# Advanced Scientific Programming in Python

a Summer School by the G-Node, the Bernstein Center for Computational Neuroscience Munich and the Graduate School of Systemic Neurosciences

August 31 – September 5, 2015. Munich, Germany

# Evaluation Survey Results

## Method

The survey has been administered with a web interface created with the LimeSurvey software available at: http://www.limesurvey.org
All answers have been submitted by October 10, 2015.
No answer was mandatory.
The free-text answers have not been edited and are presented in their original form, including typos.

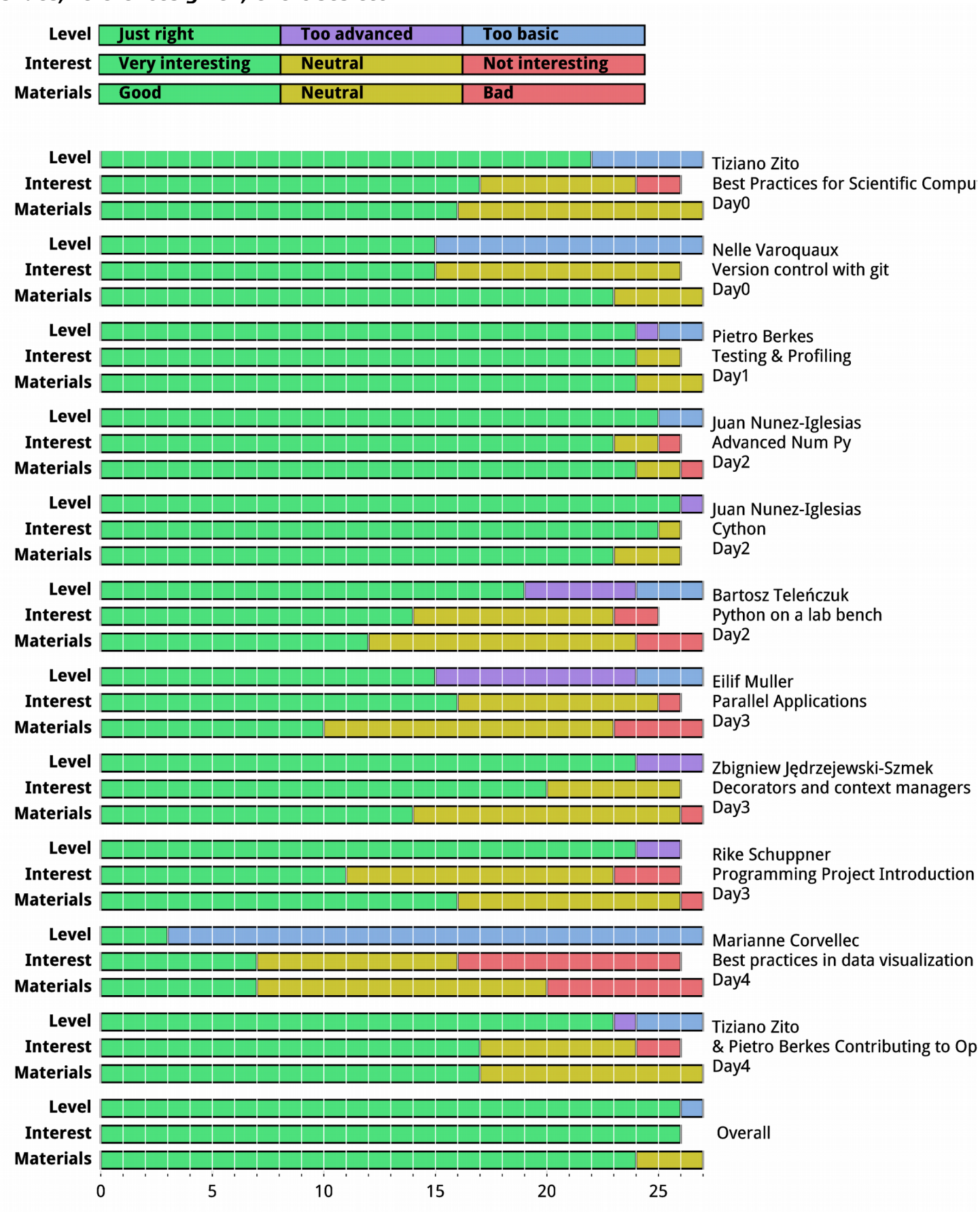
## Attendants and Applicants Statistics

| | Attendants | | Applicants | |
|---|---|---|---|---|
| | 30 | 10% | 300 | |
| Different nationalities | 21 | | 55 | |
| States of affiliation | 10 | | 36 | |
| Female | 15 | 50% | 95 | 32% |
| Male | 15 | 50% | 205 | 68% |
| Already applied | 13 | 43% | 35 | 12% |
| Bachelor Student | 0 | 0% | 10 | 3% |
| Master Student | 2 | 7% | 58 | 19% |
| PhD Students | 20 | 67% | 163 | 54% |
| Post-Docs | 6 | 20% | 34 | 11% |
| Professor | 0 | 0% | 4 | 1% |
| Technician | 0 | 0% | 1 | 0% |
| Employee | 1 | 3% | 19 | 6% |
| Others | 1 | 3% | 11 | 4% |
| Completed surveys | 27 | 90% | | |

More stats about attendants are available at: https://python.g-node.org/python-summerschool-2015/students

# Lectures & Exercises

***Q: Grade the level of the lectures***
***Q: Grade how interesting were the lectures***
***Q: Grade the quality of the teaching material provided by the lecturer, e.g. the clarity of the slides, references given, exercises etc.***



### *Q: Are some of the topics presented in the lectures not relevant for a programming scientist?*

1. This really depends on your application.
2. The lecture on graphical representation could have been less art oriented and more technical oriented, but I respect the originality.
3. I consider that without a previous knowledge of object oriented programming, decorators and wrappers make not too much of a sense since the level required is high and this python feautures are not that straight forward to understand, even for a person with some Python knowledge
4. The parallel programming lecture was not very helpful as not all of us have access to clusters and it was not clear how to use it for data analysis more than just simple computations.
5. The last lecture (contributing to open source) was 90% identical to Nelle's.

### *Q: Are there further topics relevant to the programming scientist that could have been presented, given that the total time is limited*

1. I think parallel programming is one of the most important topics. It should be covered better. I understood nothing of the lecture and even less of the exercises.
2. How to store and manage one's data (db, csv, pandas, ...)
3. I would have liked to have seen some GPU programming.
4. A more holistic lecture on visualisation would be nice.
5. Pandas, SciPy packages
6. Object oriented programming
7. I guess it was on the schedule at some point, but I did not have much experience with OOP before. A introduction to that would have been nice.
8. an intro to panda might be interesting
9. Big data analysis (e.g. Spark)
10. Object oriented programming. Advanced git practices.
11. object oriented programming
12. Would have liked to see more on data visualization especially tips and tricks to use matplotlib. A short introduction on classes would have been helpful
13. It could be useful to integrate further topics regarding all aspects of dealing with data using powerful package, pandas, from loading different data files, cleaning the data, merging data. Also, the using of Python for building and working with database system would be very interesting topic as it really demonstrates the power of Python in the era of big data. At the introduction level, I think it takes one full-day session for all of those including lectures and practical exercises.
14. I think the concept of object-oriented programming (OOP), why it is useful for a scientist, examples of OOP, as well as tools for structuring/organizing data/metadata were missing.
15. Pandas
16. Sparse Matrices
17. simulation and/or stats libraries

### *Q: Do you think that pair-programming during the exercises was useful?*

| | |
|---|---|
| Yes, I have learned from my partner / I have helped my partner | 78% (21) |
| No, it was a waste of time for both me and my partner | 0% (0) |
| Neutral. It was OK, but I could have worked by myself as well. | 22% (6) |
| Other | 0% (0) |

***Q: What do you think of the balance between lectures and exercises? When answering, please keep in mind that the overall time is limited ;-)***

| | |
|---|---|
| Lectures were too long, there should be more time for exercises | 19% (5) |
| Lectures were too short, there should be more time for lectures | 0% |
| The time dedicated to lectures and exercises was well balanced | 81% (22) |
| Other | 0% (0) |

***Q: Any further comments about the lectures and exercises?***

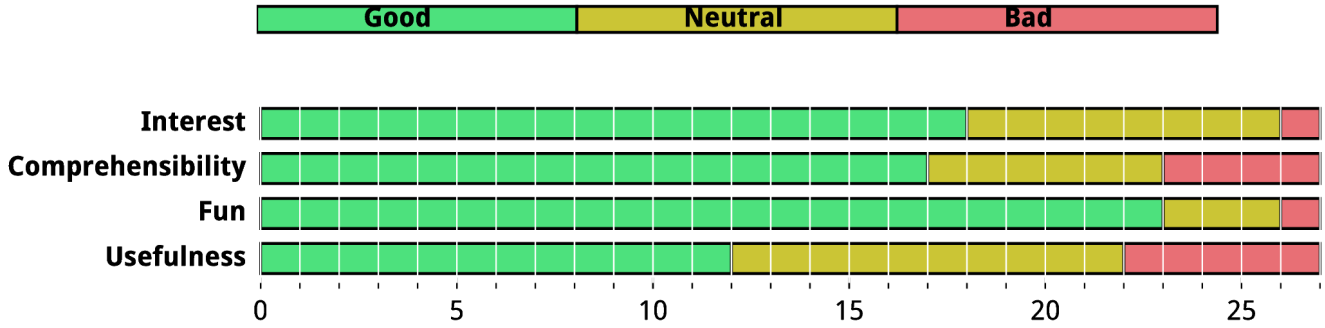
1. I very much liked the lectures with exercises in iPython notebook: it was very easy to follow and learn new python tricks. The lecture about contributing to open source software was extremely interesting and good, please, don't make it shorter, it was just the right amount of time :) It could also go very well with the lecture about git. I thought that the lecture on testing and profiling was a bit too long, although very good. Maybe 1-1.5 hours from it could be moved to the the lecture about data visualization? I wish that the lecture about data visualization was longer and had more exercises and examples of matplotlib and pandas plotting features. It can also be done on simpler datasets, so we could focus on features, not on details of the data (which for people from a different field might take too much time). In the lecture about parallel programming the exercises were focused on parallelization on a cluster, although not everyone has access to it. I think it would be more useful to focus on multithreading and multiprocessing modules. Overall the course was fantastic and I learned a lot. Paired programming worked very well, sped up the exercise completion and improved the learning curve ;)
2. I would say that all lectures, exercises are of great quality and indeed it helps me tremendously in improving programming skills. I am very grateful for this.
3. Parallelisation excersise was pretty difficult since it was a lot at the end. I think it's better to start with a small example where students have to implement themself sth, rather than giving students 5 files with parallel code. That was confusing.
4. Completely disconnected from Zbigniew's talk for some teaching material/talk problem. Our pair succeeded in solving the exercises only by googling the topics and adapting the examples on the finded docs. Eilif's talk was too long, with few exercises left to the very end. Bartosz's lecture was too quick, not enough time to solve the exercises.
5. Earlier exercises in the tutorial on parallel programming would have made it easier to follow. The one big exercise at the end didn't really work.
6. Mostly interesting, but it some of the lectures assumed too little experience with programming and dragged on quite long. The Git lecture, for example, took all day and only covered the most basic concepts, and the NumPy lecture did the same. NumPy is especially important for the scientist, though, and should have been developed more, in my opinion, even at the expense of time discussing Git.
7. The lectures were fine in contents, but as for myself, I learn programming better by trying out the script/code and see what happens. It would have been better if the lecture was more interactive. For instance, I found Juan's lecture using iPython very useful, because I could tweak here and there to see what each line of codes on his lecture meant and keep up with the lecture at the same time. If possible, developing a series of exercises with increasing difficulties that could lead students to a more or less comfort zone would be nice. Also, although I know that it is due to time constraints, it was definitely not so easy to maintain concentration throughout the week.
8. The summer school was just great. You covered all the topics that I was expecting to find. Some comments about the lectures: I would use less time for git, since everyone seemed to know it more or less, but I wouldn't suppress it. Git can be confusing sometimes. I would focus on doing exercises like the one that Tiziano and Pietro did. "Python on a lab bench" was very interesting but the exercises were a bit fast and we didn't have time to finish them. I would do something shorter. "Parallel Applications" was really interesting, but it covered too many things. If you want to give an idea of what options we have and where to look for them, then it's fine. Maybe it is because everything was new for me but, in my opinion, it could be simplified. The introduction was great, but then it was too dense. I would just explain the options there are (differences/pros/cons), and focus on doing more guided exercises. Also, I would just show code in the exercises and not in the slides, but this is a personal comment. I find code in slides hard to read and follow, unless it is a very concise example. The "visualization" class was good, but maybe too basic. It might be difficult to find good exercises because we come from different fields.
9. I think that in general the pace of the lectures could have been a bit higher. For the parts that were already familiar to me (and that I'm most interested in) I already knew most of the stuff, and there was no time to delve a bit deeper. If the level had been a bit higher I might not be able to understand everything about subjects that I don't use, but it might have presented more overall gain.

10. The general feeling I had is that I could either pay attention to the lectures of do the excessive. I know that time was limited, but besides this, there wasn't a clear cut between theoretical teaching time and doing exercises time.
11. I really enjoyed the content of the lectures and found the exercises very instructive. It was especially great to have so many people ready to help, and I always got assistance very soon in case I had questions. I felt that there was a lot of thought and effort put into developing the lectures and the exercises, and the whole school was made with love. Given that people prepare all of that in their free time and out of the altruistic desire to share their knowledge, this school was particularly excellent! The only lecture that I did not like so much was the one on visualisation. Neither the delivery, nor the content left me satisfied. The teacher didn't touch on any of the topics relevant to visualisation, such as color combinations and their effect on perception, overview of some of the existing tools and libraries, kinds of plots (beyond curves and histograms) that one could use, finally, working through some well-known mistakes and listing pitfalls that could be found in galore in the exiting papers. it would be extremely helpful to take several examples of plots and collectively discuss what could be improved in them and why such improvements are needed. I understand that we probably don't have time to implement a lot ourselves during the class, but some basic introduction into colour combinations and top 10 best practices and pitfalls solved would be something that everyone in the audience would appreciate, because we all make mistakes. This lecture should leve people inspired, by seeing the great work of others, I usually feel challenged and motivated to invest more time in improving my plots. Instead, the lecture left me disappointed on many levels. Finally, one of the greatest discoveries for me was how instructive the Python notebooks are! Sometimes the teacher would switch between several screens, and I would not be fast enough to type in the right command in my window - and then I would be lost for a while. But with the Notebooks I could concentrate on the lecture itself, and not frantically taking notes or copying code lines from the whiteboard. Great idea and very nice pedagogically! I also enjoyed a lot working with a partner and in teams! One could learn as much from an experienced teammate as from a good lecturer!
12. It would have been nice to have the tutorial consultation time a bit later in the week or even having a second one...
13. What about time for lectures and exercises, generally it was well balanced, but I had impression that some exercises were more like examples illustrating very basic applications. In other words: some exercises should be more difficult, or (taking into account diverse level of participants) there could be added some additional tasks for more ambitious.
14. I suggest to reduce the time of the lunch break (45 - 50 minutes is reasonable enough, considering the additional coffe breaks) and give a couple of hours more to data visualization. This is a fundamental topic that was not really touched during the course, we had the lecture but we didn´t have time to really put our hands-on some data and plot it.
15. Overall the lectures were great! Carefully laid out with exercises interspersed at proper intervals.
16. W.r.t to the visualitsation lecture: It would be interesting to realy focus on design principles & for each show good and bad examples (or try to improve bad examples). The lecture was a bit of that but I think that it lacked focus and I think most people are familiar with matplotlib, so implementation is not the issue but rather identifying and improving bad design.
17. I enjoyed the pair programming very much and found it useful, both for the programming experience as well as the social interaction. Thought the lectures were all presented in an engaging fashion and provided an insight into possible ways of making presentations I find it now difficult to put all lecture notes together, having to trawl through various git repositories, web sites, etc. It would be nice if at the end of the summer school al lecturers could export their lecture notes into a single format (pdf?) and have those available in one place.
18. About git: I know many people found the git lecture too basic but I was one of the people who had almost no previous experience with git and even after the talk on Monday, I often found myself struggling with it during exercises. One time, I was paired with another person who also knew very little git and I think we spent more time on getting everything to work with git than on the actual exercises we were supposed to be doing, which was a little distracting. Maybe include something on git in the introductory material people are supposed to go over?
19. The idea of putting colored stickers for help and showing understanding was a GREAT idea. The last lecture on how to contribute to open source project with a live example that we could follow on our computers was a GREAT idea.
20. Most of the lectures and exercises were really excellent. I was only disappointed by the numpy and data visualisation lectures. But these are both topics I knew pretty well beforehand. Also, while the IPython Notebook can be an excellent tool for demonstrations I thought it was not used well in the numpy and data visualisation lectures. Just executing code blocks was not as effective as writing our own scripts as we followed along.
21. I enjoyed and learned a lot in most of the lectures; however, there were some very vague at the end. For example, Bartosz's lecture was very promising at first but I could not understand the exercises and the motivation of why we do what we do. The structure could be clearer I think. Besides, lecture by Juan on "Big data in little laptop" could be expanded a bit more since handling and computing big data is very relevant topic for scientists.

## Programming Project

### *Q: Evaluate the programming project.*

**Interest**: How interesting was the programming project?
**Comprehensibility**: How clear and comprehensible was the code and the available documentation? Was it easy to work on the programming project
**Fun**: Was it fun to work on the programming project?
**Usefulness**: Was it useful to work on the programming project? Do you think you may re-use what you learned?




### *Q: Do you think the team-programming experience is relevant to your work as a programming scientist?*

Yes: 74% (20)
No: 22% (6)

### *Q: Do you think that the project should be about a real-world scientific problem instead of a video game?*

Yes: 19% (5)
No: 81% (22)

### *Q: Any further comments about the programming project?*

1. In an ideal world the project should be about a real-world scientific problem. So I ticked yes above. But I think in practice that would be extremely difficult/impossible to achieve given that we're all working on different things. So I think the video game is a good substitute. In particular it was great git practice. It really reinforced the git stuff we'd learnt during the week. It may also be a good idea to have a secondary prize for test coverage. That would have encouraged more people to use the testing procedures we'd learnt.
2. comprehensibility was not really bad but it could have been better. parts of the introduction and documentation was misleading and it took some time to actually figure out what needs to be prepared for the project.
3. I think the topic is fun and should be kept. However it may be of interest to connect it more explicitly to the lectures. One idea is to encourage people to start by writing the tests for their functions before the code. Another idea is to encourage code speed up and keep track of the time. This way people will apply the learned tools while having fun. Our team didn't do this at all. We just concentrated on the strategy and having things running without doing any test. Stress was put Pelita's documentation on the strategy by talking about Kalman's filter, manhattan distance and adjacency list. We don't care about that and we shoudn't spend our time trying to play with these things. I would encourage some control of the teams way of coding, for instance allowing merge only for a high coverage with Travis CI and controlling the speed.
4. It actually did incorporate most of the lecture materials, so it should be both fun and recapping at the same time. However, more time was used for understanding the structure of the codes for the game than the debugging, throwing in decorators, and etc.
5. The programming project was good, but the lecture introducing it was not. It took a lot of time to even understand what we were supposed to do, and most of the students had never even heard of object-oriented programming, which was a key piece of information to make use of the API.
6. I did not enjoy the Pelita project so much (the tournament was a lot of fun though :). It was too long (one and half day only for this project) and too advanced for me to keep pace with the group members. I would prefer to spend time more on exercises about the concepts we learned during the school or their application to our own projects.

7. The final programming project is a bit out-of-context from my point of view. Maybe you could give students 2 options: they can either work on the game and compete with other students who want to go for the game; or they can work on some of their own projects and e.g. improve the code, write tests, add new interactive plots, create github repo for their stuff and so on. Doing that stuff while tutors are around could actually be really useful, because if you run into problems when using you own laptop, tutors can potentially help.
8. This project was very well suited for such collaborative project as it did not involve any prior knowledge of any field. I also found that the diversity of the possible ways to try to solve this problem was consequent, which lead to very different results in the end. The number of people per team was about right for a project of this size.
9. The programming project was great fun and highlighted different approaches to programming that were at first unexpected. But it was a good exercise in team work.
10. Working in teams was a great experience and was pushing us to use repositories. Now, after two days of pushing/pulling it'll be much easier to use repos in my own project. It was also a good exercise in self-organization, dividing tasks between us and brainstorming strategies. The video game as a challenge is a great idea - you have to think outside of your usual field, it doesn't require special scientific knowledge, it's well structured into classes and methods and provides a good example of a structured code. And finally, it's a lot of fun and a rewarding process during the pelita tournament - an excellent ending to the course.
11. An intro to OOP would have been good. Also, a bit more supervision during the beginning phase of the project might help participants to design a more professional plan for setting up coding.
12. The project was my first opportunity to work on something as a team of programmers. It is very tricky - to write code together and coordinate who is doing what without everyone isolating in their own corner and trying to solve the problem on their own. Great work! And great practice to use github. It's great that that was a game, because everyone was even to start with, and it was fun to compete with other teams in the end, We had a lot of fun in our team! And the teachers were very helpful too. Thank you all!
13. Because of the fact that a lot of things were already implemented in pelita code, for my group project focused more on doc investigation than programming. Maybe you should put some traps in code to make game more python-oriented. Now it's sort of strategy oriented.
14. I enjoyed working on the project more than I thought I would. However, I would have been excited to work on an open source project as a team or to work on an interesting problem that someone is struggling with. I feel that with the talent and skills we have as a group it was a bit of a waste not to use them for great good. :)
15. I don't think it was directly relevant but it was still very useful to get to work with a group of people, see how they code and pick up stuff from them.
16. The programming project (hackathon) is indeed a wonderful final end for practical course like this one. It helps to connect people together and somehow put in practice what has been taught. I just wonder since it has been existed for several years, it may be time to think about different project where each group is provided different tasks (data set) and is required to provide insights (results in form of visualization) from those data sets. In this case, the diverse skill set of participants could be explored.
17. In my opinion the number of exisiting functions and bots should be decreased. A lot of time is spent on reading through the exisiting code base instead of writing yourself. A smaller code base could also make it easier to apply the newly learned techniques, like testing or even parallelization. It was great fun though.
18. While I like the idea of the programming project, I feel like a lot of us, due to the time limit, fell back on using elements of Python we were comfortable with rather than attempting to fully integrate the new things we learned. That said, it's a problem I'm not really sure how to solve.
19. It was fun!!
20. Changes in the project presentation. It is really important to highlight the reason why to work with a video-game instead of a "real-world" problem (different background of participants).
21. I think that the fact that it is a game allows everyone to participate and it is fun. Even if you are not the best in coding, you can think about strategy and give ideas. Then, there is a part that you can't control about how the teams work and the roles of people in the groups. In our case, it worked out quite well. Also, the contest is just the perfect way to end the school.

## The School in General

***Q: How do you overall evaluate the school?***

Good: 100% (27)
Neutral: 0% (0)
Bad: 0% (0)

***Q: How do you evaluate the general level of the school? Was it too advanced/too basic with respect to* your expectations*?***

Too advanced: 4% (1)
Just Right: 85% (23)
Too basic: 11% (3)

***Q: How do you evaluate the general level of the school? Was it too advanced/too basic with respect to* what was advertised in the announcement*?***

Too advanced: 4% (1)
Just Right: 93% (25)
Too basic: 4% (1)

***Q: Did you learn more from attending the school than you would have learned from reading books and online tutorials alone?***

Yes: 100% (27)
No: 0% (0)

***Q: How do you evaluate social interactions and social activities at the school?***

Good: 93% (25)
Neutral: 7% (2)
Bad: 0% (0)

***Q: Would you recommend this course to other students and colleagues?***

Yes: 100% (27)
No: 0% (0)

***Q: How did you hear about the school?***

Google Search: 5
Professor/Tutor/Supervisor: 7
Colleague/Friend: 13
Website/Mailing list: 5
of which:
connectionists/comp-neuro: 1
other: 4

***Q: There might not be further editions of the school unless we find a way to make it a self-supporting event. Would you have attended the school even if a fee were introduced to cover the running costs?***

Yes: 89% (24)
No: 11% (3)

***Q: If yes, do you think a fee of about 200 € would be appropriate?***

OK: 62% (15)
Too high: 25% (6)
May be higher!: 12% (3)

### *Q: Any further comments or suggestions?*

1. I really enjoyed the selection of people at the school - both participants and teachers. Everyone was very helpful and surprisingly, well-rounded in other areas except for programming. That (luckily) broke my stereotype about programmers as nerds and people who are less socially capable =))) I had many inspiring talks with people from other fields, who I would have never met otherwise. And the fact that we all came to the same school made me realise in how many areas people use programming, and how many unthinkable applications it might have. Great social events too - the lake was amazingly beautiful! And the movies night too =)) I enjoyed the school a lot! It was very profound and offered a lot to learn, and I will probably need another year to go through all the notes and apply them on practice. But also the ambience was relaxed and fun, like a summer camp =) Thank you for creating this great social micro climate!
2. Many thanks again for this very useful school, superb organization/lectures and fun!
3. The lecture on data visualization was really vague.
4. I think the school features unique characteristics where street-fighting programming skills are brought about together with wonderful social activities. I have learnt a lot from the school and other participants. The very nice thing that I notice is that the gender ratio are pretty equal among participants, which is rarely seen in any technical courses that I have been involved. I do hope that it will be the case in next year. I would like to thank organizers for their huge effort to make successful school, all the tutors for the wonderful lectures, all the participants for being so nice!
5. It was a blast. Thanks.
6. I had fun and learnt more than expected. The social part of the school was very nice too. Everyone was very nice and open. Also, the amount of people was perfect to get to know each other. I have been in bigger summer schools and you don't get to know everyone. I would try to accelerate a bit the process by doing some games during the first days to remember names etc. The team of tutors and organizers were very close to the students and it was easy to interact with them. This makes the environment very comfortable.
7. Thank you again veeeery much for organizing such a great event :)
8. Overall this is an awesome school. I learnt a lot and had fun doing it.
9. Encourage students to plan the consultation time with tutors if they came with a specific problem to solve. Organization of breaks, dinners and social events was great, thanks to the tutors. Introduction by Tiziano was funny and putting people in relaxed and confident state of mind. I expected more advanced topics of git, I don't know how this can be done given the reduced time and heterogeneous level of attendants

# Feedback from the 4 FENS/IBRO-PERC travel grant awardees

1. Taking part in Advanced Scientific Programming in Python 2015 was a great pleasure. Without hesitation I can say that it was the best course I've ever participated. I'm grateful for the time and effort tutors took to prepare their lectures and organizers to provide top-level standards. School program exceed my expectations, mostly because of great people (participants and especially organizing team) who were always eager to help and discuss about huge variety of topics. I didn't have to worry about scholarship issues, because everything was well organized and whole procedure was quick and simple. During the school FENS representatives just delivered me the stipend, which allowed me to focus more on programming than on troubling travel and living costs.
2. The course was wonderful! It was well organized and very useful for my work and career. The travel award from FENS made it much easier for me to attend this course. The communication with FENS about the award was very straightforward and clear. I do not have any ideas about how to improve because everything was very simple.
3. The Advanced Scientific Programming in Python course was really an amazing experience. The courses were very well prepared and organized. The level of the material was good and was precisely what I was expecting from the course. The logistic organization was also flawless. At all point we had maps and instructions to navigate the city and an updates schedule of the activities. But is especially noteworthy the human quality of the organizers and tutors. They were very enthusiastic and always available to help, clarify our questions and share their personal experiences. I have the feeling that I didn't have direct communication with FENS, but only through the organizers. This is not a problem. Actually the entire application procedure was quite simple. I don't have any specific recommendation to make. Overall the school was an excellent experience. Thank you very much for this opportunity.
4. The advanced scientific programming in python course is an excellent platform for scientists who spend a lot of time with Python code but are not proficient enough to take it to the next level. The teachers not only taught us advanced topics in Python, they also emphasised on the importance of contributing to the further development of this language. I thought the way they incorporated Github into the course was amazing, because we were able to learn how to collaborate, submit issues and our own contributions to existing projects.

   The course IS advanced. You find that out the first day when sit at an ubuntu laptop that only has a vim editor. For someone who has only coded using GUIs like Spyder it was hard the first couple days. But now, I have started disliking the clutter of GUIs and I am transitioning to editors. There are also participants and lecturers who are extremely competent, which intimidated me. Some lecturers were so fluent and quick that it was hard to follow but you could always ask questions and there were many TAs to help if you get stuck.

   It is hard to teach programming through lectures. So we had plenty of hands on exercises for each topic. It is also hard to teach programming with powerpoint slides. But the lecturers employed tools like ipython notebook to make sure we followed and had the code on our computers ready to be ran. I was introduced to the wonderful world of iPython notebook in this course. Its versatility has amazed me and I have been using it ever since I came back.

   Programmers usually code alone but this program insisted on pair programming. I never realised how useful it was to have another set of eyes scan your program. It was easier to figure out algorithms, easier to debug and faster to understand concepts. I would be very open to having another person program with me in future.

   The course finished with a highly fun and competitive programming projects where teams competed against each other in a strategic game. This really helped us employ the skills learnt in the course, particularly testing and debugging and pair programming.

   It was not all work either, there were activities every evening, which helped us get to know each other. The organizers did a pretty good job of making sure we were never bored. The course and the diverse group of people I met were amazing and the enthusiasm to keep it going could be seen from how many people volunteered to organise the course in 2016.

   One thing I would like to see next year, is more lectures and tips on data visualisation with matplotlib, as it applies to scientists in all fields and was not touched upon too much in this course.

   As someone who has always programmed in MATLAB and only been using Python for the past year, I never realised the power of open source programming in advancing science until this course. The fact that it doesn't have a course fee and is only a week long allows anyone who has time to spare and is eager to sharpen their Python skills to participate and benefit.