Reproducibility through environment capture: Part 1: Docker

Andrew Davison
Unité de Neurosciences, Information et Complexité (UNIC)
Centre National de la Recherche Scientifique
Gif sur Yvette, France
http://andrewdavison.info
davison@unic.cnrs-gif.fr

HBP CodeJam Workshop #7

Manchester, 14/01/2016

- what code was run?
  - which executable?
    - name, location, version, compiler, compilation options
  - which script?
    - name, location, version
    - options, parameters
    - dependencies (name, location, version)
- what were the input data?
  - name, location, content
- what were the outputs?
  - data, logs, stdout/stderr
- who launched the computation?
- when was it launched/when did it run? (queueing systems)
- where did it run?
  - machine name(s), other identifiers (e.g. IP addresses)
  - processor architecture
  - available memory
  - operating system
- why was it run?
- what was the outcome?
- which project was it part of?

Iceberg by Uwe Kils http://commons.wikimedia.org/wiki/File:Iceberg.jpg

Environment capture

- capturing all the details of the scientist's code, data and computing environment, in order to be able to reproduce a given computation at a later time.

- adapt to/extend people's existing workflow management, rather than replace it

artefact capture

store the environment
in binary format

VM/Docker

CDE/Reprozip

pre-emptive capture

create a pre-defined environment,
always run in this enviroment

Docker/Vagrant

run-time capture

capture the environment at the
same time you run the experiment

Sumatra/noWorkflow/recipy

metadata capture

store the information needed
to recreate the environment

Creating pre-defined environments

- do all your research in a virtual machine (using VMWare, VirtualBox, etc.) or in a software container (using Docker, LXC, etc.)
- ideally environment creation should be automated (shell script, Puppet, Chef, Vagrant, Dockerfile, etc.)
- when other scientists wish to replicate your results, you send them the VM/Docker image together with some instructions
- they can then load the image on their own computer, or run it in the cloud.

Example: Docker

- a lightweight alternative to virtual machines
- create portable, isolated Linux environments that can run on any Linux host
- can also run on OS X and Windows hosts through the Docker Toolkit (transparent VM)
- download prebuilt environments, or build your own with a Dockerfile

A Dockerfile for simulations with NEST

start with Neurodebian

install Debian packages

create a Python virtualenv

download NEST

build NEST

```
FROM neurodebian:jessie
MAINTAINER andrew.davison@unic.cnrs-gif.fr

ENV DEBIAN_FRONTEND noninteractive
RUN apt-get update
ENV LANG=C.UTF-8 HOME=/home/docker NEST=nest-2.6.0

RUN apt-get install -y automake libtool build-essential openmpi-bin libopenmpi-dev git vim  \
                       wget python libpython-dev libncurses5-dev libreadline-dev libgsl0-dev cython \
                       python-pip python-numpy python-scipy python-matplotlib python-jinja2 python-mock \
                       python-virtualenv ipython python-docutils python-yaml \
                       subversion python-mpi4py python-tables

RUN useradd -ms /bin/bash docker
USER docker
RUN mkdir $HOME/env; mkdir $HOME/packages

ENV VENV=$HOME/env/neurosci
RUN virtualenv --system-site-packages $VENV
RUN $VENV/bin/pip install --upgrade nose ipython

WORKDIR /home/docker/packages
RUN wget http://www.nest-simulator.org/downloads/gplreleases/$NEST.tar.gz
RUN tar xzf $NEST.tar.gz; rm $NEST.tar.gz
RUN svn co --username Anonymous --password Anonymous --non-interactive http://svn.incf.org/svn/libneurosim/trunk libneurosim
RUN cd libneurosim; ./autogen.sh

RUN mkdir $VENV/build
WORKDIR $VENV/build
RUN mkdir libneurosim; \
    cd libneurosim; \
    PYTHON=$VENV/bin/python $HOME/packages/libneurosim/configure --prefix=$VENV; \
    make; make install; ls $VENV/lib $VENV/include
RUN mkdir $NEST; \
    cd $NEST; \
    PYTHON=$VENV/bin/python $HOME/packages/$NEST/configure --with-mpi --prefix=$VENV --with-libneurosim=$VENV; \
    make; make install

WORKDIR /home/docker/
```

```
(host)$ docker build -t simenv .

(host)$ docker run -it simenv /bin/bash

(docker)$ echo “Now you have a reproducible environment
with NEST already installed”
```

```
(docker)$ …
```

```
(host)$ docker commit 363fdeaba61c simenv:snapshot

(host)$ docker run -it simenv:snapshot /bin/bash
```

```
(host)$ docker pull neuralensemble/simulationx

(host)$ docker run -d neuralensemble/simulationx

(host)$ ssh -Y -p 32768 docker@localhost

(docker)$ echo “Now you have a reproducible environment
with NEST, NEURON, Brian, PyNN, X11, numpy, scipy,
IPython, matplotlib, etc. already installed”
```

Virtual machines / Docker

Advantages

- extremely simple
- robust - by definition, everything is captured

Disadvantages

- VM images often very large files, several GB or more. Docker images smaller, but still ~1 GB
- risk of results being highly sensitive to the particular configuration of the VM - not easily reproducible on different hardware or with different versions of libraries (highly replicable but not reproducible)
- not possible to index, search or analyse the provenance information
- virtualisation technologies inevitably have a performance penalty, even if small
- the approach is challenging in a context of distributed computations spread over multiple machines.

Reproducibility through environment capture: Part 2: Sumatra

Andrew Davison
Unité de Neurosciences, Information et Complexité (UNIC)
Centre National de la Recherche Scientifique
Gif sur Yvette, France
http://andrewdavison.info
davison@unic.cnrs-gif.fr

HBP CodeJam Workshop #7

Manchester, 14/01/2016

artefact capture

store the environment
in binary format

VM/Docker

CDE/Reprozip

pre-emptive capture

create a pre-defined environment,
always run in this enviroment

Docker/Vagrant

run-time capture

capture the environment at the
same time you run the experiment

Sumatra/noWorkflow/recipy

metadata capture

store the information needed
to recreate the environment

Run-time metadata capture

- rather than capture the entire experiment context (code, data, environment) as a binary snapshot, aims to capture all the information needed to recreate the context

Example: Sumatra

```
$ python main.py input_data
```

```
$ smt configure --executable=python --main=main.py
```

```
$ smt run input_data
```

```
from sumatra.decorators import capture

@capture
def main(parameters):
    …
```

Code versioning and dependency tracking

the code, the whole code and nothing but the code

1. Recursively find imported/ included libraries
2. Try to determine version information for each of these, using

(i) code analysis

(ii) version control systems

(iii) package managers

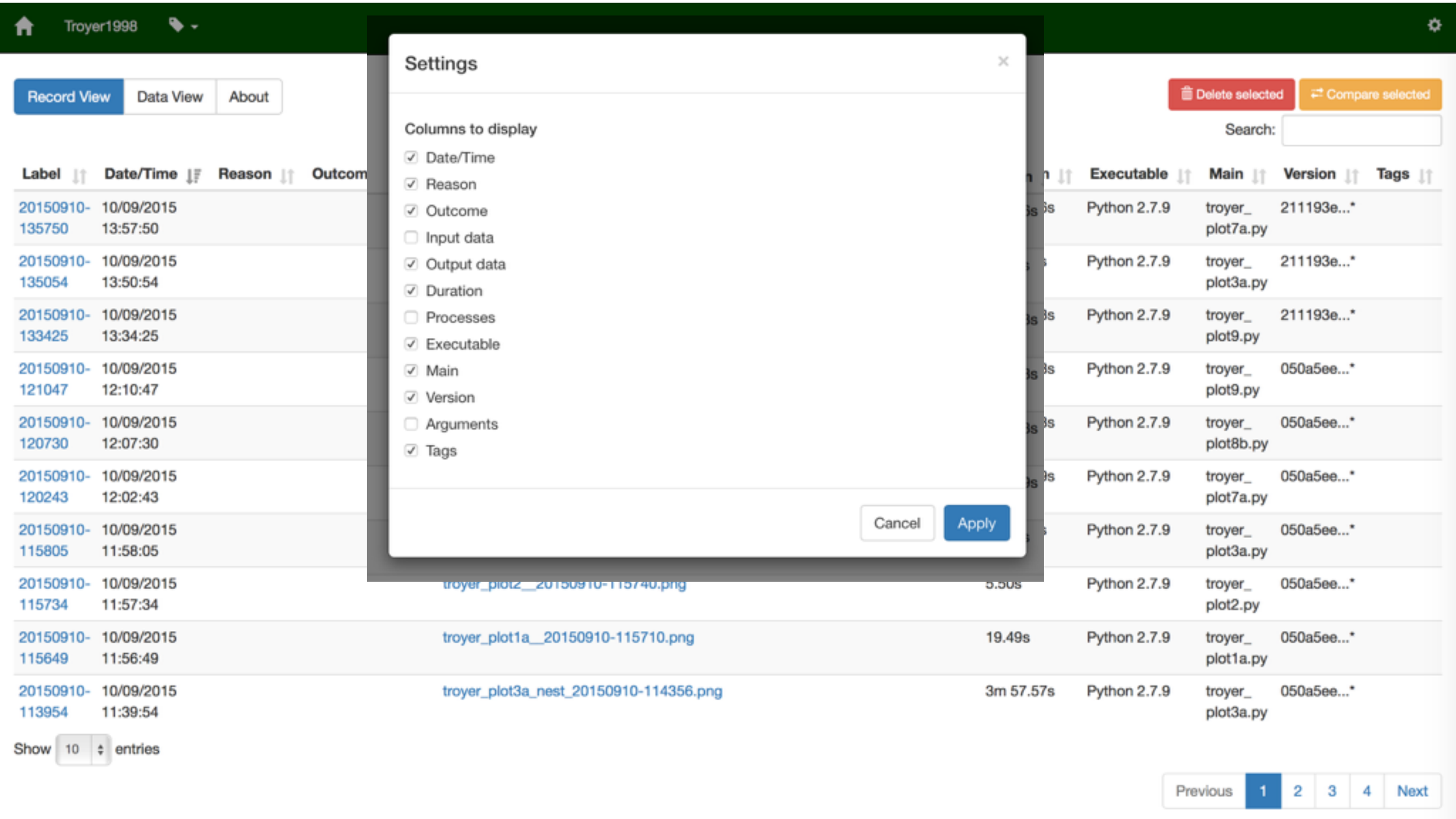
(iv) etc.

Iceberg by Uwe Kils http://commons.wikimedia.org/wiki/File:Iceberg.jpg

Configuration

- Launching computations
  - locally, remotely, serial or parallel
- Output data storage
  - local, remote (WebDAV), mirrored, archived
- Provenance database
  - SQLite, PostgreSQL, REST API, MongoDB, …

Browser interface




```
$ smtweb -p 8008 &
```

20150910-135750

Delete record

```
$ /home/docker/env/neurosci/bin/python troyer_plot7a.py --save-figures nest data/positions_on.cpickle data/positions_off.cpickle
data/0.5_spike_train_off_layer0.cpickle data/0.5_spike_train_off_layer1.cpickle data/0.5_spike_train_off_layer2.cpickle data/0.5_spike_train_off_layer3.cpickle
data/0.5_spike_train_on_layer0.cpickle data/0.5_spike_train_on_layer1.cpickle data/0.5_spike_train_on_layer2.cpickle data/0.5_spike_train_on_layer3.cpickle
```

Run on 10/09/2015 13:57:50 by

Working directory: /home/docker/projects/Troyer1998/PyNN
Code version: 211193e97496ea2b8c4abfdaadd862897c3b236d* (diff)
Repository: /home/docker/projects/Troyer1998 - cloned from https://github.com/apdavison/V1NetworkModels.git
Python version: 2.7.9
Reason: Reproduce Figure 7A from Troyer et al. (1998). Edit

Tags: Figure 7A NEST Edit

Outcome

Edit

Results are qualitatively similar (note that here we plot conductivity as a proxy for the current, which in the Troyer paper is calculated as if the voltage was clamped at threshold). Closer comparison needed.

Input data

Filename	Path	Digest	Size	Date/Time	Output of	Input to
0.5_spike_train_off_layer0.cpickle	data/0.5_spike_train_off_layer0.cpickle	f2c5bb1e2...	142.1 KB	09/09/2015 15:32:44	20150909-153244	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_off_layer1.cpickle	data/0.5_spike_train_off_layer1.cpickle	818ed5476...	142.7 KB	09/09/2015 15:32:44	20150909-153244	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_off_layer2.cpickle	data/0.5_spike_train_off_layer2.cpickle	c776f8b01...	141.5 KB	09/09/2015 15:32:44	20150909-153244	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_off_layer3.cpickle	data/0.5_spike_train_off_layer3.cpickle	516f91128...	140.9 KB	09/09/2015 15:32:44	20150909-153244	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_on_layer0.cpickle	data/0.5_spike_train_on_layer0.cpickle	58c0091d5...	114.4 KB	09/09/2015 15:55:15	20150909-155515	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_on_layer1.cpickle	data/0.5_spike_train_on_layer1.cpickle	74e437a7e...	115.0 KB	09/09/2015 15:55:15	20150909-155515	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_on_layer2.cpickle	data/0.5_spike_train_on_layer2.cpickle	d27e44085...	115.0 KB	09/09/2015 15:55:15	20150909-155515	20150910-135750, 20150910-135054, 20150910-133425
0.5_spike_train_on_layer3.cpickle	data/0.5_spike_train_on_layer3.cpickle	bbc7f8486...	114.1 KB	09/09/2015 15:55:15	20150909-155515	20150910-135750, 20150910-135054, 20150910-133425
positions_off.cpickle	data/positions_off.cpickle	7856f9bb2...	27.5 KB	09/09/2015 16:48:28	20150909-164828	20150910-135750, 20150910-135054, 20150910-133425
positions_on.cpickle	data/positions_on.cpickle	3d8027855...	27.5 KB	09/09/2015 16:48:28	20150909-164828	20150910-135750, 20150910-135054, 20150910-133425

Output data

Output data

Filename	Path	Digest	Size	Date/Time	Output of	Input to
troyer_plot7a_nest_20150910-140227.png	output_data/20150910/troyer_plot7a_nest_20150910-140227.png	310f72f25...	77.4 KB	10/09/2015 13:57:50	20150910-135750	

Dependencies

Name	Path	Version
IPython	/home/docker/env/neurosci/local/lib/python2.7/site-packages/IPython	4.0.0
OpenSSL	/usr/lib/python2.7/dist-packages/OpenSSL	0.14
PIL	/usr/lib/python2.7/dist-packages/PIL	1.1.7
_dummy_thread	/home/docker/env/neurosci/local/lib/python2.7/site-packages/_dummy_thread	unknown
_markerlib	/home/docker/env/neurosci/local/lib/python2.7/site-packages/_markerlib	unknown
_thread	/home/docker/env/neurosci/local/lib/python2.7/site-packages/_thread	unknown
builtins	/home/docker/env/neurosci/local/lib/python2.7/site-packages/builtins	unknown
cffi	/usr/lib/python2.7/dist-packages/cffi	0.8.6
chardet	/usr/lib/python2.7/dist-packages/chardet	2.3.0
configparser	/home/docker/env/neurosci/local/lib/python2.7/site-packages/configparser	unknown
copyreg	/home/docker/env/neurosci/local/lib/python2.7/site-packages/copyreg	unknown
cryptography	/usr/lib/python2.7/dist-packages/cryptography	0.6.1
dateutil	/usr/lib/python2.7/dist-packages/dateutil	2.2
distutils	/home/docker/env/neurosci/lib/python2.7/distutils	2.7.9
encodings	/home/docker/env/neurosci/lib/python2.7/encodings	unknown
future	/home/docker/env/neurosci/local/lib/python2.7/site-packages/future	0.15.1
glib	/usr/lib/python2.7/dist-packages/glib	unknown
gobject	/usr/lib/python2.7/dist-packages/gobject	unknown
gtk	/usr/lib/python2.7/dist-packages/gtk-2.0/gtk	unknown
html	/home/docker/env/neurosci/local/lib/python2.7/site-packages/html	unknown
http	/home/docker/env/neurosci/local/lib/python2.7/site-packages/http	unknown
ipython_genutils	/home/docker/env/neurosci/local/lib/python2.7/site-packages/ipython_genutils	0.1.0
jinja2	/usr/lib/python2.7/dist-packages/jinja2	2.7.3
markupsafe	/usr/lib/python2.7/dist-packages/markupsafe	unknown
matplotlib	/usr/lib/python2.7/dist-packages/matplotlib	1.4.2

Platform information

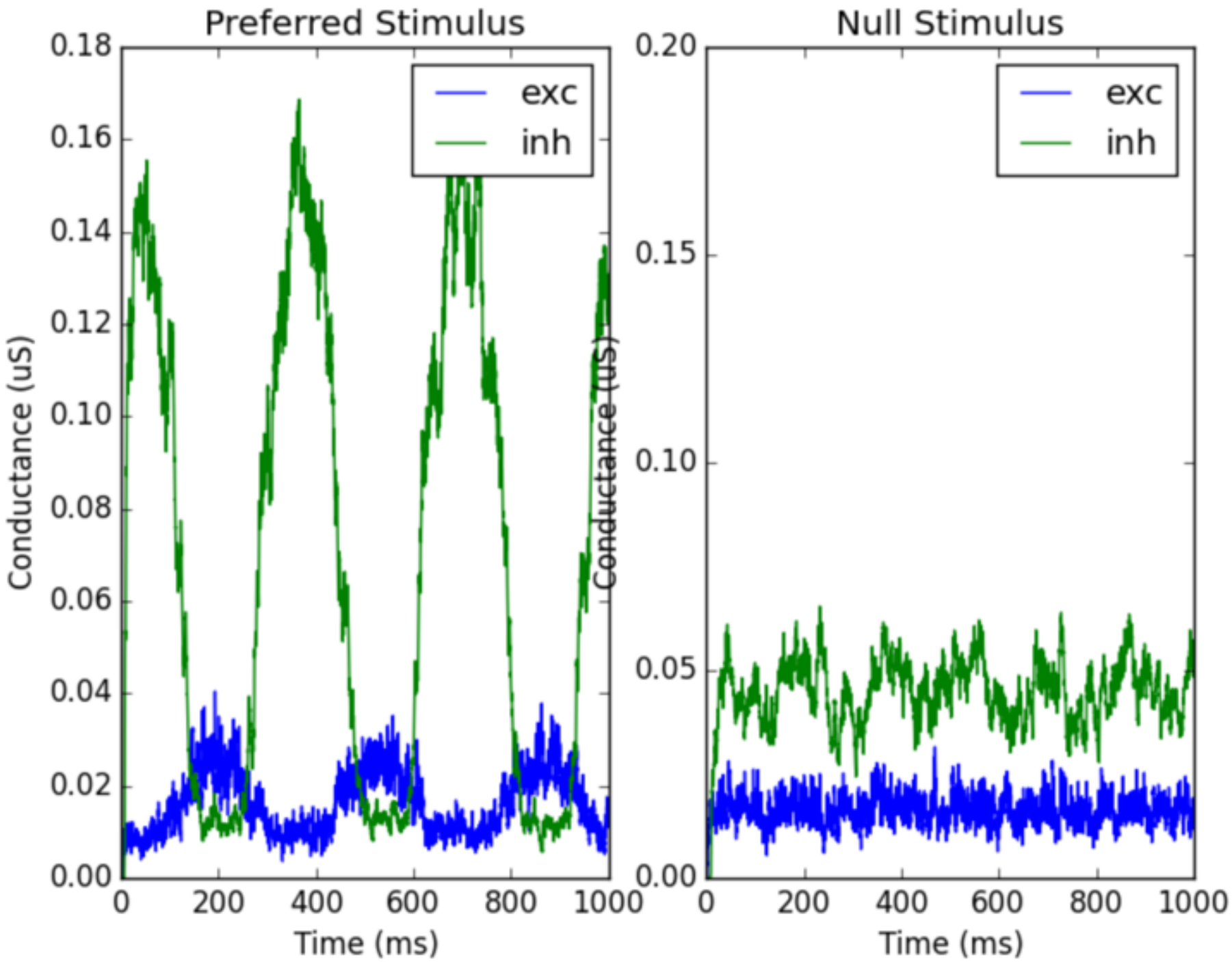
Name	IP address	Processor	Architecture		System type	Release	Version
110f3345e7e4	127.0.0.1	x86_64	64bit	ELF	Linux	3.18.11-tinycore64	#1 SMP Thu Apr 16 17:46:31 UTC 2015

Stdout & Stderr

```
-- N E S T --

Copyright (C) 2004 The NEST Initiative
Version 2.6.0 Sep 9 2015 16:24:11

This program is provided AS IS and comes with
NO WARRANTY. See the file LICENSE for details.

Problems or suggestions?
Website : http://www.nest-initiative.org
Mailing list: nest_user@nest-initiative.org

Type 'nest.help()' to find out more about NEST.
/home/docker/env/neurosci/local/lib/python2.7/site-packages/nest/hl_api.py:84: UserWarning:
ConvergentConnect is deprecated and will be removed in a future version of NEST.
Please use Connect instead!
For details, see the documentation at http://nest-initiative.org/Connection_Management
Creating connection from LGN_on_layer_0 to Excitatory layer
Creating connection from LGN_off_layer_0 to Excitatory layer
Creating connection from LGN_on_layer_0 to Inhibitory layer
Creating connection from LGN_off_layer_0 to Inhibitory layer
Creating connection from Inhibitory layer to Excitatory layer

Sep 10 13:59:38 Scheduler::simulate [Warning]:
The requested simulation time is not an integer multiple of the minimal
delay in the network. This may result in inconsistent results under the
following conditions: (i) A network contains more than one source of
randomness, e.g., two different poisson_generators, and (ii) Simulate is
called repeatedly with simulation times that are not multiples of the
minimal delay.
Construction time 102.237897158
Simulation time 116.262313128
```

output_data/20150910/troyer_plot7a_nest_20150910-140227.png 310f72f2561c0951ee5ddf335a7da5cf6983f5fe image/png 77.4 KB

Generated by 20150910-135750 on **10/09/2015** at **13:57:50**



Comparison of *20150910-135750* and *20150910-135054*

Code

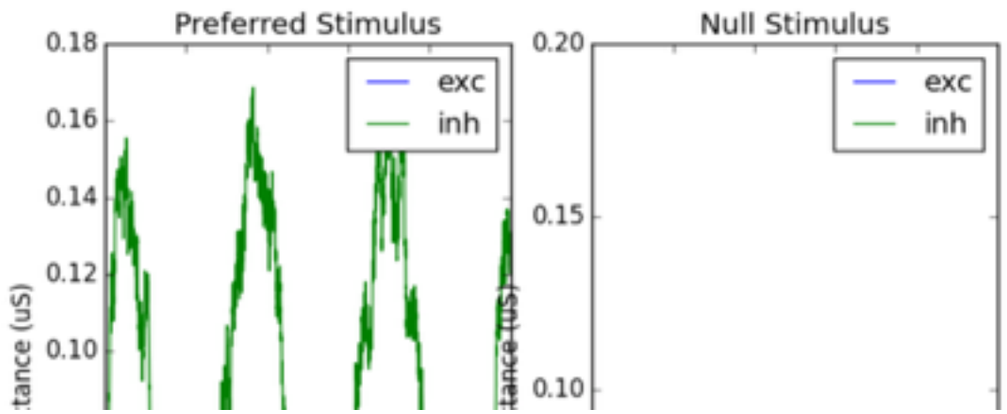
A
```
$ /home/docker/env/neurosci/bin/python troyer_plot7a.py --save-figures nest data/positions_on.cpickle data/positions_off.cpickle
data/0.5_spike_train_off_layer0.cpickle data/0.5_spike_train_off_layer1.cpickle data/0.5_spike_train_off_layer2.cpickle data/0.5_spike_train_off_layer3.cpickle
data/0.5_spike_train_on_layer0.cpickle data/0.5_spike_train_on_layer1.cpickle data/0.5_spike_train_on_layer2.cpickle data/0.5_spike_train_on_layer3.cpickle
```

B
```
$ /home/docker/env/neurosci/bin/python troyer_plot3a.py --save-figures nest data/positions_on.cpickle data/positions_off.cpickle
data/0.5_spike_train_off_layer0.cpickle data/0.5_spike_train_off_layer1.cpickle data/0.5_spike_train_off_layer2.cpickle data/0.5_spike_train_off_layer3.cpickle
data/0.5_spike_train_on_layer0.cpickle data/0.5_spike_train_on_layer1.cpickle data/0.5_spike_train_on_layer2.cpickle data/0.5_spike_train_on_layer3.cpickle
```

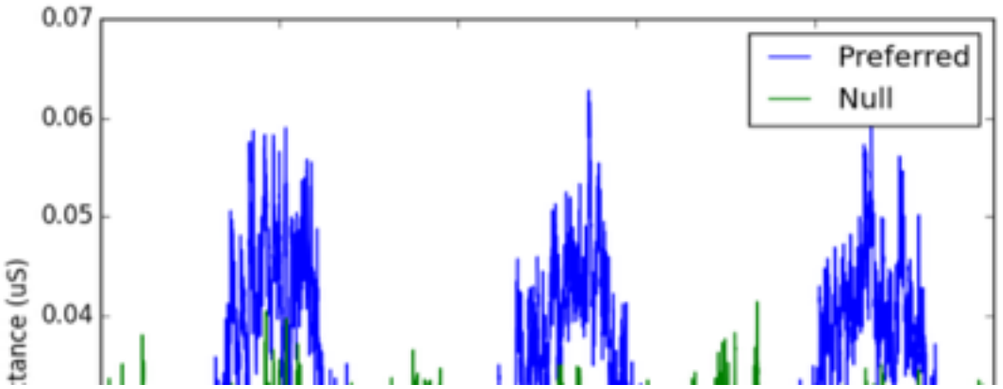
A B

Output data

A output_data/20150910/troyer_plot7a_nest_20150910-140227.png
310f72f2561c0951ee5ddf335a7da5cf6983f5fe image/png 77.4 KB



B output_data/20150910/troyer_plot3a_nest_20150910-135458.png
e4a7a497bf1d328cc419cf136adff897f3386520 image/png 80.3 KB



Linking to experiments from papers

```
\usepackage{sumatra}

Sed pater omnipotens speluncis abdidit atris,
hoc metuens, molemque et montis insuper altos
imposuit, regemque dedit, qui foedere certo
et premere et laxas sciret dare iussus habenas.
Ad quem tum Iuno supplex his vocibus usa est:

\begin{figure}[htbp]
\begin{center}
\smtincludegraphics[width=\textwidth,
                    digest=5ed3ab8149451b9b4f09d1ab30bf997373bad8d3]
                   {20150910-115649?troyer_plot1a}
\caption{Reproduction of \textit{cf} Troyer et al. Figure 1A}
\label{fig1a}
\end{center}
\end{figure}

'Aeole, namque tibi divom pater atque hominum rex
et mulcere dedit fluctus et tollere vento,
gens inimica mihi Tyrrhenum navigat aequor,
Ilium in Italiam portans victosque Penates:
incute vim ventis submersasque obrue puppes,
aut age diversos et disiice corpora ponto.
```

molemque et montis insuper altos imposuit, regemque dedit, qui foedere certo et premere et laxas sciret dare iussus habenas. Ad quem tum Iuno supplex his vocibus usa est:

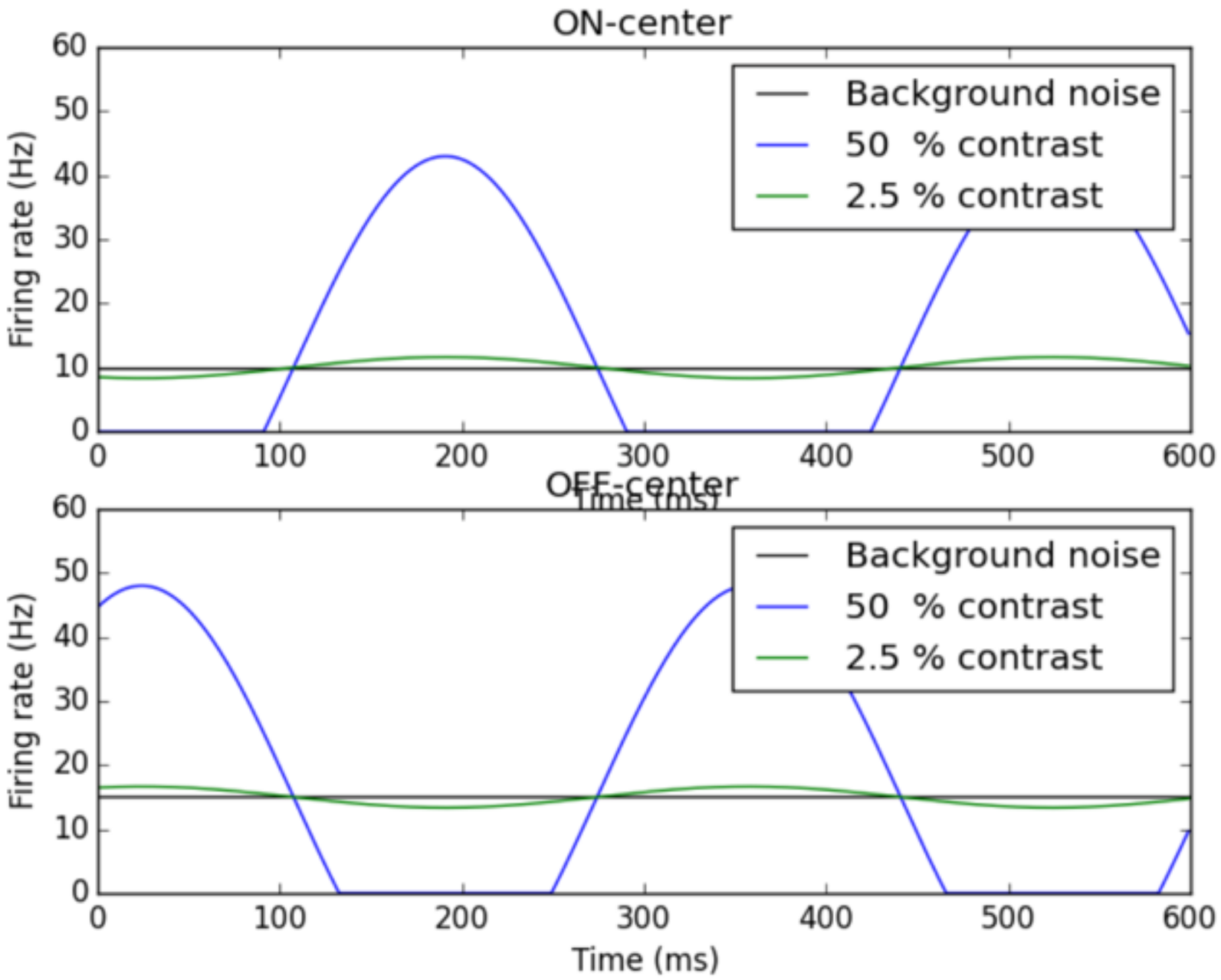


Figure 1: Reproduction of *cf* Troyer et al. Figure 1A

'Aeole, namque tibi divom pater atque hominum rex et mulcere dedit fluctus et tollere vento, gens inimica mihi Tyrrhenum navigat aequor, Ilium in Italiam portans

Run-time metadata capture

Advantages

- makes it possible to index, search, analyse the provenance information
- allows testing whether changing the hardware/software configuration affects the results
- works fine for distributed, parallel computations
- minimal changes to existing workflows

Disadvantages

- risk of not capturing *all* the context
- doesn't offer "plug-and-play" replicability like VMs, CDE

artefact capture

store the environment
in binary format

VM/Docker

CDE/Reprozip

pre-emptive capture

create a pre-defined environment,
always run in this enviroment

Docker/Vagrant

run-time capture

capture the environment at the
same time you run the experiment

Sumatra/noWorkflow/recipy

metadata capture

store the information needed
to recreate the environment

Recommendations

"Belt and braces"

Use both predefined environment and run-time capture