Welcome to The Carpentries Etherpad!

This pad is synchronized as you type, so that everyone viewing this page sees the same text. This allows you to collaborate seamlessly on documents.

Use of this service is restricted to members of The Carpentries community; this is not for general purpose use (for that, try etherpad.wikimedia.org).

Users are expected to follow our code of conduct: https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

---

Sign in: Name (Pronouns), Institution, Email & Twitter (optional)
Please sign in so we can record your attendance.

- Annika Rockenberger (she/her), University of Oslo Library, arockenberger@gmail.com, @ARockenberger (Twitter & GitHub) --- Co-instructor for the Instructor Training
- Konrad Förstner (he/him), ZB MED - Information Centre for Life Sciences, foerstner@zbmed.de, @konradfoerstner --- Co-instructor for the Instructor Training
- Christophe Klopp (he/him), INRAe - Mathématique et informatique appliquée, christophe.klopp@inrae.fr, @ChristopheKlopp --- Trainee
- Mark Abraham (he/him), ENCCS Sweden, mark.j.abraham@gmail.com, @the_mabraham (Twitter), mabraham (GitHub)
- Jesse Kerkvliet (he/him), UMC Utrecht, j.j.kerkvliet@umcutrecht.nl, @jessekerkvliet
- Simon Steinkamp (he/his) - (former) Research Center Jülich, sr.steinkamp@mailbox.org, @simon_steinkamp (twitter), SRSteinkamp (github) -- Trainee
- Luis Moßburger (he/him), private, lmossburger@t-online.de, @LuisInANutshell (twitter), @luismossburger (github)
- Andre Pfeifer (he/him), TU Darmstadt, andre.pfeifer@tu-darmstadt.de, @AndreSilasa (twitter)
- Artem Zhmurov (he/him), KTH, Sweden, zhmurov@gmail.com, @ArtemZhmurov
- Roberto Di Remigio (he/him), ENCCS, Sweden, roberto.di.remigio@it.uu.se, @_boberto_
- Fernando Pozo (he/him), CNIO Spain, fpozoc@cnio.es, @fpozoca (Twitter) @fpozoc (GitLab)
- Lilit Axner , (she/her), ENCCS, Sweden, lilit.axner@it.uu.se, @LilitAxner
- Andre Pietsch (he/him), University Library JLU Gießen, andre.m.pietsch@bibsys.uni-giessen.de
- Mikhael Manurung, Leiden University Medical Center, mikhael.manurung@gmail.com, @mikhaeldito313
- Laura Martinez Gomez (she/her), CNIO Spain, lmartinezg@cnio.es, @laumtnezg
- Joana Viana (she), University of Birmingham, UK
- Naoe Tatara (she/her), University of Oslo Library / Carpentries Regional coordinator for Nordic and Baltic countries, naoe.tatara@ub.uio.no, @naoe-tatara (GitHub), @NaoeTatara (twitter)
- Agnes Brauer (she/her), Goethe University Frankfurt, University Library, a.brauer@ub.uni-frankfurt.de
- Friederike Kühn (she/her), University Library of Freie Universität Berlin, kuehn@ub.fu-berlin.de
- Michal Michalski, Durham University, michalski.m.michalski@durham.ac.uk, github: @topographos2
- Aili Sarre, University Library, UiT The Arctic University of Norway, aili.sarre@uit.no

- Mishka Nemes, The Alan Turing Institute, mnemes@turing.ac.uk, @Mishkanemes
- Camilla Bressan (she/her), The Alan Turing Institute, cbressan@turing.ac.uk
- Luca Di Stasio, KAUST, luca.distasio@gmail.com

Please fill out the pre-training survey at
https://www.surveymonkey.com/r/instructor_training_pre_survey?workshop_id=instructor-training

You can keep track of the time in your current timezone at https://timeanddate.com/worldclock.

I. Welcome

Code of Conduct:
https://docs.carpentries.org/topic_folders/policies/code-of-conduct.html

Introductions

Information for Today's Learners

1. Add your name to the Etherpad above
2. Introduce yourselves! In your introduction, (a) explain your work in 3 words and (b) say something you are proud of (not necessarily related to research or teaching).

Our First Exercise (2 min)
In the Etherpad, write down your name, the best class you ever took (or one class from your top ten, if you can't decide), and what made it so great.

A Brief Overview of the Carpentries

https://carpentries.org/workshops/

Instructor Training Workshop Overview

- How learning works
- Building teaching skill
- Creating a positive learning environment
- Carpentry history and culture

Assessing Trainee Motivation and Prior Knowledge

Background (3 min)
Have you ever participated in a Software, Data or Library Carpentry Workshop?

- Yes, I have taken a workshop.
- Yes, I have been a workshop helper.
- Yes, I organized a workshop.
- No, but I am familiar with what is taught at a workshop.
- No, and I am not familiar with what is taught at a workshop.

Which of these most accurately describes your teaching experience?

- I have been a graduate or undergraduate teaching assistant for a university/college course.
- I have not had any teaching experience in the past.
- I have taught a seminar, workshop, or other short or informal course.
- I have been the instructor-of-record for my own university/college course.
- I have taught at the primary education level.
- I have taught informally through outreach programs, hackathons, laboratory demonstrations, and similar activities.

Key Points:

- The Carpentries are communities of practice. We strive to provide a welcoming environment for all learners and take our Code of Conduct seriously.
- This episode sets the stage for the entire workshop. The introductions and exercises help everyone begin to develop a relationship and trust.
- This workshop will cover general teaching pedagogy and how it applies specifically to the Carpentries.
- Learner motivation and prior knowledge vary widely, but can be assessed with a quick multiple choice question.
- Konrad Förstner: hosted and run serveral Library Carpentry and Software (> 30)
- Andre Pfeifer: Organized/hosted an unofficial OpenRefine Carpentry-Workshop
- Christophe Klopp : No, but I am familiar with what is taught at a workshop. I have taught a seminar, workshop, or other short or informal course.
- Jesse Kerkvliet: Helper in genomics workshops
- Mark Abraham: No, and I am not familiar with what is taught at a workshop. I have taught a seminar, workshop, or other short or informal course, particularly on GROMACS molecular simulation software.
- Laura Martinez: I've been a helper in a 'R for reproducible scientific analysis' workshop hosted at the Spanish National Cancer Research Centre.
- Artem Zhmurov: Several summer/winter schools in Molecular Dynamiics simulations and GPU programming, GROMACS workshops (not Carpentry).
- Luis Moßburger: Sadly none :((finished Bachelors in 11/2019) but spoke to instructors in depth
- Fernando Pozo: Helper in R for reproducibility.
- Agnes Brauer: LC Workshop (twice as participant, once as organizer); I have been a graduate or

undergraduate teaching assistant for a university/college course.
- Mishka Nemes: I attended a workshop on intro to programming
- Simon Steinkamp: Not participated (but familiar) with workshop contet; taugh informally
- Naoe Tatara: Have taken several 1-day self-organized workshops with focus on one lesson: SQL, R, Unix Shell, Git, Python (a bit more than first half of the programming and plotting lesson)
- Roberto Di Remigio: I have not participated in any Carpentries activities before. I have taught in other contexts, e.g. a semester-long course in theoretical chemistry.
- Andre Pietsch: Have taken a Carpentry Workshop on Spreadsheets and OpenRefine some weeks ago. I was a tutor for three semesters while I studied.
- Joana - I have been a helper and taken workshops (R)
- Michal Michalski: No, but familiar with some content relevant to my field - R, geospatial
- Mikhael Manurung: I have never had any workshop before. I mostly learn through reading books / blogposts. I have taught my lab colleagues but only for several sessions and also in an informal setting (kind of like RLadies meetups!).
- Friederike Kühn: I regularly teach in a library: information literacy, citing, citation management software, but no experience in teaching carpentries
- Aili Sarre: Not participated in any Carpentries activities before. Have taught chemistry courses, lab, seminars, theoretical courses.
- Camilla Bressan: I have limited teaching experience in informal settings and I am completely new to the carpentries project. This is the first workshop I attend
- Lilit Axner

No, but I am familiar with what is taught at a workshop.
I have been a graduate or undergraduate teaching assistant for a university/college course.
I have taught a seminar, workshop, or other short or informal course.
I have taught at the primary education level.
I have taught informally through outreach programs

II. Building Skill with Practice

https://carpentries.github.io/instructor-training/02-practice-learning/index.html

The Carpentries Pedagogical Model

Acquisition of Skill

https://carpentries.github.io/instructor-training/fig/skill-level.svg

- Novice
- Competent practitioner
- Expert

Cognitive Development and Mental Models (5 min)
In the Etherpad, write your primary research domain or area of expertise and some aspects of the mental model you use to frame and understand your work. What concepts/facts are included? What types of

relationships are included?

-

Expert:
Christophe Klopp : genome assembly, over 5 years

- beginner : sequencing technologies
- expert : graph path resolution

Agnes Brauer: Text Encoding (TEI), 5 years, started as student assistant

- beginner: the benefits of texts coded in XML
- expert: getting better in XSLT

Joana: Epigenetics, 8 years.

- Begginer: How the DNA is packed in the cells and how epigenetics is important for expression of genes
- Me: How different chromatin marks interact with each other and with DNA modifications

Artem Zhmurov: Scientific software development 10+ years.

- To the beginner: the scientific problem should guide the code design, not the other way around.
- From expert: Better understanding of C++ data management

Roberto: theoretical chemistry 7+ years (master+PhD+postdoc). C++ and Python programming 4+ years

- Beginner: we can simulate chemistry on a computer with the Schrödinger equation
- Expert: 1) storage formats for key data structures in computational chemistry software. 2) Novel and exotic simulation methods.

Lilit Axner: Computer science 15 years

- I can teach: how to use supercomputers
- I would like to learn how the DNA is packed in the cells and how epigenetics is important for expression of genes from Joana as a beginner..

Laura Martinez: genome annotation, 3 years.

- * I'd try to show the basics of what kind of technologies are used as evidence for genome annotation and their limitations
- * More in deep knowledge of specific of some of these techonologies and how to put them all together

Aili: DNA repair mechanisms, chrystallography, phd project
Beginner: Explain the instability of DNA/genomes, then tell about natures way of solving this to protect us from mutations, introduce the concept of proteins/enzymes that constantly repair DNA

Expert: Improve my knowledge on using the advanced applications in chrystallography

Andre Pietsch: Historical Linguistics/Lexicography and Computerlinguistics (studied over several years and worked in projects touching those topics)

- Beginner: Show how language developed by showing differences from hist. language to todays language
- Expert: Showing how to create historical dictionaries that can be searched with computerlinguistic

and texttechnological techniques

Jesse Kerkvliet: Bioinformatics/programming (6 years of uni)

- Beginner: Explain the principle behind comparing sequences like BLAST with visual interface
- Expert: More complex modeling, deep learning

Simon Steinkamp: Neuroimaging analysis (EEG, fMRI) ~7 years with masters:

- Beginer: fMRI: Broad foundation (conceptual) of image generation and analysis - take preprocessing as a given, clicking through
- Raw data, Computatonal modeling / causal analysis, (deep) encoding / decoding schemes, noise modeling / reduction.

Luis: Understanding Library UIs / UX from BOTH perspectives (user & library), 3 years

- Beginner: Show basic differences between library UI & common UIs (also explain term "UI"? :D)
- Expert: In-Depth CSS-Workarounds to adjust UIs that can not be updated because of missing resources ;)

Mikhael Manurung: Single cell analysis, the methods keep changing in a fast pace so not really sure.

- - Beginner: Start from a biological question and then discuss the appropriate tool/workflow to answer it. I would not bother them too much with the details behind the algorithms.
- - Expert: Discuss pearls/pitfalls to the methods so that people can choose the best tool for their problem and characteristic of their data.

Mark Abraham: C++ software development.

- Beginner - understand what the end user wants and how to express the separate ideas needed to deliver a suitable solution
- Expert - choosing the right part of the toolkit for the job, not doing unnecessary things, communicating before doing

Michal Michalski: spatial analysis, MSc plus 8 years experience - in R 2-3 years

- as beginner: break down algorithm, illustrative example, step by step
- as expert: implement the algorithm in programming language

Mishka Nemes: computational modelling, simple statistical frameworks
- Beginner: Understand basic difference between different models
- Expert: Be able to apply the relevant model as you see suited on the data
Naoe Tatara: Hard question as I am jumping from one to another all the time, but I would (probably need to say, if I need to be dare to) that I am an usability expert or mixed-method research. Took 3-4 years.

- - Beginner: what kind of factor/elements of interface, or what you see in screen, makes it difficult to understand or makes you feel uncomfortable to use, preferably with some good/bad examples.
- .- Expert: what kind of theories are behind what human being think/feel interface is easy/difficult to use.

Fernando Pozo: bioinformatics (BSc partially + MSc + PhD partially +5 years) and machine learning (3 years self learning).

- - Beginner: to show the general history of both fields (why these fields were developed in the past?). To teach the FUNDAMENTALS and FOUNDATIONS of programming, statistics and maths.

- - Expert: to focus on PARTICULAR problems that are needed to be solved with some of these techniques.

Friederike: sewing and knitting, mor than half of my life

- -beginner: how to hold needles, in a learning-by-doing-way, demonstrate and imitate
- -expert: how to use technique in creative ways, apart from concrete guides, feel free in combining techniques

Camilla Bressan: ex-computational plasma physisist and data scientist

- - beginner: statistical modelling
- - Intermediate: data analysis, deep learning, programming
- - expert: numerical simulations in the domain of plasma physiscs

The Importance of Going Slowly

https://carpentries.github.io/instructor-training/fig/mental_models.svg

How "Knowledge" Gets in the Way

Misconceptions

- Factual errors
- Broken models
- Fundamental beliefs

Identifying and Correcting Misconceptions
Formative assessment
Summative assessment

Repetition vs.Reflective practice

Formative Assessments Come in Many Forms

Identify the Misconceptions (10 min)
Choose one of the wrong answers to the question below and write in the Etherpad what the misconception is associated with that wrong answer.
Q: what is 27 + 15 ?
a) 42 correct **Answer to the Ultimate Question of Life, The Universe, and Everything**
b) 32
20+10+7+5=30+7+5=30+5+2+5=40+2, so 7+5 is over 10 and the ten needs to be remembered.
, forget to carry the 1
c) 312 (7+5=12) + (20+10=300) / (2+1), (7 + 5) = 3, 12 = 312, sum up the first two numbers and then the second two numbers and ten just merge them together
d) 33 27 + (1+5) = 27 + 6 = 33, sum up the second two numbers first and then just add them to the first number

Handling Outcomes (10 min)
Formative assessments allow us as instructors to adapt our instruction to our audience. What should we do as instructors if the class votes for:

1. mostly one of the wrong answers?
2. mostly the right answer?
3. an even spread among options?

For one of the above, enter your answer below:

-
-
-
-

Modeling Novice Mental Models (10-15 min)
Take 10 minutes to create a multiple choice question related to a topic you intend to teach. Type it into the Etherpad and explain the diagnostic power of each its distractors, i.e., what misconception is each distractor meant to identify?

Formative Assessments Should be Frequent

How Many? (5 min)
The Carpentries use formative assessments often. How many have we done since the start of this workshop? Put your guess in the Etherpad along with one example and the purpose that assessment served.

-
-
-
-

Confronting the Contradiction (Optional)
Describe a misconception you have encountered in your own learning or teaching and how to get learners to confront it.

Key Points:

- Our goal when teaching novices is to help them construct useful mental models.
- This requires practice and feedback.
- Formative assessments provide practice for learners and feedback to learners and instructors.

III. Expertise and Instruction

https://carpentries.github.io/instructor-training/03-expertise/index.html

What Makes an Expert?

What Is an Expert? (5 min)

Name someone that you think is an expert (doesn't matter what they're an expert in). As an expert, what makes them special or different from other people? What is something that you're an expert in? How does your experience when you're acting as an expert differ from when you're not an expert?

- Gael Varoquax (scikit-learn, nilearn): Increadible knowledge of the code-base and how these things are connected, as well intricacies, of bugs, formulations etc. (and where these things are). Plus: Great communicator from novice to expert level, different audiences.
- An expert has background knowledge, overview of the sitation, can predict outcomes, can be creative with the concepts and apply the knowledge to parallell situations/topics. When acting as an expert one has more confidence and can speak with authority
- Peter Gill (quantum chemist): can make the most technical topic approachable to non-experts. As an expert he has many years of practice in the field and a comprehensive overview of how different things are connected. But he is also a brilliant teacher and presenter.
- Andy Priestner (Library UX), open-minded & loads of different experiences
- François Chollet (Google AI researcher, Keras creator): Great technical knowledge in the field. It has a special talent (and lots of years of dedication) programming neural networks. He has created easier ways to use complex neural networks for the general public.
- Peter Jackson (Regisseur): he knows how to make great movies which you can see by how many people like his movies
- Jacob Nielsen (UX), long history of his special interest and experiences of many different cases (familiar to many people from novices to experts) plus theories,
- My mother in frying potatoes. She always gets it right.
- Heng Li : developper of bwa samtools minimap2 hifiasm, very good understanding of the user needs, of how-to use data structures (minimizers, graph structures), how to optimise code on the CPU level (data structure, short cuts)
- An expert ist very familiar with his topic, has self assurance, his/her teaching is well structured

- Jon Mill. He has knowledge of the broad picture as well as the detail of his field of expertise. He can put other people's work in the context of his own work and vice versa. He can keep an eye of the overall objective of his science. He can teach other with confidence, while also being able to recognise the limitations of his expertise.
- My thesis supervisor. He has the ability to look for answers beyond our normal procedures or analysis, and also he is more confident about what he knows
- My advisor, Omar May (numerical mathematics and plasma physics). He has vision, physical intuition and hard mathematical knowledge. He can be asked about practically everything. He also has the humility to keep attending new lectures and courses, given the occasion.
- Expert knows that there is even more to learn than he/she knows (!) and is able to explain what he/she knows in any elementary level to be understandable for anyone. I am expert in say project managementand organisational skils and that means I can take the reponsibility to explain the processes understandable for anyone as well as I am passionate on learning moe tips and tricks as things develop with time.
- An experts knows how to approach a topic to suit different audiences: they will know teaching a kid, talking in a public conference, or discussing with peers in a scientific conference, do require different levels of pitching their knowledge and skills.
- Prof Iain Woodhouse, Professor in Earth Observation - can explain Radar Remote Sensing in approachable way relating to everyday objects (laser beam, iphone etc) and creating informative animations.(Michal Michalski)
- selfconfidence, apply knowledge to new and unknown situations

Connections and Mental Models

Limitations of Expertise

Fluid Representations (5 min)
Give at least one example of a fluid representation that you use in your own work. If you can, also give an example of a fluid representation that might occur in a Carpentry lesson.

- The confusion between RStudio and R, in genome assembly the works 'contigs' and 'the assembly', using 'DNA bases' and 'nucleotides'
- Co-design concenpt is different for programmers and for architects for example.
-
- Metadata for librarians: related to form they type in,
- Metadata for IT-guys: XML file

Diagnosis (5 min)
What is an error message that you encounter frequently in your work? (These are often syntax errors.) Take a few minutes to plan out how you would explain that error message to your learners. Write the error and your explaination below.

- The 'No such file or directory' error in R. Encourage people to read the error message first and see what they understand the error from the criptic messages R gives. Add that when you encounter an error so hard to understand you can google it and it's likely someone already came across the same error.
- "text not allowed here; expected the element end-tag or element "ab", "addSpan", "alt"..."
- cannot recall the message itself, but I frequently got error message that tells only where the program stopped in Google Apps Script.
- "301 Moved Permanently" => MIGHT be error (new URL given?)
- oom kill. : SLURM killed your process because you did no book enough memory for you job
- 'Unexpected } on line xxx'
- C++ template compilation errors are a nightmare to debug (they can be 100s of lines long) I would suggest to a) write the whole error message to file; b) analyze the first and the last line for useful hints; c) always suggest to disregard errors from the standard library, the fault is most likely in user code and errors from standard library are red herrings to follow.
- 404 not found error: the requested webpage does not exist anymore; maybe there is a typo in the url or the page has moved and can found under another link
- segmentation fault: huge problem. The program has broken at some point, probably it was trying to access bits of memory which are not available
- File not found error. Learner may get confused when they confirmed that the file does exist in the file explorer. We can then remind the learner to be aware of the current working directory and make sure that the file path was written correctly w.r.t. the working directory.
- dplyr select() function
- Software dependencies are not installed properly.
- IndexErrror: list index out of range: The list is empty. In many cases backtracking this error, this was related to the wrong path of a preprocessing script, so no data was loaded.
- (sorry for the C++ template error messge!)
- In file included from ../src/gromacs/commandline/cmdlineinit.cpp:45:
- In file included from ../src/gromacs/commandline/cmdlineinit.h:47:
- In file included from /usr/lib/gcc/x86_64-linux-gnu/10/../../../../include/c++/10/memory:83:
- /usr/lib/gcc/x86_64-linux-gnu/10/../../../../include/c++/10/bits/unique_ptr.h:83:16: error: invalid

application of 'sizeof' to an incomplete type 'gmx::DataFileFinder::Impl'

- static_assert(sizeof(_Tp)>0,
- ^~~~~~~~~~~
- /usr/lib/gcc/x86_64-linux-gnu/10/../../../../include/c++/10/bits/unique_ptr.h:361:4: note: in instantiation of member function 'std::default_delete<gmx::DataFileFinder::Impl>::operator()' requested here
- get_deleter()(std::move(__ptr));
- ^
- ../src/gromacs/utility/classhelpers.h:179:7: note: in instantiation of member function 'std::unique_ptr<gmx::DataFileFinder::Impl, std::default_delete<gmx::DataFileFinder::Impl> >::~unique_ptr' requested here
- class PrivateImplPointer
- ^
- /usr/lib/gcc/x86_64-linux-gnu/10/../../../../include/c++/10/bits/unique_ptr.h:361:4: note: in instantiation of member function 'std::default_delete<gmx::DataFileFinder>::operator()' requested here
- get_deleter()(std::move(__ptr));
- ^
- ../src/gromacs/commandline/cmdlineinit.cpp:77:33: note: in instantiation of member function 'std::unique_ptr<gmx::DataFileFinder, std::default_delete<gmx::DataFileFinder> >::~unique_ptr' requested here
- std::unique_ptr<DataFileFinder> g_libFileFinder;
- ^
- ../src/gromacs/utility/datafilefinder.h:255:11: note: forward declaration of 'gmx::DataFileFinder::Impl'
- class Impl;
- ^
- 1 error generated.
- Nothing about the message directly suggests the actual problem (that the compiler is trying to implement a default destructor without access to the required pieces). Reading the documentation for PrivateImplPointer does provide the answer, but that is in the middle of the series of messages. Often the first and last piece are needed to guide the search for an answer, but sadly nowhere does the message hint about implementing the default destructor.
- You need access to view this - most often sharing internal links with external people without changing the access credentials
- "No such file or directory" in shell: maybe just a typo; not in general, but in this special location
- **dr**.x=**r1**.x-**r2**.x; **dr**.y=**r1**.x-**r2**.y; **dr**.z=**r1**.x-**r2**.z;
- ‘could not find function’ - missing library, or functions masked by other library
- "The markup in the document preceding the root element must be well-formed."

Expert Awareness Gap (5 min)

Is there anything you’re learning how to do right now? Can you identify something that you still need to think about, but your teacher can do without thinking about it?

-
-
-
-

Think about the area of expertise you identified earlier. What could a potential **Expert Awareness Gap** be?

-
-
-
-

Dismissive Language

Changing Your Language (5 min)
What other words or phrases can have the effect of demotivating learners? What alternatives can we use to express this meaning in a positive and motivational way? In the Etherpad, make a list of demotivating words/phrases and alternatives.

- "Yes, but" --> "Yes and"
- "You know that already"
- "All you need to do, is..."
- "Let's do an easy exercise..."
- "Oh, that's very simple"
- "As I already mentioned" when someone asks a question about something they didn't pick up on
- "I'm sure you all already know that"
- "Obviously" -> omit. "Clearly" -> omit. "As you know" -> mention why one might know. "As I mentioned earlier" -> first explain, then maybe add a pointer to previous content. "In yesterday's class" -> same. "It can be seen" -> explain.
- I've shown it 5 minutes ago
- "It's easy to show A follows from B"
- "You know that!" and expecting someone to remember when he/she asked a question
- "The rest of the proof is left as an exercise"
- "The proof in the general case easily follows from ..."
- "It's actually very simple"
- "Simply", "Easily"
- "As I already told"
- as you know from primary school/kindergarten
- You should know this
- "everyone knows"
- "It's actually very easy to do"
- you probably heard that before
- "it's common knowledge"
- "That's trivial"

You Are Not Your Learners

The Importance of Practice (Again)

Key Points:

- Experts face challenges when teaching novices due to expert blind spot.
- Expert blind spot: knowing something so well that it seems easy when it’s not.
- With practice, we can learn to overcome our expert blind spot.

IV. Memory and Cognitive Load

https://carpentries.github.io/instructor-training/05-memory/index.html

Types of Memory

Test Your Working Memory (5 min)
This website https://miku.github.io/activememory/ implements a short test of working memory. In this test, you will see about twenty words, each for a short amount of time. Try to memorize as many as you can.

What was your score? Write your answer below.

-
-
-

Strategies For Memory Management

Improving Short-term Memory with Chunking (5 min)
Repeat the memory exercise you did earlier, but this time, try to form short stories or phrases from the words you see.
Write the number of words you remembered in the Etherpad. How does this compare with your first attempt?

-
-
-
-

Active Learning Through Formative Assessment

Concept Maps as Instructional Planning Tools

Example concept maps:
https://carpentries.github.io/instructor-training/fig/array-math.png
https://carpentries.github.io/instructor-training/fig/conditionals.png
https://carpentries.github.io/instructor-training/fig/create-destroy.png
https://carpentries.github.io/instructor-training/fig/dict-set.png
https://carpentries.github.io/instructor-training/fig/io.png
https://carpentries.github.io/instructor-training/fig/git_concept_map.png
https://carpentries.github.io/instructor-training/fig/lists-loops.png

Concept Mapping (10 min)
Create a hand-drawn concept map for a part of a Carpentries lesson you would teach in five minutes (ie. the amount of material you would teach before doing a formative assessment). You can use the same subject about which you created a multiple choice question, or a different subject. Trade with a partner, and critique each other's maps. Are there any concepts missing in your partner's map that you would include? Are there more than a handful of concepts in your map? If so, how would you re-divide those concepts to avoid overwhelming your learners' working memory?
Take 10 minutes to draw the concept maps and share with your neighbor. Write "*done*" in the Etherpad chat once you have finished.

Other Uses of Concept Maps

Why Guided-Practice is Important

Faded Examples

```
# total_length(["red", "green", "blue"]) => 12
def total_length(words):
    total = 0
    for word in words:
        total += len(word)
    return total

# word_lengths(["red", "green", "blue"]) => [3, 5, 4]
def word_lengths(words):
    lengths = ____
    for word in words:
        lengths ____
    return lengths

# concatenate_all(["red", "green", "blue"]) => "redgreenblue"
def concatenate_all(words):
    result = ____
    for ____ in ____:
        ____
    return result

# acronymize(["red", "green", "blue"]) => "RGB"
def acronymize(words):
    ____
```

Create a Faded Example from a Lesson (10 min)
The following exercise should be done in groups of 2-3.

1. Pick a block of code from an existing Carpentries lesson, or from another lesson you have taught recently.
2. Replace 2-3 pieces of the code with a blank.

3. Write a question to test the student’s ability to correctly fill in that blank.
4. Paste your faded example in the Etherpad.

Summary

Key Points

- Most adults can store only a few items in short-term memory for a few seconds before they lose them again.
- Things seen together are remembered (or mis-remembered) in chunks.
- Teaching consists of loading short-term memory and reinforcing it long enough for items to be transferred to long-term memory.
- Use formative assessments to avoid overloading short-term memory.

One think you have learnt

- Listing the concepts before teaching can be very useful to keep their number limited for one lecture or lecture fragment.
- tool of diagrams.net. I think this might be extremely useful for my next github project. More generally, the importance of concept maps and formative assessment. Avoiding dismiss language is important as well.
- .
- .concept map is a diagram that depicts suggested relationships between concepts

.

- Frequent formative assessments. Online tool for drawing diagrams.
- Formative check - every 10/15 min
- Mind not using simple words like "just, you know, it is simple etc."
- formative checks and diagnostic multiple choice questions
- concept mapping is different from mind mapping +3
- Avoiding dismissive language or better even knowing that there is something like dismissive language in teaching.
- Try to reduce cognitive load as much as we can. Focus on the most important part. Now I see the importance of "filling in the blanks" type of code exercises.
- Do a formative check frequently
- keep it slowly
- Not making assumption about the learner's knowledge

Better digital types of feedback
Awareness about dismissive language
Formative checks
To be aware of the knowledge gap between the instructor and the students when facing errors, using concept maps, and talking to them
Break down an error and involve people to think about what caused the error (not only the one who got the trouble)
Good tips about assumptions for the students knowledge. Great tools for digital learning.

- Learning about dismissive language was very important, I had never thought about it.

V. Building Skill with Feedback

https://carpentries.github.io/instructor-training/06-feedback/index.html

Surveys
For links to our surveys see: https://carpentries.github.io/instructor-training/06-feedback/#surveys

Minute Cards

One-Up, One-Down

Give Us Feedback (5 minutes)
Write one thing you learned this morning that you found useful on one of your sticky notes, and one question you have about the material on the other. Do *not* put your name on the notes: this is meant to be anonymous feedback. Add your notes to the pile by the door as you leave for lunch.

Key Points

- Give your learners time to fill out the post-workshop survey at the end of your workshop.
- Take the time to respond to your learners' feedback.

VI. Motivation and Demotivation

https://carpentries.github.io/instructor-training/08-motivation/index.html

Creating A Positive Learning Environment

- Presenting the instructor as a learner.
- Establishing norms for interaction.
- Encouraging students to learn from each other.
- Acknowledging when students are confused.

Teach Most Useful First
https://carpentries.github.io/instructor-training/fig/what-to-teach.png

Authentic Tasks: Think, Pair, Share (10 min) - Breakout Room

Think about some task you did this week that uses one or more of the skills we teach,
(e.g. wrote a function, bulk downloaded data, built a plot in R, forked a repo) and explain how you would use it (or a simplified version of it) as an exercise or example in class.
In the group decide where this exercise fits on a graph of "short/long time to master"
and "low/high usefulness".
In the class Etherpad, **Share** the task and where it fits on the graph.
As a group, we will discuss how these relate back to our "teach most immediately useful first" approach.

Tasks:

- Creating an RSS-Feed (medium useful (depends on your (communities) preferences, fast to learn (if you stick with copy & paste rather than explaining XML first))
- Make a barplot in R. It would be very useful and easy to learn if the dataset and the pre-processing analysis is simple or straightforward, otherwise people might get stuck in the details of the analysis instead of learning how to actually make a barplot, which is not that hard
- Using list comprehensions instead of explicit loops in Python. This would be low usefulness/high mean time to master
- How to use NumPy arrays to "pack" similar function arguments, e.g. use a N*3 array of atomic positions as funciton argument, rather than N*3 separate arguments. This is in the middle range of both usefulness and mean time to master.
- Make a git pull request with the command line interface.
    - Having original repository as "upstream" and the forked repository in your GitHub account as "origin" (to catch up the progress made in the original repository and gets synchronized)
- Importing data/ tables, reading different formats of data, simple plots, simple statistical tests (by order of difficulty)
- Differential gene expression analysis. Show the final step of the analysis first, i.e. performing differential expression tests and then plot the results as volcano plot or heatmap, and then show how to perform a proper preprocessing of your data to ensure you have a good, clean data to be analysed. DESeq2 is actually very simple to run; they have optimised the package in a way that you only need to run two lines of code (one line to read the data and another to run the analysis).
- MPI communication patterns and parameters in the middle of the graph
- genome polisinhg : don't bother in the first place

Other Motivational Strategies

Strategies for Motivating Learners (5 min)
How Learning Works by Susan Ambrose, et al, contains this list of evidence-based methods to motivate learners.
Pick one of these points and describe in one sentence in the Etherpad how can we apply these strategies in our workshops.

- Strategies to Establish Value
    - Connect the material to students’ interests.
        - Ask for the students backgrounds in advance.
        - Know their background and try to relate to it, using examples or datasets they would feel more comfortable with
        - create dataset domain specific
    - Provide authentic, real-world tasks.
        - Bring your own data segment. Showcase how you would analyse such data through live coding.
    - Show relevance to students’ current academic lives.
        - Show how a topic is discussed in the current literature.
    - Demonstrate the relevance of higher-level skills to students’ future professional lives.
        - use anecdotes
    - Identify and reward what you value.
        - Use a step-by-step walkthrough with comments about general/personal practices
    - Show your own passion and enthusiasm for the discipline.
        - Tell an anecdote that makes people smile & appreciate what they learn :)

            - talk about your learning history, how you mastered difficult tasks
    - Strategies to Build Positive Expectations
        - Ensure alignment of objectives, assessments, and instructional strategies.
        - Identify an appropriate level of challenge.
            - Subdivide the final task into smaller, easy to tackle, chuncks. The learner will be able to grasp the corresponding knowledge one bit at a time and won't be overwhelmed by the amount of material.
        - Create assignments that provide an appropriate level of challenge.
            - Ask the participants about their previous experience (maybe with a pre-workshop survey)
        - Provide early success opportunities.
                - demonstrate something and then let the learners do it by themselves
                - start with easy exercises for each section
                - provide an advanced script in the beginning and provide details later (as in fastai, build an image classifier in 5 lines of code)
                - talk about your own experience
        - Articulate your expectations.
            - set learning outcomes at the beggining or for each task
        - Provide rubrics. Not clear for me = structuration?
        - Provide targeted feedback.
            - prepare students to expect feedback, start to make them feel safe enough to receive it, collect student work, analyze it, provide feedback either privately or anonymously *and* collectively
        - Be fair.
        - Educate students about the ways we explain success and failure.
        - Describe effective study strategies.
    - Strategies for Self-Efficacy
        - Provide students with options and the ability to make choices.
            - explain what works for you, what might work for others, and let them self-assess themselves on that is the best choice for their learning because we all have different approaches.
        - Give students an opportunity to reflect.
            - Have time on exercise to work on their own settings by applying what they learned in the workshop.

Brainstorming Motivational Strategies (5 min)
Think back to a computational (or other) course you took in the past, and identify one thing the instructor did that motivated you. *Share* the motivational story in the Etherpad.

- showing what is possible if you are good with the thing he/she teaches (for example programming with Python)
- making clear that you learn step by step and not everything from the beginning
- The teacher would always provide direct feedback on our assignments.
- I could see how passioante they were about their teaching, which infused some of that passion into my learning
- I was always amazed bt the confidence of my university professors. I took many of them as role-models and wanted to emulate their passione and their achievements

- Recently, in fastai, stressing that many "complicated" maths equations can be easily implemented and understood in code.
- The trainer showing humbleness and that they make mistakes as well
- The instructor used humor
- The instructor told a relevant anecdote from their own beginner's phase
- I remember the instructor of one programming course asking us to bring our own data to analyse
- The Swedish trainers came with salted licorice. You got to eat one if you did not find the answer.
- In my first programming course, the teachers set up an experience points system with level-ups that you would get weekly
- My coleague used to give the students an opportunity to select specific molecule to compute the binding free energy in the tutorial. That introduced an extra interest for them in a shpae of competition - who guest the best molecule..
- positive error culture
- The instructor was live-coded by saying, "I am pretending to be a good coder and Google it" when there was something that s/he did not understand why the error happened.
- In my opinion, to demonstrate live coding techniques and abilities is really motivating for the students. Moreover, mapping coding examples to real live examples is useful.
- the instructor was using stackoverflow himself, showing that he also needs help
- Showing how to perform image classification with state of the art results in five lines of code and with no background in advanced math (Jeremy Howard @FastAI)
- telling stories about failure! (positive failure)

Why Do You Teach? (5 min)

We all have a different motivation for teaching, and that is a really good thing! We want instructors with diverse backgrounds because you each bring something unique to our community.

What motivates you to teach? Write a short explanation of what motivates you to teach. **Save this as part of your teaching philosophy for future reference.**

- To pass on knowledge. I'm only to able to do what I like because others taught me what they knew. It's also a way to get out of your confort zone and to identify talent in other people.
- I would have loved to see someone teaching these things when I started learning, and I hope I can motivate other people to keep developing their own skills the same way other instructors encouraged me back then.
- I is a great way to become familiar with the topics teaching (own learning). It is great to feel that one contributes to someones development.
- Because it is the best way to learn and really go into details.
- Because I like enabling others to succeed and it helps me learn. It's also a moral obligation on those who benefitted from others' teaching.
- I like to give the skills i have to others and want to get even better by interacting with them
- Motivate people! (I have deep passion for what I do & what to share that!)
- Direct motivation: to contribute to increase teaching capacity of the university
- Intrinsic (or internal) motivation: 1) instructing at Carpentry workshops look fun and seem very useful for myself to learn more.
- Makes me feel younger.
- I like inspiring others because others inspired me - a bit of a giving back
- to pass on the knowledge I received over the years, to attract young talents into a particular discipline. Also, to understand better myself the topics I'm teaching.
- It is great to see people understand new things and using them and it is also important to learn

about the problems they have to solve.
- Brings me joy, to see others understand and maybe enjoy/appreciate a difficult topic or something new
- To continue learning and to understand better what is motivating for the student to improve teaching.
- To enable other people to succeed in technical topics
- to learn from others, to see diffrent angles on the subject
- want to improve data literacy, leaving my comfort zone
- To consolidate and structure what I have learned so far on my own and then to impart the pearls/pitfalls that I encountered throughout my self-learning journey.
- one learns better when one teaches

How Not to Demotivate Your Learners

Brainstorming Demotivational Experiences (5 min) - Breakout Room
Think back to a time when you were demotivated as a student (or when you demotivated a student). *Pair* up with your neighbor and discuss what could have been done differently in the situation to make it not demotivating. *Share* your story in the Etherpad.

- Not being aware of mental load / need for breaks of the students
- laughter on a person (taken as touching upon a personal ability and capacity)
- Overestimating ability of students
- Unprepared teachers
- Not good at entertaining the topics
- teachers using dismissive language like: "You should already know this by now."
- starting lessons with jargon
- excessive demands
- not knowing the purpose of the thing that is teached / that you should learn
- teaching person is not prepared / is not taking the teaching seriously
- technical problems
- infrastructure not robust
- When you feel everyone already knows what the lesson is about and you feel like the only one not catching anything
- instructor in a non-beginner class focuses on bringing up participants who didn't come with the expected background knowledge
- instruction doesn't acknowledge that different participants have arrived from different paths ("you should know this already by now") - some gaps are expected (so long as a student doesn't only have gaps ;-))
- learning breakdown
- assignments unrelated to topics in the lecture, e.g. more difficult than the shown examples
- not breaking it down, because enjoying how complex this is and how smart you (the instructor) are
- being told I won't be able to learn this because of my gender WHAT?! How rude.
- teacher is badly organised
- teacher is selfconcentrarted
- topic is very difficults than I thought or vice versa less informatoive than expected
- audience is more experiences and I am novice
- being laughed at and attacked at a personal level. At that point you need to detach from the

teacher, otherwise you'll never get throught that topic.
- Assuming previous knowledge and using jargon without explaining
- Not explaining why something is important
- Teacher is extreamly boring or does not care about the subject that they are teaching.
- When someone is presenting their progress in the lab group meeting and got harshly criticised. In the end students become afraid of giving a presentation because they viewed the presentation as a killing field instead of a place where people can freely share their ideas.

Psychological Demotivators

- Stereotype Threat
- Impostor Syndrome
- Accessibility Issues
- Lack of Inclusivity

Learning About Accessibility (5 min)
The UK Home Office has put together a set of posters (https://github.com/UKHomeOffice/posters/blob/master/accessibility/dos-donts/posters_en-UK/accessibility-posters-set.pdf) of dos and don'ts for making visual and web-based materials more accessible for different populations. Take a look at one of these posters and put one thing you became aware of in the Etherpad.

-
-
-
-

What Happens When Accessibility is an Issue? (Optional, 5 min)
Think of a time when you've been affected by, or noticed someone else being affected by issues with accessibility. This may have been at a conference you attended where the elevator was out of service, or maybe a class you were taking relied on audio delivery of content. Describe what happened, how it impacted your (or someone else's) ability to be involved and what could have been done to provide better accessibility in this case

-
-
-
-

Inclusivity
Key Points

- A positive learning environment helps people concentrate on learning.
- People learn best when they see the utility in what they're learning, so teach what's most immediately useful first.
- Imposter syndrome is a powerful force, but can be overcome.
- Accessibility benefits everyone.

VII. Mindset

https://carpentries.github.io/instructor-training/09-mindset/index.html

The Importance of Mindset

Does Mindset matter? (5 min)
Think: What kind of mindset do you have about different areas? Is there anything you believe you are “not naturally talented” at? Mindset often varies in different areas – someone might have a fixed mindset with respect to artistic ability, but a growth mindset with respect to computing skill. Then, think about your learners. How might a learner’s mindset about computational skill influence their learning in a workshop setting?
What are your thoughts about the influence of mindset in a workshop? Try to come up with a few different ways or situations in which mindset might be relevant.
Share in the main room and/or put a few thoughts in the etherpad

- It took time to come to the conclusion "I do not care what others think or even teacher thinks, I take my time and rythm of learning as eventually it is for me to use".
- Mindset I have developed at start of "digital semesters": have to read/learn that for myself _anyways_ later on -> influences my concentration negatively, even though i know that it's nonsense
- A positive mindset about people being able to learn new things will remove mental blocks people have about learning those things
- British people often say to me "we are really bad at learning second languages". There's no scientific reason to believe they are not good at languages. They just never had to so maybe it's now engrained in the culture that they can't learn other languages.
- I have a real mental block about learning matemathical modelling :(
- I'm not a "math person". I got headache when seeing formulas and I honestly think mathematical expressions written in LaTex tend to confuse more easily. It is much easier to understand the formulas when I attempt to write it in code!
- I am very bad at "art" techniques. It is difficult for me to draw/schematize my thoughts in an understable way.
- "Why should I try, I fail anyways"
- Not now, better tomorrow.
- I can't sing!
- I'm really bad at orientation, so I always end up letting other people give me directions or using a GPS instead of trying to remember directions by myself.
- It would take a big effort to become a low quality singer
- If the person believes him or she lack the talent for something, it may hold back the learner from trying, and the initial strugggles will feel bigger than they are.
- A learner might transfer their expertise in a related domain to a workshop and make assumptions about what they need to learn that eventually compromises their progress
- I have a fixed mindset about sport: to achieve a decent/high level, I tend to believe that you have to be born with enough talent. I tend to have a growth mindset about most of the activities only the brain is involved with

- I am very bad at reading. Reading text is always pain. Same applies to writing. This influences actually reading lesson materials. But I am relatively good at math and visualize things. In my case at least, being presented figures, an instructor talking and typing (if there is consistency), it helps a lot to overcome my bad part.
- Though, I think generally in terms of growth, implementing growth as a concept has more been an issue. It's amazing how much can be gained by doing something for a few minutes everyday/week. For promoting a growth mindset: Show that the learner is in control and has self-efficacy!

Praise Influences Mindset

Choosing our Praises (5 min)
Since we're so used to being praised for our performance, it can be challenging to change the way we praise our learners.
Which of these are examples of performance-based, effort-based, or improvement-based praise?

- I like the way you tried a couple of different strategies to solve that problem. EEEEE EeEEIPEEEE
- You're getting really good at that. Keep up the hard work! I IIIIIEEIiIPEPI
- You're really talented. PPPPPPPPPPPPpP
- That was a hard problem. You didn't get the right answer, but look at how much you learned trying to solve it!IE/I III IIE IiIIEIE

Errors are Essential to Learning

Helping Learners Learn From Mistakes (5 min)
A learner at your workshop asks for your help with an exercise and shows you their attempt at solving it. You see they've made an error that shows they misunderstand something fundamental about the lesson (for example, in the shell lesson, they forgot to put a space between ls and the name of the directory they are looking at). What would you say to the learner?
In the Etherpad, describe the error your learner has made and how you would respond.

- I would go back to the concept they misunderstood and talk them through it, making sure they und erstand it now. Help them understand the error and interpret the error message.
- Try to guide them through the error, to make them realize what triggered it and then connect it to the previous lesson when it was explained
- Ask them describe what they tried to do, so you learn where the error was made. Maybe they just forgot the space as a typo, maybe they copy-pasted from a source that didn't have space as a character, maybe they don't appreciate that spaces separate words even in shell, maybe they don't understand that a command and its arguments are things to separate (like words)
- Can you please show what you typed and tell what you tried?
- have the student exlain what they are trying to do and help them recognize what could be the mistake.
- Ask them to explain what part they were trying to do and how, and if they don't see the error themselves, go over the specific part of the materials with the focus on the fundamentals that are

mistaken
- Ask her/him what is going wrong.
- Ask her/him what the error message is indicating.
- What do you suspect is happening? Can we walk back from the error and resolve the issue?
- explain the syntax of ls
- review the `man ls` is usually the best option
- I would ask the learner to repeat what they thought they should do.
- Let them explain what they were doing so far, and what they want to achieve, and if they put effort into solving the problem themselves - Point to relevant material and come back to it.
- illustrate practically what happens when they do run the errored code and then show them what happens when they make other edits
- I would review that part personally with the learner, if the concept is brief enough that can be handled in a couple of minutes. I would suggest other material to be read if covering all that concept again would take too long.
- Ask questions that shoul guide the learner towards finding the mistake.
- Praise for the work he/she had already done; go step by step through the task and the solution of the learner to reach the point of the mistake
- "To understand this subject, I would recommend reading more on xxx." Then give a brief overview on how xxx can give you a solid foundation to understand the current task. Therefore, the student may feel compelled and curious to read more on that material.

Perseverance Predicts Success

How Are You Gritty? (Optional, 5 min)
A previous exercise asked you to think of a time when learning something was difficult for you, or you made a mistake that seemed silly or embarrassing.
How did you motivate yourself to continue learning? How did it feel to persist in the face of challenge? How do you feel now about your capabilities in this area?
In the Etherpad, describe how you could use this story to illustrate the importance of grit for your learners.

-
-
-
-

Habits of Lifelong Learners

Key Points

- Growth mindset and grit promote learning by making effort a positive thing.
- Presenting errors as essential to the learning process helps learners learn from their mistakes.
- Successful lifelong learners aren’t embarrassed to ask for help.

VIII. Teaching is a Skill

https://carpentries.github.io/instructor-training/11-practice-teaching/index.html

Lesson Study: Applying a Growth Mindset to Teaching

Giving Feedback (10 min)
We'll start by observing some examples of Carpentries-style workshop teaching and providing some feedback.
Watch this example teaching video (https://www.youtube.com/watch?v=-ApVt04rB4U) as a group and then give feedback on it. Put your feedback in the Etherpad. Organize your feedback along two axes: positive vs. to be improved/avoided and content (what was said) vs. presentation (how it was said).

Content +
+live coding
+ (I think) started with a fundamental thing

Content -
- passing function as parmeter of a function : very advanced topic not relevant for beginners (not in the carpentries scope?)
- font too small
- No real explanation of any of the code/jargon
- Fixes errors without telling what the error was
- Jumps from a fundamental thing to a sort of higher level concept of "definition".
'this is what you expect, trust me' - doesn't explain why
- no motivation on why the topic is important, as well as what the learners can expect to see in the presentation
- seems not to be prepared (jumping around a lot when searchig for examples)
- 'even excel users will understand this'
- completely unreadable font
- functions are the focus, don't talk about binding

Presentation +
+
- live coding
- acknowledged error
- asked for questions?
+sorry
+ Moving and trying to see the audience to actively involve them
+ stops to show the code
+excuses

Presentation -
-too fast
-dismissive language
-impatient

- presenting back to audiance
- jargon : foo and polymorphic
- does not explain the error and build on it
- 'This is really easy stuff'
- Dismissive language e.g. excel users
- doesn't take advantage of an error to teach
-Showing frustration because the participants weren't sitting down
- checking phone
- facing the projection, not the students
- the font size is really small +1
- going a bit too fast a would say
- even excel user would undertand that
-speaking to the wall
-"trust me"
-very fast presentation
- the screen shown is to small to be followed
- too fast, hectic behaviour+1

Feedback on Yourself (25 min)

Recommended lessons: https://carpentries.github.io/instructor-training/setup.html

1. Split into groups of three.
2. Individually, spend 5 minutes preparing to teach a 90-second segment of the lesson episode you chose before the start of the training course.
3. Get together with your group and have each person teach their segment to the group. Use visual aids if available (Note: Do not live code your lesson. There will be a chance to live code later.) Keep a strict time limit of **90 seconds per person** (one person should be responsible for the timekeeping).
4. After the first person finishes, give feedback immediately after they finish their turn teaching.
5. Then the next person teaches, etc.
6. After everyone has given feedback, return to the main group and put everyone's feedback about you into the Etherpad.

Use this rubric:

Content +
Explained each section of the code
setup was good (motivation)
proceed with knowledge from an earlier session (like "now, where we have setup Program X we will now start to import our first data in it")
starting with an example ist motivating
motivating the usefulness of the content with real examples
mentioned how to open a terminal, which is not written in the lesson material (using Spotlight function in my case with Mac OS)
explaining motivation given examples
content clearly presented

Content -

Presentation +

Good pace
Motivated audience by explaining the advantages of using the tool which is object of the training
good explanation of the coding
Shared screen to show a terminal
Presentation -
speaking too fast
speaking pretty fast
speaking too fast,
little explanation of the purpose / more context was needed
forgot to share screen
not easy to follow because there was no eye contact to the instructor (looked at second screen)
forgot to share screen
webcam should be on the screen you're working with to have eye contact with the learners
-couldnt' use screenshare, the necessary adjustment in Zoom should have been made in advance

Presentation +

- Clear speech

Presentation -

- spoke too fast
- screen resolution was unsuitable for sharing
- didn't have a teaching partner to interrupt and fix details like screen sharing

-
- Presentation-: too many 'just' words
- Presentation+ :clarity
- Content-: use of too specialised jargon without explaining it, i.e. 'branches' in git
- Content+: good to highlight the importance and applicability of the tools, in this case doing version control
- used word concatenate without explaining it
-
-

Content +

- establishing a real problem

Content -

- too much in this short time

Presentation +

- pace & explanations regarding last lessons

Presentation -

Feedback Is Hard

Using Feedback (5 min)
Look back at the feedback you received on your teaching in an earlier exercise. How do you feel about this feedback? Is it fair and reasonable? Do you agree with it?
Identify at least one specific change you will make to your teaching based on this feedback. Describe your change in the Etherpad.

- Prepare better the presentation and slow down when going through the lesson, make it more clear
- Design and test with a window that's a good size for screen sharing (first, find out what size is that?)
- better control using dismissive words; explain each concept you introduce. The feedback was very fair.
- Time the start of livecoding (screensharing) better
- Put less content in the lesson, take more time to explain and get feedback
- Yes, it was fair feedback. I would try including/starting with examples that learners may find familiar with, rather than totally sticking to the lesson material.
- better preperation, testing the technology in advance
- Won't teach a digital lesson alone, feedback from a co-instructor is necessary
- the feedback was fair
- The feedback was fair. I could have been better prepared
- For me the feedback was fair, I can improve speaking a little bit clearer.
- Prepare better, introduce every concept
- take more time for context on what i´m explaining
- Try to take the time to explain clearly, don't rush through explanation (fair, and agree)
- Start lesson with the summary of the previous one and the motivation of the current topic
- Fair & polite feedback! Will take more time to connect earlier lessons

Skill Acquisition: What level are your teaching skills? (Optional)
As with computational skills, people have a tendency to think of teaching as something you are “just good at” or not. However, teaching is a skill, and expertise develops with attentive practice. Examine the descriptions of “novice,” “competent practitioner,” and “expert.” Where do you think you fall with regard to teaching? What have you learned about teaching? What are you aware of that you still need to learn? Discuss with a partner and then write some thoughts below.

Key Points

- Like all other skills, good teaching requires practice and feedback.
- Lesson study is essential to transferring skills among teachers.
- Feedback is most effective when those involved share ground rules and expectations.

IX. Wrap-Up and Homework for Tomorrow

https://carpentries.github.io/instructor-training/12-homework/index.html

To prepare for tomorrow, please:

1. Look through these checklists to learn what hosts and instructors need to do to prepare for a workshop (https://docs.carpentries.org/topic_folders/hosts_instructors/hosts_instructors_checklist.html), and read over the differnce between a centrally-organized and self organized workshop at the bottom of this page (https://carpentries.org/workshops/).
2. Prepare for the live coding exercises. If you haven't already, pick an episode from an existing Software, Data or Library Carpentry lesson and read through it carefully. Tomorrow, you will use this to practice live coding for 3 minutes in groups of three. Your group members will comment on the delivery and content. Recommended episodes are listed here: https://carpentries.github.io/instructor-training/12-homework/.

Feedback (5 min)
The Trainer(s) will ask for feedback on the day in some form.

We'll use the "One Up, One Down" activity, i.e. one positive thing, and one thing that we can improve and/or avoid doing in the future

Reflecting on the Day (5 min)
Before we wrap up for the day, take 5 minutes to think over everything we covered today. On a piece of paper, write down something that captures what you want to remember about the day. The Trainers won't look at this - it's just for you.
If you don't know where to start, consider the following list for a starting point:

- draw a concept map, connecting the material
- draw pictures or a comic depicting one of the day's concepts
- write an outline of the topics we covered
- write a paragraph or "journal" entry about your experience of the training today
- write down one thing that struck you the most

Links from today's chat log:

https://en.wikipedia.org/wiki/Concept_map
https://app.diagrams.net/
https://firstliteracy.org/wp-content/uploads/2015/07/How-Learning-Works.pdf
https://github.com/UKHomeOffice/posters/blob/master/accessibility/dos-donts/posters_en-UK/accessibility-posters-set.pdf
https://teachtogether.tech/en/index.html

X. Welcome Back

Schedule for Day 2: Instructor Training

09:00 - Welcome Back
09:10 - Live Coding is a Skill
10:20 - Preparing to Teach
11:10 - **Morning Break**
11:25 - More Practice Live Coding
12:10 - Managing a Diverse Classroom
12:40 - **Lunch**
13:40 - Checkout Process
13:55 - The Carpentries: How We Operate
15:10 - **Afternoon Break**
15:25 - Workshop Introductions
16:05 - Putting It Together
16:25 - Wrapping Up
16:40 - Post-training survey
16:55 - **Finish**

https://carpentries.github.io/instructor-training/13-second-welcome/index.html

Questions (5-10 min)

Yesterday we asked you to read some resources about the logistics of teaching and running Carpentries workshops. Please add your questions about logistics and preparation to the Etherpad. We will answer these questions in the etherpad during your work time and will return to this list later today.

- question
- Can I, as a trainer from a participating organisation, organise other workshops for people outside of that organisation?
- What is a “community discussion”?
- Are the carpentries providing zoom and etherpad access for online courses?
- Are the carpentries courses always given in English even if all the attendees are from a non English speacking country? (I've seen carpentries-es)
- Do we trust hosts if they *say* the place is accessible? (or check in person, request fotos etc)
- What should I do if I would like to initiate a free workshop for underreached communities? Is there any facilities provided by the Carpentries that I can use free of charge (e.g. a paid Zoom account)?
- Can I teach modified content
- Find opportunities to teach / help out?

Key Points

- Instructors guide learners to construct the proper big picture (accurate mental model) of the topic rather than focus on details.
- Instructors rely on frequent feedback from learners to monitor their own presentation of the material.
- Instructors introduce a few concepts at a time to avoid cognitive overload.

- The best way to motivate learners? Show them how to do something they can immediately put to use and be enthusiastic about it.
- Teaching is a learned skill.

XI. Live Coding is a Skill

https://carpentries.github.io/instructor-training/14-live/index.html

Why Participatory Live Coding?

Up and Down (10 min)

List some advantages and challenges of participatory live coding from both a learner's and an instructor's point of view.

- + it is slower
- + attendants learn by doing. Most importantly, they see that the teacher also makes mistakes
- - it's hard to catch key combinations and/or mouse clicks. The teacher needs to remember to point these out: this is tricky, because they're likely to be in their muscle memory already!
- + it is more of a real situation that just showing a script
- - it can be hard to follow if someone has problems with one particular line, beacuse when focusing on your own computer you might get lost on what's being done in the big screen.
- + engaging, shows the actual thinking behind
- + instructor will probably also make errors
- + errors happen
- + learners can type along the instructor so that they can see what happens at the same time on their computer.
- - could be difficult for not only instructor but also for learners to pay attention to both what they are doing and talking/listening plus pay attention to learners (proably that is why this part is off-loaded to helpers)
- - difficult to pitch the right level with the participants
- + it is practical and it will last longer in their memory (finger memory as well as visual)
- + The trainees will learn from their errors (example typing errors) and be less afraid of them
- - for complicated code is take a time to go from the instructor screen to our own = oveload
- - the instructor does not always know if the trainees are in the same environment (location, shell,...)
- + Slows the instructor down to the students' speed
- + Ensures the instructor meets on-the day problems live rather that when the students go off to do the work themselves
- +/- Some students like the "railroad" and others find it restrictive
- + keep novices engaged and be able to know who is not following
- + Leaves the audience with the basic example for further reference.
- + Improves multi-tasking skill of the instructor.
- - Requires trainees to have a working infrastructure (preferably very similar to the one instructor has).
- - people who are faster may get bored and think the pace is too slow

- - can get unorganized (i.e. intructor losing train of thought)
- + live coding is somehow flexible
- - can be confusing if the teacher is creating a lot of errors all the time
- - Participants that just copy what is being typed without internalizing the command/understanding why you do something might get confused after a while
- + Shows errors made along the way, in stead of only showing a finalized script/command
- + allows to properly follow the content and to digest it
- - can be intimidating to keep up with the instructor and with the rest of the class
- - can be very demotivating, as a learner, if you see you're being left behind in learning
- - it exposes the teacher to unpredictable situations which, depending on their level of knowledge, can be extremely stressfull
- Con: it can be difficult to talk while coding. If something unexpected happens it can be stressfull. If there is a problem with only one learners computer, this person will fall behind or hold up the whole class. Pro: everybody must be hands on for all tasks. The learners will learn how to solve problems.
- (+) The way to code in real time is really illustrative. It gives an overview more realistic (than explain theory, functions, algorithms without any visual incentive) when coder has to deal with a coding problem. (-) Sometimes, it can be dissapointing for the student that is not be able to reproduce the instructor's code.
- requires better and more preporation
- - getting a feel for the audience following along can be more cogntivie load on the instructor
- - anxiety-inducing for the instructor
- + shows that expert coders can also make simple and sometimes silly mistakes
- + shows that to arrive at the fancy and concise script one would actually start with messy and ugly code
- -requires multitasking
- maybe it's too fast
- - requires an identical computing env for each student, how it is achived if students are at home? on their own computers?

Compare and Contrast (15 min)
Watch the two live coding videos as a group and then summarize your feedback on both in the Etherpad. Use the 2x2 rubric for feedback we discussed earlier.
In the videos, the bash shell *for* loop is taught, and it is assumed learners are familiar with how to use a variable, the *head* command and the content of the basilisk.dat unicorn.dat files.

poor live coding: https://www.youtube.com/watch?v=bXxBeNkKmJE&feature=youtu.be
Content +

- The content he intended to deliver is nice for a short fragment
- He started to say what he was going to do.

Content -

- poor introduction
- phone notifications going off +1
- didn't look at the participants or was engaging
- repetition of filename/animal/3
- didn't explain the error
- didn't explain step by step

- repetition of basic concepts

Presentation +

- made errors and coped with them smoothly
- Explaining intentions
- Not to fast +1

Presentation -

- phone not on silent
- Teacher didn't address the red sticky note
- He was sitting down and not pointing what was happening in the terminal step by step
- The terminal was not fullscreen, the prompt was VERY distracting
- Presentation was too fast and he didn't use the typo to help explain the for loop
- Red sticky note
- white text on black background is not well readable
- small font
- speaking while typing
- No explaining while typing +1
- Font size too small +1

good live coding: https://www.youtube.com/watch?v=SkPmwe_WjeY&feature=youtu.be
Content +

- good introduction
- explaining error/typo
- explaining one-line version when he had to correct an error +1
- intro + well explained steps and what each variable does
- he explained very well the content as well as the error +1

Content -

- did not explain what is "-n", only explained the "3"
- did not explain his use of "-3" (or maybe that was only in the first video)
- Explain the difference be animals and $animals

Presentation +

- black text on white background
- he watched out for the red sticky notes
- The way the typo was used to explain other forms of for loops was brilliant
- Checking that all sticky notes from previous sections have been addressed before moving on
- not dismissive of variable names
- good pace and engaging
- standing up position felt more engaging
- eye contact with the learners
- stood up to point out things on the board

Presentation -

-

Top Ten Tips for Participatory Live Coding in a Workshop

- **Stand up and move around the room if possible**
- **Go slowly**
- **Mirror your learner's environment**
- **Use your screen wisely**
- **Use illustrations**
- **Turn off notifications**
- **Stick to the lesson material**
- **Leave no learner behind**
- **Embrace mistakes**
- **Have fun!**

Sticky Notes

Practice Teaching (25 min)

Teach 3 minutes of your chosen lesson episode using live coding to one or two fellow trainees, then swap and watch while the other person(s) live codes for you. Explain in advance to your fellow trainee(s) what you will be teaching and what the learners you teach it to are expected to be familiar with. **Don't record the live coding sessions.** Give each other feedback using the 2x2 rubric we discussed previously and enter the feedback you received in the below.

Back in the main room 10:30 CET

Content +
Content -

- do not use unnecessarily more advanced commands (for example to edit files)

Delivery/Presentation +

- Clear, well-paced, coherent with what was shown in the screenshare
- clarity

Delivery/Presentation -

- too rushed, the pace should be more controlled
- Issues with fontsize and size of windows being shared should be addressed before starting the screenshare

Key Points

- Live coding forces the instructor to slow down.
- Coding-along gives learners continuous practice and feedback.
- Mistakes made during participatory live coding are valuable learning opportunities.

XII. Preparing to Teach

https://carpentries.github.io/instructor-training/15-lesson-study/index.html

Learner Profiles

Learner Profiles (10 min)

Read Software Carpentry's learner profiles (https://software-carpentry.org/audience/). Note that these example profiles contain more information than you will ever know about a learner; this is a creative exercise in imagining (and empathizing with) the whole people behind the faces. Now, sketch out a profile of someone you might expect to attend your first workshop. Who are they, what problems do they face, and how might this training help them? Be as specific as possible.
Enter your learner profile below.

Bench biologists that have a lot of experience in visualising and analysing data with GUI-based tools but now they would like to learn R/Python to make their analyses reproducible
Librarians or Researchers who want to learn some programming in Python to make their work/research easier
A developer of a scientific code that needs to use expand the use of MPI within a code, maybe to scale better, or parallelize more stages of a pipeline. Some kinds of problems need more than the MPI basics to handle efficiently. Different data structures need to be distributed, or collectively rather than point-to-point.
Probably someone from wet lab who wants to start analyzing their data, applying statistical tests or plotting the results.
Post-doc that does not have the enough experience with programming and data analysis but the type of his analysis has exposed him to lots of excel spreadsheets. He has to daily analyze it and needs a way to automatize this amount of stuff.
Someone who does cell and molecular biology and often applies statistical tests to their data, but has never done any programming. Often uses Excel or SPSS, some people might have used STATA before, but find R and programming really intimidating. They want to learn R to make their science more transparent and reproducible or they want to de a bit more advanced data analyses.
Molecual biologists which need now to process sequencing data and perform statistical analysis. They are mainly used to excel spreadsheets.
Librarians or Researchers who wants to structure and analyze their research data and spreadsheets. They have a lot of experience with Excel but no experience with other programs. The use of code is unsettling for them.
Diverse - psychology to engineering (PhD students / some PostDocs). Most somewhat experienced in Matlab, R or other programming languages. Need to automatize data operations, statistical analysis and visualization, but also inexperienced in version control. Data operations and version controling will be helpful, many possibly underestimate their actual programming/coding knowledge.
PhD students/postdocs with moderate programming experience in Python and/or C++ that need to run large simulations on parallel hardware. They need to learn parallel programming to improve the efficiency of their software.
PhD students at the early stage of their career with novice experience in programming, but who are

involved in a heavily computational PhD-program. Graduate students who are willing to start an internship with my institute and need to fill a gap in programming.
Researcher who needs to use some programming languages like R or Python but not for daily basis for the long time, rather for a very specific purposes in his/her research project (ex. only for the statistical analysis part) - Although all the lessons that are available in Carpentries could be useful for the different phases of the research project and his/her whole researcher life, the intention/motivation was to compensate the lack of that particular skills rather than making his/her own research practice better as a whole.
Science students and researchers that have no programming background, looking to learn to use R for data analysis and visualisation.
Archaeology students, working with spreadsheets and ArcGIS/QGIS software, learning R to create seamless workflows for non-spatial and spatial analysis
Humanities students who want to get to know digital methods and tools
Any student with some programming experience who is keen to learn about applying appropriate models to neuroscience data

Reverse Instructional Design (and Preparation!)

Working With Learning Objectives

Evaluate Carpentries Learning Objectives (10 min)

Select one learning objective from one of the lessons linked to below and then complete the following steps to evaluate it.

******[INSERT links here]******

- https://librarycarpentry.org/lessons/
- https://datacarpentry.org/lessons/
- https://software-carpentry.org/lessons/

The infographic of Bloom's taxonomy used by The Carpentries: https://carpentries.github.io/instructor-training/fig/Blooms.png
There is a revised version of the taxonomy (2011) that tries to avoid implying a hierarchy of the verbs: https://www.uky.edu/celt/instructional-resources/scholarly-teaching/blooms

- Identify the learning objective verb. How specifically does this verb describe the desired learner outcome?
- Where does this verb fit on Bloom’s taxonomy? Do you think this is an appropriate level for your learners?
- In your opinion, does the lesson do an effective job of meeting the stated objective?
- What would the next level on Bloom’s taxonomy look like for your learners? How might you be able to help them think ahead to the next level without attempting to get them there during your workshop?

- "Git - Creating a Repository" verbs are "create" and "describe". I don't know what the desired

learner outcome is. Much of the course content is about "git status" and not needlessly nesting repositories. Those topics don't relate well to the stated objectives. The stated objectives are about applying/understanding/remembering within Bloom's taxonomy, which seems suitable for beginners. The lesson does meet the objectives, but does some other stuff, too. The other stuff is useful, but not well placed, I think - perhaps because it involves aspects of evaluating and analysing how related is the proposed content. So maybe that was an attempt to provide content at a higher Bloom level.

- R for reproducible science - Data Structures. Verbs are 'identify', 'explore' and 'ask R' (something). I believe it fits in Bloom's taxonomy in the 'apply' step. The lesson meets this learning outcomes. The next stage would be to 'analyse' and learners could use these data structures (e.g. data frames) to explore their own data and see how what they have done in this lesson can be useful in manipulating their own data.
- Python (Repeating Actions, for-loop): Explain for loop and correctly write it down: This would be in "remember" and "understand", the focus is a bit less on the "apply" stage. Does this well, as it works (especially in the exercises) on these concepts. Exercises already show somer further uses, one can point towards using repetitions for many different things (higher levels), pointing towards for-loop nesting. Introduces range, but as a flyby...
- Library Carpentry | UNIX shell | Working with files & directories | "Use tab completion to limit typing"
    - Is there a desire? If so, fully, "Apply",
- Unix shell: discuss and differentiate between the basic suite of Unix commands, then execute them in your own Terminal to interpret what each command does
- Library Carpentry | Unix Shell: Navigating the filesystem - "Use" shell commands "to work/find/manipulate" -> Apply stage
    - To go to the next "analyze" stage, they should know/be aware that what they may want to do / can expect to do following by these commands introduced.
    - Not sure the material cover enought to achieve "manipulate data"
- Git - Tracking Changes: "Go through the modify-add-commit cycle for one or more files". This is to remember through repetition (hence "go through ... one or more files"). The next stage would be to understand, which is exactly what the next objective is: "Explain where information is stored at each stage of that cycle."
- "Version Control with Git": Ignoring Things. 'Configure' in the objectives is the most important verb. The lesson covers the important points in this task.
- The introduction to R : the objectives of the lecture fits with the three first steps of the Bloom's taxonomy. The purpose in the lecture is to learn the basic concepts, remember them and apply them. The lesson contain a list of objects, they are all basic and the lecture manages to include all of them.
- Programming with Python (repeating actions with for-loops) The relevant verb in the objectives is "explain" This fits in the "understand" rung of Bloom's taxonomy. The verb is specific to the learning outcome the episode will achieve and the level in the taxonomy is also adequate.
- Software carpentries | working with files and directories | "create directories" in apply category
- Exploring dataframes (R for reproducible scientific analysis) the verbs are add, remove, append... which would fit with 'create' step in bloom's taxonomy. Then the next objectives are understand, convert and display basic properties, which would fit more with the 'understand' step and also the 'apply' one. I think this lessons does exactly as the objectives say. The next step for me would be to try to apply different functions or concepts to this dataframes, this way they would consolidate their knowledge.
- Introduction to Raster Data - verbs are describe and distinguish and fit Bloom's understand and analyze taxonomy, I think next step should be to evaluate

- Library Carpentry | Python Intro for Libraries | Variables and Assignment | "Write" programs that assign values to variables and perform calculations with those values | I think it fits the "create"-step because it says "write programs" in a sense of "create programs"; I don't think it fits what a learner thinks of what creating or writing a program is.
- Git - automated version control: the objectives are around understanding the power and benefits of using git and how it works. The relevant verb of the objectives is 'understand'. The lesson is very clear in addressing these points.
- Unix lessons : Navigating Files and Directories. Objectives Explain the similarities and differences between a file and a directory. There is a definition = "It organizes our data into files, which hold information, and directories (also called ‘folders’), which hold files or other directories". I did not find the place were they present the similarities and differences.

Using Formative Assessments

3

Where are your checkpoints? (10 min)

Have a look at your lesson again. Choose a learning objective, and identify where in the lesson that objective should reasonably be achieved. How will you know that that objective has been met for all learners? Will this be clear to them?

Make a plan for where in your lesson you will use different types of formative assessment to help everyone in the room monitor their progress. Keep in mind that formative assessment can can take many forms, including multiple choice questions, faded examples, spontaneous questions and calls for sticky notes.

Write some notes or thoughts about this process below for discussion.

Ad-hoc open-ended or binary answer questions to check whether my assumptions are correct about their level of knowledge

Some thoughts: need to balance the time that it takes to teach all the material and the time for all the assessment. Zoom might make it hard for some of the forms of formative assessment. Maybe make use of online quizes (kahoot is a good and fun platform) to have different points for assessment and to see if they are following.
Watch out for "Aha" moment on faces ;)
Git: Tracking Changes: Ask everyone to add/push a file with their name as a title and last name as a content.
Probably when explaining factors in the exploring dataframes lesson, learners are supposed to already know what factors are but the way you have to manipulate them in R is tricky so I would try to make sure they get it right.

- Python for-loops episode. I would use sticky notes to signal problems, right after the for-loop constructs has been introduced.

Unix lessons : Navigating Files and Directories. Objectives Explain the similarities and differences between a file and a directory. The assessment could be placed once the trainees have tested a few commands (ls, pwd; cd, mkdir), they could explain which command is applicalble to files or directories or

both.
Library Carpentry | Python Intro for Libraries | Variables and Assignment | The formative Assessment is already built-in the lesson, I would say.
"Working with files and directories" lesson, can have a diagram for which the participants can choose which code matches that diagram (one of the learning objectives is to create a hierarchy based on a diagram, but there's no diagrams)
For assessing creating a git repo, you could have some way to look at their environment, or ask them to give feedback in some kind of form when they've done "git init." Both feel very heavy weight for the value delivered to the student and instructor - are we really going to have a form where the student ticks off every line they type? A multiple choice quiz about the content of a .git directory would probably be OK though.
"Git version control". Ignoring things. It could be important to show why sometimes it is important to hide huge datasets only to avoid problems with remote repositories. It is not only a problem of waste of disk space.
Intersperse some of the exercises in-between, like using the multiple choice question of how often a for loop executes after explaining the loop's basics, but before going into more detail (understanding of repetitions). Ask questions in between, watch out for sticky-notes. Consider more time to explain new concepts in exercises (like range).
Introduction to R: teaching about vectors, I would use sticky notes to get feedback on wheter they managed to create the vectors
Git - automated version control: asking them to list other standard tools used for collaborative work, and compare them to git.

Feedback on Your Challenges (Optional, 15 min)

With these goals in mind, pair up with a partner to discuss the MCQ and faded example problems that you wrote yesterday. Give each other specific, actionable feedback that follows our 2x2 framework. Use that feedback to make at least one modification to your exercise(s). Discuss in the Etherpad the change you made and how it will help you get more useful information about your learners.

Key Points

- To teach effectively, you have to know *who* you are teaching.
- Good learning objectives communicate the intended effect of a lesson on its learners.
- A good exercise provides useful guidance to instructors about next steps needed in teaching.

XIII. More Practice Live Coding

https://carpentries.github.io/instructor-training/17-performance/index.html

Round Two (25 min)

First, have a look at the rubric that is given to trainers as a suggested framework for evaluating the online teaching demonstration sessions that are part of instructor checkout:

https://carpentries.github.io/instructor-training/demos_rubric/. Does this rubric make sense? Take a moment to think about things you'd like to do differently with your next live coding practice. This is also a good time to ask questions about teaching demonstration.

Next, get back into the same groups you did your live coding with. Take turns re-teaching your chosen live coding session, making sure to incorporate changes based on the feedback you received and any new ideas based on reading the rubric. Give feedback to each other using the rubric this time.

When you are finished, add some thoughts on this process to the Etherpad: What did you change? Did it work better or worse with the change? How might you do it if you were to teach it again?

Key Points

- (Reflective) Practice makes perfect.

XIV. Managing a Diverse Classroom

https://carpentries.github.io/instructor-training/18-management/index.html

What Are the Challenges? (5 min)

What are some of the challenges you might expect when teaching learners with a broad range of expertise? Discuss with a partner and put your thoughts in the Etherpad.
(Until 12:22 CET)
- always the same poeple are aswering the questions (mitigate by specifying new people should answer)
- discussion the presented solutions (alternatives which are quicker)
+ advanced people could become helpers : They will also less interupt the course

- - on the other hand, how could we know that the advanced learners are teaching the correct thing to their peers?

+ they can help on hardware or other issues
+ the teacher can learn from the questions of advanced trainees
+ can involve the more competent learners for peer-learning (help learners in trouble)
- more experienced participants as teachers
+ let the different experienced two work in a group
+Ask more advanced student to help those lagging behind
+/- diverse backgrounds can lead to richness; but the proposed examples will be less effective and won't reach everybody. It might impair grasping the usefulness of the tools for each learner everyday job.
- More advanced people may loose focus while waiting for others being helped.
The group will not have the same references

Code of Conduct Violations

https://docs.carpentries.org/topic_folders/policies/index_coc.html

All about the CoC: https://docs.carpentries.org/topic_folders/policies/index_coc.html

Never Teach Alone: How to Be a Co-Instructor

Teaching Together - Nuts and Bolts

With a partner, imagine that you are planning a workshop together and answer the following questions:

- How would you prepare to teach a workshop together?
- During the workshop, what are some things the assisting instructor can do (or shouldn't do!) to support the main instructor?

As an entire group, discuss what you came up with and then compare to the recommendations below.

Minute Cards Revisited (5 min)

Use your sticky notes to write minute cards as discussed yesterday.
Make a separate vanilla user account to deliver teaching from (no bash customizations, browser windows with alerts, etc., while not requiring me to break my normal environment, and making it easy for me to switch to it if necessary)
Share a window, not the whole screen
Remember to adjust your fontsize when sharing the screen
- Don't use fancy shortcuts or custom layouts that you may use during your daily job. Try to emulate the learner's environment as much as possible
- Don't leave anyone behind. Don't try to cater to the advanced learner's. They may have one day wasted if we focus on the beginner's content but the beginners may be scarred for life and become afraid of programming if we give materials that are too advanced.
4K resolution does not work when you do screen share.
Explain concepts with a context
- Focus on presenting the usefull part in a simple way even if it is not the whole picture (limit overload)
don't squeak with the chair or better do not move too much while sitting because you're nervous or something
If you have dual monitors, place what you are focusing on in the area where the camera catches your face and eyes in a way that learners see that the instructor actually see the learners.
The importance of instructor sitting vs standing, white on black can be difficult to see from the back.
Make sure to center the command line on the screen.
High contrast or black-on-white color scheme when doing command line teaching
Type slowly and don't take any knowledge for granted. Focus on delivering a clear message even if it's not the whole content, instead of trying to cover as much as you can.
implement questions (formative feedback) early, when presenting concepts
Acting on feedback takes work and time
Standing while teaching will improve the workshop a lot!
preparing a pre-workshop test can be highly counterproductive for novices, and can potentially scare them off. Always think at how psycology works and try to put yourself in the learner's shoes
practice, practice, practice

leaving your comfort zone makes you better
errors are good +1
being slow and patient can get you far
are there community discussion curriculum specific?

Key Points

- Working with a broad range of learners can be challenging, but there are many ways to keep a classroom happy and motivated.
- Response to a Code-of-Conduct violation at a workshop is subject to instructor discretion, but all violations should be reported to the Carpentries for follow-up.

XV. Checkout Process

https://carpentries.github.io/instructor-training/20-checkout/index.html

Application Form

https://amy.carpentries.org/forms/request_training/

- Make a **contribution**
- Take part in a **community discussion**
- Prepare to teach a full Carpentries lesson, then do a 5-min **teaching demo**

Instructor Checkout

https://carpentries.github.io/instructor-training/checkout/

Checking Out Review with Questions & Answers (5 min)

Read the three checkout procedures. Make notes in the Etherpad. What questions do you still have about the checkout process?
As a beginner myself I am concerned that it may be difficult to contribute to the lecture material
-
- Will the certificate be revoked if no teaching within 12 months? (I don't intend to, but would be good to know ;))
- I don't really know how to make a contribution actually, e.g. for a german glossary or something like that.
- How does a community discussion works? Is it always in english or also held in other languages?
- Are community discussions specific for a certain workshop or lesson or general?
- what happens after we completed the three requirements? Do we contact someone or wait to be contacted?
- Is there an available list of countries and institutions that have an active Carpentry community?
- How many people are in a discussion session?

- Contribution: Does it have to be a Pull-request, or can it also be participating in the discussions (issues)?
- Could the contributions be in the area of translating lessons?
-Can we organise workshops outside of our institution? For example in the community or schools, as public engagement?

- Contribution says "to one of the Lessons". Yesterday, this illustration helped us a lot to understand "fluid representations" & I was thinking of making an illustration of that which would be CC0 instead of the New Yorker copyright illustration - would that be a valid contribution as well? (Luis, lmossburger@t-online.de)

Schedule a Discussion or Demo (5 min)

Visit the discussion Etherpad https://pad.carpentries.org/community-discussions to sign up for a session. If the session you would like to attend is full, contact the discussion host and co-host to ask if you can attend.

If you'd prefer to do your teaching demonstration before your discussion, visit the demo Etherpad https://pad.carpentries.org/teaching-demos and sign up there.

Lesson Contribution

How to make a contribution on GitHub:

- https://carpentries.github.io/instructor-training/checkout/index.html#eligible-repositories

What's in a Badge?

Check Out the Discussion (Optional)

As an instructor, your voice is important! We want you to be actively involved in discussions about the lesson materials (and other aspects of the Carpentries community). Go to the GitHub page for the lesson you worked with over the past two days and click on the "Issues" tab. Read through some of the discussions and, if you have anything to add, please add it to the conversation! If you wish to make a pull request, be sure to examine the contribution guidelines for the repository you are working in. If you do make a significant contribution to the discussion, send a link to the issue to checkout@carpentries.org. Congratulations! You've just completed one of the three remaining steps in becoming a Carpentries instructor.

Key Points

- To certify, you must contribute to a lesson, take part in a discussion, and do a teaching demo within 90 days of your training event.

XVI. The Carpentries: How We Operate

https://carpentries.github.io/instructor-training/21-carpentries/index.html

Brief History

https://carpentries.github.io/instructor-training/fig/SWCDChistory.png

Similarities and Differences Between The Carpentries Lesson Programs

https://carpentries.github.io/instructor-training/fig/carpentries-venn-diagram_20200904.svg

How a Workshop Works

- Materials
    - https://datacarpentry.org/lessons/
    - https://software-carpentry.org/lessons/
    - https://librarycarpentry.org/lessons/
- Using the Names and Logos
    - it covers the core topics (see below),
    - at least one of the instructors is a certified Carpentries Instructor,
    - you run our standardized pre- and post-workshop assessments and ensure everyone participates.
- What is the Core Curriculum?
    - Following the links to the lessons above, check "Our Core Lessons" to find out what counts as the core curriculum
- Who Can Teach What
    - Anyone can teach anything!
    - However, a branded workshop must be taught by at least one certified instructor
    - Always cite the material that you are using correctly
- Setting Up
    - Follow the instructions in the README here: https://github.com/carpentries/workshop-template

Practice With Carpentries Infrastructure (25 min)

Go to the **workshop template repository** (https://github.com/carpentries/workshop-template) and follow the directions in the README to create a workshop website using your local location and today's date. Put the link for your workshop website below.
Note: Sometimes web browsers will cache the workshop webpage, so when you make changes in Github, they don't show up on the workshop webpage immediately. Two ways to avoid this are to use a "private" or "incognito" mode in your web browser or by following the instructions at https://en.wikipedia.org/wiki/Wikipedia:Bypass_your_cache to bypass your browser cache.

https://carpentries.github.io/instructor-training/21-carpentries/index.html

To ensure unique names for each event, we use the year-month-day-venue format for the identifier, the **slug**. For instance, for a workshop that starts on Febuary 22nd, 2022 at the University of Oslo, the slug would be 2022-02-22-uio-online. If you are teaching a centrally-organized workshop, please wait for the email from your regional coordinator that will tell you the slug to use for your workshop.

Please put your workshop URL here:
- https://luismossburger.github.io/2020-12-01-lm-ONLINE/ (broken) (seems to work!)
-https://fpozoc.github.io/2020-11-13-instructor_training_test/
- https://github.com/robertodr/2021-01-30-uu
https://srsteinkamp.github.io/2020-11-13_empty_workshop/
https://lmtnezg.github.io/2020-11-13-cnio-online/
https://chklopp.github.io/2020-11-13-Toulouse-online/

Question and Answer (10 min)

What questions do you have about running and teaching at a workshop? Talk with a partner and enter your questions below. At this time we will also return to discuss questions remaining from the beginning of the day.

A Culture of Contribution

The Carpentry Community

Participating in the Carpentries: What's Your Role?

If you are at an in-person training, your instructor will hand out paper copies of a worksheet. If you are at an online training, you can get a digital copy at https://carpentries.github.io/instructor-training/files/handouts/Carpentries_roles_worksheet_v4.pdf.

Take a moment to review member community roles on the Carpentries' communitiy website (http://static.carpentries.org/community/). Working on your own, match up the roles with the descriptions. When you are done, think about the question at the bottom of the worksheet about what roles you might play, and enter your thoughts in the etherpad.

Get Connected (3 min)

Take a couple of minutes to sign up for the Carpentries discussion channels you want to stay involved with.

There are many ways to get connected with the Carpentries community:

- Our websites are:
    - Software Carpentry https://software-carpentry.org
        - Blog https://software-carpentry.org/blog/

        - Data Carpentry http://www.datacarpentry.org
            - Blog http://www.datacarpentry.org/blog/
        - Library Carpentry https://librarycarpentry.org
            - Blog https://librarycarpentry.org/blog/
        - The Carpentries http://carpentries.org/
            - Blog http://carpentries.org/blog
            - Get involved (community overview) https://carpentries.org/community/
- Our lessons are hosted on GitHub; contributions to them and discussion of changes happens via GitHub pull requests and issues, and the lessons are published using GitHub Pages. More details are given below:
    - Data Carpentry on GitHub https://github.com/datacarpentry
    - Software Carpentry on GitHub https://github.com/swcarpentry
    - Library Carpentry on GitHub https://github.com/LibraryCarpentry
- The Carpentries share public discussion lists that host everything from lively discussion on teaching practices to job postings and general announcements:
    - https://carpentries.topicbox.com
- We publish a joint newsletter. https://carpentries.org/newsletter/
- Host monthly community calls and weekly instructor discussion sessions:
    - Check out our community calendar https://carpentries.org/community/#community-events
- You can also find us on
    - Twitter:
        - Software Carpentry on Twitter https://twitter.com/swcarpentry
        - Data Carpentry on Twitter https://twitter.com/datacarpentry
        - Library Carpentry on Twitter https://twitter.com/LibCarpentry
        - Carpentries on Twitter https://twitter.com/thecarpentries
    - Slack https://swc-slack-invite.herokuapp.com
    - Gitter https://gitter.im/LibraryCarpentry/Lobby (Library Carpentry)
    - Facebook https://www.facebook.com/carpentries/

Key Points

- Carpentry materials are all openly licensed, but Software and Data Carpentry names and logos are trademarked.
- Carpentry workshops must cover core concepts, have at least one certified instructor, use our pre- and post-workshop surveys, and report attendance information.

XVII. Workshop Introductions

https://carpentries.github.io/instructor-training/23-introductions/index.html

Setting the Workshop Environment

Your Academic Past (5 min)

Think back to courses or workshops you really liked or didn't like.

- How did those courses start on the first day?
- Were you confident in the instructors abilities?
- Did you feel like they were enthusiastic about the course and invested in you?
- Was it clear what you were going to be learning?
- Were you excited about the material?
- Did you leave that first day thinking the instructor was uninterested, that you weren't the students they wanted to be teaching or you had no idea what the course was supposed to be about?

Start here:
+ eplaining and showing in the beginning deep coonection to audience (coaching)
+ instructors were passionate and made a real effort for that summer school to happen, there were more than 200 faculty members involved so altogether they mae an excellent job. i was curious about the materials and after the school, i decided i wanted to do a phd in the topic
+ I remember course on the topic no-one was interested in, but the instructor was so good that everyone liked it at the end.
- The course did not really describe properly the content and once I noticed about the materials I feel dissapointed.
Intro of instructors, with their background and affiliations. Overview of material to be covered. Prerequisites. Sometimes it felt like the instructors were a club who got together to do this teaching thing but it was more about the club than the teaching. Not being able to run the material myself made me disengage from the workshop. Not having a direct use for the material led me to disengage (ie go do something useful; I would need to Google this again later, anyway).
+instructor gave an overview of their background, they talked about how they started programming and even maybe mentioned what they used to struggle with
- I went on an unix course where the instructors didn't even introduce themselves, they just gave us the materials and stood there while we worked through them. What was the point?
+ Instructors were very positive, welcoming, entertaining and clear about the purpose and the plan of the workshop but still allowed some flexibility according to the learners' interest.
+ course start : parctical details, schedule, presentation of the trainers and trainees, code of conduct (even if not so formal), "we are here to learn together" some breathing and relaxation exercises between sessions.
- lecture (so not necessarily related to introduction): Berating students coming late for about 15 minutes
- long introductions into achievements, publications etc. -> makes one feel inferior
+ Instructors giving an overview of the material to be taught, and setting the learning goals. It is also inspiring if the instructor manages to set the material into context, how it may be usefull to the learner. It is preferable to know ahead of time when there will be breaks, and also what will be expected from the learner.
-not knowing the outline of the day/class, wondering when there will be a break or what material will come next.
+ An instructor being super enthusiastic about the course that was going to be imparted made the attendants feel eager to take it. And also another instructor who was quite shy at the beggining and told us it was her first time doing this, I think people related to her
- Diving straight into the derivations on the blackboard without giving any overarching vision for what the course would cover.

- Start a course by explaining how to do a particular step of a process (signal normalization), without any introduction to the technique or what it was supposed to be (microarrays). 2 hours later we were still unsure what a microarray was.
+ instructors showed lots of passion, dedication, and were extremely prepared on the subject; they started with getting to know their audience
- very little interest in the learners (I recall in particular university courses), very little time invested in them; almost no interaction in general, no introduction at all
- moving on too fast, using dismissive language, no interaction
+ made goals of the workshop clear

What's in an Introduction? (10 min)

Get into small groups (3-4 people) and discuss the questions below for 5 minutes. Take notes on your answers.

- **What do you hope to accomplish in a workshop introduction?**
- - I want to give a clear overview about what I will teaching and what the learners can expect to learn when they are attending the course. Also introduce the teachers and relevant parts of their backgrounds.
- - make clear, when there are breaks, so that learners are prepared beforehand with the load of information
- - building a welcoming athmosphere to learn
- - Set expectations on what the learners would accomplish after the workshop
- - set a learning atmosphere, put people at ease
- - know each others, trainers and trainees
- - explain safety features, emergency features
- - better understand the class diversity, to foster a positive learning environment
- - set clear expectations from the workshop
- - show that you are approachable (and how)
- **What information do you need to include in an introduction to accomplish these goals?**
- - clear curriculum, schedule (including breaks), outcomes, teacher background information, required materials, technical checkup before starting the lesson, expected student knowledge prerequisites
- - get to know the people you work with
- - Learner's persona and their main goals in attending the workshop
- - code of conduct
- - Rules (like how to use sticky notes, chat etc. in zoom)
- - ask the perrsons to introduce themselves and briefly mention their background/area of expertise
- - ask the class to read some material in advance

After 5 minutes, come together, and combine ideas as a large group.
Compare your ideas with the list of topics below.

- Did you miss anything?
- Did you come up with something that’s not listed below?

Goals For the Introduction

- After the introduction learners should:
    - believe in your competence to teach the workshop
    - be able to predict the type of instruction
    - know what will be taught
    - know what will be required of them
- The instructor should:
    - have an understanding of who is taking the workshop and what their expectations are

Components of the Introduction

1. Set positive first impressions
2. Introduce yourself effectively (and have other workshop leaders do the same)
3. Clarify learning objectives and expectations
4. Help learners learn about each other
5. Set the tone for the workshop
6. Collect baseline data on learners' knowledge and motivation
7. Whet learners' appetite for workshop content
8. Inform Learners of Logistics

Practice Your Introduction (10-15min)

Imagine you have completed instructor training and you are about to teach a full lesson around the material you have been practicing teaching today.

1. Write out some notes, covering some of the topics described above:
    1. Introduce yourself effectively
    2. Clarify learning objectives and expectations
    3. Set the tone for the workshop
2. Return to your groups of 2 or 3 and each give 2 minutes of your introduction.
3. After each introduction, provide 2-3 minutes of feedback.

Key Points

- A planned introduction is a helpful tool in setting the workshop environment.
- Introductions should include both practical information and start building relationships.

XVIII. Putting It Together

https://carpentries.github.io/instructor-training/24-practices/index.html

Picking up the Pieces (5min)

On your own, on a piece of paper, make a list of all the concepts you've encountered in this training. Your list can include everything from educational/teaching theories to specific in-classroom practices.

Organize Your Knowledge (5-10min)

Let's put the pieces together by creating a visual organization of information.
We suggest doing this in two ways:

1. If you are comfortable with / like concept maps, trying integrating all the topics above into a single concept map.
2. Use the provided handout to organize topics. Here are two examples:
    - Handout One (https://carpentries.github.io/instructor-training/files/handouts/Wrap-Up-doc.pdf, with example content: https://carpentries.github.io/instructor-training/files/handouts/Wrap-Up-doc-example.pdf)
    - Handout Two (https://carpentries.github.io/instructor-training/files/handouts/Carpentries_teaching_practices.pdf)

Work on this on your own. There's no "right answer" – this is about you building up a mental model, moving from "novice" to "competent practitioner".

Once you've organized your thoughts, move to the next exercise.

Parting Thoughts (5min)

If you didn't think about these issues when organizing your topics in the previous exercise, now consider:

- What is your mental model of teaching?
- Can you identify why each topic above applies to teaching for the Carpentries?

Further Reading / References

Susan Ambrose: How Learning Works. A very good introduction to science-based teaching (for higher education): https://firstliteracy.org/wp-content/uploads/2015/07/How-Learning-Works.pdf

Greg Wilson's book on **teaching "tech" to researchers** - it contains a lot of practical knowledge around teaching Carpentries workshops and transports much of the spirit of the Carpentries https://teachtogether.tech/en/index.html

Tips & Tricks **for instructors**: https://docs.carpentries.org/topic_folders/for_instructors/index.html

Recommendations for **teaching** Carpentries workshops **online**: https://carpentries.org/online-workshop-recommendations/

CarpentryCon Recordings on Youtube: https://www.youtube.com/results?search_query=CarpentryCon

Key Points

- Having a plan makes it easier for you to remember to implement the important teaching practices you've learned.

XIX. Wrapping Up

https://carpentries.github.io/instructor-training/25-wrap-up/index.html

One Up, One Down (5 min)

Provide one up, one down feedback on the entire two-day course.
Just as in our regular workshops, we collect post-instructor-training-workshop feedback. Your participation will help us evaluate the efficacy of this training and improve the content and delivery of the lesson materials.
Write your feedback below your name:

- Christophe Klopp
- + the very good "ambiance" of the course
- - some more insight about the psychology related work on which the principals are based and what fraction of the population is receptive to those prinicipals.
- Mark Abraham
- + wonderful enthusiasm and execution from both the instructors. Two rounds of live coding was useful.
- - Many exercises had too little time to both plan and deliver in a group and give feedback (advice not to try to plan and introduce a whole workshop in 2+2 minutes might have worked better for our group)
- Jesse Kerkvliet
- + Practicing the live-coding was very useful. Doing it a second time was more difficult but also very informative. Altogether it has been a great workshop with very motivating instructors.
- Simon Steinkamp
- -(why do I have to give negatives all the time? there's so little^^) I think (going back to yesterday), it might have been useful to focus a bit more on giving feedback. In being a bit more specific for positive feedback, and that negative feedback is Ok. I felt it a bit too cuddly. in the breakout rooms :)
- Luis Moßburger
- + What I could not express yesterday: You two have managed to make me a full-hearted Carpentries community member in two days. Whenever somebody will ask me about this I will eagerly share the vision and motivation behind this community! Thanks for introducing us into that world and all the best! :))
- Andre Pfeifer

- -I am very impressed and have taken a lot with me. At the same time there is nothing important I can feedback. But I have a small remark because I have to. I found it easier to listen to Annika than Konrad, because I could look her straight in the face. With Konrad we looked up from below.
- Artem Zhmurov
- +Practical exercises, although I would prefer them to be a little less time-constrained. Also, very valuable information on the workshop organization.
- Roberto Di Remigio
- -We were always short on time with the live coding and I think highlighting beforehand that this will be an exercise might help with that.
- Fernando Pozo
- +It was an amazing opportunity for me to get involved with effective learning skills and experts in the field. Thanks also Annika and Konrad for your work. Clear explanations, nice tone, very good feelings with you and the partners. Negative feedback is hard for me to construct after these 2 days...but maybe some extra real "coding" exercises (like webpage preparation with github) could be positive. All the best.
- Andre Pietsch
- -There where 3 bigger practicing tasks on the second day: live-coding, live-coding reloaded and then the introduction task. In our group the time for the introduction task wasn’t enough and I think it would be nice, if the introduction task would have been as long as the live-coding part and with a maximum of three people. Maybe it would be good to give the notes for the introduction practice also as a homework, so you just have to change the notes a bit after the theory part.
- Mikhael Manurung
- + Friendly interaction between the teachers, no one tries to outshine the other. Many opportunities to practice without any fear of harsh feedback.
- Laura Martinez Gomez
- -I really loved it when you used images and diagrams, probably because I'm used to present and see presentations with slides, so maybe I missed this format. But overall I loved the course! I really learned a lot and the atmosphere was super friendly. And I also loved it when you refered to personal experiencies when teaching. Thank you both!!
- Joana Viana
- +I've learned so much during these two days, thank you very much. My favourite part has been the discussions arount the novices mindset but also learning more about the Carpentries Community. Thank you both!
- Naoe Tatara
- + Very good training, so intensive and much to learn, but not really overloading. How well this is well designed!
- -Actually nothing negative today, but probably this particular one, i.e., one up, one down, is maybe a negative one when you cannot find anything to improve.
- Agnes Brauer
- +This was a great workshop, I learned a lot about teaching (e.g. formative assessments and engaging the learners being powerful tools)
- Michal Michalski
- - I would prefer more slides highlighting key concepts, key take-aways, even one for each part
- Aili Sarre
- +The course was very interactive
- Mishka Nemes
- -Could have made better use of the Zoom functionalities (raise hand, polls as formative assessments etc)
- Camilla Bressan

- +I really enjoyed these two days! I did not know what to expect because I took this training to teach another course (not a carpentry) at my institute. I had no idea I was about to discover such an amazing, engaging and growing community. I loved the interactive exercises with this etherpad, they are extremely effective. I loved the passion and engagemnt showed by the instructors.

Minute Cards (5 min)

In addition to giving one up, one down feedback. Instead of using sticky note feedback, go to this **Padlet** https://padlet.com/arockenberger/minutecards and fill in your "sticky notes". (The feedback will be anonymous)

Post Workshop Surveys (5 min)

Assessment is very important to us! Please complete this five-minute post-workshop survey. (https://www.surveymonkey.com/r/post-instructor-training)

Key Points

- Feedback applies to all kinds of learning, including learning how to teach.

In case of fire, contact: checkout@carpentries.org