Publishing and archiving of research data:

An Eawag-flavored hands-on guide (v1.0)

This is a hands-on guide for researchers who want to prepare their research data for depositing it in an archive or repository. It also contains practical advice regarding day-to-day data-management best practices. While framed as guide for Eawag, it should be useful for a wide range of scientists.

doi:10.25678/000066

This PDF document might be out of date. Please check with the the authoritative HTML-version.

Feedback (as GitHub-Issue or via email) is greatly appreciated.

Contents

1 Introduction

This is guide should help to organize and annotate research data to make it suitable for the deposition in the Eawag Research Data Institutional Archive (ERIC, only internal) or other repositories.

Since a good research data package is most efficiently created from data that has been well managed throughout the research process, we also provide some relevant advice for good data management practices well before the final package for archiving is prepared. As a side-effect, considering these guidelines might therefore also improve the efficiency of your work.

The guide is under constant development. We very much appreciate feedback of any kind (gaps, inconsistencies, suggested additions, etc.) to rdm@eawag.ch.

2 Scope of *research data*

The term *research data* here refers not only to numerical datasets proper, but to all kinds of information that can be used to reproduce, validate, or re-use scientific a work. If that work is the subject of a published article or report, we used the term *publication data package* to refer to the data associated with the publication.

2.1 Primary data

Strictly speaking, *primary data* is data that is the direct, unmodified result of observations, measurements, surveys, etc. In practice however, a modified form of that raw data, e.g. through format conversions or the discarding of irrelevant data, is frequently considered *primary data*. As a rule of thumb, *primary data* is data for which there exists no previous incarnation that cannot be deleted. The researcher, considering conventions in her or his field, defines what exactly to archive as *primary data*.

Archiving primary data is of the essence.

2.2 Derived data

In addition to the *primary data*, you should include as a minimum for a *publication data package* all datasets necessary to reproduce the figures, tables and individual numbers in the running text that are presented in the main paper and the Supporting Information.

Consider including more. Providing additionally data from intermediate steps in the chain of analysis and related but not crucial data for the publication in question will make your data package more useful and increase your work's credibility and exposure.

2.3 Third party data

Third party data is data that was used for the research but was not gathered within the frame of the respective research project. It is important to properly document the provenance of this data and be aware whether there are restrictions to publish it outside (or even inside) Eawag. In case there are legal problems regarding the archiving and/or publishing of such data, please contact rdm@eawag.ch.

2.4 Source code

Computer code is often a critical part of the research and necessary to validate or reproduce the work. You should include any code along with relevant information about dependencies, the platform(s) it runs on, required interpreters, compilers, libraries and the versions used. Take care to provide a reasonable degree of user documentation for your software.

2.4.1 Version control systems

It is good practice to use source code versioning systems such as Git and often a copy of the code is kept on a collaboration platform such as gitlab.com or github.com. Do not just refer to such a platform when archiving your code, as they do not qualify as "archives" and might disappear tomorrow. Instead, create a `zip`- or `tar`-archive from the version of the source code used in your work and add it to the package.

2.4.2 Reproducible computing

Taken seriously, providing the software infrastructure for other researchers to reliably run your code in the future can be very difficult for non-trivial codebases and no generally applicable recommendations currently exist. Most proposals involve virtual machines (VMs) or container technologies such as LXC/LXD, Docker, rkt or Singularity. ReproZip is a promising tool to collect the dependencies of your code into a container- or VM-format. If you are facing that challenge, please get in touch with rdm@eawag.ch.

2.4.3 Third party code

should be included in the data package if you can't refer to a copy of exactly the version you used, which is available from a reliable, well-established repository committed to long-term preservation. For example, you don't have to archive R version 3.4.4 - the The Comprehensive R Archive Network will do that most reliably for you. However, if you used this R package for extreme value analysis, which seems to be only available from a single GitHub repository, by all means archive it yourself.

2.4.4 Proprietary software

These recommendations might not be workable in full if you used proprietary third party software, libraries, languages or tool-chains. Aim at using exclusively open source software for scientific work, since proprietary tools diminish the credibility, reproducibility and long-term value of your research.

2.5 Ancillary information

Generously add any files and information that could further help to understand your work or build on your results. This includes for example photos and figures of a related publication and its Supporting Information , maps, pointers to related resources such as the project website, etc.

3 When to prepare a *research data package*?

Throughout the project, keep your research data continuously reasonably organized from the start. Experienced researchers usually do that already. If you come back after three weeks of holidays and you only need to read your own documentation to know what is where and what it means, you are OK. If you organize your data in a directory hierarchy and/or employ a file naming convention, put some effort into documenting theses structures from the start. Liberally re-factor as your project develops and these conventions and structures don't match anymore.

Primary data should be archived as soon as possible so safeguard against data-loss and because ancillary information about the measurements, observations, data gathering is still readily available.

Publication data packages, that is data tied to a specific article or report, is ideally submitted once the article is accepted. This makes it unlikely that the data change in the future and at the same time the researcher likely still has the details of the work readily available.

Leaving researchers should be reminded early to prepare and archive their research data. Depending on the sophistication of previous data organization, preparation and annotation might take much longer than expected.

4 Documentation ("*scientific metadata*")

The data in the package needs to be annotated to be useful for other researchers or your future self. Each package should contain a README - file, that describes the package at the highest level.

- The README file is a pure ASCII or UTF-8 text-file (README.txt) or is written in a common markup-language such as Markdown (README.md), or HTML (README.html).
- The README file has to reference, implicitly or explicitly, each other file in the package and thus can be used to look up purpose and description of each element of the dataset.
- The README file describes the dataset structure, that is the organization in a folder-hierarchy (if there is one) and the file-naming convention used, if applicable.
- If the package contains files of a non-common format, the README should mention that and point to required software to read them.
- The README should contain all information necessary to interpret and understand the dataset that is not recorded elsewhere, e.g. meaning of column headers, abbreviations, units of variables, etc.
- The README should contain pointers to other files in the package that contain scientific metadata.
- The README should contain a copy of text in the *Abstract* of the bibliographic metadata.
- The README should contain a copy of the text in the *Citation* field of the bibliographic metadata.
- The README should contain a reference to the associated article or report (as a DOI if possible), if applicable.

Depending on the size an complexity of the data package, it might be useful to describe parts of it in lower-level README-files, perhaps located in sub-directories.

Note: This list is likely incomplete. Your guidance should be the imagined situation of a researcher who was not involved in the project and who should be able to completely understand and make use of the package without having to contact you.

In the case of a *publication data package*, the publication proper (article, report) usually contains indispensable scientific metadata. Please include a PDF of the final publication, the accepted version, the submitted version or a draft manuscript, in that order of preference. If your publication is already registered in Lib4RI's DORA, or if it can be accessed through a DOI, it is sufficient to include the

respective URL as a resource of the package, e.g.
https://www.dora.lib4ri.ch/eawag/islandora/object/eawag:16080 or
https://doi.org/10.1177/2053019617740365

5 File naming

5.1 General rules

File- and directory-names should adhere to the following conventions to ensure interoperability across platforms and filesystems and not be a pain to process programmatically:

- Limit the file name to 32 characters (preferably less!).
- Do not use spaces in filenames! Use an underscore (_) as substitute.
- Do not distinguish files by capitalization but keep it all lowercase.
- Use only alphanumeric characters, minus (-), and underscore (_),
  don't use special characters `& , * % # ; * ( ) ! @ $ ^ ~ ' { } [ ] ? < >`.
- Use only one period and use it before the file extension.
- **Dates** in filenames (and pretty much everywhere else) must have one of two formats:
  1. YYYY-MM-DD
  2. YYYYMMDD
- **Numbers** as part of filenames should be left padded with zeros, for example:
  `site01_logger001.csv` … `site12_logger328.csv`

5.2 File naming schemes

If a file naming scheme is employed, it should be descriptive and consistent. Encode attributes of a file as alphanumeric strings separated by underscores (_). Here is a sophisticated naming convention example from the CMIP5 Model Output Requirements:

```
filename = <variable name>_<MIP table>_<model>_<experiment>
           _<ensemble member>[_<temporal subset>].nc
```

All components of the filename are either part of a controlled vocabulary or have a precisely defined format by themselves. Make sure to sensibly define and clearly document the format of the components of the filenames. This results for example in this filename:

```
tas_Amon_HADCM3_ historical_r1i1p1_185001-200512.nc
```

If mapping your content to such a convention and directory-structure becomes too complex, you should consider to employ a proper database. In particular, if you feel you spend too much effort in your analysis code to construct the paths for the data files, you are in the process of implementing a primitive database software by yourself and should step back and reconsider.

6 Folder structure and file-archives

Frequently, research data is organized in a hierarchical folder structure, for example:

```
|
- README.md
- manuscript.pdf
- data
|    |
|    - raw
|    |   |
|    |   - loggerA.csv
|    |   - loggerB.csv
|    - analyzed_1
|    |          |
|    |          - discharge.csv
|    |          - ...
|    - analyzed_2
|    |          |
|    |          - hydrographs.csv
|    |          - ...
|    - final
|         |
|         - table1.csv
|         - figure1.csv
|         - ...
- images
|      |
|      - site_A.jpg
|      - ...
- code
     |
     - python
     |     |
     |     - setup.py
     |     - extractor.py
     |     - ...
     - C
       |
       - campsci.h
       - logmod_campsci.c
```

In ERIC (as in most other repository solutions), such a hierarchical structure cannot be represented directly. The content of a *data package* are *resources*. A *resource* is either a file or a URL. There is only one "level" for all resources. The recommended way to deal with this situation is to create one or several ***archive file(s)*** which contain individual files along with the the directory-hierarchy and will re-create the latter when unpacked.

The most common type of an *archive file* is a **zip** archive. Other common formats are **tar** and **7z**.

In the simplest case, just create a (compressed) *archive file* of one full directory tree, and upload the resulting *archive file*, e.g, if the root of the directory-tree is `data`:

`zip -r data.zip data`, or

```
tar -zcvf data.tgz data
```

Always upload the README and, if applicable, the manuscript separately.

Make sure that there are no spurious, often hidden, files in the archive, which are sometimes added automatically by certain operating systems, such as files named `.DS_Store` or `__MACOSX`.

6.1 Spitting into multiple *archive files*

It might be prudent to split the directory-tree into multiple *archive files*. In general, reasons to do that are related to

a) the size of the data,
b) the applicability of the resource-related bibliographic metadata, in particular the fields *Resource Type*, *License*, and *Citation*, and
c) parts of the data that should not be compressed.

Examples are:

- One *archive file* would be too large and users might want to just use (and download) a part of the data.
- Part of the archive should be made available under a specific license (and can't be published in the public domain, as is the default for ERIC). This can for example apply to software, which often hast to be published under the GNU Public License, or for which reasons exist to retain copyright using another open source license. For question regarding software licensing, get in touch with rdm@eawag.ch.

Another case might be third-part data, which has a different license and/or which requires a separate citation.

- A large chunk of the data is of a different type than the rest. For example a sub-folder containing just images. Also, if a folder already contains mostly compressed data, e.g. just `JPG` and/or PNG files, it is advisable to create a non-compressed *archive file* from it, by either using `tar` without the `-z` option or `zip` with the option `-0`.

Note: `zip` allows to switch off compression selectively for specific file extensions, e.g.
`zip -rn .tiff:.gif:.jpg data.zip data`
will not compress the image files with the given extensions.

For **really large datasets containing large numbers of large files** that arise for example in genomics, we provide a command line tool, **`resup`**, to automate packaging, splitting into reasonably sized chunks, reliable uploading, and doing everything in reverse for downloading: https://github.com/eawag-rdm/resup

7 File formats

Try to use reliable, well-documented and universally readable file-formats. Because of the extremely large and constantly growing number of file formats, no conclusive lists of acceptable formats can be given.

7.1 Rules of thumb

- Avoid proprietary formats (prefer standardized ones)
- Prefer formats that can be read by multiple software packages, which exist for multiple platforms.
- Prefer common formats over rarely used formats.
- Avoid pseudo-standards, such as `xlsx` and `docx`, which you can recognize by the fact that there do not exist multiple implementations across different applications and platforms.
- Anything "plain text" that is not pure ASCII should be UTF-8 encoded.
- Simpler is better (as long as no info is lost).
- If your original data comes in a "low-quality" format (e.g. MP3, GIF, ...) just keep it like that, a conversion will not improve the quality.

7.2 More detailed recommendations

The Library of Congress Recommended Formats Statement is an excellent starting point. Also, feel free to get in touch with rdm@eawag.ch for help.

7.3 MS Office: The elephant in the room

Microsoft Office formats pose a particularly tedious problem because they are so prevalent but flaky and volatile at the same time.

- Never use the "Office 97-2003 formats" `doc` and `xls`. Use `xlsx` and `docx` instead, if you have to work with such software.
- Microsoft Word documents and Powerpoint presentations can always be exported as PDF. Please do that and provide the `docx` or `pptx` document additionally, if you want to make it easy for users of your data to edit it.
- Always prefer plain `csv` to `xlsx`.
- However, Saving as `csv` from Microsoft Excel is very unreliable and the result might not represent what you intended. Double-check!

To more reliably and comfortably export Workbooks even containing multiple Worksheets, we provide the tool **`xlsxtocsv`**:
https://github.com/eawag-rdm/xlsxtocsv
Double-check anyway!

- If you want to use MS Excel for data entry and display, please read the excellent short article **Data organization in spreadsheets** by Broman and Wu (2017).
- Some research groups use MS Excel as a versatile research tool that integrates data storage (in tables), presentation (through layouting- and content-embedding features), analysis software (VB macros, statistical functions and graphing). In case your results come in the form of such a sophisticated Excel Workbook, give up with regard to archiving your work reproducibly,

reliably for the long term. It is just not possible. Provide the xlsx-file as is and hope for the best.

Note: It is generally a good idea to avoid spreadsheet software, in in particular Microsoft Excel, in scientific work. The recently well publicized corruption of 20lists due to Excel - autoconversion (Ziemann et al., 2016) is just the tip of the iceberg. Other issues are, but not limited to

- 30 year old date type bugs that today interact with cell formatting under different language settings in unpreictable ways and differently in different versions of Excel (start with Joel Spolsky's entertaining anecdote.
- well documented bugs in statistical functions, random number generation and numerical algorithms, e.g. "unacceptably bad" statistical functions (Yalta, 2008) (alt. source), "it is not safe to assume that Microsoft Excel's statistical procedures give the correct answer" and "It is clear that when Microsoft claims to have "fixed" a procedure that is known to be in error, Microsoft's claim cannot be safely believed" (McCullough and Heise, 2008) (alt. source), (alt. source), "the fact that Microsoft is still marketing a product that contains known errors, some of them going back to Excel 4, released in 1994" (Mélard, 2014), (alt. source), (alt. source).
- The initially comfortable mixing of data storage, data presentation, and analysis code into one inextricable and obscure bundle that apparently is the root cause for a large number of consequential mishaps involving Excel collected by the The European Spreadsheet Risks Interest Group.

7.4 Open Document Formats (ODF)

The Open Document Formats (ODF), e.g. `odf`, `ods`, `odt`, are slightly better because they follow a real standard. However, they are still very complex and adoption is low. Therefore we advise to use them only if you can't use the simpler alternatives (i.e. `csv` for tabular data and **and if the software that was employed to create the documents in question uses ODF as its default file format (e.g. LibreOffice)** .

Do not save to ODF formats from Microsoft Office! MS Office's implementation of ODF is flawed and you might end up with files that are even less usable than the corresponding MS Office formats.

8 Bibliographic metadata

Bibliographic metadata is the data attached to the whole package and to individual ressources and is visible directly in the web-interface of ERIC. In case you make your data available to the general public as *Open Data*, this metadata will be transmitted to various public directories, search services and registries. In other words, it is metadata that is not contained with a *resource* (a file). This data includes fields such as *Title*, *Author(s)*, *Keywords*, *Curator*, *Spatial Extent*, ...

- ERIC provides **online guidance** for this metadata. Keep in mind that this metadata record mainly serves to make your data **findable**, e.g. as a source of keywords for search engines and as information to catalog your dataset in third-party directories. Therefore, scientific precision is less important than usefulness to someone who might not even know what exactly he or she is searching for. If, for example, you deployed sensors at 50 different locations across Lake Geneva, just provide a bounding box containing the lake under *Spatial Extent*. The exact coordinates should be part of the dataset proper, in a table that is uploaded as a *resource*.
- **The title for a *publication data package*** is of the form "Data for: [title_of_the_paper]"

- Make sure that *Curator* and the *Usage Contact* have email addresses.