### General feedback (email)

---

Thanks a lot for the workshop, I thought it was very good overall! I especially liked the Git/GitHub, testing, automation and documentation parts. Like I said when we talked, I think there’s room for improvement when it comes to syncing the different speakers and some of the pedagogics used/not used. I code mostly in R and bash, so a bit more R would be nice, but not absolutely required. If you keep things more general like you did with the Git / IDE parts and focus less time in total on things like Jupyter, that’d go a long way to solve it. Or maybe add some stuff for RMarkdown.   
  
It’d also be nice if you try to keep the scheduled times you so nicely added, or avoid them at all. I like that you were free to go back to previous lessons again to finish them, but the fact that you grossly deviated from the schedule can be frustrating for people looking forward to a specific topic (like me). Either don’t have a detailed schedule and feel free to hop around, or have a schedule and try to stick with it (as far as possible, of course).   
  
As far as the green/red stickers go, that’s a very nice idea but didn’t seem to be implemented that well - most of the teachers seemed to sit and do their own work rather than trying to find red stickers to help, and leave all the helping to the current lecturer.

---

And once again it was a very interesting workshop.

---

I really liked your presentation about collaborative distributed version  
control, specifically the working with remotes where you showed lots of  
pictures of commits and branches and how they develop when you do this  
and that. I forked and cloned this repository, but when I open this  
episode in Github, I don't see the pictures any more.

---

The course was really useful for me, I am so glad I got the opportunity to participate.  
I have already told most of the PhD students at my department about how useful the course is and that they have to attend your course the next time it is given, so hopefully you will get a lot pf applications from meteorologist for the next course.

---

It was a pleasure to meet you all and I had a really good time learning!  
I will keep tuned for the next upcoming events / courses from CodeRefinery, and I hope to see you another time.

---

This course is very interesting and quite useful for my future research. I learned a lots of knowledge. It should go ahead for more people. Thank you again.

---

Please do save my email and keep me updated if you plan on having coding events and meetups. I will make sure to do some pretty code for you to review by then ;)

---

Thanks so much for organizing the workshop! It was really great to learn more about stuff I have some experience with and also get to hear about and try hands on a lot of stuff I didn't know before!

### Git intro

* I liked the tempo in the module “intro to version control”
* The module “branching and merging” went a little too fast, too hard to understand for a novice
* Good to start slow at the beginning and repeat basics
* Don’t know yet
* The material is very clear for the beginners
* The explanations can be improved. In certain places it was confusing

General: The Git Intro webpage is very nice, but it should be followed more closely during the lecture. Maybe a little bit more of preparation could be necessary in order to know how much time it takes to present the material

* The course material is well illustrated
* More illustrations on the presentation would help
* I think the overall presentation needs improvement. It was very confusing. Going back and forth all the time
* None
* Perhaps keep a visualization of the repo state alongside the presentation (whiteboard or something?). Especially for the reset, merge, etc.

General: When explaining “stage” in git, mention it’s local, i.e. you could do that on the plane (no connection to others)

* Concise examples
* Really rushed “branches” section
* organization/rythm/explanations
* None
* Online document is really good to follow, a lot of info
* Interactive session is good way of learning
* Presentation was a bit chaotic
* Could do with a bit more “why” we are doing it (“when you want to remove/edit …” e.g.)
* The details in the explanation was really good
* Taking up more time in some [...] white [...]
* Clear structure of course and content of this course in the beginning
* Code example [...] foo on website was not presented, instead hello -> confusing for people not used to python
* Explanation hard to follow, little chaotic sometimes
* Fix projector resolution/windows setup before starting; a lot of unnecessary time went by with nothing happening
* Since there were instructions of things to do before the workshop, please do assume that people have done it, instead of repeating all the things
* It seems that some basic instructions of text editors would have been useful for some of the participants (in the instructions before the workshop)
* Having red on black background can be hard for some people to see (color scheme on terminal)
* Very good introduction (for me as a total beginner)
* Add info on ssh-keys to the instruction website (was missing for windows, not sure about the others)
* The tutorial online page is very didactic!
* Format of the room is bad for seeing and hearing the teachers
* Time! Better to skip things rather than to rush through because then you run the risk of loosing people
* Nice examples, good review
* In the end it became a bit fast. Maybe spend less time on setup (include this in the installation instructions)
* Good feedback on questions
* Show visually the branches beside, while discussing (two screens)
* Speak a bit louder
* More emphasis when changing subject
* It was a bit fast, sometimes
* The checkout/resent things could have been more clear
* Good introduction to git
* Could be a bit more organized
* The online instructions are very good
* The actual implementation of exercises was a lot messier, use the insturctions more and more directly!

General: would it be possible to get a history of all the actual commands shown + given on screen, please

### Code complexity

* Explanations are good
* Instead of so many examples, 2 to 3 examples and how they fit into a slightly larger code would be very helpful to see which version is better
* Very nice presentation! Easy to follow, pedagogical.
* Would be nice with a bit more R, and examples/thoughts on highly linear workflows
* Great to follow presentation.
* Nice with examples and questions in multiple programming languages
* Some of the things will only be proven useful in later sessions like “this will be good for testing…” is a bit harder to grasp now but wets the appetite for the next days :-)
* Very good concise examples in last part
* Although concise, many example needed experience in the language -> not language independent

General: try using more small examples/analogies in first part

* Well explained
* Spend more time in comparing examples
* Real code examples and discussion
* Would have taken notes if I had had printouts with note section
* Too long, got tired, especially after lunch!
* Very general introduction but specific examples that illustrated the problem for people with different backgrounds
* For me nothing, but not sure about people with more experience in software development
* Quiz was very good, passive active learning
* With the different functionalities of different languages it was a little hard to have a “red thread”. Maybe less code specific things in first part, and save code specificities for quiz
* Very good walkthrough and especially involvement of audience
* Again maybe too much packed into one session
* Covering many languages and their opportunities + pitfalls
* Not always clear (beyond intuition, “elegance”) why a choice is better than another
* I like the discussion of the examples and good presenter
* I didn’t like the explanation of classes being bad, it made me confused. Should I use classes or not
* Nice examples and coverage of concepts
* Difficult for me to keep focus for long stretches of time (have noticed that many other workshops and courses at KTH are organized so)
* good.
* As I’m not very familiar with python, some examples went a bit too fast or over the head. But not too bad
* interesting
* Too many python examples
* Very clear, focused
* Great slides
* No strong connection to version control and collaborative code development, would be nice if it connected more with previous presentation
* Good concepts regarding how to think about code
* Could be more in-depth
* Nice presentation and slide illustrations
* It would be nicer to have the quiz examples in at least 1 main language: ex: python (75% of audience). I did not follow well the fortran examples…
* Good presentation and interaction
* Maybe try to use more language independent examples
* Summarized, content
* Too much python oriented
* Very nice presentation.
* Not much to say other than that maybe it were too many short/simple examples, I would have preferred some more complex or real life examples

### DevOps

* Overview of tools that I didn’t know even existed
* Learned that it is possible to do some kind of code testing
* Emphasize that this is an overview
* Try to present in a simpler language that even people understand who have never heard anything about devops at all - why important?
* Interesting subject, easier to grasp when periodic table was shown
* But, it would be easier with a consistent example, relating to research, being used throughout each part in the “workflow”
* Very interesting:
* CI
* Docker / virtualization
* Need more info about unit testing
* Glad that we really clarified stuff and managed to understand each other
* Somehow match the level of the audience. We did that in the end, and this is different for every audience. So, a more basic introduction to DevOps was needed for us, this time, I guess.
* I still think it turned out good, lots of discussion
* Too theoretical and vague. Better to start with a tool and show what it can do. Then presenting theory would be helpful
* Good topic to explain things that many have heard about but don’t understand
* Needs a more gentle introduction
* I think this presentation is scheduled too early in the workshop. Maybe it could be nice to “emulate” the workflow of the development process, to show how, why and where you need all these tools. It doesn’t have to be very deep here, better to maintain an overview at this stage, and then have more specific sessions for the different tools.
* DevOps and CI etc are very good tools. Very nice connection to simple vs easy from previous presentation
* Don’t use terms not yet introduced, e.g. DevOps, CI, push/pull
* Explain abbreviations, and connect better with previous presentations. E.g. push/pull not yet introduced which will confuse git newbies
* Explain better the abbreviations and jargons used in the beginning
* Improvement on the written material since it is not self explanatory
* I felt a bit confused about the topic
* Afterwards it was clear that this was a great overview of what we will look into more during the week.
* Periodic table link is super useful
* At least my background was not big enough to get the presentation at the beginning. Definitions were just floating in the air
* Dont be scared to be more guiding, we have no clue where to start
* I liked your introduction to DevOps. It made me want to go home and google the “periodic table”
* (still unclear) The steps in CI is still unclear for me, please explain again tomorrow what programs should be used in each step
* Very good attempt to give overview over code development toolchain,
* … unfortunately little overwhelming
* Unclear why i should care, what the advantage is, example would have been good
* Intention to give an overview is very good -> make this more clear [i.e. the intention]
* Give more specific definitions, better introduction to the general topic
* Maybe prepare 2-3 different more specific examples from projects in fields represented in the audience
* Showing useful tools
* Better contextualization needed
* Uses of terms without explanation (“what does CI stand for). What DevOps stands for exactly (the abbreviation) is still unclear to me
* It would be nicer with a presentation instead of scrolling down a webpage, and more clear what you are talking about at the moment
* This session contaied very little actual things for me to take home and actually work with.
* I know Docker and Vagrant, but only the names. No examples or anything… At least give one example software for us to start with!
* none
* Lack of context presentation, not clear idea of the use. Lack of connection with the rest
* Start with the background and the purpose and what problem we have that this solves.
* Now it was new words lots of them, without any particular meaning
* But in the end it was OK :-)
* Bring out the periodic table a bit later when we need it not when we are overwhelmed already
* We need an overview of tools so that is a good idea
* But we need very clear information that we are speaking about certain wide categories of tools, or concrete examples, not general questions what we should answer
* New concepts, I will recognise these terms when I encounter them next time (this is really important for me!)
* Could use some sort of structure or hierarchy to make it easier to follow. Maybe some clearer examples etc. Maybe put it a bit later
* Good to get an overview of tools
* A little frustrating to not delve into anything. Might be good to choose one or a few tools and dig into them

### Git collaborative

* Loved the whole class fork/code review project! Also like the detailed step-by-step instructions and box sketches on the website
* Don’t have any suggestions
* Good pace, good management of questions
* Maybe too slow for a 3-day course. Increase course? More preparation exercise?
* A lot of specific examples that lets you get familiar with git
* Could maybe focus a little more on hands-on stuff
* Great exercise! Really got a feel for what collaborative development with Git looks like. It was also good to have the conflict from group 3 and see how to resolve it.
* Very clear presentation, good examples, good speed.
* Way too much content for half day, reserve full day next time for just that.
* Very detailed explanations that could be practised during the hands-on
* The first hands-on on the morning was very fast
* Good exercises and structure
* I had trouble following which set of slides we were on on my computer
* Nice documentation and illustrated material
* The graphic illustration could be improved in a more intuitive [drawing with arrows moving forward instead of backwards and smiley instead of HEAD pointer]
* Everything was very well prepared! Nice figures.
* Take a look at Atlassian’s git tutorial for inspiration on how to make really clean, succinct commit graph figures
* Good exercise
* But I think it would have been easier if we each could have done it instead of having groups.
* Very good session! Learned a lot
* Wouldn’t actually hurt to repeat the actual task (what we are supposed to do) and maybe give more hints, 10 minutes into the exercise or so
* Great explanation to forks
* Maybe a more advanced exercise?
* Good practical exercise
* Be more realistic with the scheduling!
* Very nice repetition in the beginning of yesterday’s lesson!
* Nice explanation of arrows! Maybe take that in beginning. I really like the practical exercises!
* When showing how pull/push/rebase can be confusing it might be good to show both the initial pull + error attempt and the e.g. rebase solution at the same time.

### Jupyter Notebooks

* The whole lecture. Explanations and exercises
* None
* Very good hands on test/exercises
* Learned about a completely new tool - exciting! Now I have to figure out how I would possibly incorporate it into my workflow
* Include in intro more general information:

1. How would a researcher use jupyter notebook in his/her workflow?
2. Felt a bit lost in beginning what this is actually useful for in daily work life

* Nice module scheme, it was easy to follow the lecturer or go ahead with the instructions
* No complaints
* Organization, explanations, exercises
* The explanation of the use vs git
* Good lecture, I learned a lot about magic and widgets that I did not know from before
* I did not really get how you use the debugger in magic
* I think too much material was covered, and I also think we spent too much time on Jupyter. It might be useful for specific purposes, but I would have preferred more time on version control, testing, building etc.
* It is an interesting tool and the exercises are very clear
* We don’t get the big picture of how it can be used

General: Jupyter seems interesting, but what can we use it for? Can you give examples on cases where in serves a purpose?

* The topic, jupyter is great
* Don’t know
* Nice documentation and presentation. Very clear
* Would be nice to have more time for it
* Maybe more code optimization could be inserted here as exercise
* Clear presentation with very good examples, easily hands-on
* Installing the line profiler didn’t work well under Windows (MS Visual C++ missing), the notebooks are powerful but still appear like a sandbox to me
* Overview what is possible in jupyter
* Went a bit too fast in the end (but maybe only because I got stuck on one of the exercises and couldn’t really follow afterwards)
* Very good run-through with multiple examples
* Not enough time is a bit frustrating
* Nice layout, quite easy to follow, good exercises
* Some examples relies a lot on what is written on screen. Text is quite small and can be hard to see
* Very comprehensive but still easy to follow intro to jupyter notebooks
* Research how to best use notebooks in collaborative environments
* Great session. Lots of functionality covered, seems like a great thing to try out, will definitely look into lessons 2 and 3
* Would like to spend time on them in class also but the workshop is short for that, so it’s good to have them.
* Practical examples were great!
* Given that ~ ⅓ of all participants don’t use Python but rather R, plus the fact that RMarkdown is (like mentioned) similar and is not replaced by Jupyter, there seemed to be a bit too much time spent on Jupyter. Git, GitHub, etc. are all independent of language.
* Super fun to see how much is possible with it, had seen it used for simple python before, but not for all kinds of languages and data types!
* A bit much for the time but probably better this way, as long as we have resources to try out by ourselves what we wouldn’t cover now

### Make/CMake

* Very interesting and useful
* Too quick, missed detail explanations
* Make and CMake are useful tools and important to know for anyone working with code. Good with real examples
* Better motivation. Explain why we need it. What does it do? Explain structure of makefile better. What is a target? What is a dependency? Why? Better explanation what is make/CMake syntax, what is Bash, what is variables etc. I think this was very confusing to people not already familiar with make,bash,unix.
* Could use a better intro as to why we should use Make. Would also be better to include more basic theory on how Make works (dependencies, time stamps, etc.) since many questions made it clear it was hard to understand for many people. LOADLIBS,LDLIBS, etc were badly explained at first, and not very good answered when questioned. I still do not know what they are for! General confusion about executables, objects etc for us that only work in scripting languages. This session was in general very hard to follow and it is hard to know why we are doing things and what we’re supposed to learn from it.
* Very useful hands-on examples
* It went too fast to keep up on our own computers
* Nice topic, I didn’t know about cmake
* More examples from real-life where CMake would be useful
* 1st session: Nice build up + follow-able speed. I’d never seen any C++ or make or anything and was still able to follow along with the exercises
* 1st session: Could do with a little more “the goal of this is to..”
* 2nd session:Goal of the exercises was nice , good explanations
* 2nd session:Very fast, but ok for just listening, not for doing along
* A lot far-from-my-land show :-)
* Overview of what is possible in CMake
* Depending on the audience a brief general introduction int what compiling is could be very useful to bring everyone on the same page
* Nice introduction, make became a lot clearer to me
* Big CMake example went a bit fast , there were many things
* Nice presentation and written materials
* Maybe more illustrations or visualisations of how cmake works would help.
* CMake seems to be a completely new language to be learned before taking the course
* The step-by-step execution of CMake example is very good.
* In the end it was to fast that I couldn’t follow. Maybe we might have focused only on CMake , leaving make out of the schedule.
* Good overview with nice type-alongs
* Slightly too heavy on the C++ stuff so if you have no experience it’s difficult to follow along
* I like the high tempo and realistic example
* Improve: ….
* To be clear, for me it turned out good -- with questions was able to clarify everything I wanted to ask
* I think the order of the material was good, and motivating CMake with “make” first was really good, but many were inexperienced with “make”. It might have been fast since they wouldn’t have experienced the downsides of “make”. On the other hand, of course, you don’t want to spend a lot of time explaining it cause we won’t be use it anyway.
* It was really good to include CMake in the program! Lot of things are clearer to me now and how where to start if/when I need it. Would really have liked to do all the exercises at a slower pace, but this is very tricky to do for a workshop of this length and also because it maybe won’t be useful for many people. On this note, I think it is useful for almost anyone, actually, that has ever needed to run make so that when there’s errors we’ll have a better idea of what is wrong with what we’re trying to build
* At the beginning it somewhat flew over my head. I had no clue about what “make” does, and had forgotten how coupling in C++ looks like (source -> object file -> executable), so needed to think more and ask more.( Which I also think is good, btw). But for future if most of the class really has no clue it might be useful to really state obvious things so that nobody gets lost.
* I have only coded in Python and Matlab before, so for me the whole lecture was difficult to follow. Some more basic examples of make code might have made it easier to follow.
* Still, the lecture was useful for using cmake files in the future.

### IDEs

* Really nice overview of some IDE capabilities I’ve never known they existed at all, e.g. extract method of extract constant
* Could be better: explanations connected to what you show in IDE could be a little slower - sometimes it got so fast I could hardly see the mouse moving or knew what you have done
* Nice introduction to IDE
* Nice presentation and “hands on” showing the software
* Would be nice to use more illustrations on the written material
* More detailed written material would be good
* Great with an IDE introduction, I think it is useful for many researchers, especially if working on larger projects.
* Lesson material somehow messy. Contents is ok but hard to use as resource material after the presentation.
* Interesting introduction.
* Could be improved: went a bit fast, so I’m still daunted. Probably will continue with vim. (Will try this IDE though … maybe)
* This session was the most useful session for me!
* NA
* Great presentation, seems like good stuff. Will definitely try it! Great that we had the git stuff before this, makes us able to see usefulness of having it integrated in PyCharm.
* Was a bit fast to follow :-)
* Great lecture that made me see the benefit with using an IDE instead of Sublime. I will definitely give it a chance. Thanks!
* Good combination of general overview and specific features
* Maybe something language agnostic would be more widely appreciated

### Jupyter parts 2 and 3

* Very nice examples and well prepared notebooks!
* Part 1: Very clear presentation, techniques seem to work across Python
* Part 2: Intro into powerful panda, very clear present.
* Part 1: When does it not work? Not addressed
* Part 2: Less on panda, more on “interactivity” of mpl3d would have been good
* Very nice Jupyter summary! It would be nice to have a similar thing for most (all?) sessions. Lesson 3 was nice for us who use R (because we use R since we analyse data)
* I like the topic, nice basics and good inspiration for future
* Try to use more code to relate to, take an actual research data and do the data analysis hands on. Don’t have them pre-written.
* Great work on explaining pandas.
* Still unclear when I can use Cython.

### Testing

* Nice overview, easy to follow the general idea.
* Very nice workflow exercise! Collects lots of what we’ve learnt.
* It seems that the languages taken into account is Python, C/C++ and Fortran, but ~⅓ of participants (2nd largest) is R. So … maybe add more R stuff? The rest of the course (so far) has been quite nice with at least acknowledging R, but here it was only “use this package”, but nothing else.
* Very good session! Learned a lot :-)
* This should prob come before the more broad DevOps session, so everyone will have something to hold on to
* Very well planned! Great hands-on!
* I can’t think of anything …
* Good motivation and examples
* Good topic. Good exercise
* Should have been before the last topic of the first day
* Very nice with a practical exercise that ties together several tools
* Best hands-on session so far!
* No copy/paste to/from Git Bash in Win
* Very interesting.
* The testing scenario felt real.
* It would have been good to set up .gitignore file also, so no wrong files can be committed by mistake.
* Also great exercises! Really got a feel of how issues are raised and resolved and how having someone submit a pull request looks like.
* Nice documentation and presentation. Very well outlined!
* More exercises would be nice. Maybe something as “homework” or “after course” exercise.
* Really good exercise. I learned a lot!
* Still don’t know how to write and come up with a good test function.
* Example exercise was fun and good
* ?
* In general very useful information & easy to follow even without much experience in software development.
* --
* Very interesting material
* I only suggest that apart from only Python testing, it would be helpful if there are some examples on C++. We need not discuss them in class but something for us to take.
* Super useful to go through the whole thing a controlled situation. Also got the whole point of forking, pulling, committing, pushing, pull request better now, not just see the test.
* None

### Documentation

* Very useful and clear illustration of the doc workflow
* Extremely fast, hoping for good material (couldn’t even check that :-) )
* Nice presentation and documentation
* Would be nice to have more time to discuss licenses
* Good presentation
* Could use more time. I think this subject is very important in research
* Good choice with RTD and Sphinx. Great motivation.
* Good use of both whiteboard and projector
* It would be nice to have an example code that we worked with all through the course and used all tools with. For example, make a scientific toolbox, code it -> git it -> test -> … -> sphinx it.
* A bit fast talk, slow down and take the time
* Super useful! Explanation on the website really clear.
* Fast :-)
* Maybe a bit more on “what should be documented?” “how do I get it in a working routine?” “how do I convince my supervisor that it is useful to put time in it?” :-)