nest::

&

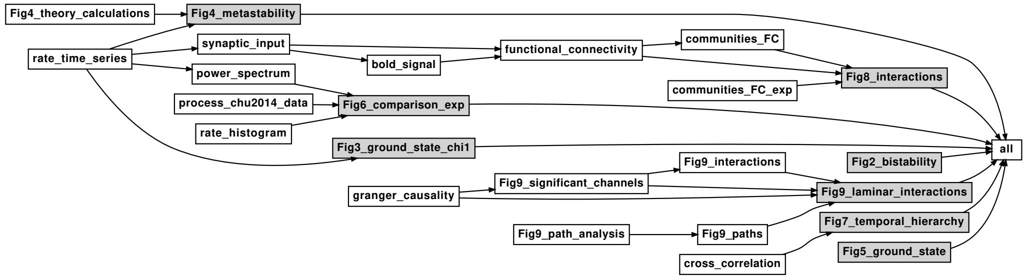
PART 2: SNAKEMAKE

Introduction to the simulation of structurally detailed large-scale neuronal networks

13 July 2019 | Alexander van Meegen, Dennis Terhorst | INM-6, IAS-6, INM-10; Jülich Research Centre

Motivation

- real world problems are complex (many steps, applications, scripts, ...)

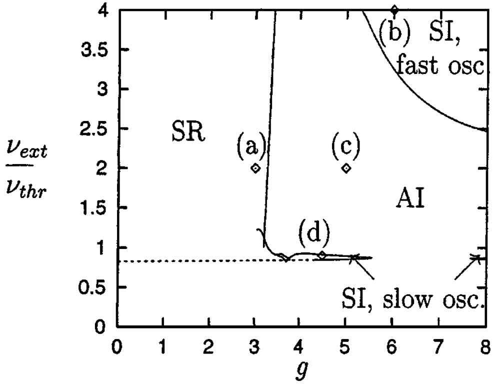


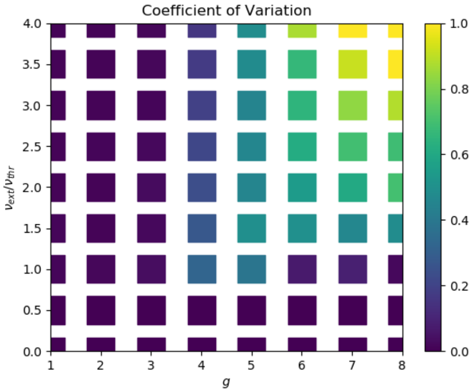
$\Rightarrow$ workflow management

- what leads to this complexity?
  - parameter scans
  - data preprocessing
  - analysis pipeline
  - ...
- reproducibility $\Rightarrow$ one-click to generate all results (figures, ...)

Example

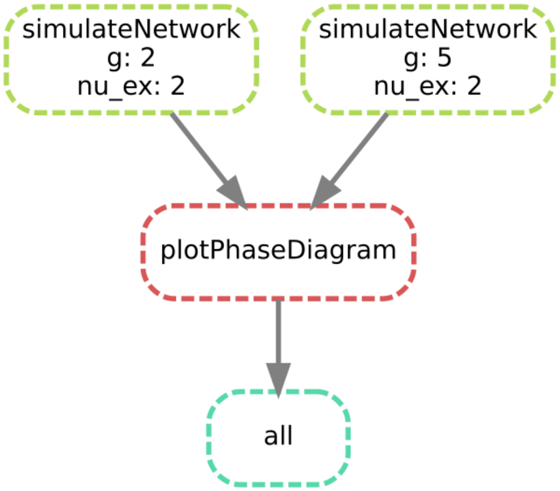
Brunel (2000) phase diagram:





$\Rightarrow$ parameter scan of *CV* $= \sigma_{\mathrm{ISI}} / \mu_{\mathrm{ISI}}$

Workflow


simulateNetwork
g: 2
nu_ex: 2
simulateNetwork
g: 5
nu_ex: 2
plotPhaseDiagram
all

Snakefile

```
rule all:
    input:
        'phase_diagram.png'

rule simulateNetwork:
    input:
        'brunel_parameters.yaml'
    output:
        'data/spikes_{g}_{nu_ex}.npy',
        'figures/raster_{g}_{nu_ex}.png'
    shell:
        'python3 scripts/simulateBrunel.py --g {wildcards.g} --nu_ex {wildcards.nu_ex} {input} {output}'

rule plotPhaseDiagram:
    input:
        expand('data/spikes_{g}_{nu_ex}.npy', g=G, nu_ex=NU_EX)
    output:
        'phase_diagram.png'
    shell:
        'python3 scripts/plotPhaseDiagram.py {output} {input}'
```

Parsing arguments

- we want to pass parameters from snakemake to the Python script:

```
python3 scripts/plotPhaseDiagram.py {input} {output}
```

- solution (here): the docopt package

```
"""Plot the phase diagram of the Brunel network.

Usage:
    plotPhaseDiagram.py [options] <plotfile> <spikefile>...

Arguments:
    plotfile     Output file for plot.
    spikefile    Input file(s) with spike data.

Plotting options:
    --markersize=<markersize>   Markersize [default: 500]
"""
from docopt import docopt
args = docopt(__doc__)
print(args['<plotfile>'])
print(args['--markersize'])
```

Further considerations

- important for workflow: standardized output

```
plt.savefig(args['<plotfile>'])
```

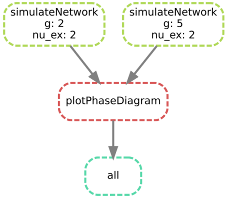
- best practice: modular structure of Python scripts

```
def simulateBrunel(simtime, dt, network_config):
                # do magic here ...
    return (ids_e, times_e), (ids_i, times_i)


if __name__ == '__main__':
    from docopt import docopt
    # parse command line parameters
    args = docopt(__doc__)
    # simulate network (magic parameters network_config)
    (ids_e, times_e), _ = simulateBrunel(
        simtime=float(args['--simtime']), dt=float(args['--dt']),
        network_config=network_config
    )
    # save spikes
    np.save(args['<spikefile>'], [ids_e, times_e])
```

Advanced features

- one line graph visualization: `snakemake --dag | dot | display`



- cluster support (e.g. SLURM)

```
snakemake --jobs 100 --cluster-config cluster.json
          --cluster "sbatch --job-name={cluster.job-name} --ntasks={cluster.ntasks}
                                --time={cluster.time}"
```

- integrated package management (e.g. rule-specific conda environments)

```
snakemake --use-conda
```

Hands on

- all files in `part2_snakemake`
- detailed instructions in `part2_snakemake/README.md`

- enjoy and feel free to ask questions :)

- snakemake not installed?
  ```
  pip3 install --user snakemake
  ```