Tools for Transparency and Replicability of Simulation in Archaeology

Mark E. Madsen and Carl P. Lipo

University of Washington, Seattle

California State University at Long Beach

Session: **Open methods in archaeology: how to encourage reproducible research as the default practice**

Why We Simulate

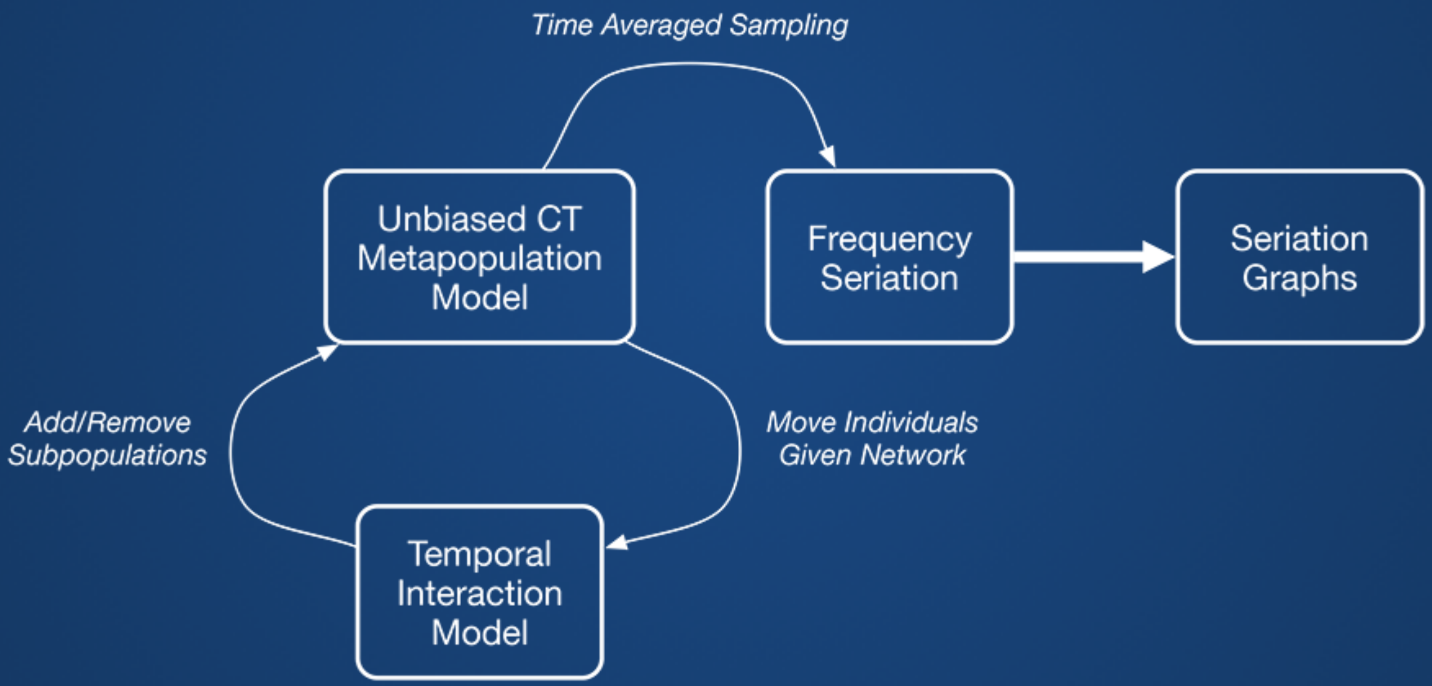
- Express models of social and evolutionary dynamics
- Understand model outcomes
- Predict archaeologically visible patterns
- Compare our data to the patterns

Doing Simulation Well Is Hard

- Difficult to demonstrate correctness
- Hard to manage data, software, parameters
- Hard to separate exploration from rigorous experimentation

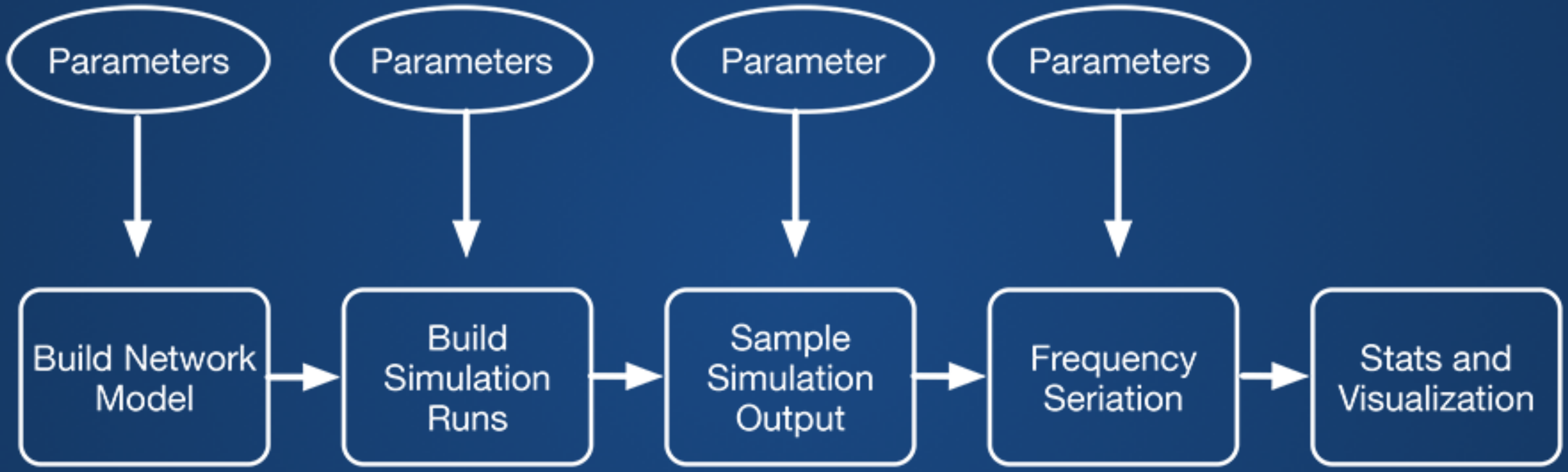
Example Project: SeriationCT

https://github.com/mmadsen/seriationct



Time Averaged Sampling
Unbiased CT Metapopulation Model
Frequency Seriation
Seriation Graphs
Add/Remove Subpopulations
Move Individuals Given Network
Temporal Interaction Model

Workflow Pipeline



- Each stage generates intermediate data in DB or files
- Each stage may have input parameters

Traceability

Consistency

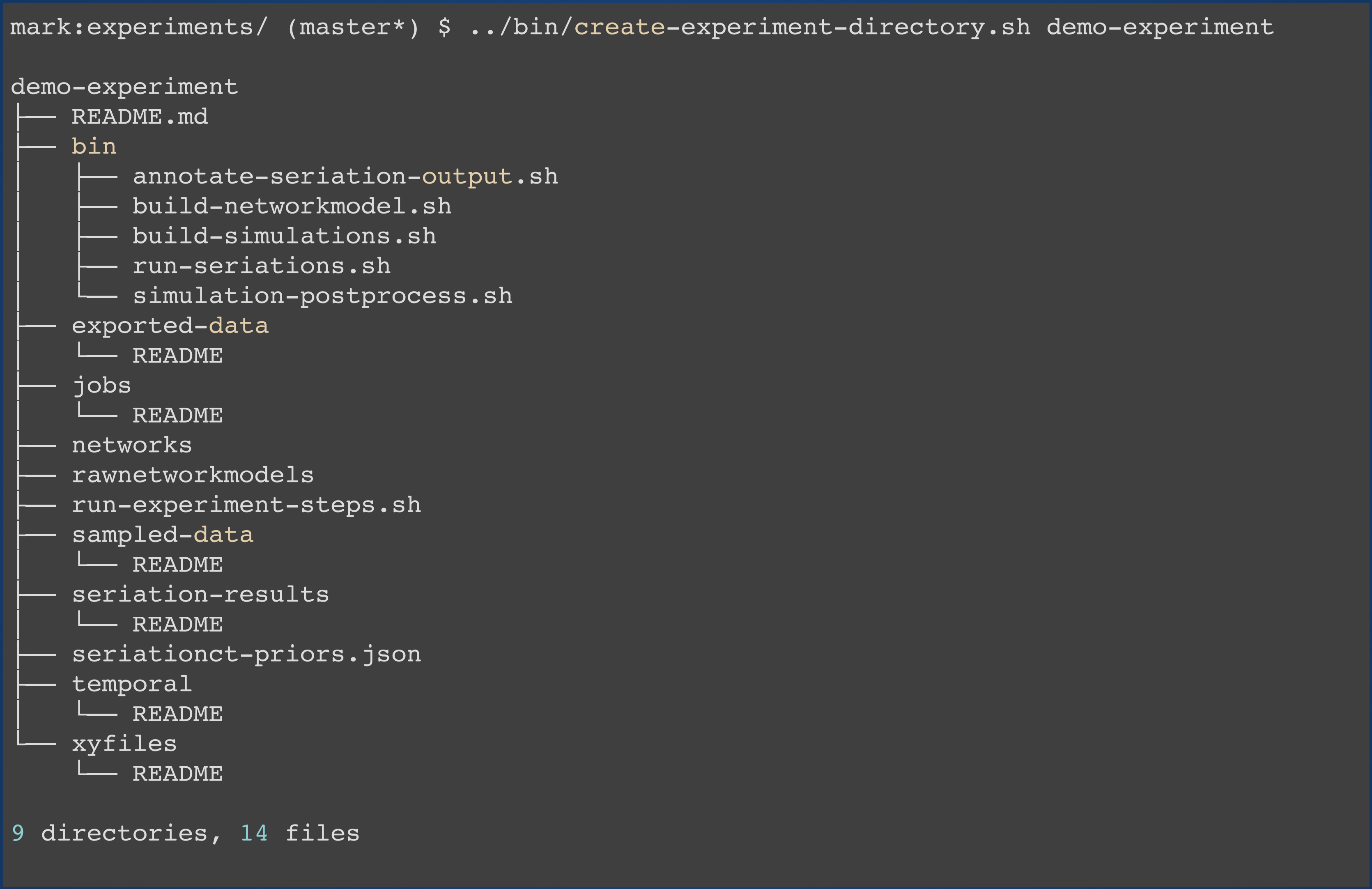
Replicability

Best Practices

- Everything lives in a revision control system (Git/Github, Subversion, Mercurial)
- Exploration is separated from production simulation work
- Production work is templated and scripted
- As a result, GUI tools are for exploration, not production

- Each unit of work is an "experiment"
- Experiments start with a template
- Graphics and tables are generated, not hand made

Creating New Experiment

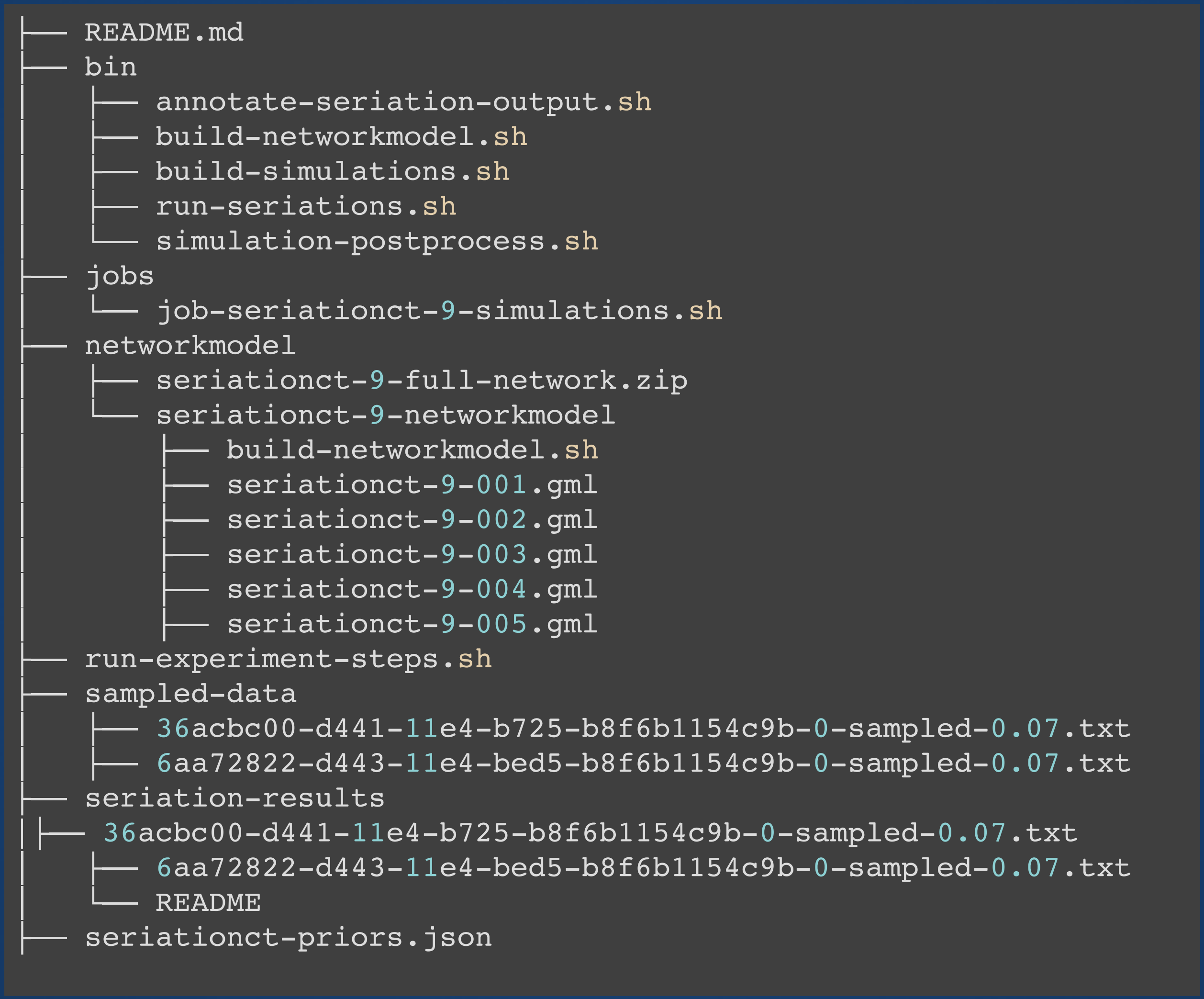


```
mark:experiments/ (master*) $ ../bin/create-experiment-directory.sh demo-experiment

demo-experiment
├── README.md
├── bin
│   ├── annotate-seriation-output.sh
│   ├── build-networkmodel.sh
│   ├── build-simulations.sh
│   ├── run-seriations.sh
│   └── simulation-postprocess.sh
├── exported-data
│   └── README
├── jobs
│   └── README
├── networks
├── rawnetworkmodels
├── run-experiment-steps.sh
├── sampled-data
│   └── README
├── seriation-results
│   └── README
├── seriationct-priors.json
├── temporal
│   └── README
└── xyfiles
    └── README

9 directories, 14 files
```

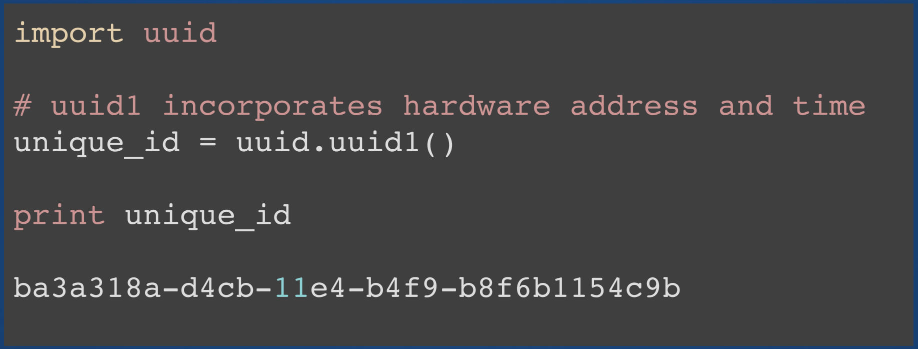
Experiment In Progress...



```
├── README.md
├── bin
│   ├── annotate-seriation-output.sh
│   ├── build-networkmodel.sh
│   ├── build-simulations.sh
│   ├── run-seriations.sh
│   └── simulation-postprocess.sh
├── jobs
│   └── job-seriationct-9-simulations.sh
├── networkmodel
│   ├── seriationct-9-full-network.zip
│   └── seriationct-9-networkmodel
│       ├── build-networkmodel.sh
│       ├── seriationct-9-001.gml
│       ├── seriationct-9-002.gml
│       ├── seriationct-9-003.gml
│       ├── seriationct-9-004.gml
│       ├── seriationct-9-005.gml
├── run-experiment-steps.sh
├── sampled-data
│   ├── 36acbc00-d441-11e4-b725-b8f6b1154c9b-0-sampled-0.07.txt
│   ├── 6aa72822-d443-11e4-bed5-b8f6b1154c9b-0-sampled-0.07.txt
├── seriation-results
│ ├── 36acbc00-d441-11e4-b725-b8f6b1154c9b-0-sampled-0.07.txt
│   ├── 6aa72822-d443-11e4-bed5-b8f6b1154c9b-0-sampled-0.07.txt
│   └── README
├── seriationct-priors.json
```

Universally Unique Identifiers

Internet RFC 4122:
https://www.ietf.org/rfc/rfc4122.txt



```
import uuid

# uuid1 incorporates hardware address and time
unique_id = uuid.uuid1()

print unique_id

ba3a318a-d4cb-11e4-b4f9-b8f6b1154c9b
```

- Component of all file names
- Field in all database records
- Primary means of tying data elements together

Random Number Generation

- Need the RNG seed for replicability
- Want good randomness during simulation

- Generate seed randomly for each simulation run
- Store the generated seed with each data record

Simulation Metadata

```
{
    "simulation_run_id" : "urn:uuid:eaf71706-ce8c-11e4-a9ac-b8f6b1154c9b",
    "random_seed" : 2127774500,
    "elapsed_time" : 257.4463579654694,
    "experiment_name" : "seriationct-1",
    "full_command_line" : "sim-seriationct-networkmodel.py -mf 0.0938
    --popsize 250 --nm hier-1.zip"
}
```

Simulation Output Data

```
{
    "_id" : ObjectId("5514e910544bd6744cae8aec"),
    "simulation_run_id" : "urn:uuid:36acbc00-d441-11e4-b725-b8f6b1154c9b",
    "random_seed" : 1601673696,
    "replication" : 0,
    "class_freq" : {
        "0-3-4" : 0.6857142857142857,
        "2-4-1" : 0.1428571428571428,
        "0-4-4" : 0.1714285714285714
    },
    "simulation_time" : 3000,
    "subpop" : "assemblage-33-6",
    "mutation_rate" : 0.00668494110834,
    "population_size" : 250,
    "class_richness" : 3
}
```

Other Tools

Sumatra

http://neuralensemble.org/sumatra/

Numerical analysis or simulation project tracking and replicability tool

Lancet

http://ioam.github.io/lancet/

Strong parameter management and experiment execution library

Where We're Headed

- Sumatra needs files as "data" capture, extend to handle database as data store, requires archival scheme
- Lancet replacing our simple execution scripts and parameter JSON files
- Extend Sumatra and Lancet to include real UUIDs and capture seeds with data

Thank You