Writing and reviewing reproducible scientific research. Review of some good practices

First Workshop on Reproducible Software Environments for Research and High-Performance Computing
Montpellier, France, November 8–10, 2023.

Miguel Colom
http://mcolom.info

1. Introduction: some definitions[1]

Repeatability and Reproducibility

Capacity to perform the same experiment as many times as needed.

→ **Repeatability**: Same team, same experimental setup

→ **Reproducibility**: Different team, same experimental setup

Example: is distilled water electrically conductive? Is salt water conductive? We can perform the experiment many times and get results (https://www.dailymotion.com/video/x2lcg6a).

Replicability

Capacity to obtain the same results when repeating an experiment by following a detailed procedure.

→ Different team, different experimental setup

In computational sciences (deterministic code, digital data): results obtained by following a detailed and correct pseudo-code description must be equivalent if the same input data is provided.

[1] There are different terminologies, see ACM Artifact Review and Badging, we use the version 1.0

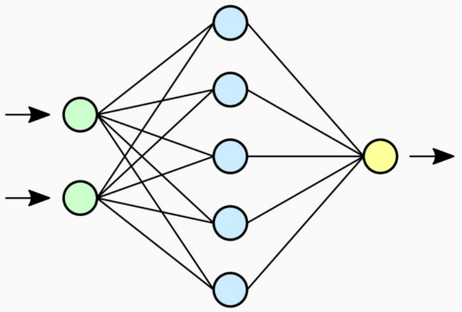
1.1 Definition: Repeatability Examples

Repeatable

Obtaining the classification results with a neural network.

We can repeat the experiment as many times as we want.

We just need the weights of the network and the input data.



Not repeatable:

Detection of the merger of two black holes from gravitational waves. We can't repeat the experiment as needed.

1.1 Definition: Reproducibility Examples

Reproducible:

Given:

- a detailed pseudo-code (or the source code itself),
- any associated learning or initialization data,
- the input data,

we should obtain exactly the same results each time we run the algorithm.

$\Rightarrow$ Exactly the same denoised image, classification results, etc.

Not reproducible

In a paper that shows

- a pseudo-code without all the details, or its initialization,
- the source code is not available,
- neither the learning data,

other researchers can't compare with the proposed method.

$\Rightarrow$ We can't be sure about anything on the method, nor test it with their own data.

1.2 Motivation: Example on Biomedical Research

Main Keys Points

- 2009: David Donoho points out a credibility crisis in scientific research
- 2012: the director of the oncology division at Amgen: tried to reproduce 53 of the most important papers on oncology. He failed to reproduce 47 of them.
  Sources:
  https://www.nature.com/articles/nature.2016.19269
  https://www.nature.com/articles/483531a
- Bayer HealthCare Germany confirmed: only 25% of cancer research is reproducible.
  Source: https://www.nature.com/articles/nrd3439-c1

1.3 Implementation of Reproducible Research

- Non-exact sciences (biology, medicine, . . .): difficult (but *desirable*). Hard to have exactly the same conditions along experiments.
- Computational sciences: no excuse!

3. Advanced Editorial Investment: IPOL Publications

Peer-reviewed

- Both the article (PDF) and the source code.
- Reproducibility: the reviewers check carefully that the source code matches the pseudo-code.

Each publication:

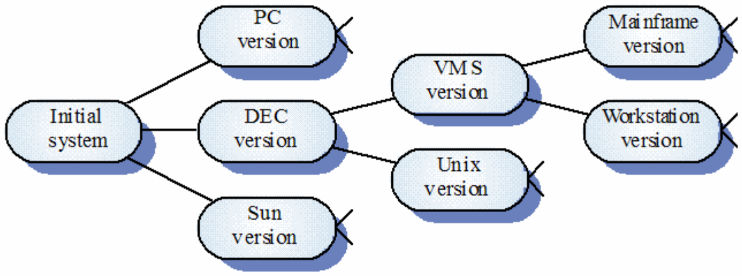
- A text describing the method in detail, including pseudo-codes.
- The source code, under an open-source software license.
- An online demo which allows users to test the method with their own data.
- An archive of experiments.

- No need to be an original work. We're interested in the math details, reproducibility, and understanding.
- ISSN, DOI, indexed by SCOPUS. Not yet an "Impact Factor".

Reviewing reproducible articles

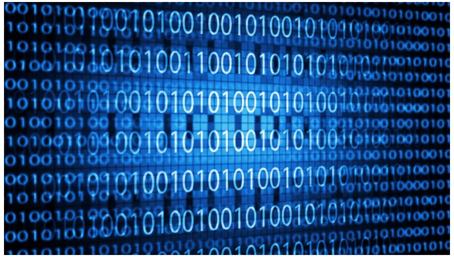
Pre-requisites (1/2)

- **Existence** of a **detailed procedure** for both the **compilation** and **execution**.
- The **exact environment** must be **declared** also (for example, to reconstruct a Docker container)
- The **exact version** of the **code** (commit, SWHID, …) must be given. The review is only valid for an specific status of the code



Pre-requisites (2/2)

- The **data** must also be **referenced**. The **datasets** must be **public** and **reusable** and allow for comparison.
- **FAIR** principles:
  - **Findability;**
  - **Accessibility;**
  - **Interoperability;**
  - **Reusability**.



- The **reviewer** must be able to **obtain the same** or **comparable results** as in the **paper**. For example: the values in any figures or tables.

Pre-requisites

- The **reviewer** must be able to **obtain** the **same** or **comparable results** as in the paper. For example: the values in any figures or tables.

→ **Let's try!** With **IPOL's DCT denoising**

Checking the pseudocode and code

- As part as the **detailed procedure**, a **pseudocode** must be **available**
- The pseudocode **must describe exactly what the code does. The reviewer must check this.**
    - No hidden hyperparameters
    - No unexplained *magic* numbers
- The **names** of variables and functions **should match** the ones in the **article**
- **Comments** must be added to understand ***why*** the code does some operations (**not *how*!**)

Checking the pseudocode and code

- **As part as the detailed procedure, a pseudocode must be available. Input and outputs.**

Algorithm 2: DCT Denoising - Hard thresholding

Function DCTDENOISINGHARD(Y, σ, s)
    input : noisy image Y, noise level σ, and patch size s
    output: denoised image
    $X, W \leftarrow 0$
    $Y \leftarrow$ DECORRELATECOLORS(Y)
    for each *patch domain* $\Omega_{patch} \subset \Omega$ *of size* $s \times s$ **do** // Ω is the image support
        $b_{tmp} \leftarrow 0$ // color patch temp variable
        $N_P \leftarrow 0$
        for each *color channel* c **do**
            $\widehat{b} \leftarrow$ DCT(EXTRACTPATCH(Y, Ω_{patch}, c)) // uses DCT/IDCT defined in (10)-(11)
            for $\omega \in (\{0, \cdots, s-1\} \times \{0, \cdots, s-1\})$ **do** // scan patch frequency domain
                if $\omega \neq \vec{0}$ **then** // don't filter the zero frequency
                    if $|\widehat{b}(\omega)| < 3\sigma$ **then** $\widehat{b}(\omega) \leftarrow 0$
                    else $N_P \leftarrow N_P + 1$ // # of nonzero coefficients of $\widehat{b}$
            $b_{tmp}[c] \leftarrow$ IDCT$(\widehat{b})$ // store channel c of color patch
        $X(\Omega_{patch}) \leftarrow X(\Omega_{patch}) + b_{tmp} \cdot (1 + N_P)^{-1}$
        $W(\Omega_{patch}) \leftarrow W(\Omega_{patch}) + (1 + N_P)^{-1}$ // Adaptive weights, see Section A
    $X \leftarrow X/W$
    return UNDODECORRELATECOLORS(X)

Checking the pseudocode and code

- **Comparison code / pseudocode**

```
inline void ExtractPatch(const Image &src, int pr, int pc, DCTPatch *dst) {
  // src is padded, so (pr, pc) becomes the upper left pixel
  for (int chan = 0; chan < dst->channels(); ++chan) {
    for (int row = 0; row < dst->rows(); ++row) {
      // // the following line copies a line interval to the patch and
      // // is equivalent toi (but faster):
      //   for (int col = 0; col < dst->columns(); ++col) {
      //     dst->space(col, row, chan) = src.val(pc + col, pr + row, chan);
      //   }
      copy(&(src.val(pc, pr + row, chan)),
           &src.val(pc + dst->columns(), pr + row, chan),
           &dst->space(0, row, chan));
```

$$\widehat{b} \leftarrow \mathrm{DCT}(\textsc{ExtractPatch}(Y, \Omega_{patch}, c))$$

- Names OK. Signature of the function different.

Checking the pseudocode and code

- **Comments must be added to understand *why* the code does some operations (not how!)**

Algorithm 2: DCT Denoising - Hard thresholding

Function DCTDenoisingHard(Y, σ, s)
  input : noisy image Y, noise level σ, and patch size s
  output: denoised image
  $X, W \leftarrow 0$
  $Y \leftarrow$ DecorrelateColors(Y)
  for each *patch domain* $\Omega_{patch} \subset \Omega$ *of size* $s \times s$ **do** // Ω is the image support
    $b_{tmp} \leftarrow 0$ // color patch temp variable
    $N_P \leftarrow 0$
    for each *color channel* c **do**
      $\widehat{b} \leftarrow$ DCT(ExtractPatch(Y, Ω_{patch}, c)) // uses DCT/IDCT defined in (10)-(11)
      for $\omega \in (\{0, \cdots, s-1\} \times \{0, \cdots, s-1\})$ **do** // scan patch frequency domain
        if $\omega \neq \vec{0}$ **then** // don't filter the zero frequency
          if $|\widehat{b}(\omega)| < 3\sigma$ **then** $\widehat{b}(\omega) \leftarrow 0$
          else $N_P \leftarrow N_P + 1$ // # of nonzero coefficients of $\widehat{b}$
      $b_{tmp}[c] \leftarrow$ IDCT($\widehat{b}$) // store channel c of color patch
    $X(\Omega_{patch}) \leftarrow X(\Omega_{patch}) + b_{tmp} \cdot (1 + N_P)^{-1}$
    $W(\Omega_{patch}) \leftarrow W(\Omega_{patch}) + (1 + N_P)^{-1}$ // Adaptive weights, see Section A
  $X \leftarrow X/W$
  return UndoDecorrelateColors(X)

Checking the pseudocode and code

- Any **pre/post processing must** be **explained** in the **paper**
- The **structure** (functions) of the code should be **reflected** in the **pseudocode**
- The **pseudocode** should have a **proper granularity**
  - No need to describe how to compute a cosine with a Taylor series!
  - However, any significant computation should be described. Example: in DCT denoising it's important that the DCT matrix is orthogonal to keep the isometric property.

Checking the pseudocode and code

- **Granularity**

$$\widehat{b} \leftarrow \mathrm{DCT}(\textsc{ExtractPatch}(Y, \Omega_{patch}, c))$$ `// uses DCT/IDCT defined in` (10)-(11)

$$\widehat{g} \leftarrow \mathrm{DCT}(\textsc{ExtractPatch}(G, \Omega_{patch}, c))$$

Checking the pseudocode and code

Isometric DCT transform. The type-II DCT transform implemented in the FFTW library and its inverse (type-III) are not isometric, so in order to implement the frequency domain denoising they must be normalized. The FFTW transforms (identified by the w superindex) compute for $k = 0, \cdots, N-1$

$$\mathrm{DCT}^{\mathrm{w}}(X)_k = 2\sum_{j=0}^{N-1} X_j \cos\left[\pi\left(j+\frac{1}{2}\right)\frac{k}{N}\right], \tag{5}$$

$$\mathrm{IDCT}^{\mathrm{w}}(Y)_k = Y_0 + 2\sum_{k=1}^{N-1} Y_k \cos\left[\pi\left(j+\frac{1}{2}\right)\frac{k}{N}\right], \tag{6}$$

which are unnormalized, hence $\mathrm{IDCT}^{\mathrm{w}}(\mathrm{DCT}^{\mathrm{w}}(X)) = 2N\,X$.
The isometric transforms $Y = \mathrm{DCT}(X)$ and $X = \mathrm{IDCT}(Y)$ that satisfy Parseval's equality $\sum_k |Y_k|^2 = \sum_j |X_j|^2$ are obtained as

$$Y_k = \mathrm{DCT}(X)_k = \alpha_k\ \mathrm{DCT}^{\mathrm{w}}(X)_k = \alpha_k\, 2\sum_{j=0}^{N-1} X_j \cos\left[\pi\left(j+\frac{1}{2}\right)\frac{k}{N}\right], \tag{7}$$

$$X_j = \mathrm{IDCT}(Y)_j = \mathrm{IDCT}^{\mathrm{w}}(\beta\cdot Y)_j = \beta_0\, Y_0 + \sum_{k=1}^{N-1} \beta_k\, 2Y_k \cos\left[\pi\left(j+\frac{1}{2}\right)\frac{k}{N}\right], \tag{8}$$

$$\text{with}\quad \alpha_k = \begin{cases} \sqrt{1/(4N)}, & k=0 \\ \sqrt{1/(2N)}, & k=1\dots,N-1 \end{cases} \quad\text{and}\quad \beta_k = \begin{cases} \sqrt{1/N}, & k=0 \\ \sqrt{1/(2N)}, & k=1\dots,N-1. \end{cases} \tag{9}$$

The normalization factors corresponding to the 2D-DCT of a $N \times M$ image are given by

$$Y_{k,m} = \alpha_k\, \alpha'_m\ \mathrm{DCT2D}^{\mathrm{w}}(X)_{k,m}, \tag{10}$$

$$X_{j,l} = \mathrm{IDCT2D}^{\mathrm{w}}(\widetilde{Y})_{j,l} \qquad \text{with}\quad \widetilde{Y}_{k,m} = \beta_k\, \beta'_m\, Y_{k,m}, \tag{11}$$

where α' and β' are defined as in Equation (9) but for the range $[0\dots,M]$.

Checking the pseudocode and code

- **Optimizations**: the might cause the **source code** and **pseudocode look** quite **different**
- **Potential problem**: the pseudocode is **describing something different** to what has been **implemented**
- The paper need to **explain carefully** why **they're equivalent**. **Not granted**.

Checking the pseudocode and code

- **The paper need to explain carefully why they're equivalent. Not granted.**

```
for (int chan = 0; chan < dst->channels(); ++chan) {
  for (int row = 0; row < dst->rows(); ++row) {
    // // the following line copies a line interval to the patch and
    // // is equivalent toi (but faster):
    //   for (int col = 0; col < dst->columns(); ++col) {
    //     dst->space(col, row, chan) = src.val(pc + col, pr + row, chan);
    //   }
    copy(&(src.val(pc, pr + row, chan)),
         &src.val(pc + dst->columns(), pr + row, chan),
         &dst->space(0, row, chan));
```

Checking the pseudocode and code

- The **pseudocodes must** be **referenced** in the **paper**
- Each **pseudocode** must contain a **brief explanation** what's about, with its inputs and outputs.

Checking the pseudocode and code

- **The pseudocode must be referenced in the paper**

Using (3) and (4) for this procedure guarantees that white Gaussian noise remains so under the DCT transform, so the noise model remains the same in every layer of the pyramid. A scaling factor s used (Algorithm 4 lines 14 and 25) to guarantee that the values of the image remain on the same ange after resizing, which also implies that the standard deviation of the noise gets halved at each

Checking the pseudocode and code

- **Each pseudocode must have a brief explanation what's about, with its inputs and outputs.** Not really found here.

Algorithm 5: Multiscale DCT Denoising

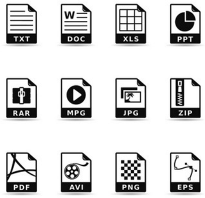
Function MultiscaleDCT($Y, \sigma, s, n_{scales}, f_{rec}$)
    input : noisy image Y, noise level σ, patch size s, number of scales n_{scales}, and multiscale recomposition factor f_{rec}
    output: denoised image
    for $l \leftarrow n_{scales} - 1, \ldots, 0$ **do**
        $Y_l \leftarrow$ ExtractScale(Y, l)
        $X_l \leftarrow$ DCTdenoising2step($Y_l, \sigma/2^l, s$)
        if $l == n_{scales} - 1$ **then** *combined* $\leftarrow X_l$
        else *combined* $\leftarrow$ MergeCoarse(X_l, *combined*, f_{rec})
    return *combined*

Recommendations when writing a reproducible article

Recommendations about code availability, referencing, and environment

- **Make your code available**:
    - Github, Gitlab
    - Software Heritage → Version tracking. Easy referencing. Permanent archiving.
    - ...
- **Data repositories**
    - Zenodo
    - ...
- **Control** and **describe** your **environment**
    - **Guix** :-), Nix
    - Docker
    - Singularity
    - Virtual machines
    - TerraForm
    - ...

Recommendations about formats

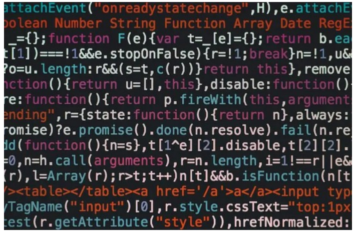


- Use **standards** and **reusable formats**. For both **documentation, code**, and **data**. **Avoid proprietary formats**.
- You can use, for example:
  - CSV
  - HDF
  - LaTex
  - And many others

- Use **public datasets**. Make your own **research artifacts FAIR** (for example, in Zenodo)

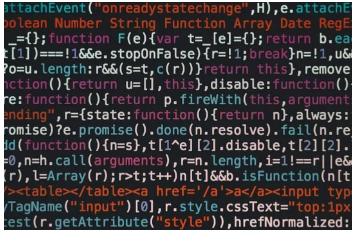
Recommendations about code quality (1/2)

- Use **asserts** to **control errors**. Specially during active development.
- Save an example of **execution** and **compare** with the **output** if you change anything
- **Comment** the **code**: **why it does** something, **not how!** "# sum a and b" vs "# Compute the accumulated cost"



Recommendations about code quality (2/2)

- **Document** your software. To avoid that it gets **unsynced** with the code you can use **automatic documentation** (Doxygen and others)
- Give a **version number** or **commit version** to your **released software**
- **Ask** your **colleagues** to **review** your code and article before submitting it



Recommendations to write the article

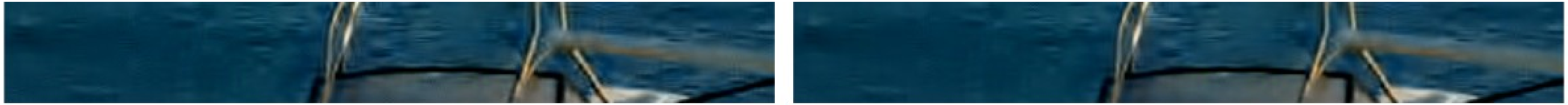
- **Cite** the **work** of **others**. Statements must be **cited** or **proven**! **Reproducing** existing work **without citation** may be considered **plagiarism**!
- **Scientific writing** should be **factual**, **concise** and **evidence-based**, but that doesn't mean it can't also be **creative**, **appealing to the readers**. **It must be.**
- **Avoid speculation** in the **discussion** section. You can **add** some in the **conclusions**. For example, about the evolution of the field.
- **Focus** your paper on a **single and clear key message or claim**. The **title** should reflect this, and it should be **clear** in the abstract.

Recommendations to write the article

- Use **institutional emails**. You're **working in a group**!
- The **abstract** typically is 1**50-200 words**, but check the journal/conf. rules
- A proposal to **structure** the **abstract**: 1. the **purpose** of the study (the central question); 2. a brief statement of **what was done** (Methods); 3. a brief statement of **what was found** (Results); 4. a brief statement of **what was concluded**.
- **Avoid *"I"*** and use ***"we"*** (even if you alone! *On the shoulders of giants*)
- **Tense**: **methods** section in **past tense**. **Conclusions** in **present tense**.

Recommendations to write the article

- The **captions** of the **figures** must be **complete**, even if some **text is repeated from a section**. They should explain **what the figure is showing**, along with any **information needed** for the **interpretation**. Help the lazy reader.



no aggregation weights (27.89 dB) | aggregation weights (27.99 dB)

Figure 1: Detail of a result from MS DCT denoising with 8×8 patches computed without and with aggregation weights for a noise level $\sigma = 50$. Note the reduced oscillations in the sky.

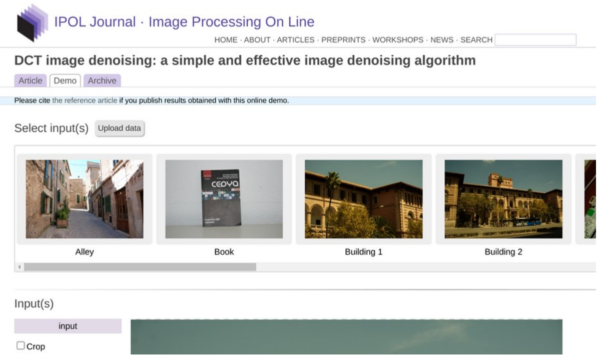
Recommendations to write the article

- Any **important equations** must be **numbered**, and **referenced** in the text.
- **Graphics**: use **vector graphics** whenever possible (PDF, SVG)
- **Review** the **bibliography**. Review the **format** of the **citations.** Check that it's **complete**.

Recommendations for the online demos (1/3)

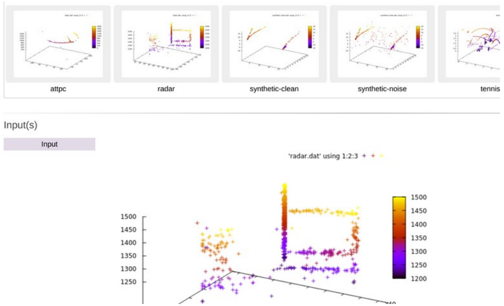
→ **Online demos** are very **useful**. They allow other researchers to quickly **obtain results** and **compare**. They increase the **impact** of your **publication**.

- **Minimize** the number of **parameters**: it's a demo, not a complete app. If needed, add an **"expert mode"** to show the rest of the parameters.



Recommendations for the online demos (2/3)

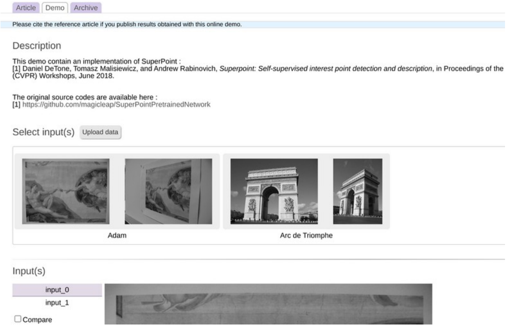
- Add a **short explanation** of each **parameter**
- Choose **typical default values**
- Choose a **reasonable range** of values (min, max, default) for the **parameters**
- **Limit** the **range** of the **parameters** which cause **too-long executions**. A user typically waits no more than **30 seconds**. (“*who waits forever, anyway?*”)



Recommendations for the online demos (3/3)

- **Show results** in a way that they **illustrate** the **method** and are **easy** to **interpret**
- Add a **small introduction** in the **demo**. Some users might land directly there from a Google's search. The **demo** must be **auto-contained**.
- Check the **online archive** now and then, since you'll find **unexpected results** which will bring you **insights** for **your research**.

→ (Take a look at an IPOL demo, if time)



References

- Vers une recherche reproductible. https://rr-france.github.io/bookrr
- Mack, C. (2014). How to write a good scientific paper: structure and organization. J. Micro/Nanolith. MEMS MOEMS, 13(4), 040101.
- Alston, J. M., & Rick, J. A. (2021). A beginner’s guide to conducting reproducible research. Bulletin of the Ecological Society of America, 102(2), 1-14.
- Steingraber, S., Jolls, C., & Goldberg, D. (1985). Guidelines For Writing Scientific Papers. Honors Organismal Biology Laboratory Manual. Web. Retrieved July, 11, 2014.
- Nicola Pierazzo, Jean-Michel Morel, and Gabriele Facciolo (2017). Multi-Scale DCT Denoising. Image Processing On Line, 7, 288–308.

Thank you for your attention

This work is under the Attribution-ShareAlike 4.0 International (CC BY-SA 4.0) license.
For more details: https://creativecommons.org/licenses/by-sa/4.0/

The images used in these slides are under the *Fair Use* provision, given that they're used only for this particular scholarly purpose.
Please contact me if any of the images should be removed.