An Overview of Platforms for Reproducible Research and New Ways of Publications

Miguel Colom$^1$, Bertrand Kerautret$^2$
August 20, 2018, Beijing, 2nd RRPR Workshop

$^1$CMLA, ENS Paris-Saclay, France

$^2$LORIA, Université de Lorraine, France
1. Introduction

   1.1 Quick overview of RR

   1.2 Benefits of RR

2. Main Platforms for Reproducible Research

   Galaxy, IPython, Jupyter, Code Ocean, Research Compendia, RunMyCode, DAE, IPOL

3. New Ways of Publications

   3.1 IPOL Journal

   3.2 ReScience journal

   3.3 JOSS Journal

4. Conclusion
1. Introduction
1.1 Quick overview of RR

- **RR redefines** the result of the research. Not just a paper, but also all the procedures needed to obtain the same published results.

- Which items? The **article** itself, the **source** code, and the **data**.

- Definition of a “**reproducible scientific publication**” given by Claerbout and followed also by Buckheit and Donoho [Buckheit & Donoho 95]: "An article about computational science in a scientific publication is not the scholarship itself, it is merely advertising of the scholarship. The actual scholarship is the complete software development environment and the complete set of instructions which generated the figures."
1.2 Benefits of RR

- **Less prone to errors**, since all the methods are **clear, public, and therefore auditable**.

- **Good practices. Reliable science.** The descriptions **match** what the published descriptions.

- The descriptions, methodology, source code, and data is **available** to the scientific community.

- **Comparison** of methods easier (or even possible!).

- For authors, possibility of publishing research of **high quality**.
  Counterexample: the case of a copy-pasted image in a disputed biology article.

- Also for authors: increase of the **visibility** of the publications.
2. Main Platforms for Reproducible Research
2. Main Platforms for Reproducible Research

- Galaxy - https://galaxyproject.org
- IPython - https://ipython.org
- Jupyter - http://jupyter.org
- Code Ocean - https://codeocean.com
- Research Compendia - http://researchcompendia.science
- DAE - http://dae.cse.lehigh.edu/DAE
- IPOL - https://www.ipol.im
## 2. Main Platforms for Reproducible Research: Galaxy

**Description:**

- Platform for *genomic research*.
- It makes available *tools* which can be used by *non-expert* users too.
- Galaxy defines a *workflow* as a *reusable templates* which contains different algorithms applied to the input data.
- In order to achieve *reproducibility* the system stores:
  - the *input dataset*,
  - the *tools and algorithms* which were applied to the data along the *chain*,
  - the *parameters*,
  - the *output* result.

[https://galaxyproject.org](https://galaxyproject.org)
2. Main Platforms for Reproducible Research: IPython

**Description:**

- **Generic tool** that can be used for Reproducible Research.
- **Mature tool**: created in 2001.
- Allows creating **reproducible articles** by not only editing text in the notebook, but allowing **code execution** and creating figures *in situ*.
- Follows closely the definition of a “**reproducible scientific publication**” of Claerbout, Buckheit, and Donoho [Buckheit & Donoho 95].

https://ipython.org
2. Main Platforms for Reproducible Research: Jupyter

Description:

- **Spin-off of IPython** in 2014.
- **Main goal:** separate the Python language used in IPython **from all the other functionalities** needed to run the notebooks (for example, the notebook format, the web framework, or the message protocols).
- **Languages:** execution kernels in Jupyter.
- Nowadays it supports **more than 40 languages** that can be used as kernels.

http://jupyter.org
2. Main Platforms for Reproducible Research: Code Ocean

Description:

- Stated in 2014 as part Runway Startup Postdoc Program at the Jacobs Technion Cornell Institute.
- Sponsored by IEEE.
- Defined by themselves as a computational reproducibility platform.
- Not a journal itself. It only runs code, but not publishes articles.
- Assigns a DOI to each source code.
- Several languages accepted: Python, R, Julia, Matlab, Octave, C++, Fortran, Perl, Java
- They claim to view and download for everyone for free. The free plan is limited and the other are paid.
- Plans based on CPU time and storage usage. For example, the researcher plan allows 1h CPU and 5GB of storage per month.
- No statistics on usage. Seems low by website inspection.

https://codeocean.com/
2. Main Platforms for Reproducible Research: Research Compendia

Description:

- A dissemination platform, not a journal.
- Reproducibility in computational research.
- Share and archive the data, codes, documentation, parameters, and environmental settings.
- Free. Non-profit organization.
- No DOI for the moment. Future plan.
- Supported languages: R, MatLab, Python, and Cactus.
- No online demos, just shared files.

http://researchcompendia.science
2. Main Platforms for Reproducible Research: RunMyCode

Description:

- Confusing: two different services with the same name! runmycode.online, www.runmycode.org.
- Both dissemination platforms, not a journals.
- runmycode.online: run online code from Github, Gitlab, BitBucket, Go by Example, Github Gist, Gitlab Snippets, Bitbucket Snippets.
- runmycode.online languages: C/C++, Java, Nodejs, Python 2/3, Ruby, PHP, Go, Kotlin, Scala.
- runmycode.org: shares source code and data associated to a publication.
- runmycode.org: no code execution. All languages and data formats accepted.

http://www.runmycode.org
2. Main Platforms for Reproducible Research: DAE

Description: [Lamiroy & Lopresti 16]

- Allows to run document analysis algorithms and apply comparisons.

http://dae.cse.lehigh.edu
2. Main Platforms for Reproducible Research: DAE

**Description:** [Lamiroy & Lopresti 16]

- Platform for **Document Analysis** and **Exploitation**.
- Allows to run document analysis algorithms and apply comparisons.
- RR framework for document analysis with image data base.
2. Main Platforms for Reproducible Research: IPOL Journal

Description: [Arevalo et al. 16]

- A complete peer-reviewed journal can be considered as a platforms.
- Image Processing domain.
- Focused on mathematical rigorness. Detailed descriptions.
- Fast to create new demos for editors: automatic system.
- Accepted languages: C/C++, Python, MATLAB, Octave.
- Free to use/submit.
- Next move: machine learning applications. Servers with GPU.

https://www.ipol.im
3. New Ways of Publications
3. New Ways of Publications

### Recent original journals

- **IPOL** (as presented the morning)
  
  [https://www.ipol.im](https://www.ipol.im)

- **ReScience**
  
  [http://rescience.github.io](http://rescience.github.io)

- **JOSS**
  
  [https://joss.theoj.org](https://joss.theoj.org)
3.1 IPOL Journal: Image Processing On Line

Origin: http://www.ipol.im

- Journal started in October 2009.
- Initiative of Nicolas Limare and Jean-Michel Morel (CMLA).
- First article published in 2010.
- Domain of Image Processing.
3.1 IPOL Journal: Image Processing On Line

**Origin:** [http://www.ipol.im](http://www.ipol.im)

- Journal started in October 2009.
- Initiative of [Nicolas Limare](http://www.ipol.im) and [Jean-Michel Morel](http://www.ipol.im) (CMLA).
- First article published in 2010.
- Domain of Image Processing.

**Motivations**

- Reproducible Research.
- New way to publish research results.
- Allows everybody to test the algorithms (with their own images).
- **Free online demonstration** (user-platform independant) and source code.
3.1 IPOL Journal: short overview

Characteristics

- Journal publishing **algorithm** description, **source code**, online demonstration with experiment archives.

![IPOL Journal - Image Processing On Line](image)

**Automatic Detection of Internal Copy-Move Forgeries in Images**

Thibaud Ehret

**Abstract**

This article presents an implementation and discussion of the recently proposed 'Efficient Dense-Field Copy-Move Forgery Detection' by Cozzolino et al. This method is a forgery detection based on a dense field of descriptors chosen to be invariant by rotation. Zernike moments were suggested in the original article. An efficient matching of the descriptors is then performed using PatchMatch, which is extremely efficient to find duplicate regions. Regions matched by PatchMatch are processed to find the final detections. This allows a precise and accurate detection of copy-move forgeries inside a single suspicious image. We also extend successfully the method to the use of dense SIFT descriptors and show that they are better at detecting forgeries using Poisson editing.

**Download**

- full text manuscript: PDF (5.6M)
- source code: TAR/GZ

**Preview**

Loading takes a few seconds. Images and graphics are degraded here for faster rendering. See the downloadable PDF documents for original high-quality versions.
3.1 IPOL Journal: short overview

**Characteristics**

- Journal publishing algorithm description, source code, **online demonstration** with experiment archives.

![IPOL Journal - Image Processing On Line](image)

**Automatic Detection of Internal Copy-Move Forgeries in Images**

Select input(s) ▶️ Upload data

Input(s)

Parameters ▶️ Reset

- Test for flipped internal copies
- Number of iterations of patchmatch: 8
- Minimum distance: 8

The peer-review process includes the article, and source code.

Open Science journal and Reproducible Research.

Like classic journal: ISSN, DOI, indexed by: SCOPUS, DBLP, Scirus, Google Scholar, DOAJ, SHERPA/RoMEO, Héloïse, WorldCat, CrossRef, Ulrich, Index Copernicus, PBN, JGate, VisionBib, CVonline, JournalSeek and NewJour.

Recent evolution

- New system to automatically create our own online demonstration (see next session).
- Extended topics to sound and video processing with 3D processing.
3.1 IPOL Journal: short overview

Characteristics

- Journal publishing algorithm description, source code, online demonstration with experiment archives.
### 3.1 IPOL Journal: short overview

#### Characteristics

- Journal publishing algorithm description, source code, online demonstration with experiment archives.
- The peer-review process includes the article, and source code.
3.1 IPOL Journal: short overview

**Characteristics**

- Journal publishing algorithm description, source code, online demonstration with experiment archives.
- The peer-review process includes the article, and source code.
- Open Science journal and Reproducible Research.
3.1 IPOL Journal: short overview

**Characteristics**

- Journal publishing algorithm description, source code, online demonstration with experiment archives.
- The peer-review process includes the article, and source code.
- Open Science journal and Reproducible Research.
- Like classic journal: ISSN, DOI, indexed by: SCOPUS, DBLP, Scirus, Google Scholar, DOAJ, SHERPA/RoMEO, Héloïse, WorldCat, CrossRef, Ulrich, Index Copernicus, PBN, JGate, VisionBib, CVonline, JournalSeek and NewJour.
3.1 IPOL Journal: short overview

**Characteristics**

- Journal publishing algorithm description, source code, online demonstration with experiment archives.
- The peer-review process includes the article, and source code.
- Open Science journal and Reproducible Research.
- Like classic journal: ISSN, DOI, indexed by:
  - SCOPUS, DBLP, Scirus, Google Scholar, DOAJ, SHERPA/RoMEO, Héloïse, WorldCat, CrossRef, Ulrich, Index Copernicus, PBN, JGate, VisionBib, CVonline, JournalSeek and NewJour.

**Recent evolution**

- New system to automatically create our own online demonstrations (see next session).
- Extended topics to sound and video processing with 3D processing.
3.2 ReScience Journal

**Philosophy** (http://rescience.github.io)

- Context of Reproducible research [Buckheit & Donoho 95].
- Explicit replication: propose a new implementation of an existing work.
- Motivated from replication problems in computational science [Hinsen 15], [Topalidou et al. 15], [Hinsen 14].
3.2 ReScience Journal

**Philosophy** ([http://rescience.github.io](http://rescience.github.io))

- **Context of Reproducible research** [Buckheit & Donoho 95].
- **Explicit replication**: propose a new implementation of an existing work.
- **Motivated from replication problems in computational science** [Hinsen 15], [Topalidou *et al.* 15], [Hinsen 14].

**Details:**

- **Origin**: first volume in 2015.
- **Editorial Board**:
  - **Editors-in-Chief**:
    - Konrad Hinsen (Molecular Biophysics - Python, C, Racket, Clojure).
    - Nicolas P. Rougier (Comp. Neuroscience, Computer Science - Python, C/C++).
  - **11 Associate Editors with roles in**: Bioinformatics; Cognitive Modelling; Computational Ecology; Computational Physics; Image processing; Ecology, High-Performance Computing; Physics; Robotics; Signal Processing.
3.2 ReScience Journal: short overview

Characteristics

- Same presentation as in a “classic” journal.
3.2 ReScience Journal: short overview

Characteristics

- Same presentation as in a “classic” journal.
3.2 ReScience Journal: short overview

Characteristics

• Same presentation as in a “classic” journal.

Overview of the submission process

The ReScience editorial board unites scientists who are committed to the open source community. They are experienced developers who are familiar with the GitHub ecosystem. Each editorial board member is specialised in a specific domain of science and is proficient in several programming languages and/or environments. Our aim is to provide all authors with an efficient, constructive and public editorial process.

Submitted entries are first considered by a member of the editorial board, who may decide to reject the submission (mainly because it has already been replicated and is publicly available), or assign it to two reviewers for further review and tests. The reviewers evaluate the code and the accompanying material in continuous interaction with the authors through the PR discussion section. If both reviewers managed to run the code and obtain the same results as the ones advertised in the accompanying material, the submission is accepted. If any of the two reviewers cannot replicate the results before the deadline, the submission is rejected and authors are encouraged to resubmit an improved version later.

Criteria for Publication

To be considered for publication in ReScience, any given submission must satisfy the following criteria:

• Replicability
• Rigorous methodology
3.2 ReScience Journal: short overview

Characteristics

- Same presentation as in a “classic” journal.
3.2 ReScience Journal: short overview

Characteristics

- Same presentation as in a “classic” journal.
- Journal living on GitHub.
- Original submission process through Pull Request on GitHub.
### 3.2 ReScience Journal: short overview

**Characteristics**

- Same presentation as in a “classic” journal.
- Journal living on *GitHub*.
- Original submission process through *Pull Request* on *GitHub*.
- Peer reviewed journal (*reviews* and *reviewer name* given in the paper).
3.2 ReScience Journal: example of publication process

Example of publication [Rougier 17]

- Already published work "Weighted Voronoi Stippler" [Secord 02]
- Code no more available on author webpage:
  https://mrl.nyu.edu/~ajsecord/stipples.html.
3.2 ReScience Journal: example of publication process

Example of publication [Rougier 17]

- Already published work "Weighted Voronoi Stippler" [Secord 02]
- Code no more available on author webpage: https://mrl.nyu.edu/~ajsecord/stipples.html.
- Authors propose a new implementation defined in another context.
3.2 ReScience Journal: example of publication process

Example of publication [Rougier 17]

- Already published work "Weighted Voronoi Stippler" [Secord 02]
- Code no more available on author webpage: https://mrl.nyu.edu/~ajsecord/stipples.html.
- Authors propose a new implementation defined in another context.
- New description of the choice made to replicate the initial work.
3.2 ReScience Journal: example of publication process

Example of publication [Rougier 17]

- Already published work "Weighted Voronoi Stippler" [Secord 02]
- Code no more available on author webpage: https://mrl.nyu.edu/~ajsecord/stipples.html.
- Authors propose a new implementation defined in another context.
- New description of the choice made to replicate the initial work.

Conclusion of ReScience Journal

- Original new way of publish replication.
- Contains actually 22 published papers and 2 under review.
- Author of original work can not submit their own replication.
- Can potentially present works that were not successfully replicated ... → not yet present
3.3 JOSS Journal

Journal of Open Source Software (https://joss.theoj.org)

- Origin: founded by Arfon M. Smith in May 2016.
- Free and Open Access.
- Peer reviewing.
- Motivated by the fact that [Smith et al. 17]: "Current publishing and citation do not acknowledge software as a first-class research output".
3.3 JOSS Journal

**Journal of Open Source Software** ([https://joss.theoj.org](https://joss.theoj.org))

- **Origin**: founded by Arfon M. Smith in May 2016.
- **Free and Open Access**.
- **Peer reviewing**.
- **Motivated by the fact that [Smith et al. 17]: "Current publishing and citation do not acknowledge software as a first-class research output"**.

**Details**

- **Design**: defined in the current merit system of science.
- **Aim**: can be considered as a "journal for research software packages".
- **Editorial Board**:
  - Arfon Smith (@arfon), Editor-in-Chief.
  - **19 Topic Editors**: representing: Astronomy; Biodiversity Informatics; Bioinformatics; Computational Science; Data Science; Engineering; Computational Combustion; Computational Social Science; Fluid Dynamics; Energy Engineering; Geophysics; Geoscience; High Perf. Computing; Image; Information Sciences; Machine Learning; Neuroimaging; Nuclear Engineering; Open Science; Psychology; Semantic Web; Social Sciences; Software Deployment; Reproducible Research.
3.3 JOSS Journal: motivations

**Motivation of JOSS Editor in chief [Smith et al. 17]:**

- Software more and more present in numerous disciplines:
  → from a 2014 survey 90% mention to use software and 70% indicates that they were obligatory [Hettrick et al. 14].

- Software leak of scholarly support: no ecosystem of publication, citation, acknowledge.

- JOSS is the contribution to offer modern computational research results.
3.3 JOSS Journal: motivations

Motivation of JOSS Editor in chief [Smith et al. 17]:

- Software more and more present in numerous disciplines:
  ⇒ from a 2014 survey 90% mention to use software and 70% indicates that they were obligatory [Hettrick et al. 14].
- Software leak of scholarity support: no echosystem of publication, citation, acknowledge.
- JOSS is the contribution to offer modern computational research results.

As mentioned by Buckheit and Dohono [Buckheit & Donoho 95]:
"An article about computational science in a scientific publication is not the scholarship itself it is merely advertising of the scholarship"
### 3.3 JOSS Journal: characteristics

**Specific form:**

- **Form:** voluntary short: short abstract length (author names, list of key references, a link to software repository and a short description of the content).
3.3 JOSS Journal: characteristics

Specific form:

- **Form:** voluntary short: short abstract length (author names, list of key references, a link to software repository and a short description of the content).

- **Not allowed:** API or novel research descriptions.
## 3.3 JOSS Journal: characteristics

### Specific form:

- **Form**: voluntary short: short abstract length (author names, list of key references, a link to software repository and a short description of the content).

- **Not allowed**: API or novel research descriptions.

- Same characteristics than other journals: ISSN, Crossref DOI.

- Code Review: direct visibility
  - ⇒ collaboration with classic development tools (based on GitHub).
3.3 JOSS Journal: characteristics

Specific form:

- **Form:** voluntary short: short abstract length (author names, list of key references, a link to software repository and a short description of the content).
- **Not allowed:** API or novel research descriptions.
- Same characteristics than other journals: ISSN, Crossref DOI.
- Code Review: direct visibility  
  ⇒ collaboration with classic development tools (based on *GitHub*).

Content requested:

- Software need to be open source.
- Research application.
- Submitter needs to be main software contributor.
- Significant new contribution.
- Feature-complete (not partial).
3.3 JOSS Journal: publication process and cost (1)

Review Process

- Submission page: simple web application.

![Image of JOSS submission page]

- Crossref membership: $275. This is a yearly fixed cost for the JOSS parent entity—Open Journals—so that article DOIs can be registered with Crossref.

- Crossref article DOIs: $1. This is a fixed cost per article.

- JOSS web application hosting (currently with Heroku): $19 per month

Assuming a publication rate of 100 articles per year results in a core operating cost of ⇐ $6 per article. With 200 articles per year—which seems possible for the second year—the cost drops to ⇐ $3.50 per article:

\[
\frac{275 + (1 \times 100) + (19 \times 12)}{100} = 6
\]

\[
\frac{275 + (1 \times 200) + (19 \times 12)}{200} = 3.51
\]

Submitting authors retain copyright of JOSS articles and accepted articles are published under a Creative Commons Attribution 4.0 International License [22]. Any code snippets included...
3.3 JOSS Journal: publication process and cost (1)

**Review Process**

- Submission page: simple web application.
- Each submission generates an associated *GitHub* issue.

https://github.com/openjournals/joss-reviews

![The JOSS submission page. A minimal amount of information is required for new submissions.](image)

- Crossref membership: $275. This is a yearly fixed cost for the JOSS parent entity—Open Journals—so that article DOIs can be registered with Crossref.
- Crossref article DOIs: $1. This is a fixed cost per article.
- JOSS web application hosting (currently with Heroku): $19 per month

Assuming a publication rate of 100 articles per year results in a core operating cost of \( \sim \$6 \) per article. With 200 articles per year—which seems possible for the second year—the cost drops to \( \sim \$3.50 \) per article:

\[
\text{(1)} \quad \frac{\$275 + (\$1 \times 100) + (\$19 \times 12)}{100} = \$6.
\]

\[
\text{(2)} \quad \frac{\$275 + (\$1 \times 200) + (\$19 \times 12)}{200} = \$3.51.
\]

Submitting authors retain copyright of JOSS articles and accepted articles are published under a Creative Commons Attribution 4.0 International License [22]. Any code snippets included...
3.3 JOSS Journal: publication process and cost (1)

Review Process

- Submission page: simple web application.
- Each submission generates an associated GitHub issue. [https://github.com/openjournals/joss-reviews](https://github.com/openjournals/joss-reviews)
- Review interface: joss-review GitHub repository

Figure 1: The JOSS submission page. A minimal amount of information is required for new submissions.

- Crossref membership: $275. This is a yearly fixed cost for the JOSS parent entity—Open Journals—so that article DOIs can be registered with Crossref.
- Crossref article DOIs: $1. This is a fixed cost per article.
- JOSS web application hosting (currently with Heroku): $19 per month

Assuming a publication rate of 100 articles per year results in a core operating cost of $6 per article. With 200 articles per year—which seems possible for the second year—the cost drops to $3.50 per article: $(275 + (1 \times 100) + (19 \times 12)) / 100 = 6$. 

Submitting authors retain copyright of JOSS articles and accepted articles are published under a Creative Commons Attribution 4.0 International License [22]. Any code snippets included...
3.3 JOSS Journal: publication process and cost (1)

**Review Process**

- Submission page: simple web application.
- Each submission generates an associated GitHub issue. [https://github.com/openjournals/joss-reviews](https://github.com/openjournals/joss-reviews)
- Review interface: joss-review GitHub repository

**Figure 1:** The JOSS submission page. A minimal amount of information is required for new submissions.

- Crossref membership: $275. This is a yearly fixed cost for the JOSS parent entity—Open Journals—so that article DOIs can be registered with Crossref.
- Crossref article DOIs: $1. This is a fixed cost per article.
- JOSS web application hosting (currently with Heroku): $19 per month

Assuming a publication rate of 100 articles per year results in a core operating cost of $6 per article. With 200 articles per year—which seems possible for the second year—the cost drops to $3.50 per article:

\[
\frac{275 + (1 \times 100) + (19 \times 12)}{100} = \text{\$6.03}
\]

\[
\frac{275 + (1 \times 200) + (19 \times 12)}{200} = \text{\$3.51}
\]

Submitting authors retain copyright of JOSS articles and accepted articles are published under a Creative Commons Attribution 4.0 International License [22]. Any code snippets included...
Review Process

- Submission page: simple web application.
- Each submission generates an associated GitHub issue.
  
  https://github.com/openjournals/joss-reviews

- Review interface: joss-review GitHub repository
- Submission handled by a collection tools from a RubyGem library: Whedon.
  
  https://github.com/openjournals/whedon

Figure 1: The JOSS submission page. A minimal amount of information is required for new submissions.

- Crossref membership: $275. This is a yearly fixed cost for the JOSS parent entity—Open Journals—so that article DOIs can be registered with Crossref.
- Crossref article DOIs: $1. This is a fixed cost per article.
- JOSS web application hosting (currently with Heroku): $19 per month

Assuming a publication rate of 100 articles per year results in a core operating cost of \( \approx \$6 \) per article. With 200 articles per year—which seems possible for the second year—the cost drops to \( \approx \$3.50 \) per article:

\[
\frac{\$275 + (\$1 \times 100) + (\$19 \times 12)}{100} = \$6.
\]

\[
\frac{\$275 + (\$1 \times 200) + (\$19 \times 12)}{200} = \$3.51.
\]
### 3.3 JOSS Journal: publication process and cost (2)

<table>
<thead>
<tr>
<th>Main review steps</th>
</tr>
</thead>
<tbody>
<tr>
<td>• Mainly handled from issue labels.</td>
</tr>
<tr>
<td>• Example of review flow (extracted from [Smith et al. 17]).</td>
</tr>
</tbody>
</table>
3.3 JOSS Journal: publication process and cost (2)

Main review steps

- Mainly handled from issue labels.
- Example of review flow (extracted from [Smith et al. 17]).

Make software available in repository with OSI-approved license: https://opensource.org/licenses

Author short Markdown paper: paper.md

Submit to JOSS by filling out short form

Editor assigns ≥1 reviewers, who review submission

Reviewer(s) raise comments and issues following guidelines: http://joss.theoj.org/about#reviewer_guidelines

Authors fix issues

Editor accepts paper, authors archive software

Paper published & receives JOSS DOI: 10.21105/joss.####

Comparisons with other journals

- Journal of Open Research Software (openresearchsoftware.metajnl.com)
- SoftwareX (journals.elsevier.com/softwarex/)
  - Both journals review papers and software.
  - Anonymous reviews in the contrary to the contrary public review and public authors/reviewers interaction in JOSS.
### 3.3 JOSS Journal: publication process and cost (2)

**Main review steps**

- Mainly handled from issue labels.
- Example of review flow (extracted from [Smith et al. 17]).

**Cost [Smith et al. 17]**

- Minimum cost from volunteer editors and reviewers;
- Around 6$ per papers (with a base of 100 paper/year)
  → cross ref membership 275$ + crossref DOI 1$/paper + 19$/month.
3.3 JOSS Journal: publication process and cost (2)

Main review steps

- Mainly handled from issue labels.
- Example of review flow (extracted from [Smith et al. 17]).

Cost [Smith et al. 17]

- Minimum cost from volunteer editors and reviewers;
- Around 6$ per papers (with a base of 100 paper/year)
  → cross ref membership 275$ + crossref DOI 1$/paper + 19$/month.

Comparisons with other journal

- Journal of Open Research Software (openresearchsoftware.metajnl.com)
- SoftwareX (journals.elsevier.com/softwarex/)
  → both journals review papers and software.
  → anonymous reviews in the contrary to the contrary public review and public authors/reviewers interaction in JOSS.
### 3.3 JOSS Journal: one year overview and future [Smith *et al.* 17]

<table>
<thead>
<tr>
<th>Synthetic review of first year anniversary</th>
</tr>
</thead>
<tbody>
<tr>
<td>• 110 published articles for the first year <em>(actually 350)</em>.</td>
</tr>
<tr>
<td>• 45.6 days between submission and publication.</td>
</tr>
<tr>
<td>• 1.11 reviewers (total of reviewer count 93).</td>
</tr>
</tbody>
</table>
3.3 JOSS Journal: one year overview and future [Smith et al. 17]

Synthetic review of first year anniversary

- 110 published articles for the first year (actually 350).
- 45.6 days between submission and publication.
- 1.11 reviewers (total of reviewer count 93).

Future plan and open orientations

- Complete the Whedon RubyGem library for an automatic final paper publication.
- Question about what to do to handle version number and publication?
  → Select the MAJOR version from a normalized MAJOR.MINOR.PATCH?
- JOSS is now under NumFOCUS (charity supporting "world-class, innovative, open source scientific computing.").
- Make more adverts in various communities like in Pattern Recognition.
4. Conclusion
## Reproducible Research in Science

- Essential for credible science.
- In particular in Computational Science.
- Motivated by the credibility crisis pointed out by Donoho.
- Allows to follow the scientific method: claims can be verified (and eventually disproved).
4. Conclusion

Reproducible Research in Science

- Essential for credible science.
- In particular in Computational Science.
- Motivated by the credibility crisis pointed out by Donoho.
- Allows to follow the scientific method: claims can be verified (and eventually disproved).

Evolution

- Development of several platforms and journals.
- New tools allows comparisons and establish the real state of the art.
- New metric of assessing the impact of the research (instead classic citation indices).
- Increasing interest in this area.
References

Thank you for your attention!

感谢您的关注！

Githinji, G., Gymrek, M., Huff, K., Madan, C., Cabunoc Mayes, A.,
Moerman, K., Prins, P., Ram, K., Rokem, A., K Teal, T., Valls Guimera,
R., T Vanderplas, J.:
4 (2017)

[Hettrick et al. 14] Hettrick, S., Antonioletti, M., Carr, L., Chue Hong, N.,
Crouch, S., De Roure, D., Emsley, I., Goble, C., Hay, A., Inupakutika, D.,
Jackson, M., Nenadic, A., Parkinson, T., Parsons, M.I., Pawlik, A., Peru,
G., Proeme, A., Robinson, J., Sufi, S.:
UK Research Software Survey 2014.
(2014)

[Buckheit & Donoho 95] Buckheit, J.B., Donoho, D.L.
In: WaveLab and Reproducible Research. Springer New York, New
York, NY (1995) 55–81
References (2)

[Hinsen 2015] Hinsen, K.: 
**Writing Software Specifications.**

[Topalidou et al. 15] Topalidou, M., Leblois, A., Boraus, T., Rougier, N.: 
**A long journey into reproducible computational neuroscience.**

[Hinsen 14] Hinsen, K.: 
**Computational science shifting the focus tools to models.**
F1000Research (2014)

[Rougier 17] Rougier, P, N.: 
**Weighted Voronoi Stippling.**
ReScience 3 (2017) 1–8
1. Introduction
2. Main Platforms for Reproducible Research
3. New Ways of Publications
4. Conclusion

References (3)

[Secord 02] Secord, A.: 
**Weighted Voronoi stippling.**

[Lamiroy & Lopresti 16] Lamiroy, B., Lopresti, D.P.: 
**The DAE platform: A framework for reproducible research in document image analysis.**
In: RRPR@ICPR. Volume 10214 of Lecture Notes in Computer Science. (2016) 17–29

**The IPOL demo system: A scalable architecture of microservices for reproducible research.**
In: RRPR@ICPR. Volume 10214 of Lecture Notes in Computer Science. (2016) 3–16