe, where possible	hoc	trunk	task	apdavison *	01/27/09	apdavison
#157	Split `common` into multiple files	common	trunk	task	apdavison *	02/16/10	apdavison
#51	More sophisticated error handling for writing to file	common	trunk	defect	apdavison	03/19/08	apdavison
#124	setup.py does not recompile .mod files	hoc	trunk	defect	apdavison	01/01/09	apdavison
#165	Temporary files are not closed correctly by Recorder class	nest	trunk	defect	apdavison	05/21/10	bruederle
#24	Implement user-specified output formats for spikes, etc	unspecified	trunk	enhancement	somebody	05/16/07	apdavison
#35	Access to neuron variables via properties of the ID class	common	trunk	enhancement	apdavison *	11/14/07	apdavison
#65	Add a `get_simulator()` function	common	trunk	enhancement	apdavison *	04/09/08	apdavison
#69	Change `set(cells, params, val)` to `set(cells, **parameters)`	common	trunk	enhancement	apdavison	04/09/08	apdavison
#114	`dummy` or `test` module to check argument types, etc in code before starting a time-consuming run	all	trunk	enhancement	apdavison *	09/09/08	apdavison
#132	The selection of dynamic synapse models that is done in `Projection.__init__()` would be better done in the `SynapseDynamics` class	all	trunk	enhancement		06/01/09	apdavison
#133	Add functions `run_for()` and `run_until()`, with `run()` an alias for `run_for()`	all	trunk	enhancement		06/01/09	apdavison
#139	Add a FromArrayConnector class	common	trunk	enhancement	apdavison	06/05/09	apdavison
#142	When creating a Projection it is not clear how to define if it is a excitatory or inhibitory synapse (in the case of conductance based synapses)	all	trunk	enhancement		06/12/09	JensKremkow
#164	In the `random` module, we should use mpi4py instead of relying on the correct rank, etc. being passed	random	trunk	enhancement	apdavison	05/14/10	apdavison
#1	Combine print() and print_v() into a single write() that takes what-to-print as an argument	unspecified	trunk	task	apdavison *	04/24/07	apdavison
#8	Create a NEURON equivalent of Eilif's adapting I&F model in NEST	hoc	trunk	task	apdavison *	04/24/07	apdavison
#9	Add an Izhikevich standard model	unspecified	trunk	task	somebody	04/24/07	apdavison
#10	Improve NEURON implementation of IF_curr_alpha, etc	hoc	trunk	task	apdavison	04/24/07	apdavison
#20	Performance benchmarks	unspecified	trunk	task	somebody	04/24/07	apdavison
#54	Thoroughly test with native models	unspecified	trunk	task	apdavison	03/20/08	apdavison
#67	Move the `max_delay` argument of `setup()` into `extra_params`, since not all simulators use it	common	trunk	task	apdavison	04/09/08	apdavison
#73	Compatible output should save times in voltage files	common	trunk	task	apdavison	04/09/08	apdavison
#151	Additional information regarding pyNN installtion with NEURON	Documentation	trunk	enhancement	apdavison *	12/08/09	bkaplan

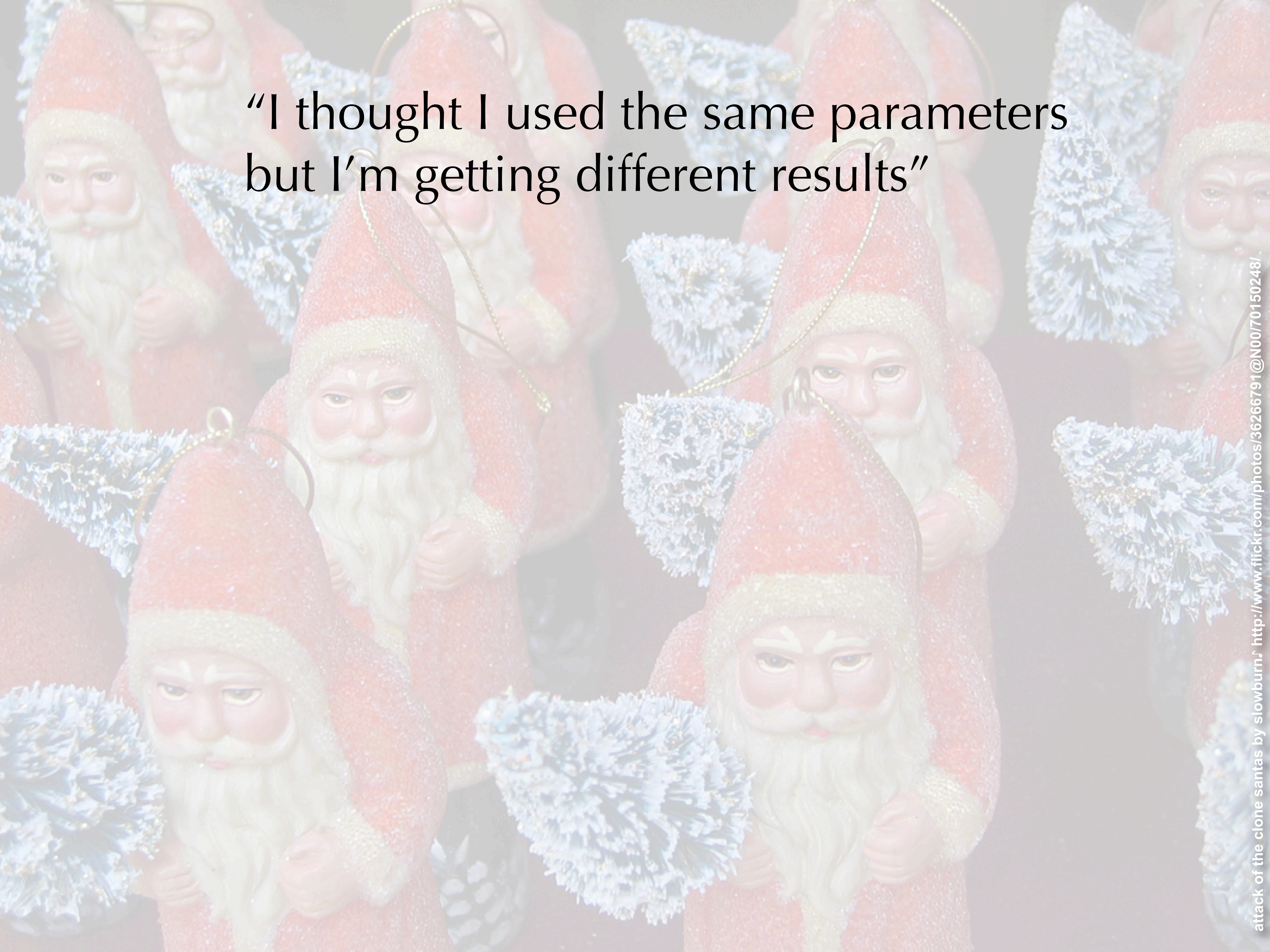
Note: See TracReports for help on using and creating reports.

Sumatra

Rumah Gadang Minangkabau in West Sumatra by CharlesFred http://www.flickr.com/photos/charlesfred/2870828972/

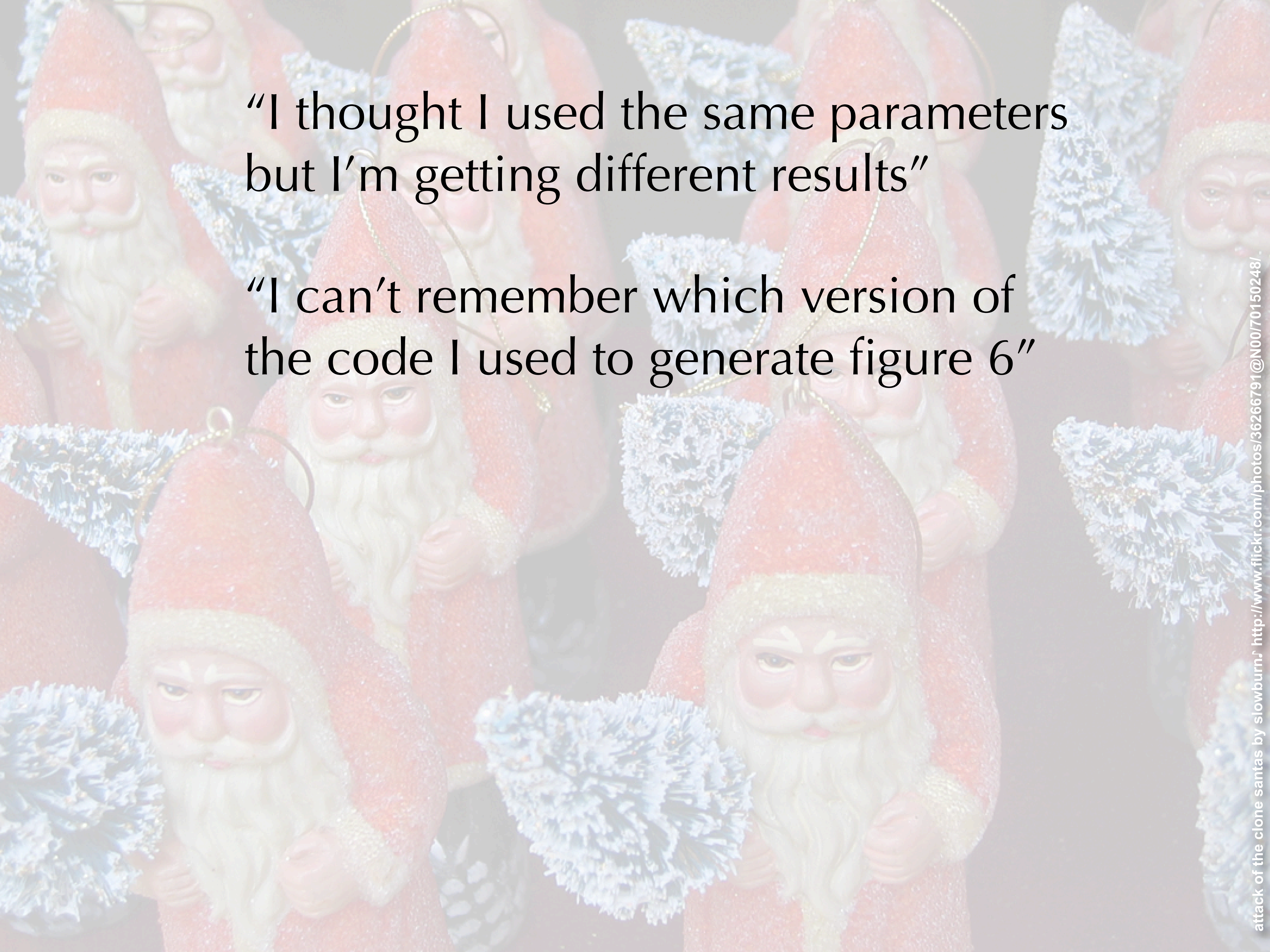
Replicability

attack of the clone santas by slowburn” http://www.flickr.com/photos/36266791@N00/70150248/



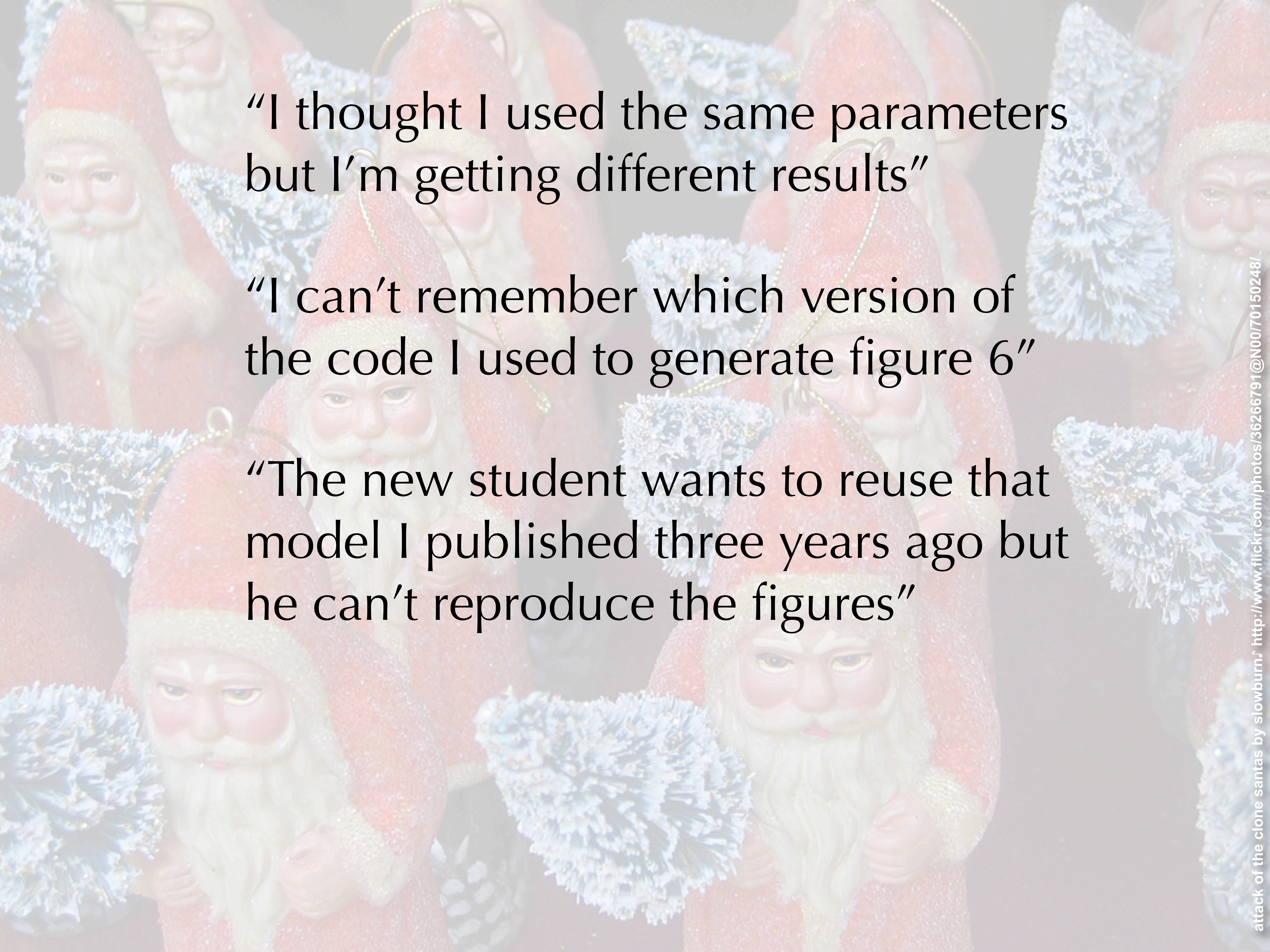
“I thought I used the same parameters but I’m getting different results”

attack of the clone santas by slowburn, http://www.flickr.com/photos/36266791@N00/70150248/



"I thought I used the same parameters but I'm getting different results"

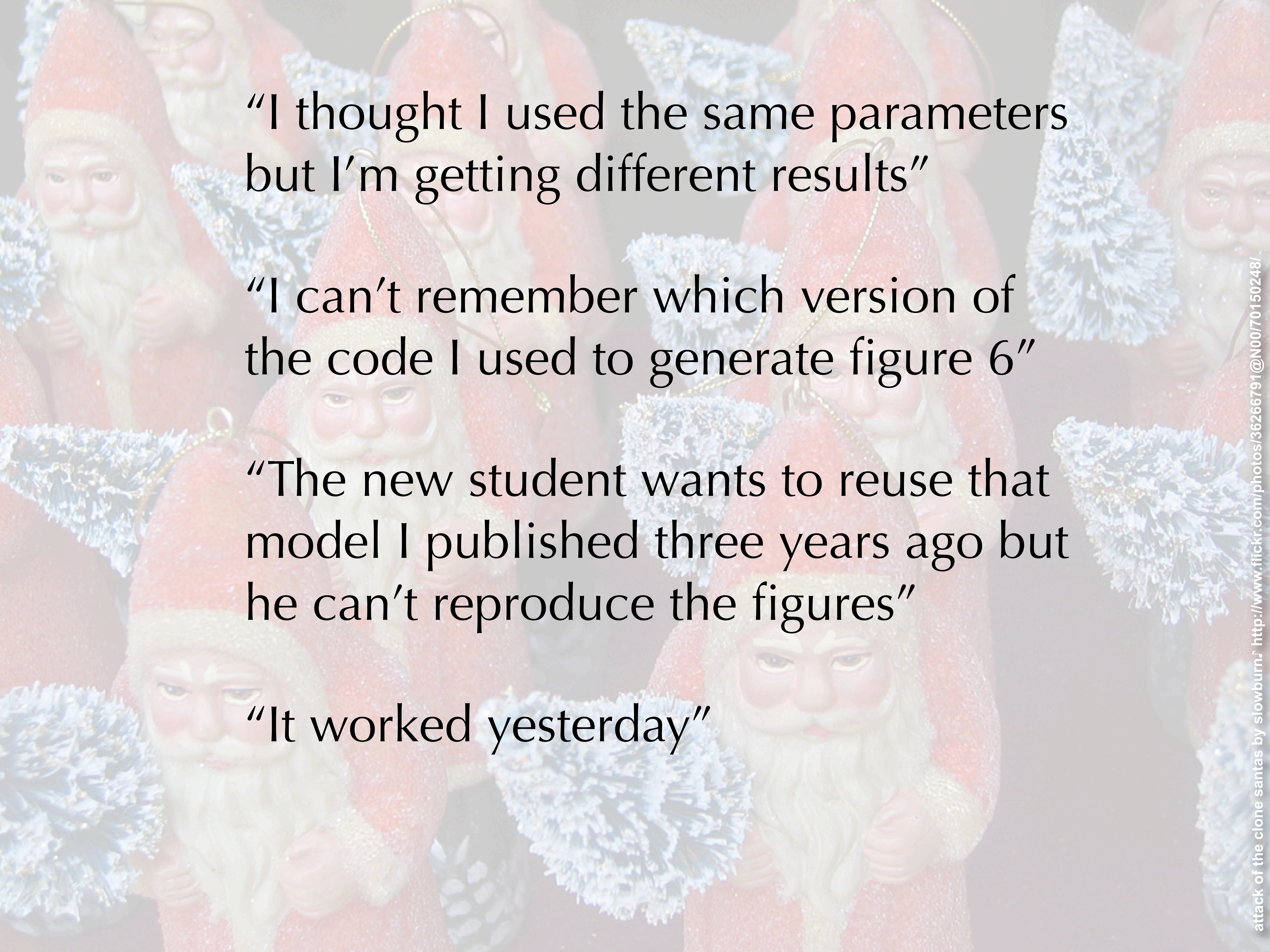
"I can't remember which version of the code I used to generate figure 6"



"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"

"The new student wants to reuse that model I published three years ago but he can't reproduce the figures"

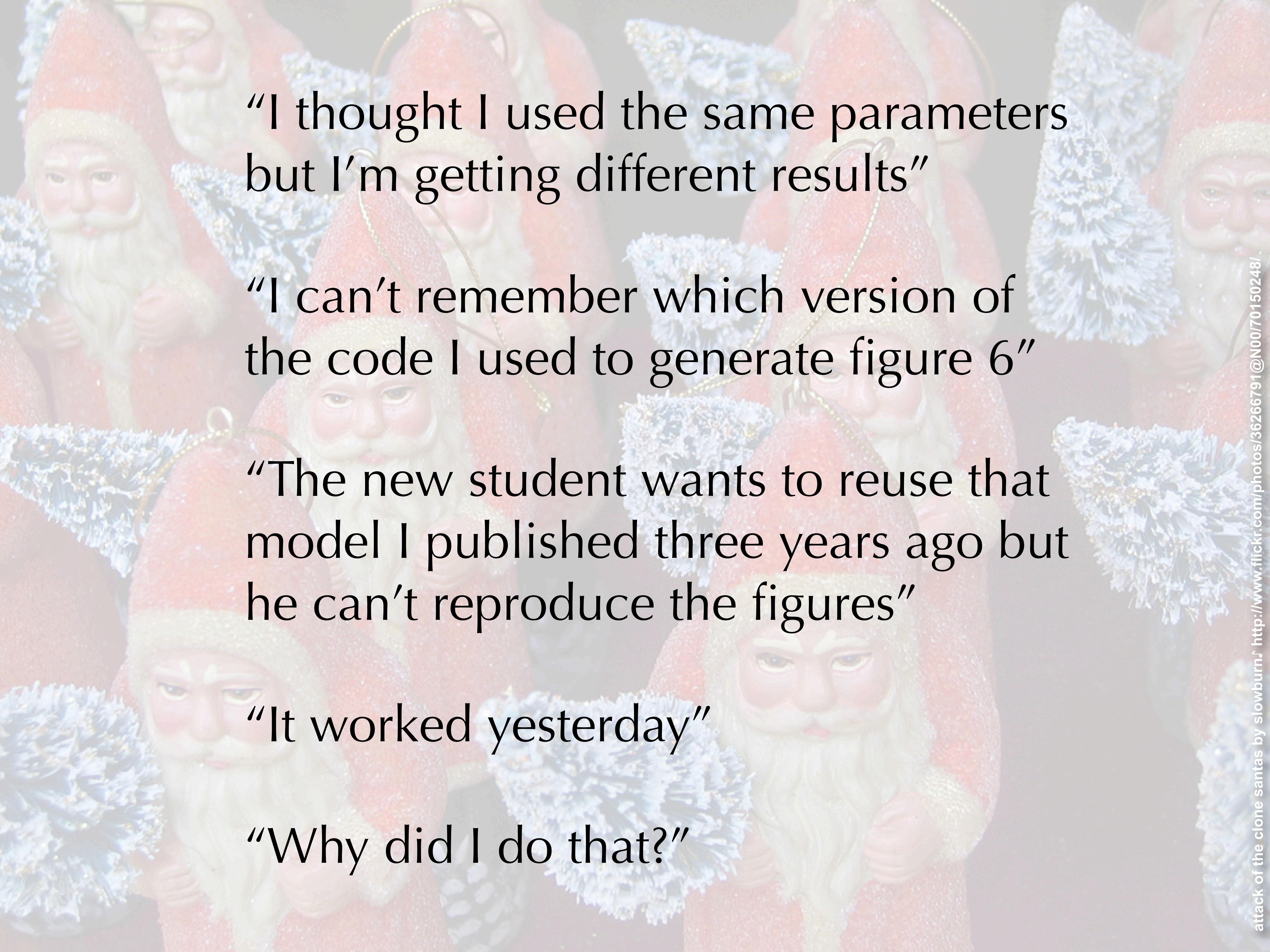


"I thought I used the same parameters but I'm getting different results"

"I can't remember which version of the code I used to generate figure 6"

"The new student wants to reuse that model I published three years ago but he can't reproduce the figures"

"It worked yesterday"



"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"

"The new student wants to reuse that model I published three years ago but he can't reproduce the figures"

"It worked yesterday"

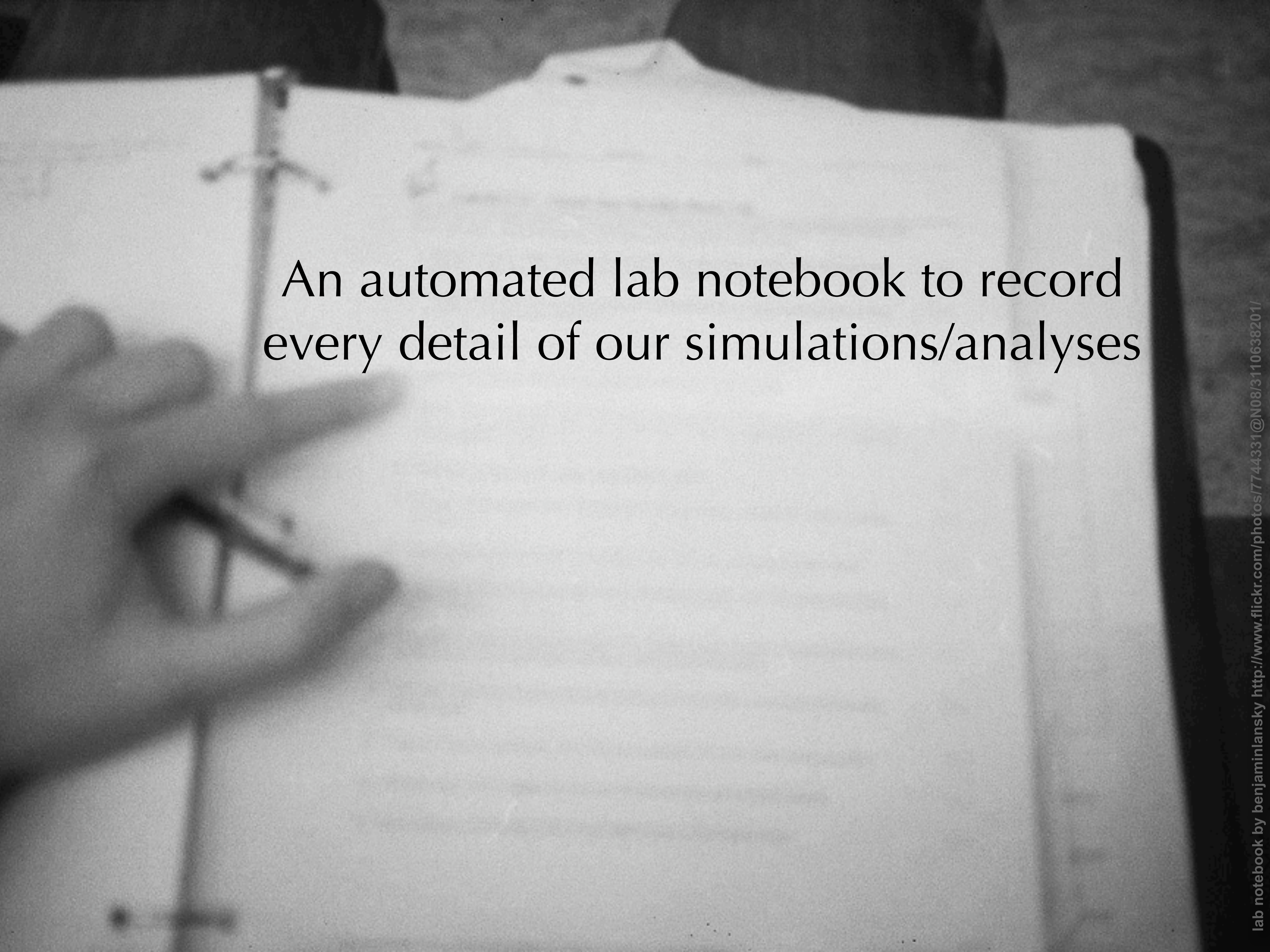
"Why did I do that?"

Why isn't it easy to reproduce a computational experiment exactly?

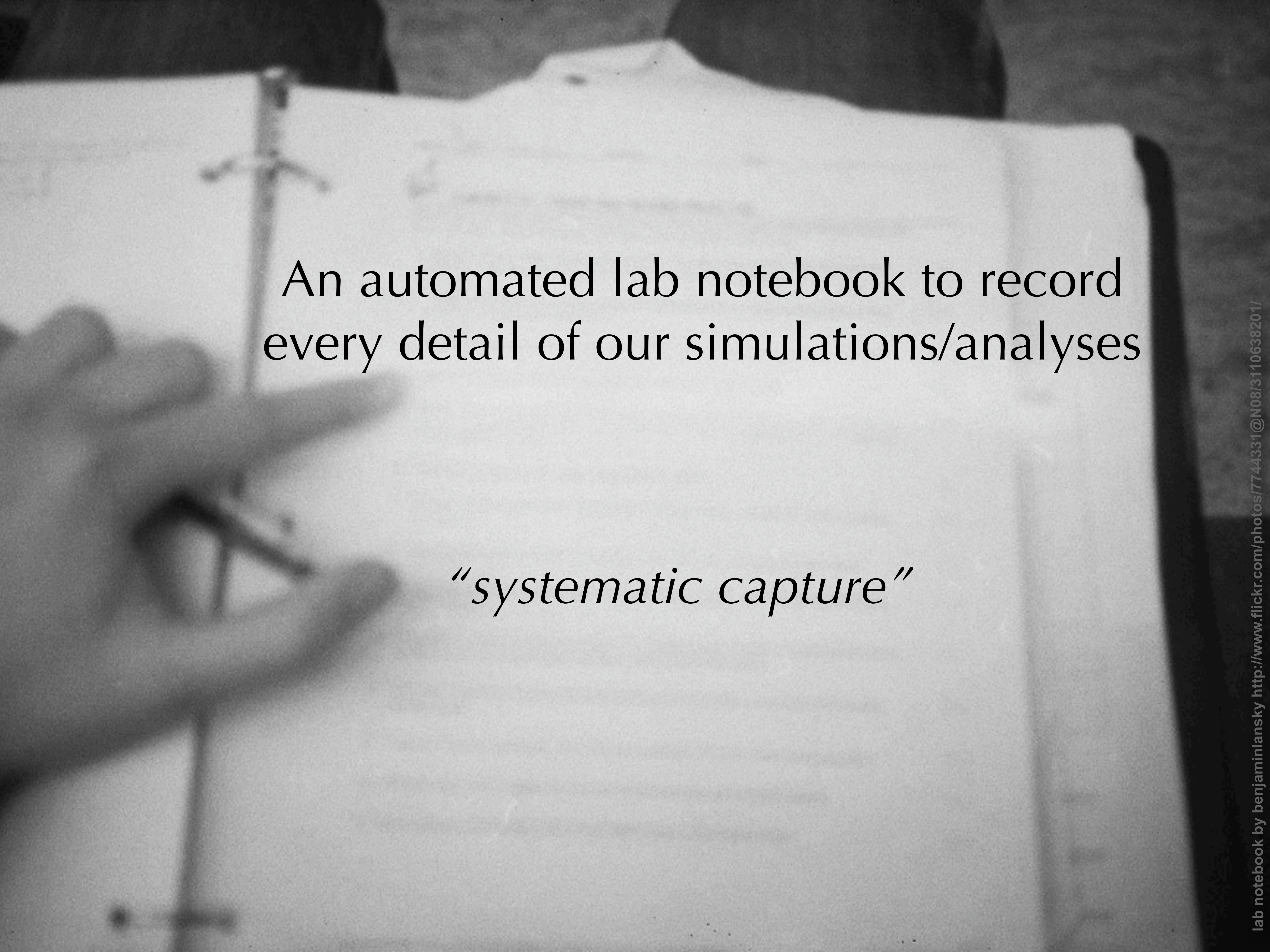
Cute clones by jurvetson http://www.flickr.com/photos/44124348109@N01/3327872958/

What can we do about it?

Cute clones by jurvetson http://www.flickr.com/photos/44124348109@N01/332787295/



An automated lab notebook to record every detail of our simulations/analyses



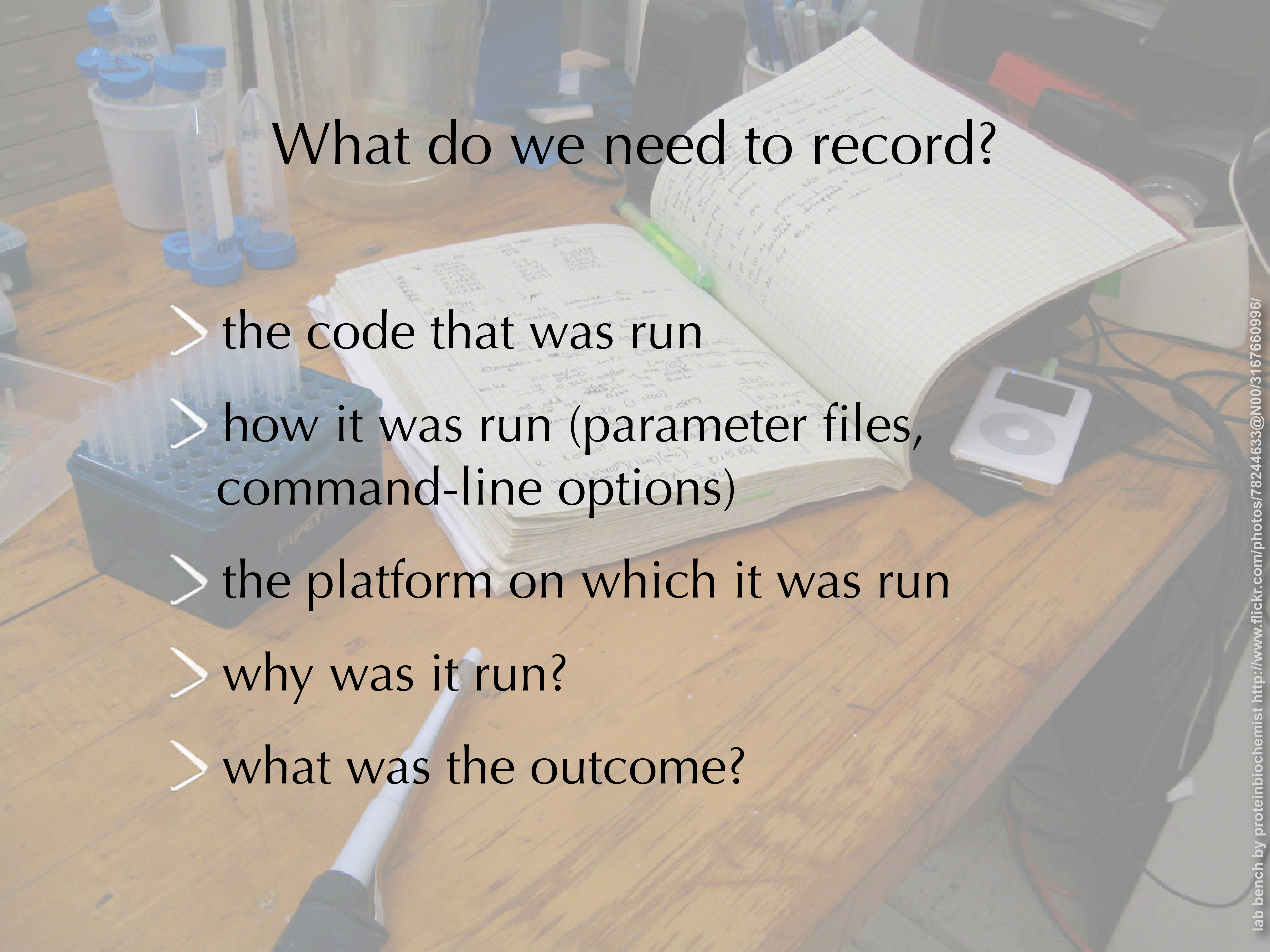
An automated lab notebook to record every detail of our simulations/analyses

“systematic capture”

What do we need to record?

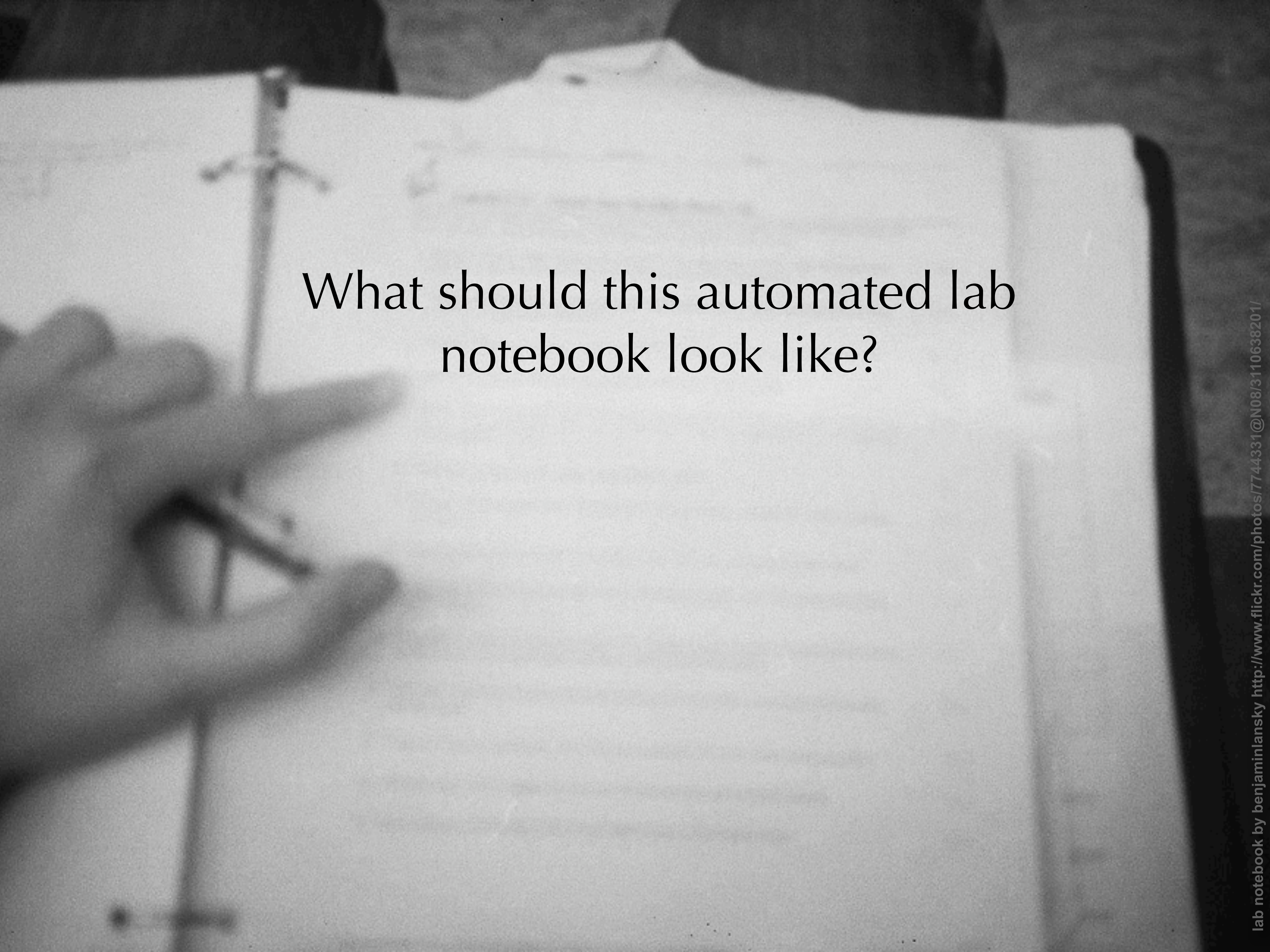
lab bench by proteinbiochemist http://www.flickr.com/photos/78244633@N00/3167660996/

What do we need to record?



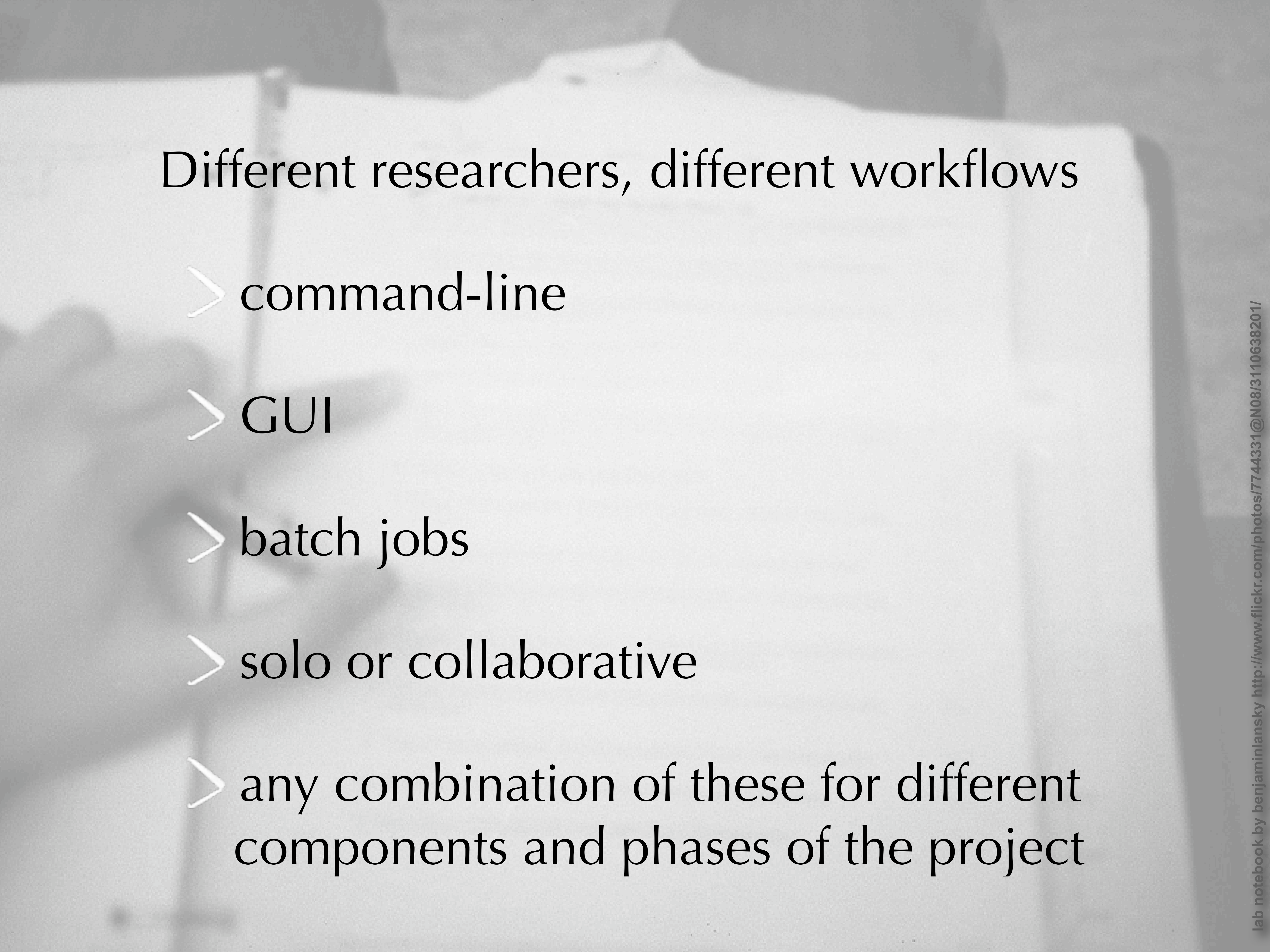
- the code that was run
- how it was run (parameter files, command-line options)
- the platform on which it was run
- why was it run?
- what was the outcome?

lab bench by proteinbiochemist http://www.flickr.com/photos/78244633@N00/3167660996/



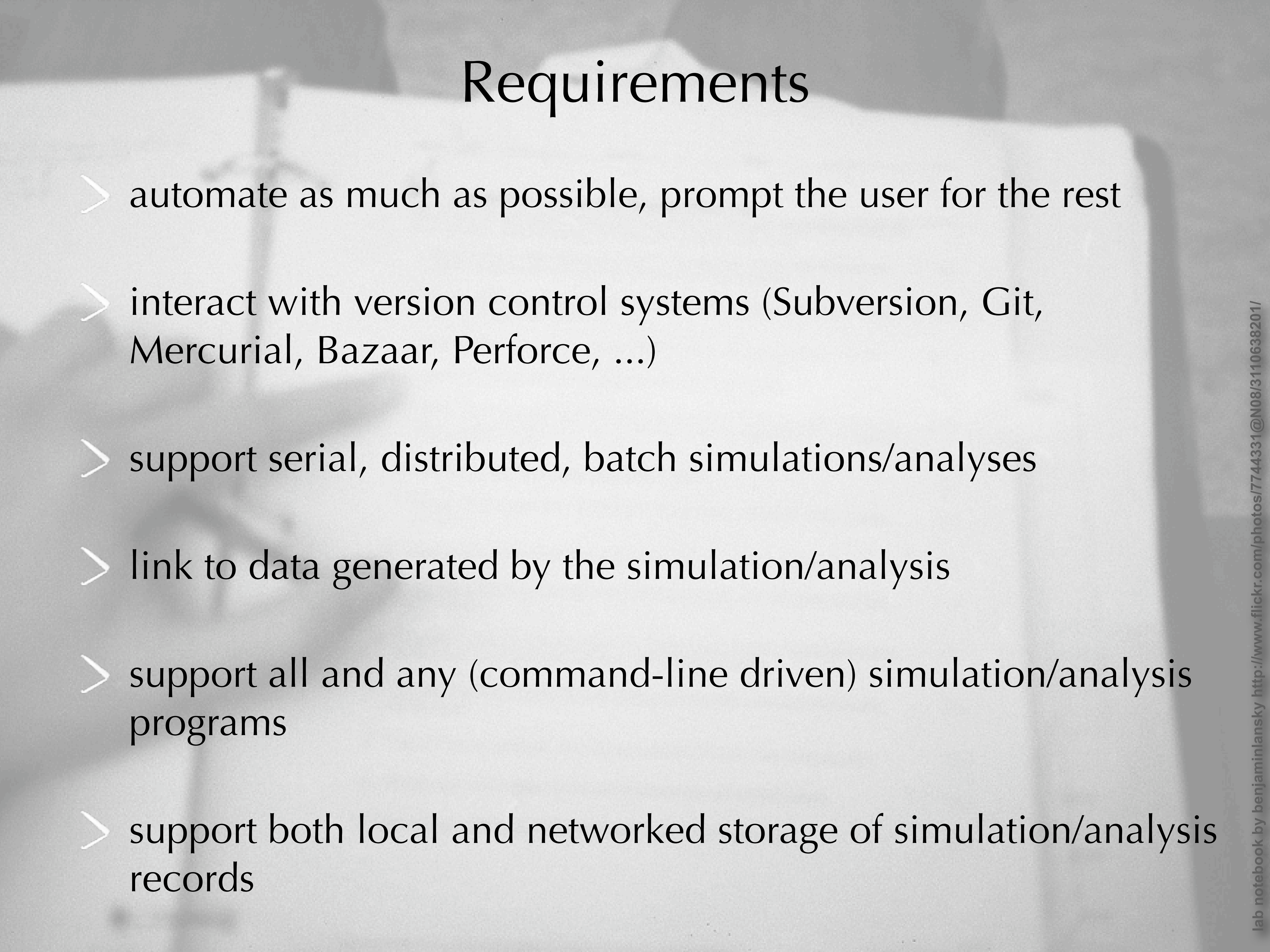
What should this automated lab notebook look like?

lab notebook by benjaminlansky http://www.flickr.com/photos/77443331@N08/3110638201/



Different researchers, different workflows

- command-line
- GUI
- batch jobs
- solo or collaborative
- any combination of these for different components and phases of the project

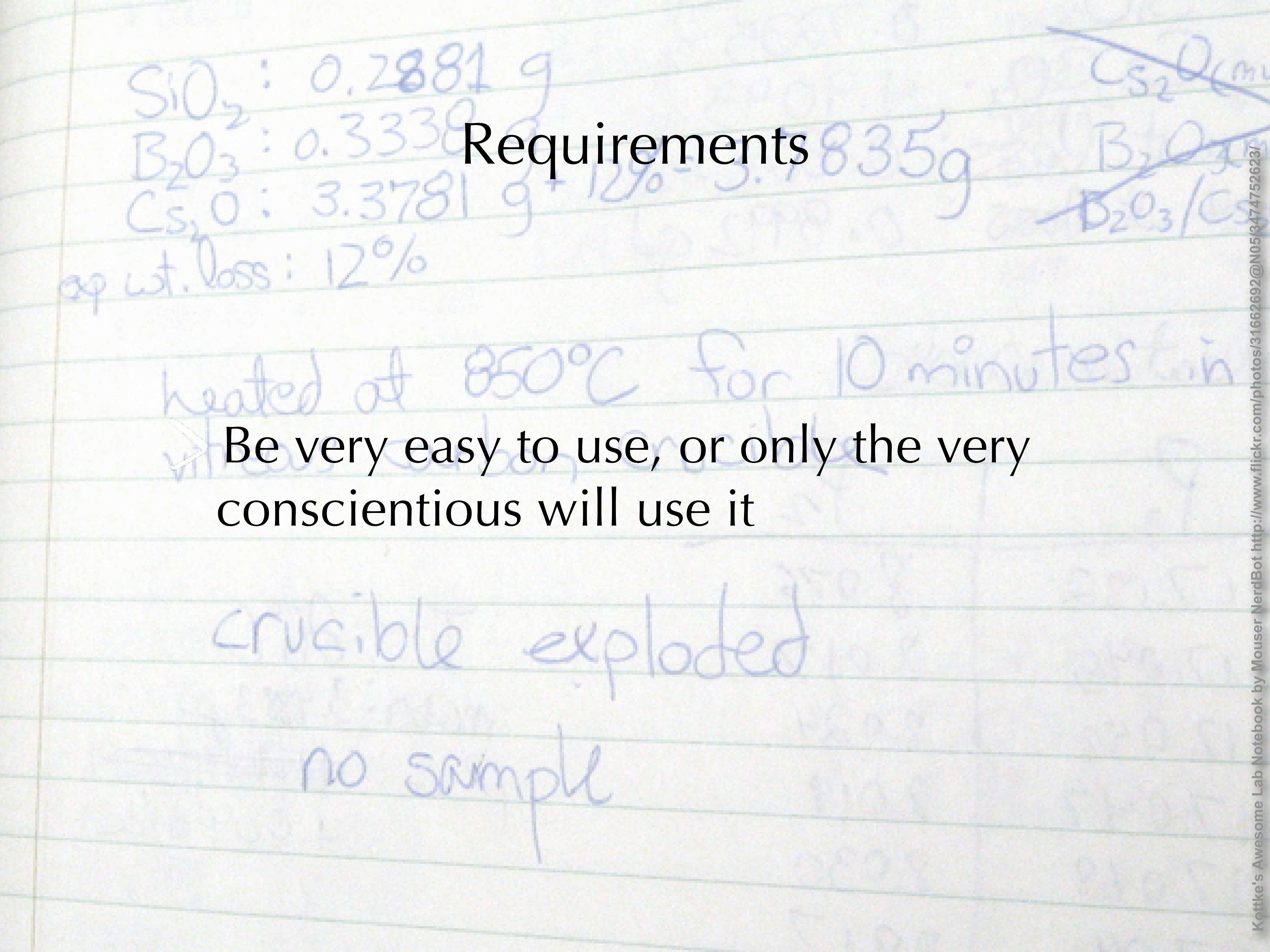


Requirements

- automate as much as possible, prompt the user for the rest
- interact with version control systems (Subversion, Git, Mercurial, Bazaar, Perforce, ...)
- support serial, distributed, batch simulations/analyses
- link to data generated by the simulation/analysis
- support all and any (command-line driven) simulation/analysis programs
- support both local and networked storage of simulation/analysis records

Requirements

- Be very easy to use, or only the very conscientious will use it

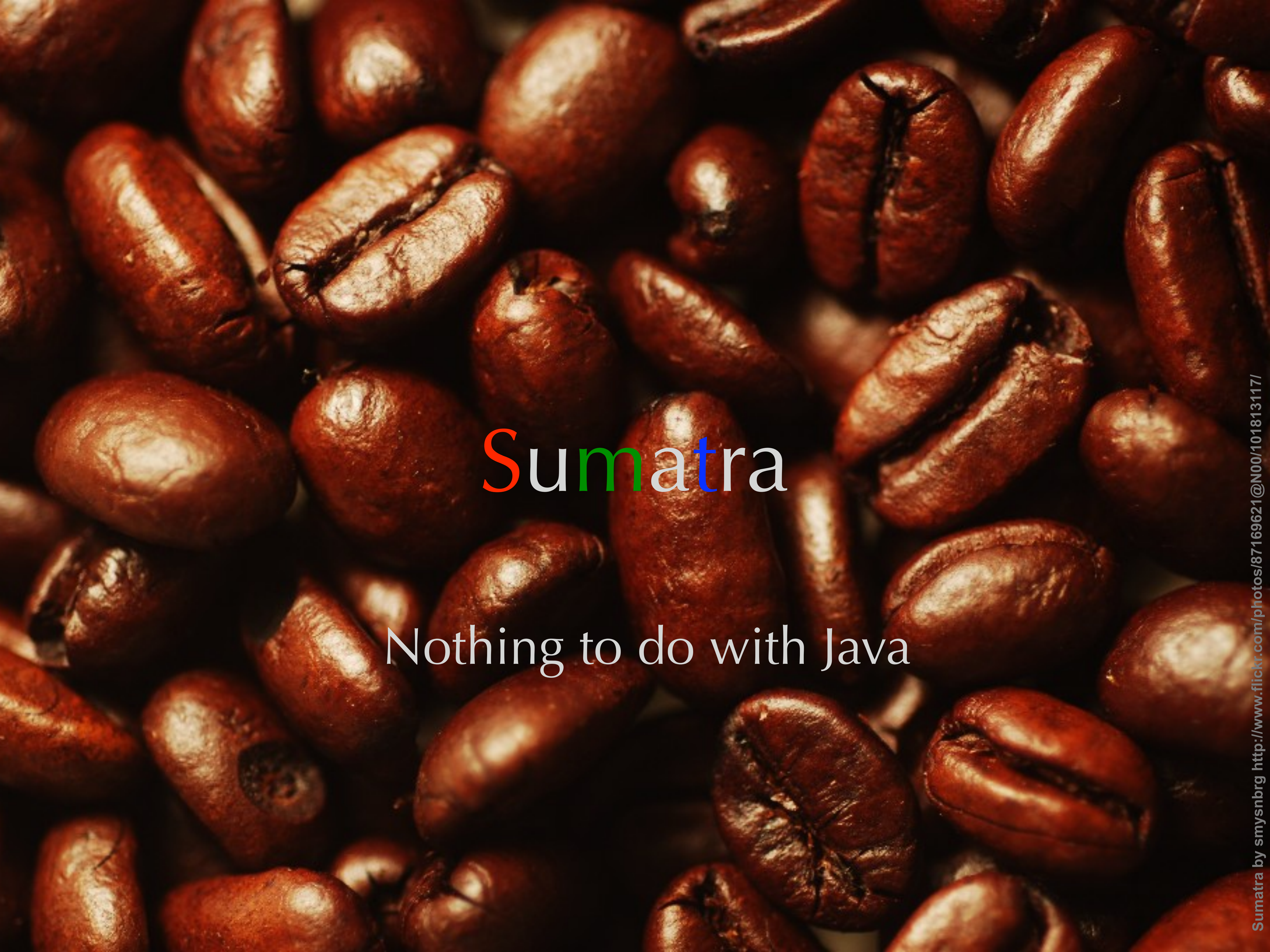


Sumatra

Simulation Management Tool

http://neuralensemble.org/trac/sumatra

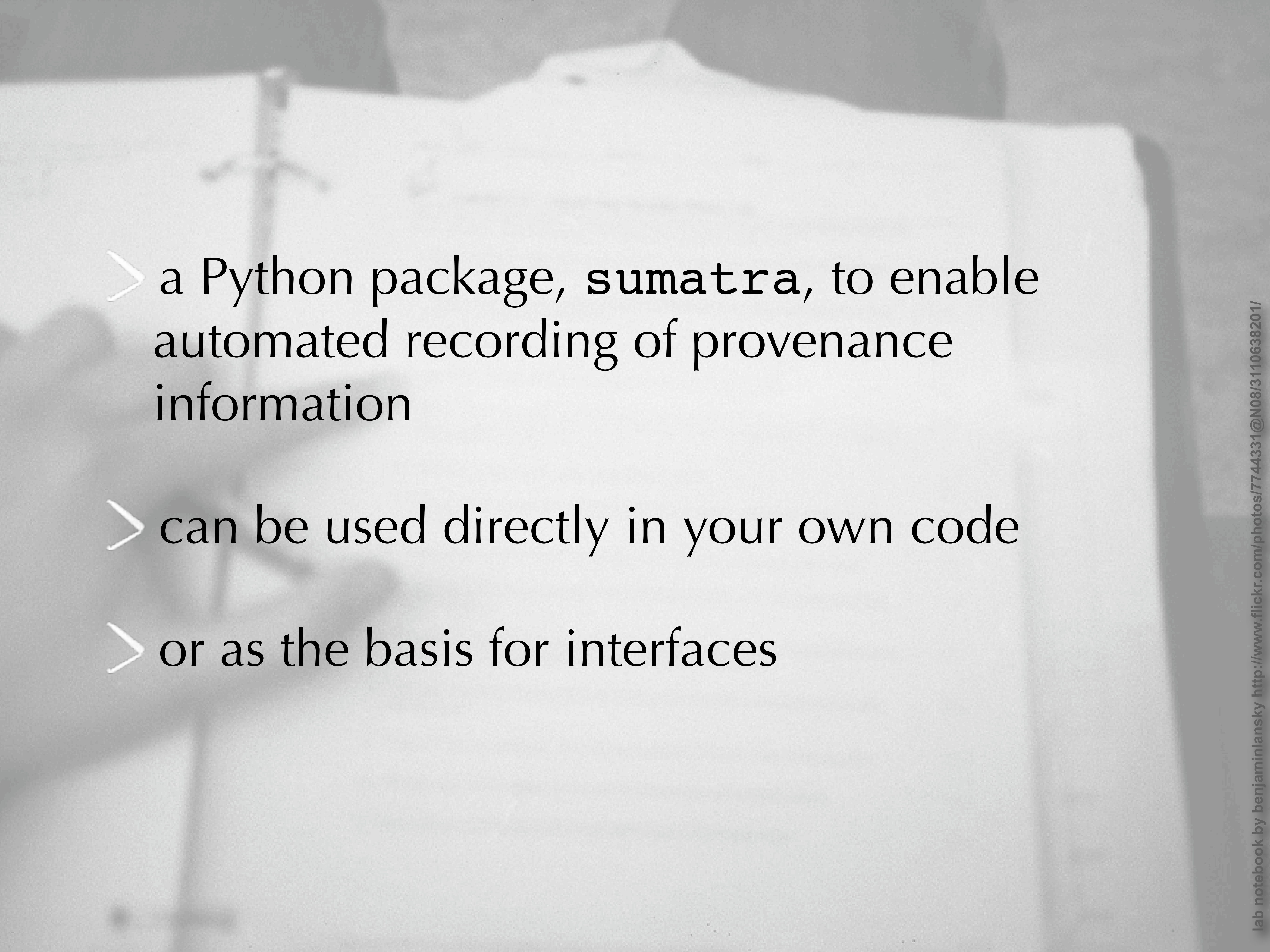
Sawahs in West Sumatra by CharlesFred http://www.flickr.com/photos/charlesfred/2869003149/



Sumatra

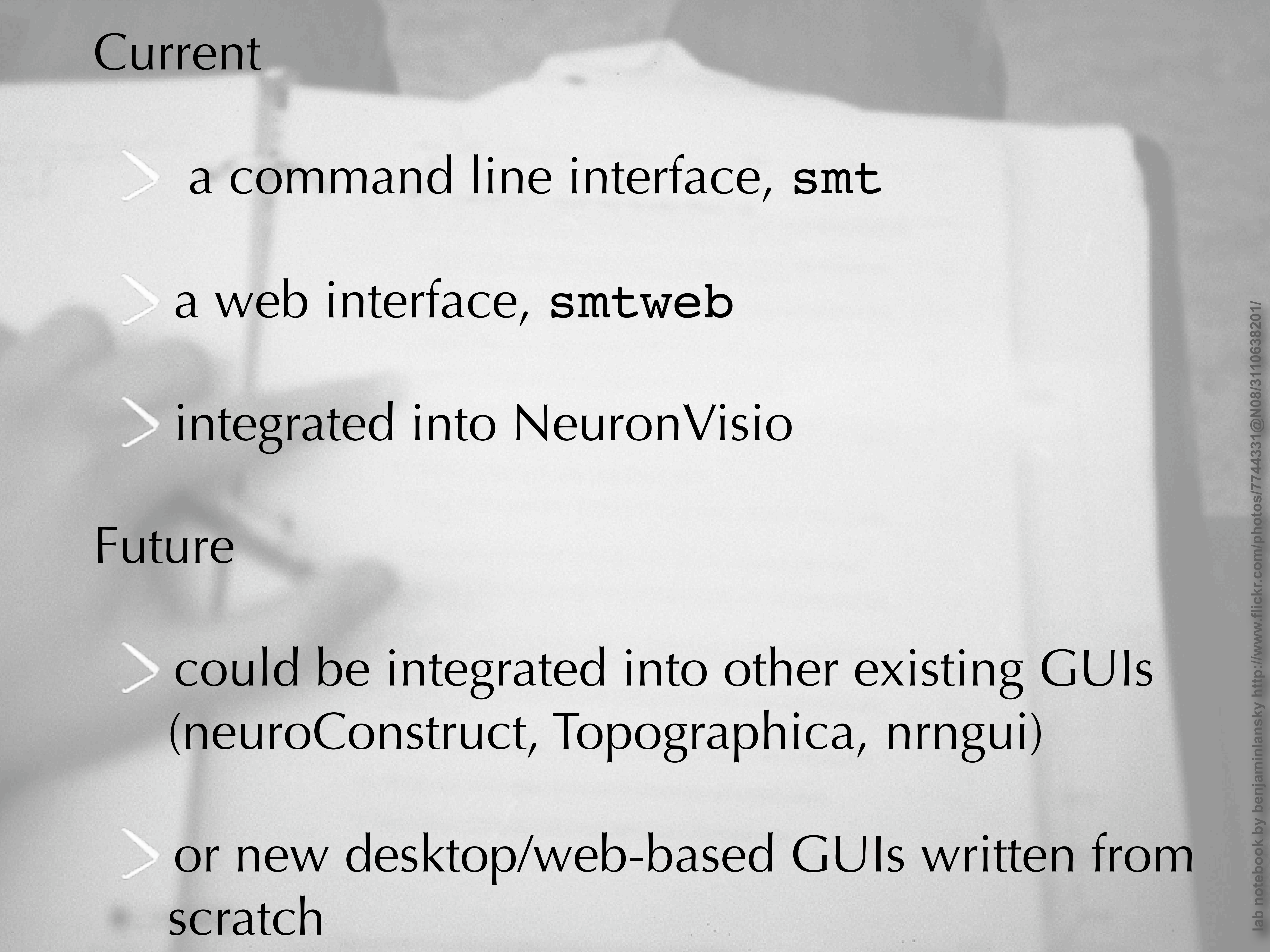
Nothing to do with Java

Sumatra by smysnbrg http://www.flickr.com/photos/87169621@N00/1018131117/



- a Python package, `sumatra`, to enable automated recording of provenance information
- can be used directly in your own code
- or as the basis for interfaces

lab notebook by benjaminlansky http://www.flickr.com/photos/7744331@N08/3110638201/

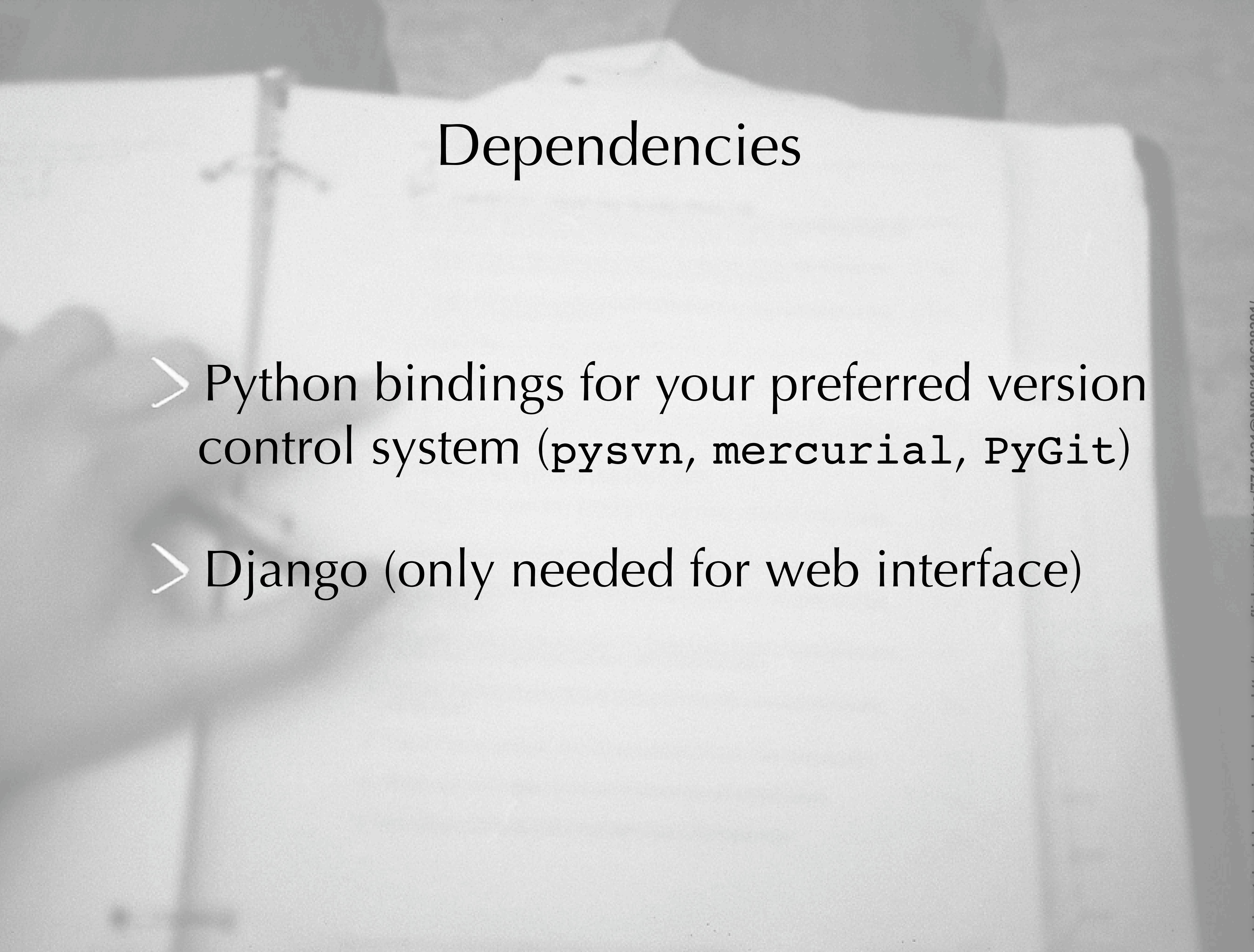


Current

- a command line interface, **`smt`**
- a web interface, **`smtweb`**
- integrated into NeuronVisio

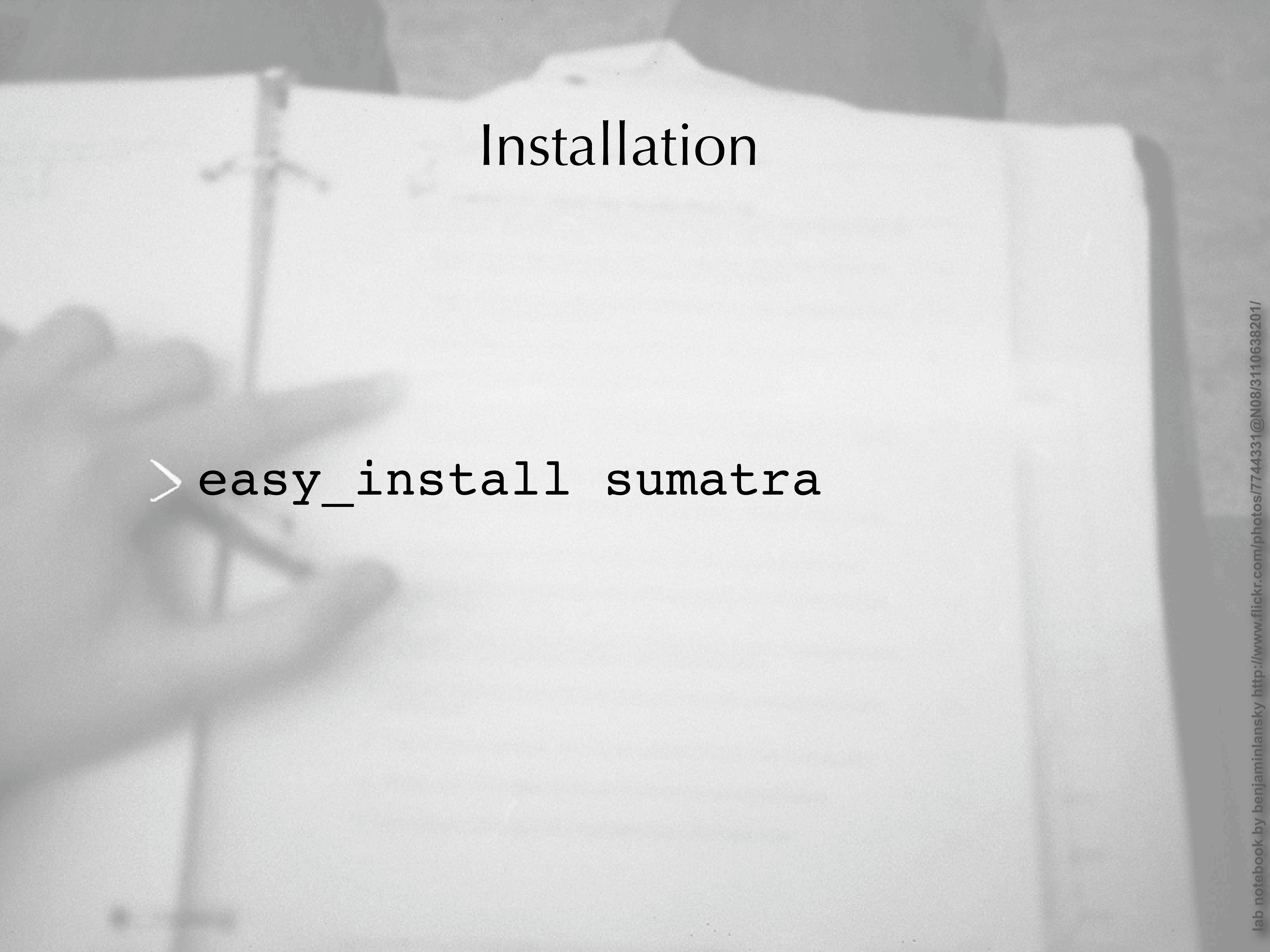
Future

- could be integrated into other existing GUIs (neuroConstruct, Topographica, nrngui)
- or new desktop/web-based GUIs written from scratch



Dependencies

- Python bindings for your preferred version control system (`pysvn`, `mercurial`, `PyGit`)
- Django (only needed for web interface)



Installation

```
> easy_install sumatra
```

lab notebook by benjaminlansky http://www.flickr.com/photos/7744331@N08/3110638201/

Using sumatra within your own scripts

```python
import numpy
import sys

parameter_file = sys.argv[1]
parameters = {}
execfile(parameter_file, parameters) # this way of reading parameters
                                     # is not necessarily recommended
numpy.random.seed(parameters["seed"])
distr = getattr(numpy.random, parameters["distr"])
data = distr(size=parameters["n"])

output_file = "example.dat"
numpy.savetxt(output_file, data)
```

```python
import numpy
import sys
import time
from sumatra.projects import load_simulation_project
from sumatra.parameters import build_parameters

project = load_simulation_project()
start_time = time.time()

parameter_file = sys.argv[1]
parameters = build_parameters(parameter_file)

sim_record = project.new_record(parameters=parameters,
                                main_file=__file__,
                                label="api_example",
                                reason="reason for running this simulation")

numpy.random.seed(parameters["seed"])
distr = getattr(numpy.random, parameters["distr"])
data = distr(size=parameters["n"])

output_file = "%s.dat" % sim_record.label
numpy.savetxt(output_file, data)

sim_record.duration = time.time() - start_time
sim_record.data_key = sim_record.datastore.find_new_files(sim_record.timestamp)
project.add_record(sim_record)

project.save()
```

smt

```
$ cd myproject
$ smt init MyProject
$ python main.py default.param
$ python main.py default.param
$ smt run --simulator=python --main=main.py default.param
$ python main.py default.param
$ smt run --simulator=python --main=main.py default.param
$ smt configure --simulator=python --main=main.py
$ python main.py default.param
$ smt run --simulator=python --main=main.py default.param
$ smt configure --simulator=python --main=main.py
$ smt run default.param
$ smt list
default_20090930-174949
default_20090930-175111

$ smt list -l
---------------------------------------------
Label      : default_20090930-174949
Reason     :
Outcome    :
Duration   : 0.0548920631409
Script     : MercurialRepository at /path/to/myproject
             rf9ab74313efe (main file is main.py)
Executable : Python (version: 2.6.2) at /usr/bin/python
Timestamp  : 2009-09-30 17:49:49.235772
Tags       :
.
.
.
```

```
$ smt run --label=haggling --reason="determine whether
the gourd is worth 3 or 4 shekels" romans.param
```

```
$ smt comment "apparently, it is worth NaN shekels."
$ smt comment default_20090930-174949 "Eureka! Nobel
prize here we come."
```

```
$ smt tag “Figure 6”
$ smt run --reason="test effect of a smaller time
constant" default.param tau_m=10.0
```

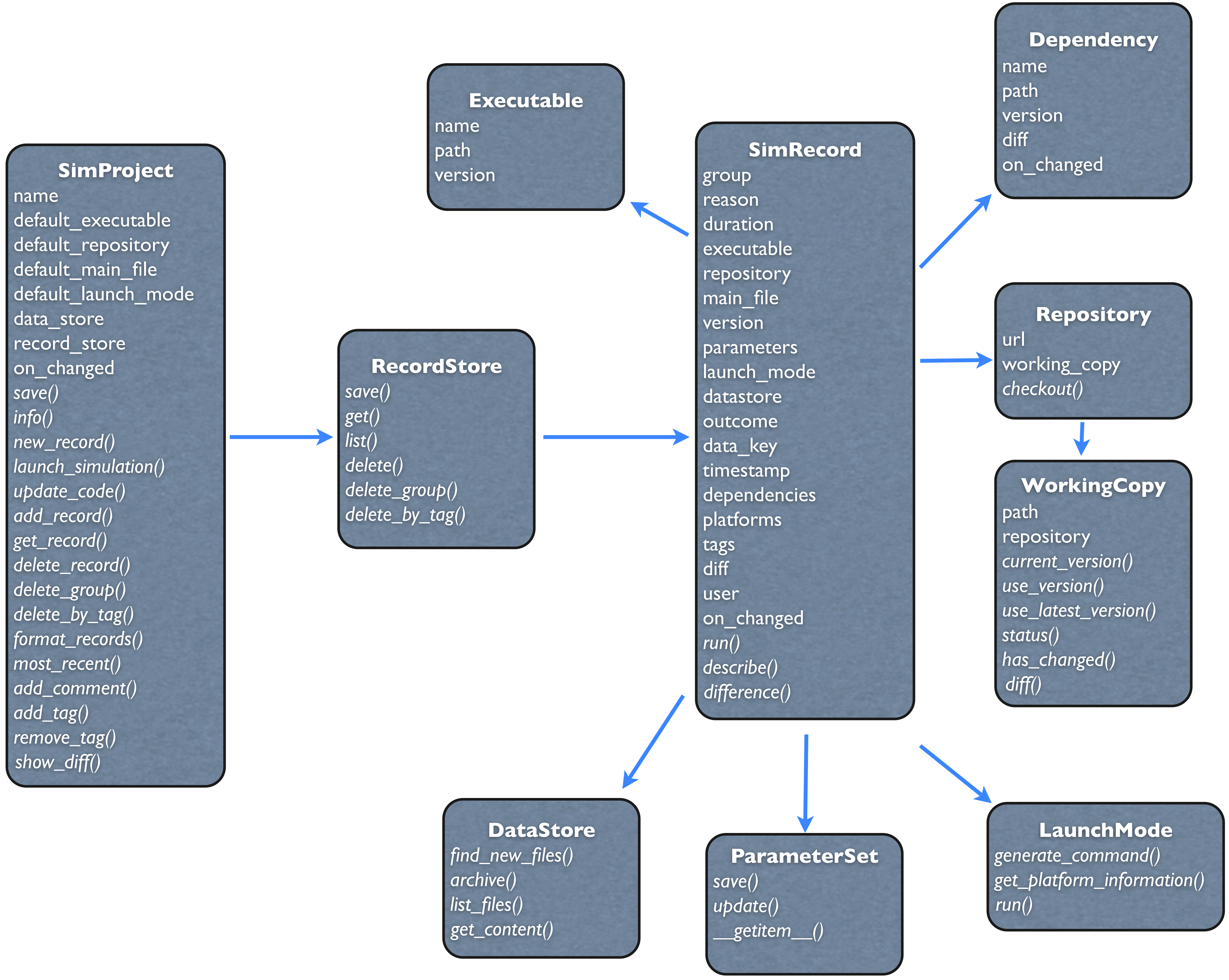
```
$ smt repeat haggling_2009101002
The simulation results match.
```

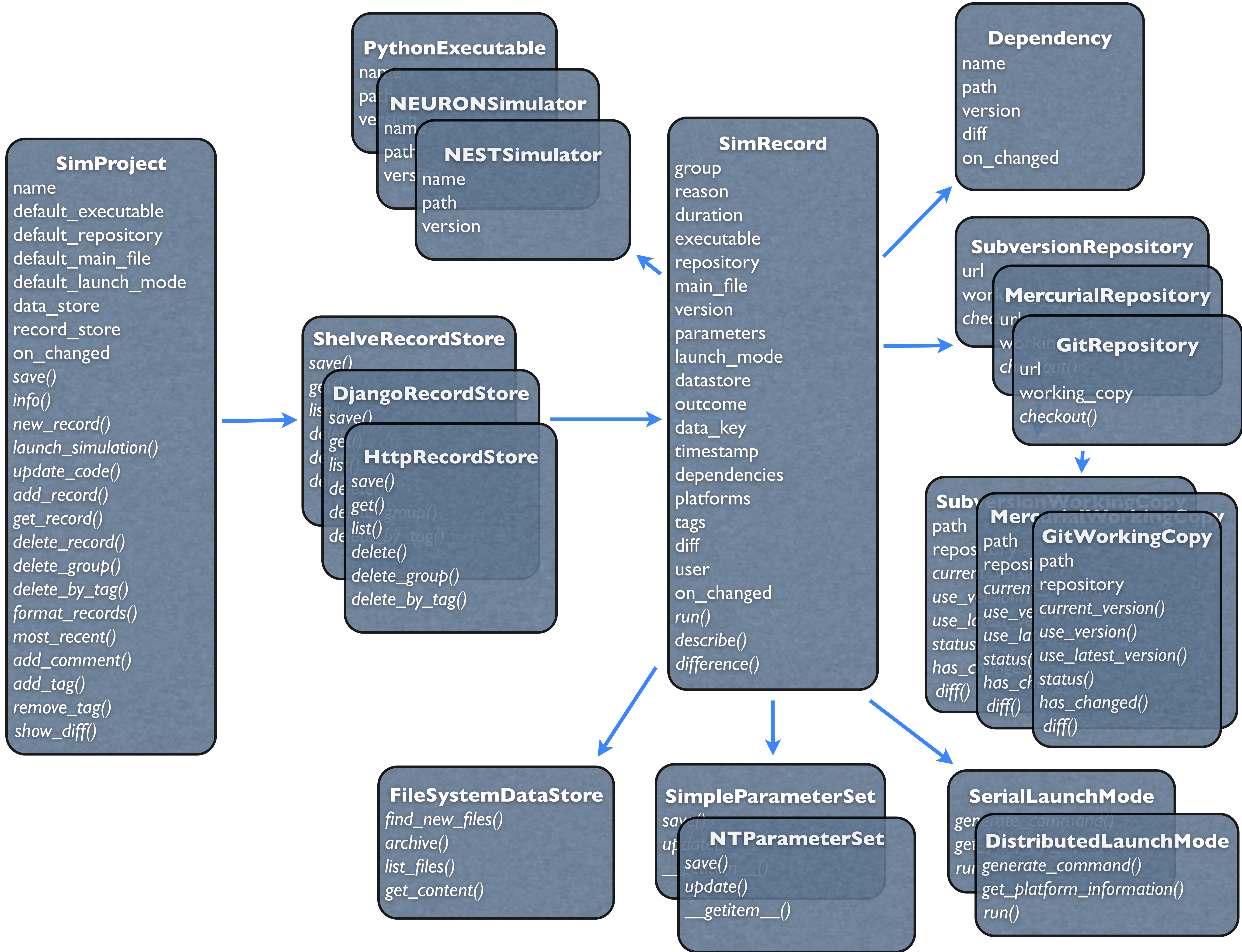
```
$ smt
Usage: smt <subcommand> [options] [args]

Simulation management tool, version 0.1

Available subcommands:
  init
  configure
  info
  run
  list
  delete
  comment
  tag
  repeat
  help
```

```
$ smtweb 8002 &
```


SimProject
name
default_executable
default_repository
default_main_file
default_launch_mode
data_store
record_store
on_changed
save()
info()
new_record()
launch_simulation()
update_code()
add_record()
get_record()
delete_record()
delete_group()
delete_by_tag()
format_records()
most_recent()
add_comment()
add_tag()
remove_tag()
show_diff()
RecordStore
save()
get()
list()
delete()
delete_group()
delete_by_tag()
Executable
name
path
version
SimRecord
group
reason
duration
executable
repository
main_file
version
parameters
launch_mode
datastore
outcome
data_key
timestamp
dependencies
platforms
tags
diff
user
on_changed
run()
describe()
difference()
Dependency
name
path
version
diff
on_changed
Repository
url
working_copy
checkout()
WorkingCopy
path
repository
current_version()
use_version()
use_latest_version()
status()
has_changed()
diff()
DataStore
find_new_files()
archive()
list_files()
get_content()
ParameterSet
save()
update()
__getitem__()
LaunchMode
generate_command()
get_platform_information()
run()


SimProject
name
default_executable
default_repository
default_main_file
default_launch_mode
data_store
record_store
on_changed
save()
info()
new_record()
launch_simulation()
update_code()
add_record()
get_record()
delete_record()
delete_group()
delete_by_tag()
format_records()
most_recent()
add_comment()
add_tag()
remove_tag()
show_diff()
PythonExecutable
NEURONSimulator
NESTSimulator
name
path
version
ShelveRecordStore
save()
DjangoRecordStore
save()
HttpRecordStore
save()
get()
list()
delete()
delete_group()
delete_by_tag()
SimRecord
group
reason
duration
executable
repository
main_file
version
parameters
launch_mode
datastore
outcome
data_key
timestamp
dependencies
platforms
tags
diff
user
on_changed
run()
describe()
difference()
Dependency
name
path
version
diff
on_changed
SubversionRepository
url
MercurialRepository
GitRepository
url
working_copy
checkout()
MercurialWorkingCopy
GitWorkingCopy
path
repository
current_version()
use_version()
use_latest_version()
status()
has_changed()
diff()
FileSystemDataStore
find_new_files()
archive()
list_files()
get_content()
SimpleParameterSet
NTParameterSet
save()
update()
__getitem__()
SerialLaunchMode
DistributedLaunchMode
generate_command()
get_platform_information()
run()

What’s new since CodeJam #3

- Released version 0.1 (not vaporware anymore)
- Michele Mattioni added git support
- `RecordStore` can now contain records from multiple projects/users
- Added `HttpRecordStore`* enabling storing records on a remote server

**almost finished*

http://neuralensemble.org/trac/sumatra

@apdavison

http://www.andrewdavison.info

Sumatran orangutan by BelalangJantan http://www.flickr.com/photos/7164478@N07/3575735482/