# Publication of data and computations with ActivePapers

Konrad HINSEN

Centre de Biophysique Moléculaire, Orléans, France
and
Synchrotron SOLEIL, Saint Aubin, France

December 2, 2016

# Publication of computational results

- Archiving → preservation
- Make sense to future scientists → documentation
- Embedding into the scientific record → references, provenance

# ActivePapers

## Idealist phase (2010-2011)

- Do the best possible job with available technology.
- ... even if this makes it difficult to use.
- ActivePapers JVM edition
- Finalist in the Executable Papers Challenge at ICCS 2011

## Pragmatist phase (2012-)

- Compromises to make it usable with today's software.
- Priority: biomolecular simulations
- ActivePapers Python edition

Full story: K. Hinsen, F1000Research 2015 **3** 289

# Biomolecular simulation: data lifecycle

## Small input data

- Experimental data
- Software
- Parameters

## Large intermediate data

- Molecular Dynamics trajectories
- Typically 1 GB to 100 GB
- Can be recomputed, but at high cost ($\approx$ weeks)

## Mid-size output data

- Analyses of MD trajectories
- Typically 1 MB to 100 MB
- Can be recomputed if the MD trajectory is available.

# Biomolecular simulation: knowledge lifecycle

- MD trajectories: few years
- Analyses of MD trajectories: few decades
- Longer timescales: models derived from concrete results

# Biomolecular simulation: data models

## Simple data ($N - d$ arrays)

- Time series
- Images
- Volumetric data

## Complex structured data

- Molecular structures
- Relations between molecular structures
- Relations between other types of data

# Biomolecular simulation: data models

## Simple data ($N - d$ arrays)

- Time series
- Images
- Volumetric data

## Complex structured data

- Molecular structures
- Relations between molecular structures
- Relations between other types of data

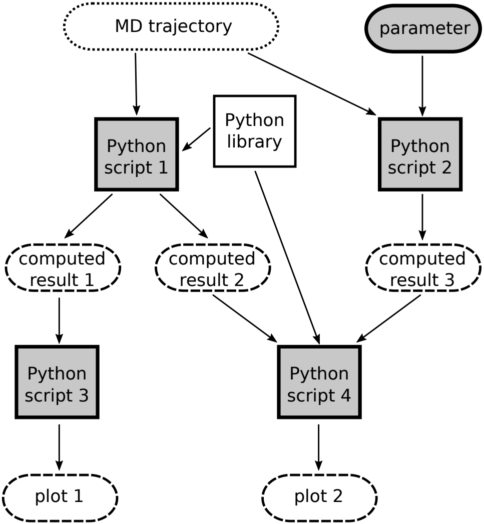
**No standard data models for structured data.**

→ Store the program that generated the data for documentation.

# ActivePapers in a nutshell

- ActivePaper: an HDF5 file that respects the ActivePapers conventions
- Can contain any HDF5 data
- Code: Python modules and scripts
- References permit reuse of data/code from published ActivePapers
- Publication on DOI-granting servers (Zenodo, figshare)
- Dependency graph is stored as HDF5 metadata
- Recomputable data can be deleted to save space

# A dependency graph example



# ActivePapers in practice

- Used in five research projects
- 12 ActivePapers published on Zenodo
- 5 ActivePapers published on figshare
- Two types of published ActivePapers:
  - Software libraries
  - Data plus scripts

# Open problems

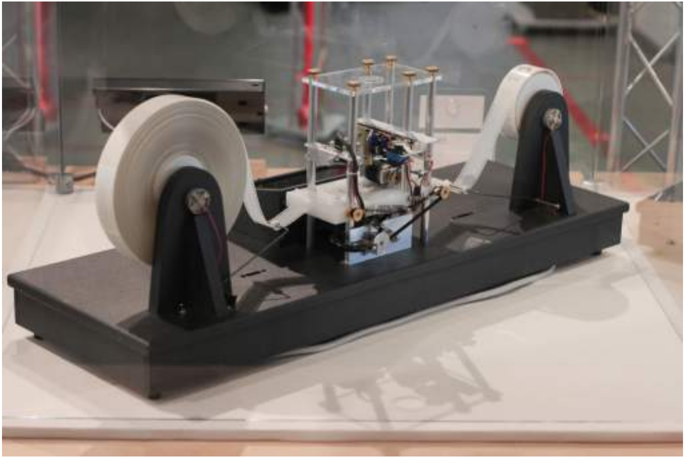
## Code representation

- "Python only" is very restrictive.
- More general code representations (x86 code) create a risk for users.
- The JVM is a nice compromise but it's not popular in scientific computing.
- Fundamental issue: **no stable code platform**

## Tool support

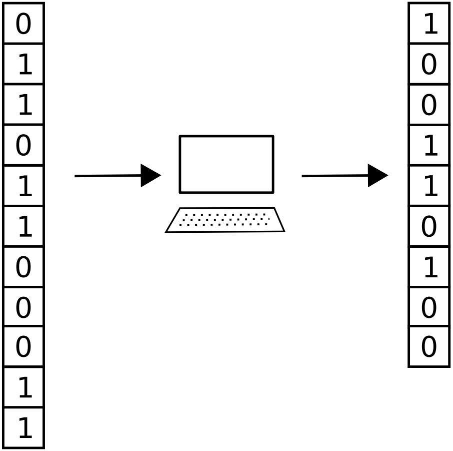
- Command-line `aptool` - minimalist but functional.
- Users need a good understanding of the ActivePapers architecture.

# Remember Turing?



By Rocky Acosta (Own work) [CC BY-SA 3.0], via Wikimedia Commons

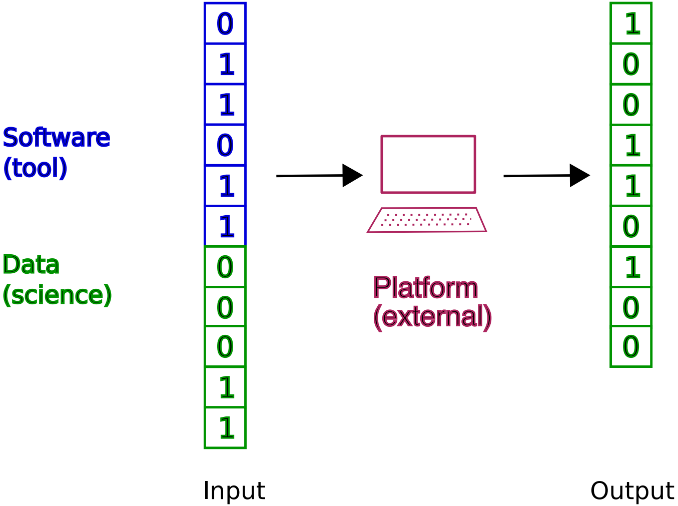
# Computation in science



Input

Output

# Computation in science



Software
(tool)

Data
(science)

0
1
1
0
1
1
0
0
0
1
1

Platform
(external)

1
0
0
1
1
0
1
0
0

Input

Output

# Software vs. data

## In computation

- Impossible to clearly define a distinction
- Software → semantics of data
- Platform → semantics of software

## In science

- Real distinction: tool vs. knowledge
- Knowledge in both software and data

Never store complex data without the software that produced it!