cBRAIN open research practice guide

Anja Betz, cBRAIN research coordinator
Alberto Villagran, cBRAIN computer scientist
Dr. Malika Ihle, LMU Open Science Center and Switch to Open Program coordinator
Reema Gupta, LMU Open Science Center FAIR data management consultant
Prof. Dr. med. Inga Katharina Koerte, M.D., cBRAIN Principal Investigator

Last edited: July 2025

This handbook is for students and researchers joining the cBRAIN team. It provides guidance for those involved in research processes that may lead to the publication of scientific articles. This manual covers each part of the research cycle, starting from formulating a research question to publishing articles, data, and code. Members are expected to check in advance that they have sufficient knowledge to perform the necessary steps by reviewing the self-paced tutorials linked in this handbook and to allocate sufficient time to learn the skills required (in total, between 6 and 35 hours; time estimation is indicated for each tutorial). All tutorials are also listed in an overview table at the end of this document. This is a collaborative living document; all cBRAIN members are asked to suggest updates or improvements when they notice the need for it. The cBRAIN research coordinator, computer scientist, and PI listed above will approve your suggestions so they become part of the guidelines for the rest of the group. Parts of these guidelines are marked as required, some as recommended. Some are only relevant for specific cases. We recommend you read the entire handbook upon onboarding to understand how you can effectively contribute to the team, and come back to specific relevant sections as you work through your own project.

The figure below summarises, along the research cycle, the steps (in the white boxes) that need to be performed by the project team between each checkpoint (in colored boxes) that take place with the rest of the team or direct supervisor. The source file is available here to make any adaption on a per project basis: cBRAIN-lab-handbook-figure.pptx . We recommend you print a copy for your office to serve as a memory prompt and active checklist.

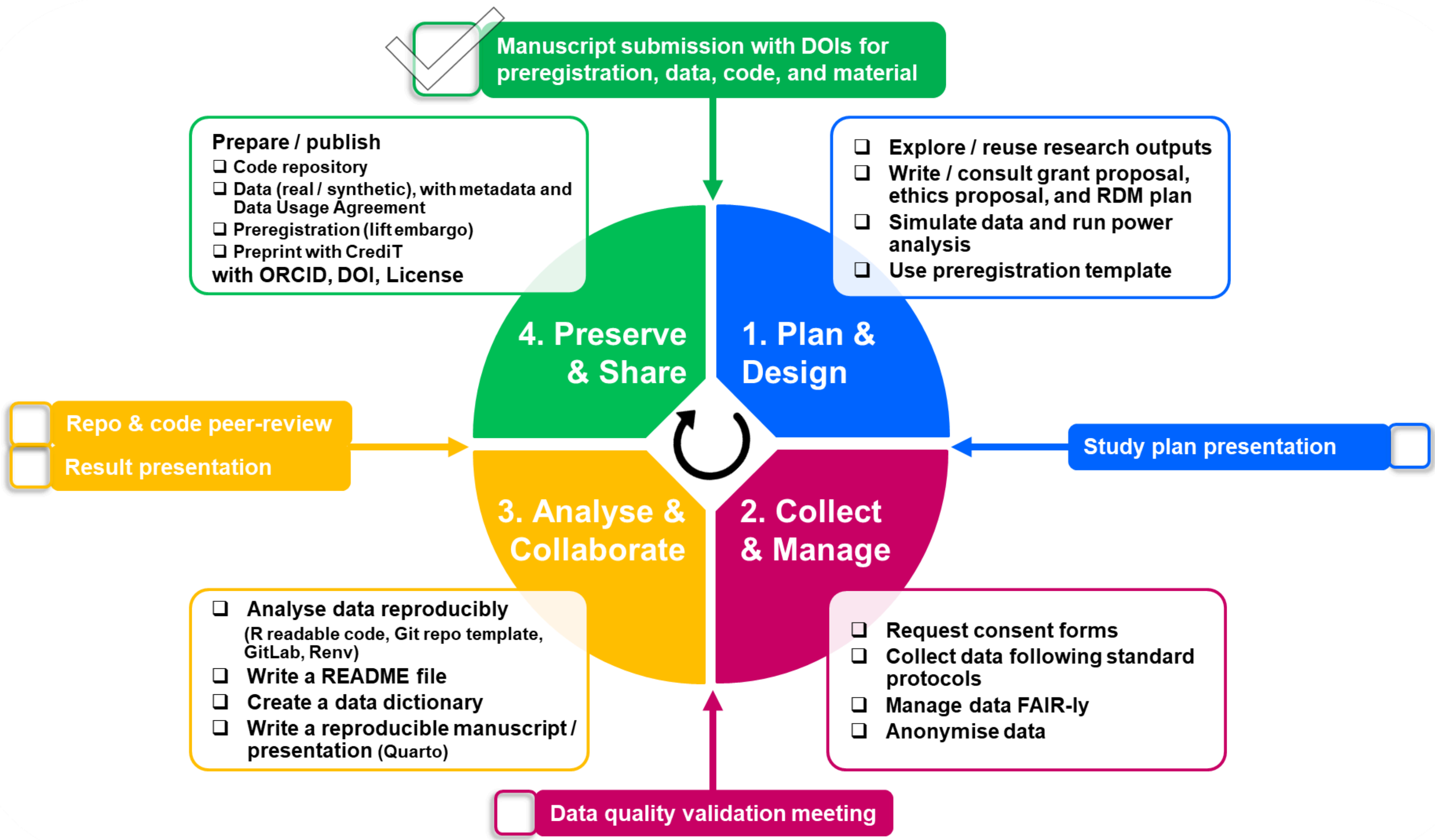

Figure 1. Open Research Cycle. Detailed customizable checklist of open research practices placed along the research cycle. To use as a memory prompt for each research project. Based on Ihle, M. (2025). Open research cycle. LMU Open Science Center. https://doi.org/10.5281/zenodo.15630230

1. Plan & Design

a. Explore/Reuse existing research outputs

LEARN (recommended)

- reference management https://lmu-osc.github.io/introduction-to-zotero/ (1h)

i. Existing articles

To review the existing literature and come up with a well-founded research question, you can search for articles and preprints (reviews, meta-analysis, original articles, study cases, etc.) from the medical field on PubMed or EuropePMC, and the "cBRAIN journals list" (TODO: HYPERLINK). Search registries like openAlex and Google Scholar will also include book chapters or dissertations. Note that your primary sources should be peer-reviewed articles. It is recommended to keep track of your bibliography with a reference manager, which integrates with your reproducible workflow for writing manuscript. We recommend Zotero (see tutorial) because it is free and open source, and it can be integrated within RStudio to write reproducible manuscripts seamlessly.

ii. Existing preregistered projects

To have some insight into projects that are currently ongoing, you can check https://osf.io/registries and https://clinicaltrials.gov/ for projects that were preregistered and should get published in the following years (or should get accompanied by a note if the project is abandoned). A preregistration typically consists of a hypothesis and predictions, a plan for data collection (when relevant), and a plan for data analysis, that researchers upload before starting their projects, often in order to increase the rigor of confirmatory research.

iii. Existing datasets or code

To find existing datasets, you can explore neuroimaging specific repositories, such as InTBIR, OpenNeuro, EBRAINS, NIMH Data Archive, NITRC Image Repository, find other discipline-specific repositories on re3data which is a central registry of all repositories, or explore subject agnostic repositories such as DataCite or Zenodo (oftentimes with the corresponding analysis code). These platforms enable you to use datasets that already exist and are open for reuse, or provide metadata and provide explanations on how to request access to the data. Have you found something you would like to work on? If the data is not openly licensed, but requires you to make a request to the owners following specific guidelines, please discuss with your supervisor/PI beforehand. For instance, following a Data Use Agreement, you can obtain access to large US-based datasets like ABCD on the NIH Data Archive. Finally, you may also find code available for reuse on Zenodo (archived) and/or possibly still in active development on Github.

Any resource you use, in order to get inspired, or to reuse and/or adapt, must minimally be cited, using their DOIs if they have one (and otherwise URL with author, date and time of access) and you must follow the licence and/or usage agreement provided by the authors.

b. Write/Check bounding frameworks such as grant proposal, ethics proposal, and Research Data Management (RDM) plan

i. LMU Guidelines for Safeguarding Good Scientific Practice

In August 2019, the LMU implemented the "Guidelines for Safeguarding Good Scientific Practice" of the German Research Foundation (DFG) - the so-called "Kodex". This is a mandatory step for all institutions that want to be eligible for DFG funding. The derived regulations (see Ordnung der Ludwig-Maximilians-Universität München zur Sicherung guter wissenschaftlicher Praxis vom 17. November 2023, in German), are legally binding for all academics, researchers, research support staff, teachers, and students at LMU. Only the original text in German prevails, but here is an English summary of relevant aspect for this guide:

Appropriate level of documentation and standards to allow reproduction:

- Reproducible methods must be used. (§11)
- When research software is developed, its source code must be documented. (§12)

Appropriate level of documentation and standards to allow replication:

- All information relevant to the production of a research result must be documented comprehensively to enable replication. (§7 and §12)
- If specific professional recommendations exist for review and evaluation, the results must be documented in accordance with these respective specifications. (§12)
- Individual results that do not support the hypothesis must also be documented; a selection of results is not permitted. (§12)

Public access to research results:

- Apart from specific exceptions, all findings should be made public. For this, they must be described in a detailed and comprehensible manner which includes making available the research data, materials and information on which the results are based, as well as the methods used and the software employed (including appropriately licensed self-written software) according to the FAIR principles*. (§13)

- Data, material, software made publicly accessible must be appropriately archived, usually for a period of 10 years (§17).

**Note: The FAIR principles are defined as:*
- Findable: metadata should be deposited in a searchable repository and be assigned a permanent identifier
- Accessible: the data is either open, or accessible upon some authentication process, or closed, but with open metadata.
- Interoperable: the data is described with a standard terminology (so the dataset can be merged with other ones) and saved in a stable file format
- Reusable: the data is richly documented and is accompanied by a data usage license
See https://www.go-fair.org/fair-principles/ for more information.

ii. Write grant / check funders requirements

➢ Writing your own grant?

Typically, only supervisors and PI will write grants. With their agreement, you may draft a grant application that they will need to approve.

Check the additional open and reproducible science requirements of the call you are applying to (e.g. extent and timing of data sharing). You might want to test some subjects or use open datasets and analyze them before the study even starts, to use as *pilot data* and show already how promising your idea is.

After approval from your supervisor, you may have your proposal reviewed by experts of the Research Funding Unit who know the latest development of open science requirements of funders and how they should be addressed in a proposal (Research Funding Unit contact points with Open Science expertise: Florian Schreck for national calls, Laura Kropf for international calls; Research Funding Unit contact points for fields of Medicine: Jörg Teuber for national calls, Susanne Troppmann for international calls).

➢ Working on a funded project?

Ask your PI (the grant holder) to see the sections of the grant proposal that are relevant to frame your work, and possibly the funders' funding principles that will outline expectations for researchers working on the project, including those relating to open and reproducible research practices.

iii. Write / check Research Data Management (RDM) proposal and ethic proposal

➢ Are you acquiring your own data?

LEARN (required)

- how to write a Research Data management (RDM) plan: https://lmu-osc.github.io/FAIR-Data-Management/dmp.html (1h)

Write a Research Data Management (RDM) plan. This will include your detailed plan on how to acquire, store, use and share your data. Much of this will be needed for the ethics approval, so draft this first. The RDM plan can be updated as needed throughout the course of a project. The LMU University Library provides the RDMO tool to create RDM plans that correspond to funders requirement and their team can support you in writing it: rdm@ub.uni-muenchen.de.

Any data acquisition and data analyses you want to do first has to be approved by the ethics committee. To prepare an ethics proposal, please refer to the instructions of this official website: https://www.med.lmu.de/de/fakultaet/wer-wir-sind/gremien-kommissionen/ethikkommission/, and to our additional guidelines about writing appropriate informed consent forms and examples of approved applications (TO DO HYPERLINK WITH LRZ SSO). Additional questions may be addressed to the research coordinator. Only the PI can submit these documents to the ethics committee. This will take approximately 4 weeks to first review, a week to review, and another 3 for edits.

➢ Are you querying existing data?

Familiarize yourself with the dataset. Ideally, in order not to introduce confirmation or hindsight bias, you should familiarise yourself with the metadata and not start plotting the data itself. Specifically, there may be a data dictionary (also called 'codebook'), or other extensive documentation on which variables and their description are included. Make sure you know where your data comes from, how the data was collected and processed, and reflect on whether any of it poses problems for your research question. Finally, check what requirements and resources the data sources have for you (e.g., the open dataset ABCD has many publications advising you on how to handle the data, e.g. population weighing for representativeness (https://doi.org/10.1016/j.neuroimage.2021.118262), integrity of the measures and retest reliability (https://doi.org/10.1016/j.dcn.2023.101311)).

c. Design study

LEARN (required)

- what can make science irreplicable: https://osf.io/mguqj/files/osfstorage (1h)
- how to make science credible: https://osf.io/rhtgn/files/osfstorage (1h)

To understand what may affect the credibility of your results, please go through this introduction to the replicability crisis, and this introduction to open research practices (which also introduces all the tutorials mentioned in this guide and that you will have to go through).

i. Preregistration

LEARN (recommended)
- how to prepare a preregistration: link tba in Dec 2025 (2h)

A lot of research at cBRAIN is exploratory. However, if you do have a hypothesis on the nature, direction and mechanism of your effects, discuss with your supervisor, and consider following a pre-registration template to draft your project*. Ultimately, evaluate with your supervisor the possibility of publicly pre-registering your research project on e.g. the Open Science Framework (OSF) (see the Center for Open Science preregistration page, and this preregistration tutorial tba). If you are concerned about your plan being scooped, you can embargo it for maximally 4 years after which it will automatically become public in order to leave a trace of all projects that were undertaken, whether published or not. You can withdraw any preregistration by adding a public withdrawal note.

* Even if you are performing exploratory research and do not want to submit a preregistration, using one of the pre-registration templates is recommended to support the planning of your project, as it provides prompts for questions you should ask yourself when designing a study.

ii. Data simulation in R to plan your study

LEARN (recommended)
- the R programming language: https://lmu-osc.github.io/introduction-to-R/ (3h)
- how to simulate data in R: https://lmu-osc.github.io/Introduction-Simulations-in-R/ (2h)

To create a detailed pre-analysis statistical plan without looking at existing data (which could introduce confirmation or hindsight biases), it is recommended to simulate data in R based on e.g. the expected distribution of certain variables or based on the metadata of existing datasets. If you haven't already, learn R by following this tutorial. R will be required in upcoming sections (unless you are able to create a completely reproducible workflow using another software like e.g. Python). You can simulate data and add the code to analyse it as part of your preregistration, and reuse it during the analyses of your real data.

iii. Data simulation for power analyses for complex statistical models

LEARN (recommended)

- how to perform power analyses for complex statistical models: https://lmu-osc.github.io/Simulations-for-Advanced-Power-Analyses/ (6h to complete entirely but only the chapter on the relevant type of analyses can be picked, advanced)

Power analysis is relevant for anyone, whether you are designing a project from scratch or just running an analysis on existing data. There are two main types: *a priori* and *post-hoc* power analysis.

1. A priori: If you collect your own data, you may need to think about how many samples or participants you need to collect or recruit. To perform a power analysis to calculate the smallest sample size required to detect the smallest effect of interest, it is recommended to simulate data following this advanced tutorial. For a very basic power calculation, you can also use R or G*Power if you know 3 out of 4 of these parameters: required sample size n (usually the one missing), desired power (default 0.80), the alpha level (default 0.05), and the expected effect size (has to be estimated or extracted from the literature on the form of d, f, etc.). To get support with pre-analysis planning, you can book a consultation with the statistical consulting unit by filling out this form, contact their team at kontakt@stablab.stat.uni-muenchen.de and visit the StaBLab website for more information.
2. Post-hoc power: Oftentimes, you may not be able to control the sample size for your project (e.g., because data collection has ended and you only have a certain number of good quality scans). In this situation, you can compute a so-called “post-hoc power”. **Beware: This power computation comes in two flavors - one is legitimate, and one is flawed and not defensible.** The legitimate post-hoc power is computed with your actual *n*, and *the same effect size that you plugged into your a-priori power analysis*. This analysis gives you the achieved power to detect your assumed effect.
   The flawed version of post-hoc power (see O’Keefe, 2010, or Hoenig & Heisey, 2001) is called “observed power”: If an analysis yields a non-significant result, some researchers calculate the post-hoc power, but plug in the *observed* effect size. “Observed power”, however, is just a one-to-one function of the p-value (a non-significant p-value returns a low power < 50 %, a just significant p-value of .05 always yields a power of exactly 50%). Observed power adds no new information to the p-value and is essentially meaningless. Do not compute this type of post-hoc power!

Check Point 1: Present your plan to your primary supervisor, to your PI and then to the lab!

Short updates (~15min) can be given during the lab meeting “project updates” and should be given at least once per quarter. For the final presentation of your plan* please schedule a Lunch & Learn. During this meeting, a discussion on possible co-authorships and contributorships given the anticipated roles of team members as the project is currently planned should also take place.

* Needs to include: background, literature review, hypothesis, methodology, timeline.

2. Collect & Manage

a. FAIR data management

LEARN (required)

- manage your data following the FAIR principles: https://lmu-osc.github.io/FAIR-Data-Management/ (2h)

Before starting to work with data (folders and files), you are required to read up on FAIR data management and apply these standard practices of organisation and documentation throughout your research to allow yourself, other lab members and collaborators and, in some cases, external researchers, to access, understand and reuse your data.

b. Follow lab standard protocols for data acquisition and storage

i. Data operating / acquisition

Text to add: to list type of data we acquire, and to cite the respective acquisition manuals

ii. Data backup and storage

For all research data, you should follow the 3-2-1 rule of data management: 3 copies of the data, the copies are on 2 different media types, and 1 copy is stored “off-site” or on a completely separate location. Currently the 3 copies are generally: an external drive, the DSS (data science storage, accessible via virtual machines aka VMs) and the KJP data storage centre (login details created for you upon onboarding). These storage locations are specifically chosen and may not be changed. There should not be any data on personal laptops or storage devices. Ideally, the data should be transferred from the drives to the VM in a .zip format every few weeks.

iii. MRI operating

When collecting data at the MRI scanner, always follow the MRI operating manual (TODO: HYPERLINK REQUIRING LRZ SSO) corresponding to your specific project. You should follow closely the file structure and naming described in the manual. Should you encounter problems with the data transfer or saving to the USB sticks or the drives, please notify the cBRAIN computer scientist and the researcher responsible for your project as soon as possible. Within the week after testing, the MRI data collection sheets of all participants you scanned should be entered in REDCAP (https://cbrain.med.uni-muenchen.de/redcap2/).

When working with imaging data, once on the VM, the MRI DICOM files should be passed through the **BIDS conversion pipeline** (see manual TODO HYPERLINK WITH LRZ SSO). After having followed the BIDS conversion manual, you should double check that we have the following backups of the data:

- Raw MRI DICOM files in ".zip" format
- NIfTI format files in ".zip" format (output of the BIDS conversion)

We keep a backup of the raw MRI DICOM files in .zip files, and the converted files after BIDS conversion (in nifti format) also in .zip files. These two formats of the raw data need to be kept in the three backups. This is to ensure that there will always be a copy of the raw data and the converted nifti data in case of any data loss. From this stage onwards for all processing pipelines, you will only be working with the BIDS converted data (NIfTI data). The cBRAIN computer scientist takes care of conservation of the datasets and formats, and your respective questions.

iv. MR Modality-specific processing

The different modalities of MRI sequences each have different processing pipelines. For any of the pipelines you will start working with the data in BIDS format. The updated list of current processing pipelines implemented at the lab are in the wiki (LINK list pipelines WITH LRZ SSO). To this day there are processing pipelines for anatomical/structural (LINK), diffusion MRI (DTI, kurtosis, NODDI TODO LINKS WITH LRZ SSO), ALS (LINK), MRS (LINK), fMRI (LINK), and MRE (LINK). The states of the different pipelines vary as some of them are work in progress. Manuals are available for each modality, together with the respective quality control.

c. Anonymise dataset

LEARN (required) ● when relevant: anonymize your dataset: link tba

Most of our projects involve human data, which requires extra careful handling. It's important to stay aware of the type of information being collected and how it is stored, processed, analyzed, and—if allowed—shared.

Sensitive information, such as participant names, addresses, phone numbers, or facial features, can be accidentally disclosed if anonymization is incomplete. This includes checking tables, image files, and metadata (e.g., DICOM or NIfTI headers).

Depending on the data type, cBRAIN provides specific manuals to guide you in performing or verifying anonymization. (TODO: Link manuals WITH LRZ SSO)

It is your responsibility to review, test, and confirm that the data you are using is properly (pseudo)anonymized. If you are unsure or have questions about your project, please consult the lab's computer scientist or research coordinator.

d. Check data quality

The data quality checks (QC) should be done prior to starting any statistical analyses (needed for the "Data quality validation meeting"). This step is used to exclude subjects early on that might severely bias your results. Depending on the modality you are using, QC is needed for the raw data before processing and/or after pre-processing and/or sometimes after post-processing, this is specified in the respective manuals (TODO LINKS WITH LRZ SSO). There are modality-specific QC manuals that you can find here:

- Structural MRI
- Diffusion MRI
- MRS
- ASL
- fMRI

Some general principles and good practices when QC'ing (visually), especially if many people are helping you out:

- Schedule a meeting with all team members involved in the QC of your data to walk them through the QC manual
- Set up a progress sheet to track the evolution of the QC (shared document accessible to everyone in the team -TODO LINK TEMPLATE WITH LRZ SSO)
- Split the tasks to each team member and make sure they know what they should be doing
- At the end of the QC process, review all the flagged cases and then schedule a meeting with the project leader to discuss them

If there are any QC steps in any of the manuals that are not clear or do not work properly, please reach out to the project leader or the computer scientist, or the people that wrote and designed the QC manuals.

Before moving forward, please make sure that all your data has undergone all the QC steps and that you have a clear list of the datapoints/cases you are excluding and which ones you are keeping for your analyses.

For your data quality validation meeting, you should check that:

- ☐ the data is stored in the appropriate format where it is supposed to be
- ☐ the data is sufficiently anonymised to enter the next stage of collaborative data analyses
- ☐ the data acquisition notes and all flagged subjects from the QC output, and make a preliminary list of cases that should be excluded (and why)

Check Point 2: Data quality validation meeting: with your project's subteam, demonstrate that your data has gone through quality checks, is stored where it is supposed to be in the appropriate format, and decide on final exclusion criteria

Short updates (~15min) can be given during the lab meeting "project updates" and should be given at least once per quarter. Please schedule a Lunch & Learn to present this milestone.

3. Analyse & Collaborate

The data processing and analysis of each project should be reproducible, independently of the programming language, software, and operative system used. Reproducibility is most easily achieved by avoiding point-and-click processes (which are 'manual' and cannot be traced back) and favouring automated processes through programming*. Minimally, this requires adequate documentation and the sharing of code that allows to rederive your results on a different computer, by a lab mate or any other researcher.

* Note: your scripts and data analysis projects are dynamic objects, they can (and will) change due to errors correction, new ideas, feedback from pairs, supervisors and journal reviewers, etc. Therefore, process automatization is key during the script development to save time down the line.

In this section we provide the recommended workflow for users of the R programming language.

a. Process and analyse data reproducibly

LEARN (required)

- R programming language: https://lmu-osc.github.io/introduction-to-R/ (3h) - if not done previously
- Writing readable code: https://osf.io/sbncz/files/osfstorage (1h)
- Version control with Git within RStudio: https://lmu-osc.github.io/Introduction-RStudio-Git-GitHub/ (2h)
- Collaborative version control with Git and GitHub within RStudio: https://lmu-osc.github.io/Collaborative-RStudio-GitHub/ (1h)
- R package management system Renv: https://lmu-osc.github.io/introduction-to-renv/ (1.5h)

Each R project should be based on a .Rproj file created with the free and open source software RStudio (see R tutorial). The folder containing this file is your project directory, or code repository. Your repository should include a standard folder structure, e.g. based on the template https://github.com/jonas-hag/analysistemplates - new link for OSC template (easier to use) https://github.com/lmu-osc/research-project-template to be provided by reema.vbu@gmail.com in July 2025.

R scripts themselves must be structured, commented, and styled following current standards (see lecture on readable code). Namely, you have to use relative paths (i.e. "./subfolder", where . represents the root of your .Rproj directory) or the library here, so the project stays portable to another environment. You have to use meaningful names for variables, functions, and scripts. You also have to add comments to your code explaining why you made a decision, and define your own function rather than copy and pasting pieces of code which makes it hard to maintain error-free. Finally, each R project repository should have renv activated to keep track of all packages and their version (see renv tutorial), so that you or someone else can indeed reproduce your results on another computer at a later time.

Each repository has to be under version control, i.e. each change made to any file should be traced and reversible thanks to the software Git (see Git tutorial). Git is a version control system that tracks changes in source code. The Git software and your git repositories should be, respectively, installed and located in your local environment (i.e. on your computer, not on a drive), and backed up on LRZ GitLab. LRZ GitLab is a cloud-based hosting platform that uses Git for version control, providing collaboration features like pull requests and issues. LRZ GitLab works exactly the same as GitHub, the most popular, but proprietary and US-based cloud-based platform for software development with Git, see GitHub tutorial. However, LRZ GitLab is free and open source and is installed on the LRZ servers and can therefore be considered secure when the repository is private: https://gitlab.lrz.de/users/sign_in (you have to use your LRZ account e.g. ra34tzu). You will, similarly as with GitHub, create a secure connection between your local environment (e.g. your laptop) and LRZ

GitLab with a SSH key (see GitHub tutorial). While your LRZ GitLab account is associated with your LRZ account, your GitHub account can be associated with your private email, be included in your CV, and be used for the code publishing stage (see Figure 2 and section 4).

Your **LRZ GitLab** repository should contain your raw data and/or your processed data, when possible. You are not allowed to have any sensitive data on **GitHub** even in a private repository (Figure 2).

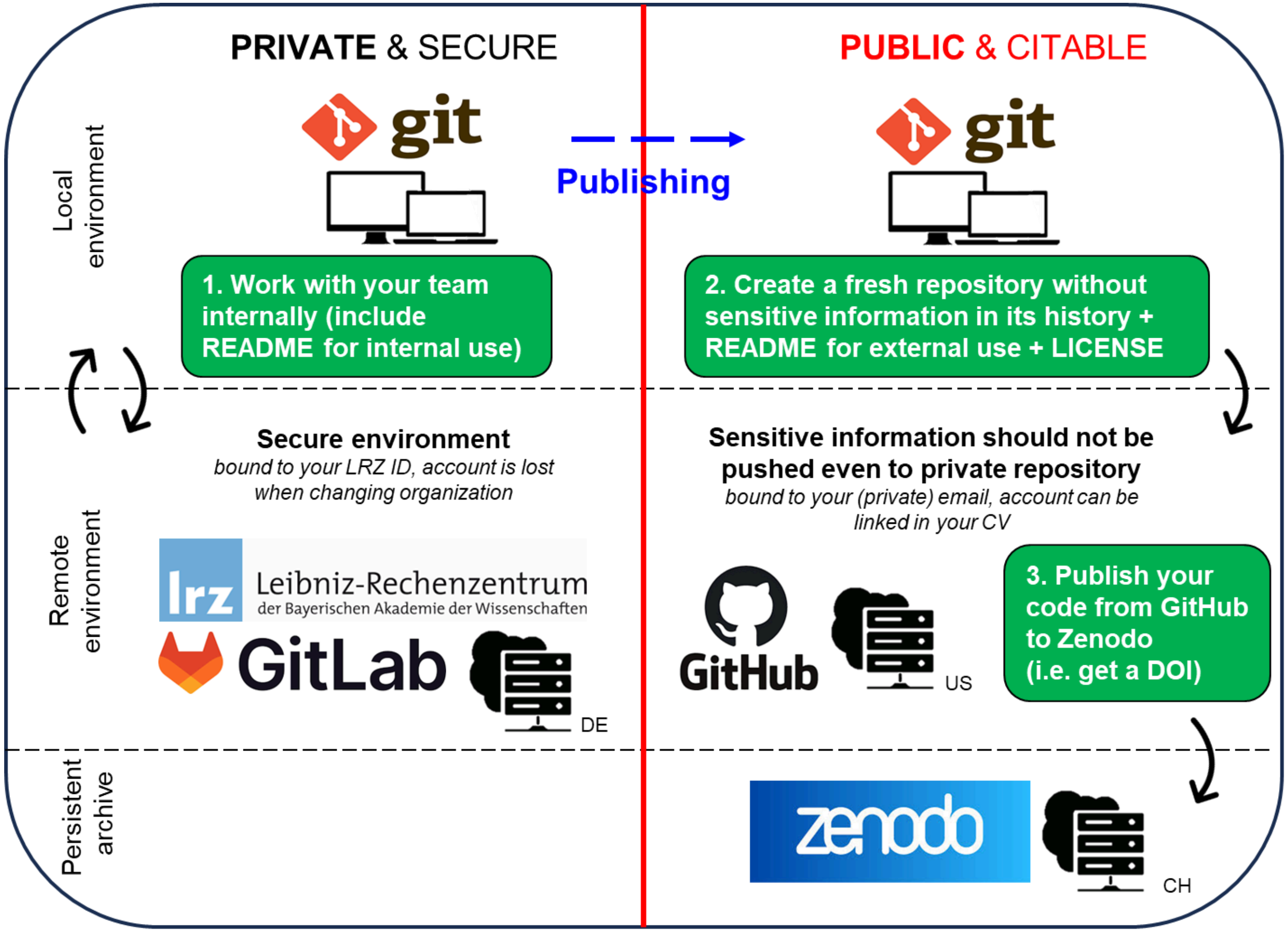

Figure 2. Workflow connecting Git, LRZ GitLab, GitHub, and Zenodo. Start working in your local computer environment (step 1) and as long as you actively work on your project and are writing code, stay in the left part of the figure (PRIVATE & SECURE). Once you are in the publishing process of your manuscript and code, following the instructions of section 4, you can move to the right part of the figure (step 2 and 3). This figure is editable here: cBRAIN-lab-handbook-figure.pptx .

b. Create a data dictionary and write a README file

REVIEW

- manage your data following the FAIR principles: https://lmu-osc.github.io/FAIR-Data-Management/ (2h)

LEARN (required)

- create a data dictionary: https://lmu-osc.github.io/code-publishing/data.html (30 min) - new link after separation from code publishing tutorial to be provided by reema.vbu@gmail.com - and update text below accordingly

As you have read in the FAIR data management tutorial, README files and codebooks (also called 'data dictionaries', which contains each variable and description of what they are) are crucial documentation to accompany any dataset and code repository. You should start them early and regularly update them, first for your own use, then for internal use, then before publication for external use. Detailed guidance on how to create a final data dictionary and a README file for external use is presented in the code publishing tutorial in section 4. Optionally, you can describe the data in detail using the data descriptor template from EBRAINS (template link).

c. Write a reproducible report

LEARN (required)

- Literate programming with Quarto: https://lmu-osc.github.io/introduction-to-Quarto/ (2.5h)

LEARN (recommended)

- reproducible reference management integrated within RStudio: https://lmu-osc.github.io/introduction-to-zotero/ (1h)

Literate programming consists in alternating plain common language (e.g. English) and programming language to provide context to results, tables, and figures generated by the code. With Quarto, one can create reproducible presentations, reports, websites, executable and dynamic manuscripts, using R and /or other programming languages in RStudio or in other software (e.g. Jupyter notebooks). In these various forms of report, results and figures get automatically updated when the R code changes. These presentation and report files are written with simple Markdown syntax to format the text (e.g. heading, bold, ect), and can be rendered into html, pptx, docx, or other file formats.

At cBRAIN, the code repository will eventually be published with simulated, synthetic, or real data, depending on the project, together with the respective paper (see section 4.). You are requested to provide the analysis code that creates the results of the manuscript as a Quarto file to serve as documentation for the code (see Quarto tutorial) and facilitate reproducibility checks (sometimes required by journals) and/or reuse.

Reaching this required step will have given you the necessary knowledge for the following *optional* steps:

- creating a reproducible presentation of the results in Quarto (and rendering it into html/reveal.js or pptx)
- writing your entire manuscript in Quarto (and rendering it as a word document to receive feedback from collaborators and manually integrate them into your Quarto document, or have your collaborators work through collaborative version control in LRZ GitLab). The reference management software Zotero can be integrated in RStudio to cite articles and have them automatically formatted in any journal standards. Bibliography can also be done through .bib files directly sourced in your Quarto document. (see Quarto tutorial and Zotero tutorial). Some journals offer Quarto templates (see list of Quarto extensions) for further formatting requirements. Some journals offer LaTeX or Word templates; you can adapt these by modifying Quarto's YAML header and markdown content to create your own Quarto template.

d. Overview of a code repository structure

At this stage, each data analysis project / code repository should:

- ☐ consist of a R project with the lab's standard folder structure
- ☐ be shared with the LRZ cBRAIN GitLab organization by using the Git version control system, with meaningful and navigable commits
- ☐ use Renv for package management
- ☐ contain raw data and/or processed data
- ☐ have a descriptive README file explaining how to use and run the codes, and in what order
- ☐ contain a codebook/data dictionary
- ☐ contain a Quarto reproducible report

In addition, R scripts should have appropriate:

- ☐ metadata / header to e.g. identify author, state the purpose of the script
- ☐ structure to increase readability
- ☐ informative comments, which do not contain any sensitive information (e.g. password, patient information)
- ☐ meaningful and consistent naming convention

- ☐ style following a programming language guide (e.g. using styler or lintR)
- ☐ functions breaking down actions in readable snippets, and avoiding duplication
- ☐ seeds for any random number generation to allow reproducibility

You must have your repository and your code peer-reviewed by a lab member, your direct supervisor, and/or the computer scientist by giving them access to your LRZ GitLab repository (link to OSC help tool tba).

Any co-authors may additionally want to check:

- ☐ the functionality of the code (e.g. does it technically do what is stated in the header? Are there any errors?)
- ☐ the reproducibility of the results

Check Point 3: Repository and code review with a team member or supervisor

Short updates (~15min) can be given during the lab meeting "project updates" and should be given at least once per quarter. Please schedule a Lunch & Learn to present this milestone.

4. Preserve & Disseminate

When prompted on repositories and article submission system, in order to get credit where credit is due, link all your research outputs to (required):

- your researcher unique identifier ORCID https://orcid.org/ (CREATE YOUR PROFILE)
- your institution unique identifier ROR ID (LMU Klinikum: https://ror.org/02jet3w32)
- your funders ID / ROR ID:
  - ERC (NEUROPRECISE): 804326 - https://ror.org/0472cxd90
  - ERA-NET NEURON (Neu-vasc): 01EW2011
  - BMBF (German Center for Child and Adolescent Health (DZKJ)): 01GL2406A - https://ror.org/04pz7b180
  - NINDS (R01 Sex Differences): R01NS100952 - https://ror.org/01s5ya894

a. Prepare code repository for publication with simulated or synthetic data

LEARN (required)

- steps for code publishing: https://lmu-osc.github.io/code-publishing/ (reproducible report, data dictionary, README, licence, 2.5h)

LEARN (recommended)

- simulation of data in R: https://lmu-osc.github.io/Introduction-Simulations-in-R/ (2h)
- create synthetic dataset link tba July 2025 (2.5h)

So far, your code repository was on a secured LRZ Gitlab server for internal use only. Before the submission of a manuscript to a journal and/or upon the acceptance of a manuscript, there are new steps that need to be done to make your code available on GitHub and publish it (i.e. archive its current version and get a DOI) on Zenodo. Please request the support from the cBRAIN computer scientist as needed.

➢ Do you want to make your code available to reviewers (sometimes required by journals which have reproducibility checks in house, or sometimes requested by individual reviewers)?

This means if reviewers request new analyses to be performed on your real data, you will need to work with your LRZ GitLab version of your repository, and later update your Github public version (see Figure 2) and possibly create a new release (same DOI, but new version, e.g. 2.0).

➢ Do you want to only make your final code available, upon acceptance of your manuscript?

You will keep working in LRZ GitLab until acceptance, and only make the code available publicly and get a DOI for it at the very end. The steps below to publish your code require at least ~ 2 hours to double check everything *if* you had worked reproducibly and completed all preparation prior to that point as was recommended in the previous steps of the handbook, including data dictionary, and simulated dataset generation. If not, this may require days/weeks.

In both cases, the steps to publish your code as are follow:

1. if you work with sensitive data:
    a. (required) create a new local repository that contains all but your raw/processed data (or add this folder in the .gitignore file of this new version controlled repository). Note that if you remove the data of an existing version controlled repository, one can revert the project to a state where the data was still there so it is important to create a fresh repository that never contained the data or where

the data was ignored from the very beginning of its history. If a mistake occurs and some sensitive data does get pushed, you can still rewrite your history retrospectively following this procedure, but best to avoid this altogether.

   b. (recommended) provide a simulated toy dataset (see simulation of data tutorial) that allows to run the code (but not to find the same results) or a synthetic dataset that has similar property as the real dataset, allows to find the same or similar results, and could be reused by other researchers to plan a new analysis of your data (see synthetic data tutorial, tba). Synthetic datasets are superior as they allow more reuse than simulated datasets, but may present a trade off with anonymity. Review the synthetic data tutorial carefully to identify whether you could create a synthetic dataset.
   c. update your code accordingly to use this new data
2. update the README and finalise the data dictionary for external users (see code publishing tutorial /data dictionary tutorial tba)
3. add a licence (see code publishing tutorial)

Before crossing the red line that makes your code public, it is recommended that you check-in with the cBRAIN computer scientist to double check that no sensitive information remains in your repository (e.g. code comments, history of sensitive data).

---

4. create a new empty repository on your Github account and push your new repository to GitHub (see GitHub tutorial)
5. publish on Zenodo with a DOI (see code publishing tutorial)
   a. create a Zenodo account

   ➢ If you want the possibility to create new releases: link your GitHub repository to Zenodo

   b. navigate to your profile > account settings > link external accounts > GitHub > authorize Zenodo to access your GitHub account > a list of your GitHub repositories should appear
   c. enable the repository by toggling the switch next to it > refresh the page to check to see if the visual indicator that the repository is connected appears
   d. create a release on GitHub: When you create a new release on your GitHub repository, Zenodo will automatically archive it. A DOI will be generated for this specific release, making it citable. If you make a new release, it will create a new version on Zenodo, under the same overall DOI, with a 'sub-DOI' to identify a specific version.
   e. update your GitHub README file: copy the DOI for your Zenodo record and paste it into your GitHub repository's README file. This allows others to easily find and cite your archived code

- ➢ If you do not need the possibility to create new releases: create a new upload on Zenodo and indicate the link to your GitHub account. The DOI can be reserved to share in your manuscript for review, prior to making your code repository public (see e.g. https://help.zenodo.org/docs/deposit/create-new-upload/) if you do not want reviewers to access your repository before acceptance.

6. Add the Zenodo entry to the cBRAIN Zenodo community page https://zenodo.org/communities/lmu_cbrain

b. Prepare or publish your real data and/or real metadata

Research data is just as important as the results it produces. Sharing it helps others reuse it, build on your work, and make science more impactful. Publishing your data along with your code also makes your research reproducible and trustworthy and will often lead to collaborations. However, sensitive data, like clinical neuroimaging, must be shared carefully. Privacy, ethics, and legal rules come first. Some data can be fully shared, some only with restrictions, and in some cases, only metadata (information about the dataset) should be published. The data sharing decision tree from OxWIN can help you assess if you are ready to share your data and make some decisions about the sharing process.

TODO: differentiate what it is expected from the students to do vs the computer scientist

Case A: When Data Can Be Shared Openly (e.g. using OpenNeuro) -- THERE IS CURRENTLY NO PROJECT TO WHICH THIS APPLY GIVEN PROPER CONSENT WAS NOT GIVEN - NEED TO SEE IF THIS COULD APPLY TO FUTURE PROJECT WITH APPROPRIATE CONSENT

1. **Data Anonymization and De-identification:** Ensure all personally identifiable information (PII) is removed as per Step 2c of this handbook. Assess re-identification risks, especially when combined with other data.
2. **Prepare BIDS-Compliant Dataset:** Organize your data strictly according to the Brain Imaging Data Structure (BIDS) standard, including a comprehensive dataset_description.json file with rich metadata. This preparation, if done following Step 2b, is essential as OpenNeuro requires BIDS validation. You can use this online BIDS validator for this purpose. No data is uploaded in this process, only the directory structure and file names are used.
3. **License:** Select a permissive open license, such as CC0 (if using OpenNeuro, CC0 is the only permitted license) or CC BY, to maximize reusability.

4. **Ethical and Regulatory Compliance:** Confirm data sharing aligns with the approved ethics proposal (Step 1). Obtain necessary approvals from the relevant ethics committee, ensuring compliance with GDPR and ethical standards before uploading.
5. **Upload to OpenNeuro (or another repository):** Upload your validated BIDS dataset. If using OpenNeuro, the platform will perform further validation checks.
6. **Obtain DOI:** Upon successful validation and publication on OpenNeuro, the platform will assign a persistent Digital Object Identifier (DOI) to your dataset, making it easily citable. For other data repositories, you might need to do this explicitly, for instance, G-node INfrastructure requires you to fill in a metadata file and submit a request.
7. **Update Code Repository:** Add the dataset DOI and clear instructions for accessing and downloading the data to your code repository's README file. Ensure scripts can programmatically download and structure the data as needed for analysis.

Case B: When Data Requires Restricted Access (Using EBRAINS for Metadata & LRZ for Data Storage) -- THIS PROCESS IS NOT IN PLACE YET, IT NEEDS FURTHER INFRASTRUCTURE FOR CONTROLLED ACCESS TO LRZ DSS, APPROVAL FROM ETHICS, LEGAL FRAMEWORK AND SETTING UP AN INTERNAL PROCESS OF THE REQUESTS, BUT IS BEING WORKED TOWARDS - THIS MAY BE A TASK ONLY FOR THE COMPUTER SCIENTIST?

For sensitive data that cannot be shared openly, we use a restricted access model. This involves sharing detailed, non-sensitive metadata via EBRAINS for discoverability, while the actual data remains securely stored on institutional servers (LRZ). Access is granted on a case-by-case basis after a formal request. Critically, EBRAINS curators do not access the sensitive data itself during curation; only the metadata provided by you is reviewed. *It is your responsibility to ensure no sensitive information is accidentally included in the metadata shared for curation.* Please check the terms and policies laid out by EBRAINS here.

1. **Submit a curation request:** EBRAINS offers curation support for step-by-step data sharing. The first step is to submit a request using the provided form. Clearly state in the form that data access is restricted and describe the general conditions or process for requesting access.
2. **Submission review and personal curator:** Within 5 working days, you will be notified if your curation request **is** accepted. If accepted, a personal curator will be assigned to guide you through the metadata submission and dataset annotation process.
3. **Comprehensive, non-sensitive metadata:** This typically involves you providing comprehensive **non-sensitive** metadata, which the curator will then curate using EBRAINS' structured schemas. This metadata must describe the study, methods, and nature of the data without revealing any sensitive or identifying information. A crucial piece of information here is the dataset custodian who will handle requests forwarded by EBRAINS.
4. **Secure data storage (LRZ):** Store the actual sensitive research data securely on the approved institutional servers provided by the Leibniz Supercomputing Centre (LRZ). Ensure storage complies with GDPR and institutional data protection standards. Do not upload the sensitive data to EBRAINS.

5. **Public metadata record:** Upon publication, this curated public metadata record can be used for future reference, making your study findable even when the data itself is not accessible (private). The public dataset page will display only non-sensitive metadata and provide a form for users to request access to the dataset.
6. **Specify access request mechanism (EBRAINS):** Configure the EBRAINS dataset entry so that interested researchers can use a request form. EBRAINS will forward these access inquiries directly to the designated data custodian(s) in our lab.
7. **Ethical and regulatory compliance:** Ensure the entire process (metadata sharing, data storage on LRZ, access request procedure) strictly adheres to the original ethics approval obtained from ethics committee and complies fully with GDPR.
8. **Link code repository to metadata:** In your code repository's README, provide the DOI of the EBRAINS metadata record. Clearly state that the associated data has restricted access and direct researchers to the EBRAINS dataset page for information on how to request access.

Granting access to restricted data (post-request via EBRAINS) - THIS MAY BE A TASK ONLY FOR THE COMPUTER SCIENTIST

When you receive a data access request forwarded from EBRAINS as the dataset custodian:

1. **Consult the PI:** This is a mandatory first step. Discuss the request and the proposed data sharing plan.
2. **Evaluate the request:** Assess the request based on its scientific merit, the requestor's affiliation and stated purpose, and alignment with ethical guidelines and the original consent/ethics approval.
3. **Prepare data subset (If applicable):** If approved, prepare only the specific data subset required to fulfill the request. Re-verify anonymization before proceeding.
4. **Draft Data Access Agreement (DAA):** Use the standard institutional or lab template for a DAA. This legally binding document should clearly define:
    - The specific data being shared.
    - Permitted uses of the data.
    - Data security requirements for the recipient.
    - Prohibition of further sharing or attempts at re-identification.
    - Publication acknowledgments/citation requirements.
    - Duration of data access.
    - Data destruction procedures post-project.
5. **Obtain approvals:** Secure documented approval from your supervisor. Depending on the nature of the request or data, re-consultation with the ethics committee might be necessary, as well as with the legal department.

6. **Generate secure sharable link (LRZ):** Use an approved LRZ service (e.g., LRZ Sync+Share) to create a secure, trackable, and potentially time-limited download link for the prepared data subset. Do not use email attachments or insecure cloud services.
7. **Execute DAA and share data:** Once the DAA is signed by both parties (requestor and authorized institutional representative), provide the secure LRZ download link to the requestor.
8. **Document:** Maintain a clear record of the request, evaluation, decision, signed DAA, data shared (subset details), and the sharing mechanism (link details) for compliance and tracking purposes.

c. Lift your preregistration's embargo (when relevant)

If you did submit a preregistration on e.g. the OSF with an embargo, you may want to lift that embargo, i.e. make the preregistration public which will also create a DOI that you can cite in your article.

d. Publish preprint/postprint/open access articles

The landscape for publishing article open access is constantly changing and depends on the contracts negotiated by the LMU and each publisher. Typically, in legacy / traditional journals, authors need to pay publishers to have their articles made open access upon publication. In such a case, unless your PI can and wants to pay for the article processing charges (APCs) to make your article open access through the publishers (process called 'gold open access'), it is recommended to publish a preprint and/or a postprint and/or release the published article on an institutional repository once the embargo from the publisher is lifted. These procedures, which are free to the authors, fall under the label 'green open access', and are detailed below.

However, note that, since 2025, at LMU, for Elsevier, Wiley, and Springer Nature, authors need to pay for publishing in hybrid journals even for closed articles (for the same price as for open access articles). Therefore, authors choosing to publish there and paying the fee may as well request an open access article from the publisher directly. The current contract conditions and procedures (and any changes that may happen to them) can be accessed here: https://www.ub.uni-muenchen.de/schreiben/open-access-publizieren/publikationsgebuehren/index.html

Recommended steps to increase the accessibility of your article, for free:

- publish your manuscript's conceptual figures under a CC-BY 4.0 licence (e.g. on the Open Science Framework (OSF)) to cite in the submitted manuscript so you can reuse them in future manuscripts without asking permission from the publisher, by indicating the reproduce from DOI under LICENSE in the figure legend.

- submit a preprint (manuscript before acceptance) to your field repository (e.g. MedRxiv) or your institutional repository (LMU Open Access) when allowed by the publisher/journal as checked on https://openpolicyfinder.jisc.ac.uk/ or, if the policy is missing there, as checked on the self-archiving page of the journal website; either
    i. at first submission when e.g. you, the author, want feedback beyond the designated peer reviewers of the publisher. You should then update the preprint later with new revised versions.
    ii. when receiving the review 'minor revisions' so it is already close to the final version
- update the preprint with a postprint (accepted version of the manuscript, non-formatted by the publisher) when allowed by the journal as checked on https://openpolicyfinder.jisc.ac.uk/ or self-archiving page of the journal website. If the publisher only allows you to publish a postprint on your institutional repository, submit it on LMU Open Access
- when given the choice, request your publisher to publish the manuscript under a CC.BY 4.0 license (https://deal-konsortium.de/warum-ccby), so the right to distribution is included (possibly your publisher will request CC.BY. NC so you can't commercialise it but at least you will then be allowed to deposit the manuscript on preprint/postprint server or institutional repository)

To get support with Article Processing Charges (APCs), open access policies, and the institutional repository LMU Open Access, you can contact the Open Access team at open-access@ub.uni-muenchen.de and visit the University Library Open Access page for more information.

Figure 3 summarises some of the steps and terminology used for these different manuscript versions.

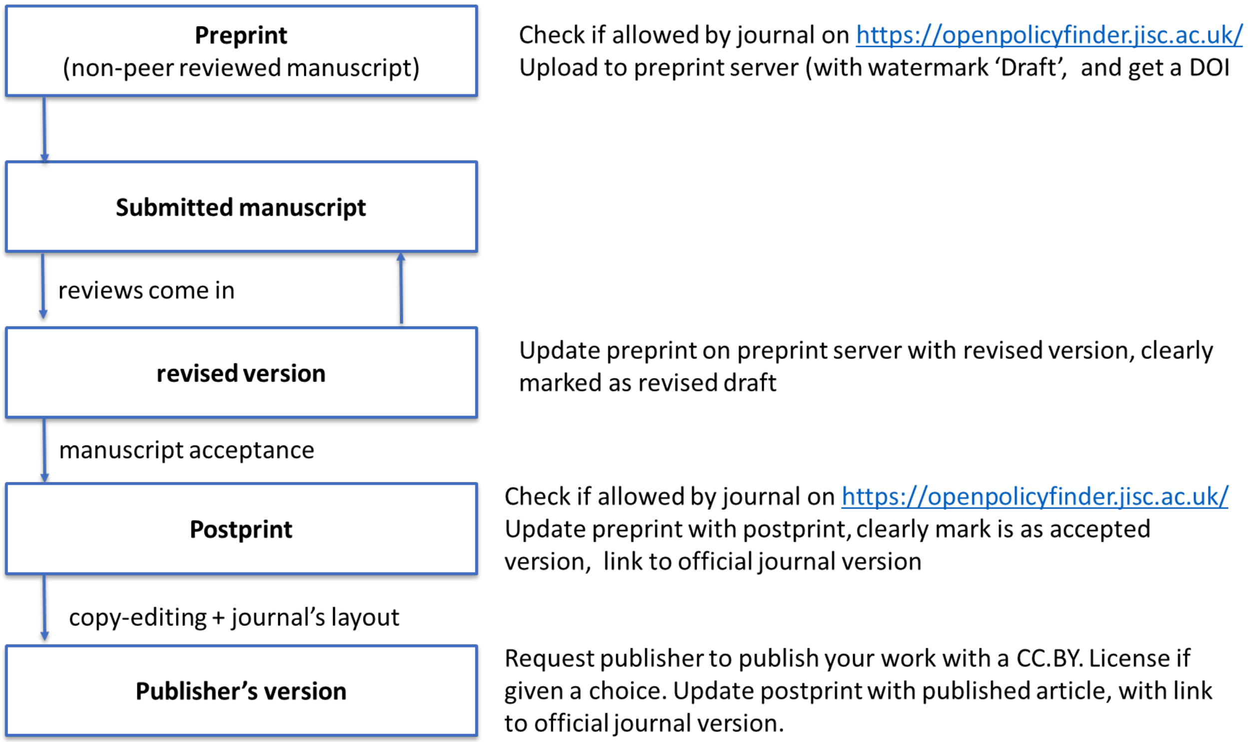

Figure 3. Green open access options for various stages of the manuscript. This figure is editable here: cBRAIN-lab-handbook-figure.pptx.

e. Acknowledge everyone's contribution

Finally, to increase transparency and accountability, and acknowledge what contribution to a research project each contributor has made, we recommend to use the CRediT taxonomy ('Conceptualization', 'Data curation', 'Formal analysis', 'Funding acquisition', 'Investigation', 'Methodology', 'Project administration', 'Resources', 'Software', 'Supervision', 'Validation', 'Visualization', 'Writing—original draft', and 'Writing—review & editing') which is used by many publishers. When not prompted for these contributor roles when submitting your article, you can add the CRediT statement in the acknowledgement section of your manuscript. When relevant, we also recommend to use 'Method Reporting with Initials for Transparency (MeRIT)' i.e. to add initials to the Methods section identifying who did what when several team members contributed to e.g. data curation or software.

Check point 4: Submit manuscript with DOI links to (preregistration), code repository, metadata/data

Typically the DOI are inserted in parenthesis in the text when these resources are very specific to this paper. If you publish a larger dataset or reuse someone else's dataset, you cite it like an article. Journals may have their own standards for this. Before submission, you need to get the consent of all co-authors for all parts of the manuscripts.

Short updates (~15min) can be given during the lab meeting "project updates" and should be given at least once per quarter. Please schedule a Lunch & Learn to present this milestone.

Once submitted, your article will undergo peer-review which may require you to go back to previous steps in the research cycle (e.g. collect new data, perform new exploratory analyses, revise the text of your manuscript). The more reproducible and documented your workflow was, the easiest it will be to revise your manuscript. Once published on a preprint/postprint server, and/or accepted, please provide the citation and the pdf of the article to all co-authors.

Additional neuroimaging resources (suggestions are welcome!)

- NeuroImaging Tools & Resources Collaboratory (NITRC), a hub for software, datasets, and tools in neuroimaging research: https://www.nitrc.org/
- Andy's Brain Book for fMRI beginners: https://andysbrainbook.readthedocs.io/
- Neurohackacademy lecture collection on tools and methods in neuroimaging: https://neurohackademy.org/course_type/lectures/
- Reproducibility in different neuroimaging research scenarios - https://how-would.repronim.org/

Overview homework

These homework are categorised along the research cycle, and need to be done the order specified.

0. Foundations

Order of completion	Level of need to be able to continue the program	Topic	Tutorial or resource link	Estimated duration for completion
1	strongly suggested	Lecture: Replicability crisis	https://osf.io/mguqj/files/osfstorage	1h
2	**necessary**	Lecture: credible research	https://osf.io/rhtgn/files/osfstorage	40 min
3	*optional*	Lecture: Assessing research replicability	https://osf.io/dnym9/files/osfstorage	1h

1. Research planning

Order of completion	Level of need to be able to continue the program	Topic	Tutorial or resource link	Estimated duration for completion
5	strongly suggested	Tutorial: Data simulation in R	https://lmu-osc.github.io/Introduction-Simulations-in-R	2h

6	*optional*	Tutorial: Data simulation in R for advanced power analyses	https://lmu-osc.github.io/Simulations-for-Advanced-Power-Analyses/	6h

2. Computationally reproducible workflow in R

Order of completion	Level of need to be able to continue the program	Topic	Tutorial or resource link	Estimated duration for completion
4	**necessary**	Tutorial: Introduction to R	https://lmu-osc.github.io/introduction-to-R/	3h
7	**necessary**	Tutorial: Introduction to Git version control within RStudio	https://lmu-osc.github.io/Introduction-RStudio-Git-GitHub/	1.5h
8	**necessary**	Tutorial: Introduction to GitHub	https://lmu-osc.github.io/Collaborative-RStudio-GitHub/	1h
9	strongly suggested	Practice: Logging in to your LRZ GitLab account (with LRZ username), connect your SSH key to it (as in the 1st git tutorial when connecting to GitHub), create a remote repository as a test repo, clone it locally in a RStudio project, create a file, commit it, push it, invite I		30 min

		and a colleague as collaborators and explore what access rights you can give us. I'l also add you to the onboarding survey result repository.		
10	*optional*	Video tutorial: Collaborative coding and code peer review	Git Branching and Me…	1h
11	**necessary**	Lecture: clean code	https://osf.io/sbncz/files/osfstorage	45 min
12	**necessary**	Tutorial: Literate programming	https://lmu-osc.github.io/introduction-to-Quarto/	2h
13	**necessary**	Tutorial: package manager	https://lmu-osc.github.io/introduction-to-renv/	1h
14	strongly suggested	Tutorial: reproducible bibliographic reference manager	https://lmu-osc.github.io/introduction-to-zotero/	1h
15	**necessary**	Tutorial: code publishing, readme files, licenses	https://lmu-osc.github.io/code-publishing/	2h

3. Data collection, management, and sharing

Order of completion	Level of need to be able to continue the program	Topic	Tutorial or resource link	Estimated duration for completion
16	strongly suggested	Lecture: reproducible protocols (1)	Report detailed meth…	30 min
17	strongly suggested	Lecture: reproducible protocols (2)	How to write a reusab…	30 min
18	strongly suggested	Lecture: Data sharing	https://osf.io/wv6zy/files/osfstorage	1h
19	**necessary**	Tutorial: FAIR data management	https://lmu-osc.github.io/FAIR-Data-Management/	2h
20	**necessary**	Tutorial: data dictionary	FIXME-REEMA	1.5h
21	**necessary**	Resources: data sharing in your discipline-specific repository	FIXME-REEMA	1.5h

4. Publishing

Order of completion	Level of need to be able to continue the program	Topic	Tutorial or resource link	Estimated duration for completion
22	**necessary**	Lecture: Open access , preprint, postprint	https://osf.io/k52ny/files/osfstorage	40 min
23	**necessary**	Practice: create ORCID and populate https://orcid.org/ + explore https://openpolicyfinder.jisc.ac.uk/ for your favorite journals		20 min