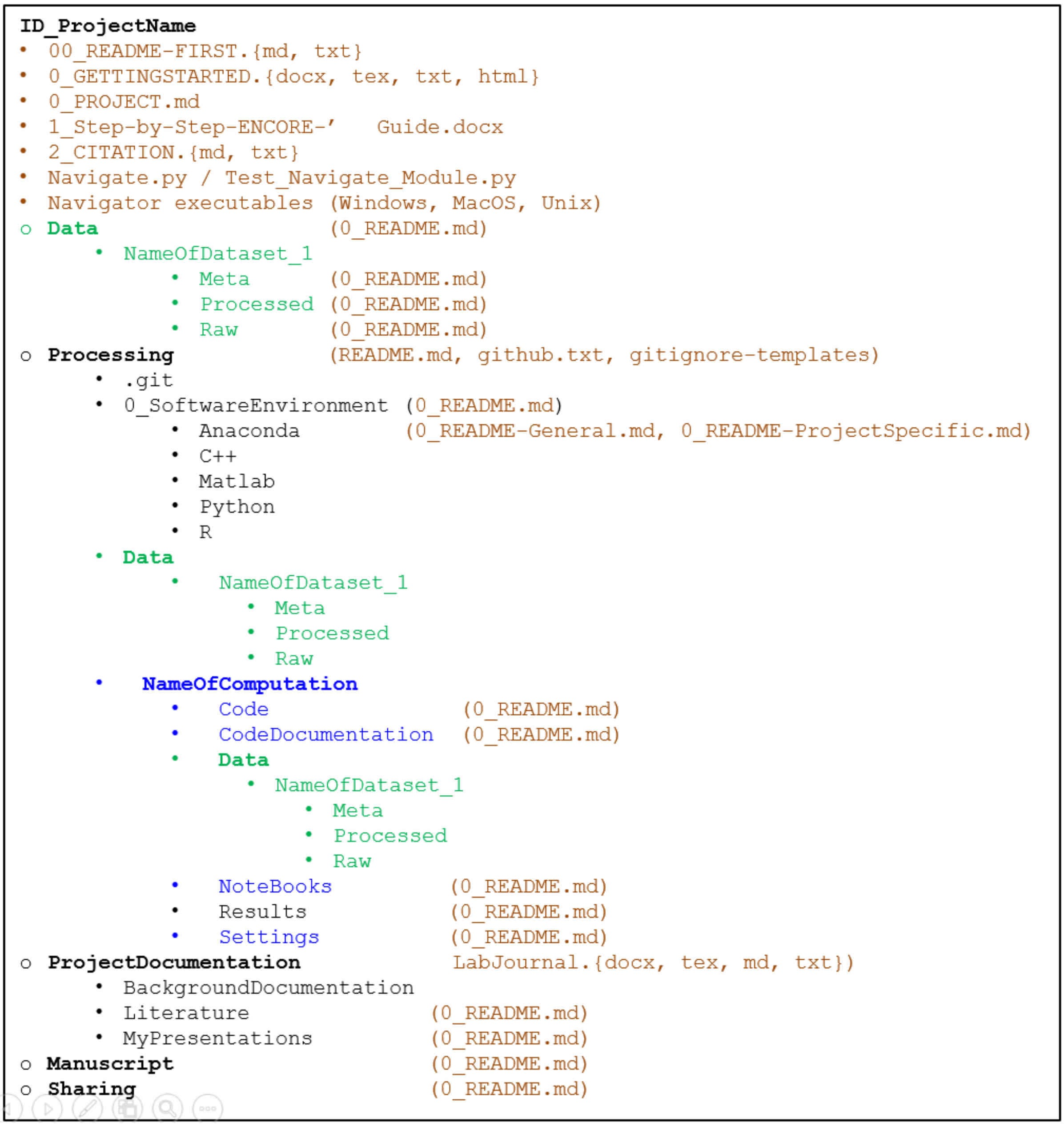
ENCORE. A practical implementation to improve reproducibility and transparency of computational research

Supplementary Information

Pre-defined sub-directories and files in the standardized file system structure (sFSS)

The README markdown files in the sFSS describe the content of the sub-directories and the content of the pre-defined files (including the README files) in more detail. The list below provides a summary of all sub-directories and pre-defined files.



```
ID_ProjectName
•  00_README-FIRST.{md, txt}
•  0_GETTINGSTARTED.{docx, tex, txt, html}
•  0_PROJECT.md
•  1_Step-by-Step-ENCORE-'   Guide.docx
•  2_CITATION.{md, txt}
•  Navigate.py / Test_Navigate_Module.py
•  Navigator executables (Windows, MacOS, Unix)
o  Data                   (0_README.md)
      •  NameOfDataset_1
            •  Meta       (0_README.md)
            •  Processed  (0_README.md)
            •  Raw        (0_README.md)
o  Processing             (README.md, github.txt, gitignore-templates)
      •  .git
      •  0_SoftwareEnvironment (0_README.md)
            •  Anaconda         (0_README-General.md, 0_README-ProjectSpecific.md)
            •  C++
            •  Matlab
            •  Python
            •  R
      •  Data
            •   NameOfDataset_1
                  •  Meta
                  •  Processed
                  •  Raw
      •   NameOfComputation
            •   Code                  (0_README.md)
            •   CodeDocumentation    (0_README.md)
            •   Data
                  •  NameOfDataset_1
                        •  Meta
                        •  Processed
                        •  Raw
            •   NoteBooks           (0_README.md)
            •   Results             (0_README.md)
            •   Settings            (0_README.md)
o  ProjectDocumentation           LabJournal.{docx, tex, md, txt})
      •  BackgroundDocumentation
      •  Literature               (0_README.md)
      •  MyPresentations          (0_README.md)
o  Manuscript                     (0_README.md)
o  Sharing                        (0_README.md)
```

Figure 1 (copied from main text). The standardized File System Structure (FSS) and associated pre-defined files. Directory structure of the sFSS containing pre-defined files (brown), which include markdown README files that provide a documentation template and instructions. Note that the pre-defined files in the data directory (green) and the 0_SoftwareEnvironment subdirectories are only shown once. Directories 'NameOfDataset_1' and 'NameOfComputation' provide placeholders and should be replaced with more descriptive names. These are duplicated if multiple datasets are used and if different computation procedures are performed. Subdirectories shown in blue are version'd using Git/GitHub. The optional '0' prefix ensures that these files/directories are always on top of the file list. The README.md in 'Processing' is the default GitHub repository README file and therefore does not have the '0' prefix.

Description of the sFSS sub-directories

\Manuscript

This directory contains the (draft) manuscript(s) corresponding to this computational project including figures, tables, and supplementary information.

\Data

\Data\NameOfDataset1

\Data\NameOfDataset\Raw

\Data\NameOfDataset\Processed

\Data\NameOfDataset\Meta

These directories contain the raw, processed, and meta-data. Raw data comprises unprocessed data that come from the physical measurement device. Processed data comprises data obtained from collaborators or public databases and, consequently, not produced as part of your computational analyses. If the data (pre)processing is part of your computational analyses then, preferentially, it should be placed in the \Results directory within \Processing. However, ENCORE allows the flexibility to store your own processed data in the \Data\Processed directory. Meta-data is the description of the data including the data license, description of the samples, experimental design, content and format of the data files, etc.

\Processing

Contains all sub-directories and pre-defined files related to the computational part of the project.

\Processing\0_SoftwareEnvironment

\Processing\0_SoftwareEnvironment\Anaconda

\Processing\0_SoftwareEnvironment\C++

\Processing\0_SoftwareEnvironment\Matlab

\Processing\0_SoftwareEnvironment\Python

\Processing\0_SoftwareEnvironment\R

One challenge that is only partially addressed by ENCORE concerns the preservation of the full computing environment. This environment is defined by (interdependencies of) the operating system, software tools, versions and dependencies, programming language libraries, etc. Gruning and co-workers proposed a software stack of interconnected technologies to preserve the computing environment (Gruning, Chilton, et al., 2018). This stack comprises (Bio)Conda (Anaconda Software Distribution, 2020; Gruning, Dale, et al., 2018) to provide virtual execution environments addressing software versions and dependencies, container platforms such as Docker (Nust et al., 2020) to preserve other aspects of the runtime environment, and virtual machines using cloud systems or dedicated applications such as VMware, to overcome the dependencies on the operating system and hardware. We are currently investigating how to best approach this within the ENCORE environment.

However, some basic information about the computing environment (e.g., export of Anaconda environments) can be stored in this directory.

The sub-directories provide basic information about different environments (e.g., R/Rstudio, Python/PyCharm, Anaconda) for peers not familiar with the used computing environment. In addition, you may find other files such as cheat sheets, tutorials, and exports of (Anaconda) environments.

\Processing\Data

See \Data

\Processing \NameOfComputation

- This directory should also contain a conceptual description of applied methodology to improve transparency. For example,
- Brief description of used pre-existing methods (version) including specification of the mathematical/statistical model, parameters, variables, references, etc.
- If a new method is developed that this method should be described in full detail.
- Describe why the selected or developed computational approach is valid for your research question.
- This allows your peers to make their own judgement about the approach and results.
- Considered alternatives?
- Detailed description of all data filtering, reduction, normalization, etc steps that are performed prior to the downstream analysis.
- Avenues of exploration examined throughout development, including information about negative findings.

\Processing \NameOfComputation\Code

Contains the (in-house developed) software used for the computational analysis.

\Processing \NameOfComputation\CodeDocumentation

External (user) documentation of the code. Possibly automatically generated with documentation tools such as Sphinx.

\Processing \NameOfComputation\Data

See \Data

\Processing \NameOfComputation\NoteBooks

Notebooks ((web-based) interactive computing platform that combines live code, equations, narrative text, visualizations etc.) should be placed in this sub-directory.

\Processing \NameOfComputation\Results

(Intermediate) results (e.g., figures, tables) from the computational analysis.

Record of intermediate results (preferable in a standardized format). Generate hierarchical analysis output, allowing layers of increasing detail to be inspected. This can reveal discrepancies toward what is assumed, and can in this way uncover bugs or faulty interpretation that are not apparent in the final results. It also allows any inconsistency to be tracked to the step where the problem occurs. It also allows critical examination of the full process behind a result. Clearly document the intermediate/final results and the imposed hierarchy.

For any figure or table that ends up in a publication, report, or presentation at meeting, the underlying data and a stand-alone piece of code should be available to regenerate the figure. It also allows easy modification of a figure and to retrieve the data of the figure (instead of having to redo a complete analysis). Equally important, the data of the figure can be further analyzed or inspected.

\Processing \NameOfComputation\Settings

This file/sub-directory concerns settings/parameters for the algorithms you have developed. For settings related to the computing environment see \0_SoftwareEnvironment for further instructions.

\ProjectDocumentation

This subdirectory contains (background) information about any part of the project. However, as a general rule, documentation should be close to component (e.g., data, code) that is described. For example, the documentation about the data should be in the \Data directory. However, more general information can be placed in the sub-directories in \ProjectDocumentation.

\ProjectDocumentation\BackgroundDocumentation

Documents relevant as project background documentation. For example: the project proposal, presentations from collaborators or peers (thus, not from the project team), or relevant tutorials about applied methodology.

\ProjectDocumentation\Literature

This subdirectory should contain relevant scientific literature (e.g., pdf files) obtained from PubMed, BioRxiv, etc. The pdf files should be named using the Author-Year-Journal (or similar) to allow easy retrieval during project discussions. In addition, it should contain the reference manager (e.g., EndNote, Mendeley, Bibtex) file and an export to the standardized RIS format. The README file in this sub-directory should briefly describe the relevance of each paper (e.g., parameters used in the computational analyses).

\ProjectDocumentation\MyPresentations.

This sub-directory contains oral or poster presentations (and abstracts) given by the project team during meetings (e.g., progress meetings, seminars, conferences).

\Sharing

Rationale. In general, the complete file system structure (FSS) and its contents should be shared unmodified with peers that aim at reproducing the computational analysis. However, in specific cases it might be desirable to share only parts of the FSS and/or restructure the FSS. The reduced and/or restructured FSS is then stored (as a compressed file) in the /Sharing directory. This file should at least indicate what you did (not) share and how/why you restructured the FSS. In addition, document when and who you shared with.

Typical use. A typical use of the \Sharing directory is for support projects in which computational analyses were performed for other researchers as a service (e.g., biomedical, clinical researchers). These researchers might only be interested in the final results (figures and tables) and not in the code that produced these results. That is, they will not aim to repeat the analyses. In such situation, the results and tables can be shared in a (flat) structure that is more convenient for them to browse and use, and that leaves out all code and background documentation.

Restrictions. Sharing an FSS with an (external) colleague may be restricted due to, for example, copyright on pdf files of papers, sensitive/private information (in labjournal.docx), non-open-access of data, etc. Make sure you remove such information from \Sharing*

Description of the sFSS pre-defined files

00_README-FIRST.{md, txt}, 0_README.md / README.md

Throughout the sFSS there are README files that explain the content of the sub-directories and provide instructions and a template to guide documentation of the project. Most of these files are so-called Markdown files that can be opened in any text editor but require a Markdown viewer (e.g., Notepad++, Typora) to show the markup.

To facilitate first-time users the 00_README-FIRST file is also provided as a text file that contains instructions w.r.t. the Markdown files.

All 0_README files start with the ‘0_’ or ‘00_’ prefix to ensure it appears at the top of the file list. The only exception is the README.md file in the /Processing directory which is the default GitHub README file that should not contain a prefix. Because the sFSS (and not GitHub) is the entry point for a project, the GitHub README.md file does not necessarily have to contain a project description. However, you may want to copy the information from 0_PROJECT.md (see below) into this README.md file. More importantly, it should provide a explanation of the code in the processing directory and instructions about its execution.

0_PROJECT.md

Short description of project, contact person, and project team. This file is used by the FSS Navigator.

0_GETTINGSTARTED.txt.

Template document (plain text format). Copy this file to your favorite editor to add content.
Examples:

- **0_GETTINGSTARTED.docx.** Template document (Microsoft Word format to show how to include links). The docx file can be saved as html (make sure you use utf-8 encoding).
- **0_GETTINGSTARTED.tex.** Template document (LaTex format to show how to include links). The LaTex file can be converted with [Pandoc](https://pandoc.org/index.html) to html.

- **0_GETTINGSTARTED.html.** Example of exported html file used by the FSS navigator.

The main use of the GETTINGSTARTED files is to guide a first-time user of a finished project to the most important aspects (e.g., results, code) of the project before he/she explores are information contained in the sFSS manually. The GETTINGSTARTED files provide links to the relevant sub-directories and files. 'Getting started' templates are provided in different file formats, which can be converted to html once finished. The 0_PROJECT.md and 0_GETTINGSTARTED.html files are used by the FSS Navigator.

Help files

- **1_Step-by-Step-ENCORE-Guide.{pdf, docx}.**
  User guide to use ENCORE (the File System Structure (FSS) and setting up a corresponding GitHub repository).

General

- **2_CITATION.{md,txt}.** How to cite ENCORE and the FSS Navigator
- **.FSSignore.** Currently not used but to be used with an application that selects all files needed for sharing.

FSS Navigator

- **Navigate.html.** Open in your browser to navigate the standardized file system.
- **Navigate.py.** Standalone Python 3 script to generate Navigate.html to navigate the FSS. Can be run from the command line (Navigate.py -h)
- **Navigate_U.sh.** Shell script to run Navigate on Unix/Linux systems. Change the first line (#!/usr/bin/Python) if necessary. Make executable using chmod +x

There are also executables available for Windows and Mac OS. These are available from the GitHub *release* and also from Zenodo (DOI: https://doi.org/10.5281/zenodo.7985655; https://zenodo.org/record/7985655)):

- **Navigate_W.exe.** Windows executable if you don't have Python installed (Navigate.exe -h).
- **Navigate_M.** MacOS executable (macOS 13.3.1 (Ventura), Apple M1)
- **Navigate_MacIntel.** MacOS executable (macOS 10.13.6 (High Sierra), Intel Core i5)

Test_Navigate_Module.py.

Python script to show how to use Navigate.py as module in other Python scripts. This may help to keep Navigate.html up-to-date without manually executing Navigate.py.

Navigate.conf.

Configuration file for the FSS navigator.

\ProjectDocumentation\LabJournal.{docx, md, tex, txt}.

Templates in different file formats for the lab journal.

In general, any documentation should be kept in the sub-directory where it belongs. Thus, use the

0_README.md and/or additional (e.g., PowerPoint) files to document the data, software, and results in their respective directories.

The lab journal will contain more general documentation. For example,

- General information and concepts
- Summaries of project discussions
- Steps to be taken
- New (future) research ideas
- Pointers to the location of certain pieces of information

In addition, it may also contain integrated parts of the various readme files whenever useful (but keep it consistent with the source files). Include figures and tables when necessary.

Although, strictly speaking, a lab journal is not for recording new/future ideas or proving summaries of discussions, it is important for the group to also have a record of this. Therefore, they can also be included in the lab journal.

Parts of the lab journal that should not be shared with peers (e.g., new research ideas) should be clearly labelled with 'Not for sharing' such that we can easily remove these parts. Alternatively, you may maintain two separate documents.

If necessary, ENCORE allows to maintain lab journals in multiple sub-directories. For example, one may decide to have a separate lab journal in '\Processing\NameOfComputation' for a specific part of the computational analysis.

\Processing\github.txt.

Provides the URL to the corresponding GitHub repository and any other relevant information about the repository. This file is used by the FSS Navigator.

\Processing\gitignore-FSS-template.txt and .gitignore

This file should be adapted (if needed) and renamed to .gitignore. It contains instructions for git about files not to synchronize with the GitHub repository (e.g., data and results). This file should be modified depending on the contents of \Processing. Optionally, you can use other language-specific templates that are also found in \Processing*:*

- gitignore-R-template.txt
- gitignore-C++-template.txt
- gitignore-FSS-template.txt
- gitignore-JetBrains-template.txt
- gitignore-Matlab-template.txt
- gitignore-Python-template.txt

To use any of these templates, simply merge its content into .gitignore. It is considered good practice to keep a single .gitignore in the top-level directory and not in individual subdirectories, which would make debugging more troublesome.