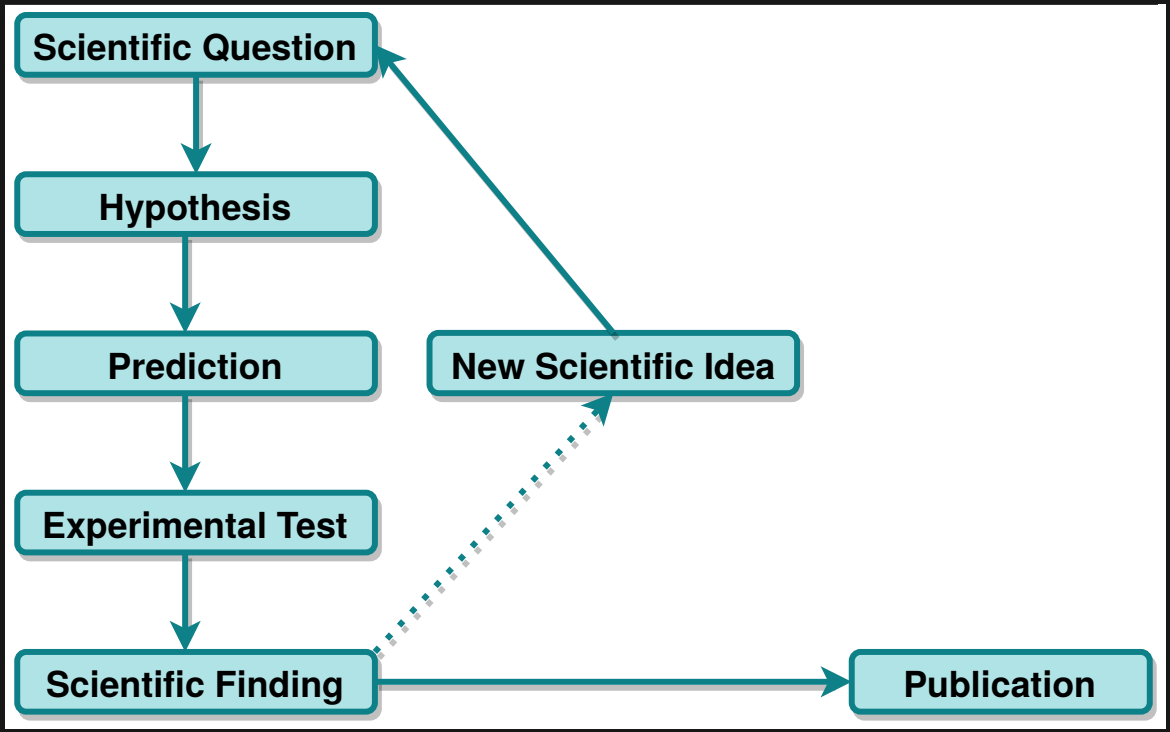
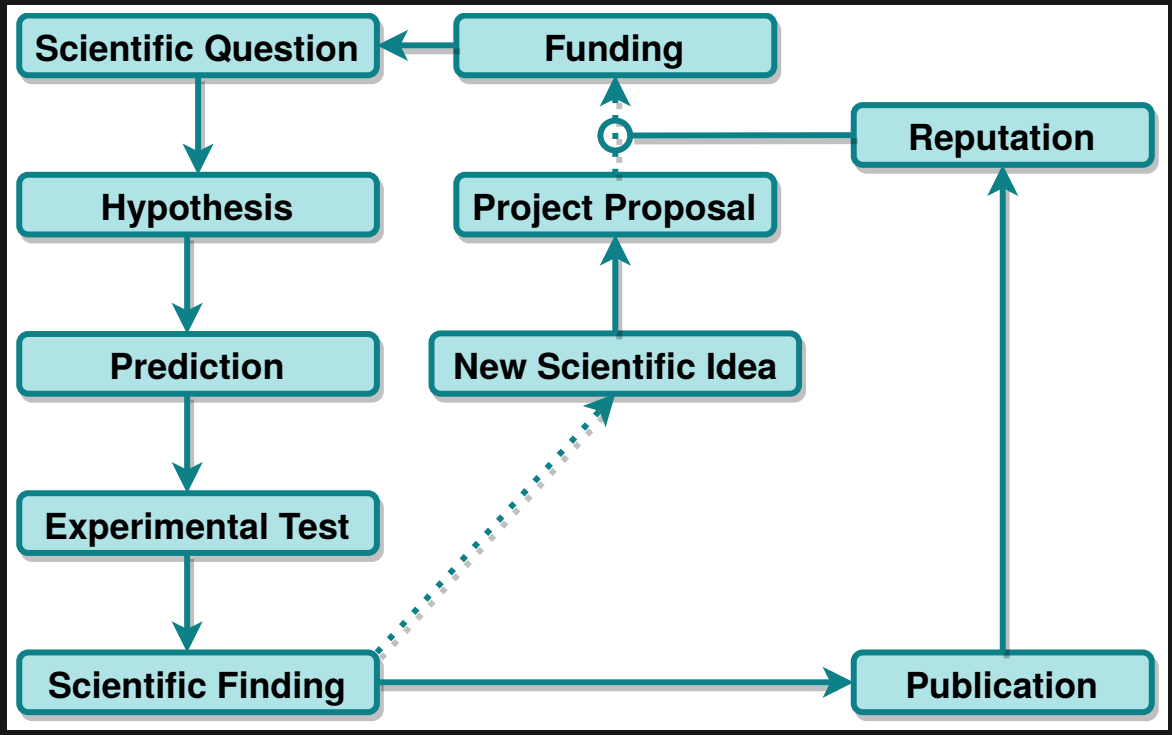
SCIENCE

Aim: understand <system>* better (* molecule, species, earth, universe, ...)



SCIENTIFIC REALITY



WHAT IS NOT PART OF THIS

TIME FOR IMPROVEMENTS NOT DIRECTLY CONTRIBUTING TO THE SCIENTIFIC PROGRESS, E.G.

- infrastructure software development
- refactoring / restructuring of code
- long term maintenance of code

EDUCATION OF SOFTWARE DEVELOPMENT SKILLS

- best practices for code style
- quality assurance
  - test driven development
  - continuous integration
- version control
- validation

MONEY FOR

- software development training courses
- non-scientific software developers
- software infrastructure (IT, self hosted services, ...)

ADDITIONAL ISSUES

TRUST

- for **seemingly** small projects software is quickly self-implemented
- for **complex** projects commercial software **seems** more reliable

TRUST

- publication comes first, software release *maybe* later
- making errors is taboo, reputation issue
- publishing code pushes scientific progress
  - Pioneering in code publication
    Izhikevich (2006) Polychronization: Computation with Spikes
  - Reimplementation and examination
    Pauli et al. (2018) Reproducing Polychronization: A Guide to Maximizing the Reproducibility of Spiking Network Models

DEDICATION

- software development is a side occupation
  - no extended time reserved for this
  - not continuous task, but on demand
  - has a low priority compared to scientific findings and publications
- rapid changes in staff & small / not clearly defined user base
- no dedicated team per software project
  funding usually supports individuals
- focus on numeric precision

EXAMPLES OF OPEN SOURCE SCIENTIFIC SOFTWARE PROJECTS

ODMLTABLES

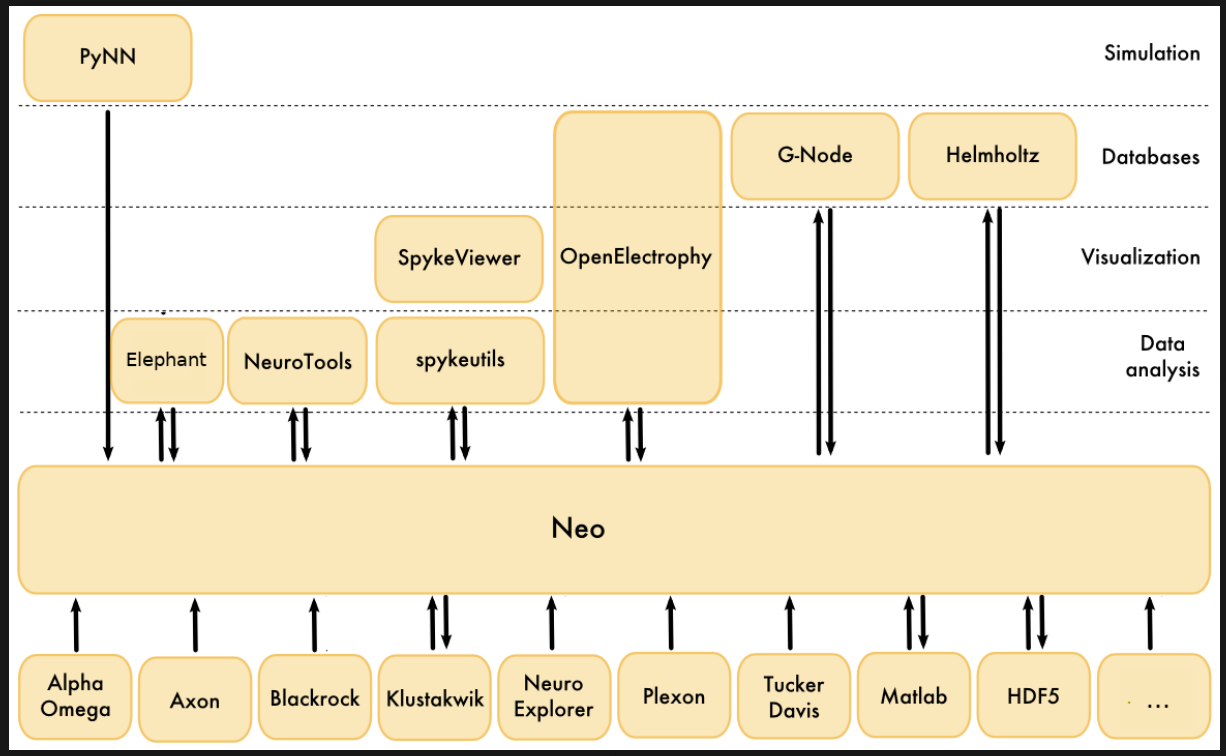
- extends usability of open metadata markup language (odML)
- interface between common laboratory formats (xls/csv) and odML
- additional utility functionalities

ODMLTABLES

- initiated in 2015
- 1 developer
- 0 contributors
- 1-10 users
- 1 publication

NEO

- interface between 30+ electrophysiology dataset formats (proprietary & open)
- standardized data representation



NEO

- initiated in 2009
- successor of previous electrophysiology data handling packages
- used in 130+ repositories
- 3 developers (3 labs)
- 55 contributors (8 active)
- ~100 direct users
- presented at scientific conferences & workshops

NEST

- simulator for spiking neural network models
- scales from laptops to exascale computers [Jordan et al. 2018]
- community-standard

NEST

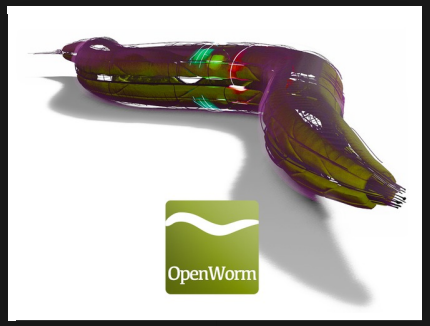
- initiated in 1993
- owned by *The Neural Simulation Technology Initiative*
- > 10 developers (5 active)
- 1 funded position for project documentation
- 79 contributors (>10 active)
- dedicated website (news, announcements, tutorials, videos, brochures)
- publication list (user / software development)
- dedicated nest conferences & user workshops & hackathons

THERE'S A LOT MORE

Neuroscience

Human Brain Project

Biology



Physics

ROOT
Data Analysis Framework

... and a large and growing number of smaller projects

WHAT CAN YOU DO?

... AS SOFTWARE DEVELOPER

- talk to scientists to discover new interesting problems
- get involved in existing scientific projects and provide feedback
- make your documentation readable by non-experts, *'installation for dummies'*
- advertise existing software within potential user community (conferences, workshops)

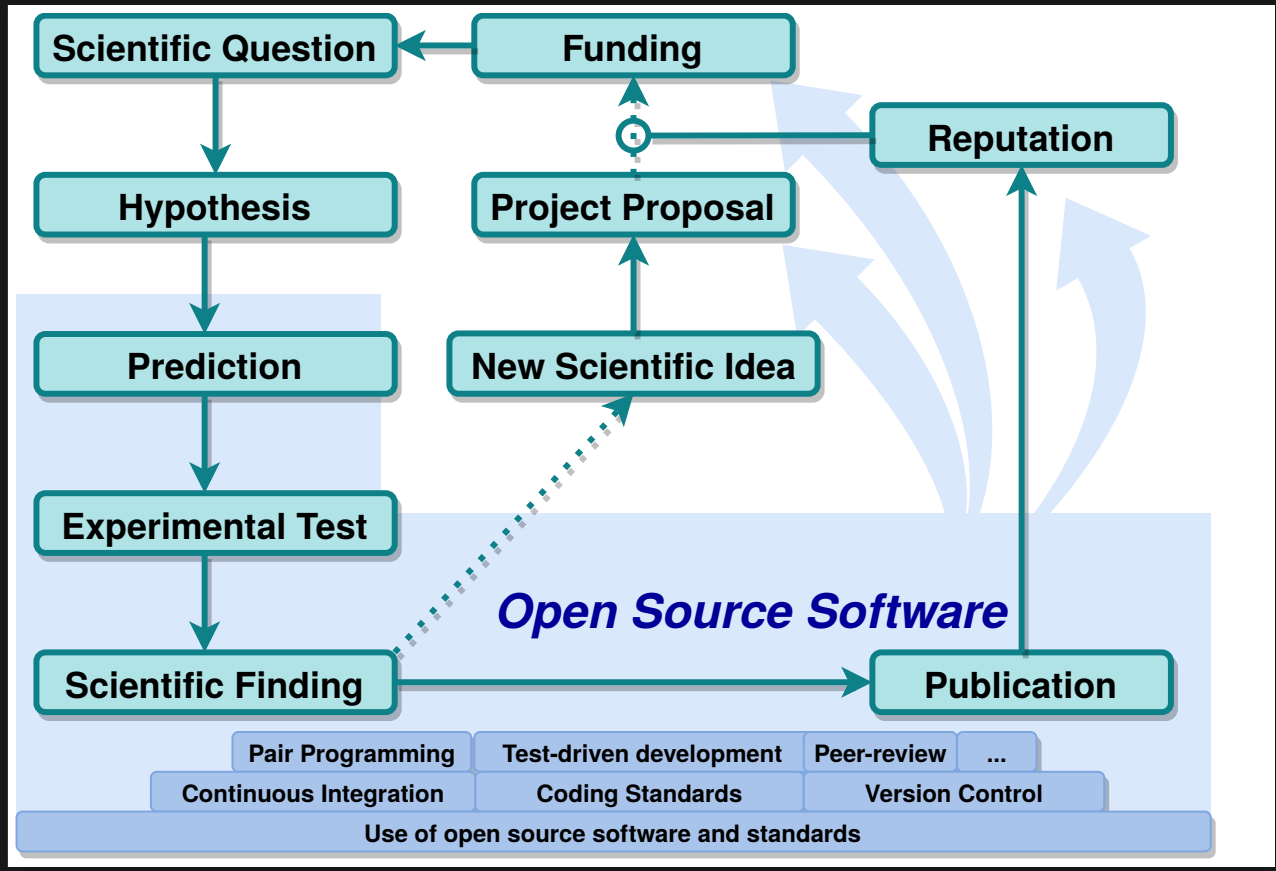
... AS SCIENTIFIC OPEN SOFTWARE PROJECT

- involve a large user (and potential contributor) base
- simplify usage and contribution on all levels
  - user & developer guide
  - tutorials
  - solve simple issues first
  - define project standards and contribution guide
- advertise within potential user community

... AS SCIENTIST

- use existing open source tools, don't start from scratch
- make sure your software outlives your career
  - bestpractices.coreinfrastructure.org
- create links to other packages
- integrate your project into larger framework
- consider software development aspects from the beginning

SCIENTIFIC VISION



THANK YOU!

BACKUP SLIDES

RECOMMENDATIONS FOR INSTITUTES

- provide kickstart in basic software development techniques
  - version control, coding standards, CI
  - pair programming, test driven development, code review
  - ~~scrum~~, kanban
- have a dedicated software coordinator
- define project standards
- organize hackathons / topic weeks / workshops

ADDITIONAL REFERENCES

USING MARKDOWN AND PANDOC FOR PRESENTATIONS

- Creating slides with pandoc
- Example markdown presentation
- Using pandoc to create reveal.js slides
- From markdown to manuscripts
- Pandoc examples

DIFFERENCES ACADEMIA & BUSINESS

- Academia
  - reputation of an individual counts
  - longer software development cycles
  - main focus: scientific results & paper publications
- Business
  - reputation of company counts
  - scrum style software development cycles
  - main focus: product development & marketing

Polychronization: Computation with Spikes

Eugene M. Izhikevich
Eugene.Izhikevich@nsi.edu
The Neurosciences Institute, 10640 John Jay Hopkins Drive, San Diego, CA 92121, U.S.A.

We present a minimal spiking network that can polychronize, that is, exhibit reproducible time-locked but not synchronous firing patterns with millisecond precision, as in synfire braids. The network consists of cortical spiking neurons with axonal conduction delays and spike-timing-dependent plasticity (STDP); a ready-to-use MATLAB code is included. It exhibits sleeplike oscillations, gamma (40 Hz) rhythms, conversion of firing rates to spike timings, and other interesting regimes. Due to the interplay between the delays and STDP, the spiking neurons spontaneously self-organize into groups and generate patterns of stereotypical polychronous activity. To our surprise, the number of coexisting polychronous groups far exceeds the number of neurons in the network, resulting in an unprecedented memory capacity of the system. We speculate on the significance of polychrony to the theory of neuronal group selection (TNGS, neural Darwinism), cognitive neural computations, binding and gamma rhythm, mechanisms of attention, and consciousness as "attention to memories."

Reproducing Polychronization: A Guide to Maximizing the Reproducibility of Spiking Network Models

Robin Pauli[1†], Philipp Weidel[1†], Susanne Kunkel[2,3] and Abigail Morrison[1,4]*

[1] Institute of Neuroscience and Medicine (INM-6) and Institute for Advanced Simulation (IAS-6) and JARA BRAIN Institute I, Jülich Research Centre, Jülich, Germany, [2] Faculty of Science and Technology, Norwegian University of Life Sciences, Ås, Norway, [3] Department of Computational Science and Technology, School of Computer Science and Communication, KTH Royal Institute of Technology, Stockholm, Sweden, [4] Institute of Cognitive Neuroscience, Faculty of Psychology, Ruhr-University Bochum, Bochum, Germany

Any modeler who has attempted to reproduce a spiking neural network model from its description in a paper has discovered what a painful endeavor this is. Even when all parameters appear to have been specified, which is rare, typically the initial attempt to reproduce the network does not yield results that are recognizably akin to those in the original publication. Causes include inaccurately reported or hidden parameters (e.g., wrong unit or the existence of an initialization distribution), differences in implementation of model dynamics, and ambiguities in the text description of the network experiment. The very fact that adequate reproduction often cannot be achieved until a series of such causes have been tracked down and resolved is in itself disconcerting, as it reveals unreported model dependencies on specific implementation choices that either were not clear to the original authors, or that they chose not to disclose. In either case, such dependencies diminish the credibility of the model's claims about the behavior of the target system. To demonstrate these issues, we provide a worked example of reproducing a seminal study for which, unusually, source code was provided at time of publication. Despite this seemingly optimal starting position, reproducing the results was time consuming and frustrating. Further examination of the correctly reproduced model reveals that it is highly sensitive to implementation choices such as the realization of background noise, the integration timestep, and the thresholding parameter of the analysis algorithm. From this process, we derive a guideline of best practices that would substantially reduce the investment in reproducing neural network studies, whilst simultaneously increasing their scientific quality. We propose that this guideline can be

OPEN ACCESS

Edited by:
Sharon Crook,
Arizona State University, United States

Reviewed by:
Antonio C. Roque,
Universidade de São Paulo, Brazil
Thomas Nowotny,
University of Sussex, United Kingdom

****Correspondence:***
Robin Pauli
r.pauli@fz-juelich.de

† These authors have contributed equally to this work.

Received: *01 March 2018*
Accepted: *26 June 2018*
Published: *03 August 2018*

Citation:
Pauli R, Weidel P, Kunkel S and

STAGES OF SCIENTIFIC SOFTWARE

1. custom code for experiment specific task (e.g. experiment control)
    - for current use only
    - no documentation
    - no reuse possible
    - not maintainable
    - not shareable

GROWING NUMBER OF SCIENTIFIC OPEN SOFTWARE PROJECTS & CODE

- Increasing awareness of software as scientific basis Katerbow & Feulner (2018): Recommendations on the Development, Use and Provision of Research Software, Research Software Working Group of the Alliance of German Science Organisations
- Funding initiatives will require code & software publication