Advanced Scientific Programming in Python

a Summer School by the G-Node and the Faculty of Electrical Engineering, Mechanical Engineering and Naval Architecture (FESB), University of Split

September 8–13, 2014. Split, Croatia

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 04, 2014.
No answer was mandatory.
The free-text answers have not been edited and are presented in their original form, including typos.

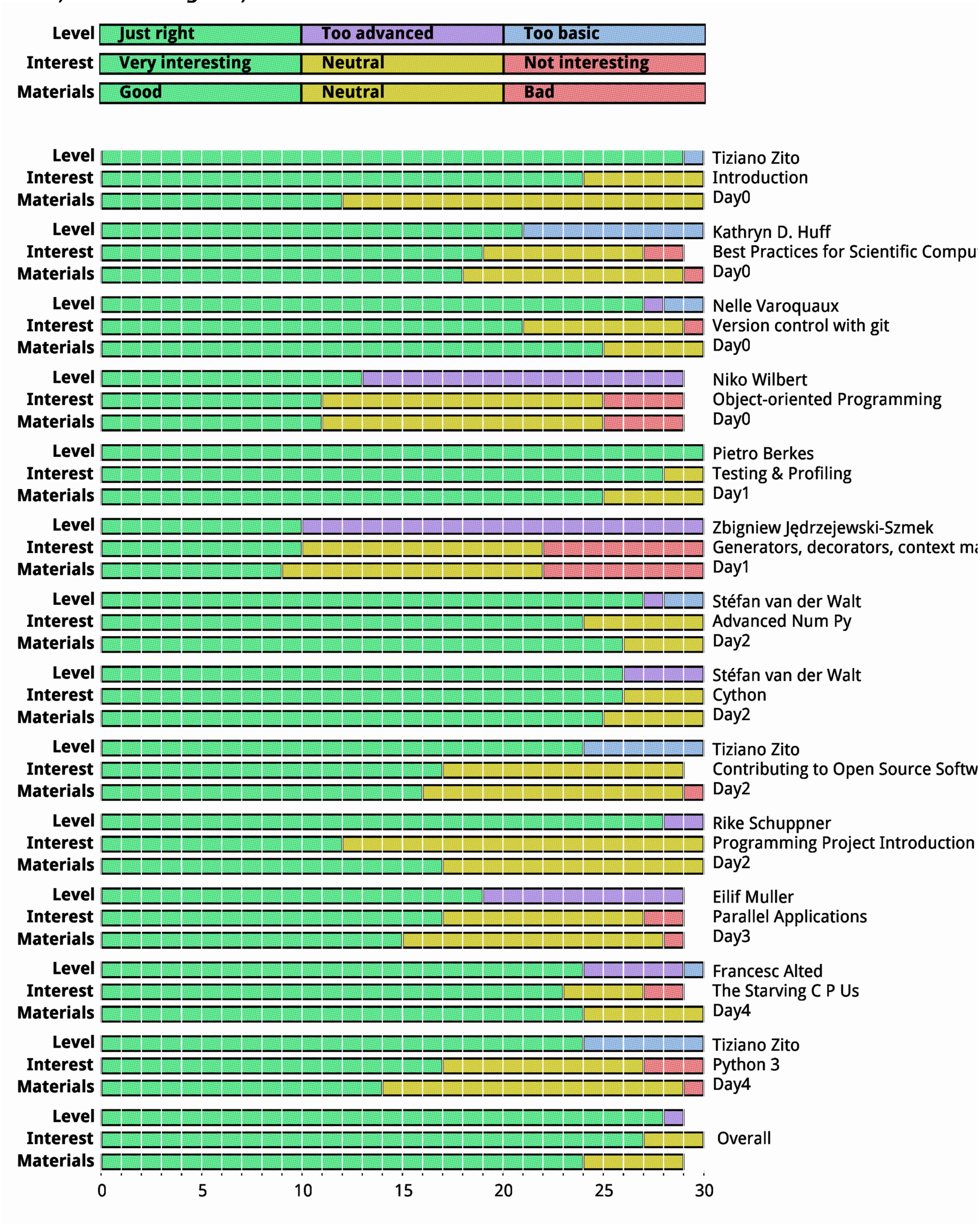
Attendants and Applicants Statistics

	Attendants		Applicants	
	30	18%	168	
Different nationalities	15		38	
States of affiliation	10		30	
Female	13	43%	36	21%
Male	17	57%	132	79%
Already applied	10	33%	24	14%
Bachelor Student	0	0%	14	8%
Master Student	5	17%	24	14%
PhD Students	17	57%	68	41%
Post-Docs	3	10%	29	17%
Professor	1	3%	4	2%
Technician	0	0%	1	1%
Employee	1	3%	12	7%
Others	3	10%	16	10%
Completed surveys	30	100%		

More stats about attendants are available at: https://python.g-node.org/python-summerschool-2014/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. Generators, etc.
2. Object oriented programming is very relevant, but the way it was exposed I found not the best one. The lecture was too dry and theoretical, an improvement will be to demonstrate the concepts with more exercises.
3. In my opinion all the topics were relevant, but but Zbigniew Jędrzejewski's and Niko Wilbert's lectures were perhaps a bit too advanced (given my level, at least) and it's maybe harder to find a direct application for them in scientific context (without previous background in computer science).
4. I think object oriented programming is not really relevant in the extend it was presented - some science-related examples could have helped to clarify how this would be useful in science . the starving cpu lecture was interesting but i think could have been covered in the advanced numpy lecture, i.e. not necessary in this extend
5. I think Niko Wilbert (particularly the design patterns and detailed OO stuff) and Zbigniew Jędrzejewski-Szmek were not immediately relevant for scientific programming.
6. All has been perfect! I don't need some stuff at the moment, however right now I know what is available out there and I know where to look up if I need it!
7. Overall, the choice was good. I think you should talk more about development time reduction and less about computational costs. For many scientists, some of the topics of the "starving CPUs" talk are too far from what they can care about.
8. For me the goal of the generators was not completely clear.
9. Object oriented programming.
10. Maybe Generators and Decorators are a little to bit advanced. I accept, however, that I am in a point where I still do not need them and that explains my views.
11. Object oriented programming was not very relevant to scientific computing - it could have been made more relevant by focusing on concrete examples of its usefuleness instead of abstract concepts.
12. The generators and decorators lecture. I don't see why it is relevant, it should be emphasized what for this objects are actually useful.

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

1. Plotting data. Organizing data (Pandas)
2. I think build systems (make, scons etc.) would be very useful to most scientists. Very handy for automating data analysis, plot generation, document generation etc.
3. Some examples of algorithms used for real data analysis which are already properly written in Python, clearly coded, efficient and so on.
4. pandas (instead of one of the optimization lectures [cython, parallel, starving])
5. Some ideas: - more git (e.g. debugging by diff's with old commits, merging in practice etc.); - more on a typical data analysis workflow (ipython notebook vs. usual scripts, how to organize plotting, how to build an efficient analysis chain etc.) Maybe Rike or Pietro can present some examples from their companies, to show which approaches work and which don't. I think this is more relevant than reducing computational time by a factor of 1.5.; - pandas/matplotlib/mayavi
6. pandas
7. It would be good to elaborate more about working in pairs or in small groups in scientific env. I think in science, it has to be that one of programmer is a leader (and gets a first authorship) but is it possible that another one is helping?
8. plotting libraries (e.g., matplotlib + seaborn and mpltools); cython debugging
9. Maybe some of the advices given Kathryn in the first lecture could have been expanded further throughout the course . I somehow thought they were going to be expanded, at least. In particular, I'd have been interested in a bit more information on API design, adequate workflows (e.g. for analysis and creation of figures; use of makefiles) and an overview on tools to create publication-ready figures.
10. GPU programming (maybe focus on a single approach to parallelism (MPI4py?) and integrate GPU)
11. Maybe a little bit of data structure and algorithms ? Is something that scientists are usually not leached and is quite handy.
12. Sure, but current lineup seems well rounded.

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

Yes, I have learned from my partner / I have helped my partner	80% (24)
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.	7% (2)
Other	13% (4)

Other:

1. I would have prefered to work work in pairs but with one computer each
2. Neutral, but it was nice to get to know people
3. as a good programmer, it would have been more helpful for me to always work with another good programmer
4. Paired programming with partners at similar levels was great; paired programming with different levels not so helpful

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	7% (2)
Lectures were too short, there should be more time for lectures	0%
The time dedicated to lectures and exercises was well balanced	80% (24)
Other	13% (4)

Other:

1. excersises in itinere were very useful, unfortunately not all the teachers adopted this method
2. Parallel lecture and exercise will be ideal
3. I would have preferred more time for exercises and less for the project
4. the timing would have been just right if the instructors respected the time limit

Q: Any further comments about the lectures and exercises?

1. I would repeat my comment about team/pair programming in science. I think there should be an extra lecture before starting work in teams. How to work in teams? How to discuss progress? How to see if you are about to be done before deadline? How should lead a team? How to communicate between pairs in a team? etc. I think the time for working on project could be reduced and a lecture (and some exercises) could be introduced. BTW, it's me. Marcin :-) I was to late to join you for first 2.5 day so please don't count me fake answers for scoring lectures that I have not been participating. My real scoring goes > Stéfan van der Walt Cython Day2
2. Well, as a personal opinion I would have preferred using the time of the project for more exercises related to the lectures.
3. I enjoyed it all !
4. Extremely nice course. However, I did not like the exercises of object oriented programming part. I realize it's very difficult to do them because of the large differences in experience. Nevertheless, if there are people with little or no experience, in my opinion the first exercise should present an intuitive, clear, well designed program instead of a program, which is non-intuitive and badly designed. Also, the explanation of how object oriented programming works was very fast. I would have given an exercise right after the basic explanation where objects are used in a common way. Then people with experience can explain something to people without experience. The second part of the lecture was very interesting. I just thought there was a bit much code on the slides and I didn't have enough time to follow that code. The part of the generators etc. could be improved in my opinion by starting with the advantages of using them. The exercises were good.
5. On structure: I think the classes worked slightly better when presented as a lecture followed by exercises. When we did exercises interspersed with commentary, as with Francesc, the structure of the class fell apart a little, because the students didn't all arrive at the "barrier" together and the lecturer had difficulty speaking clearly over ongoing conversations. On content: I enjoyed all of it. I found the iPython notebooks really useful to work through, specifically when the lecturer was trying to make a linear story for us to work through, and when it's more important to see the result than actually do the coding. I wouldn't suggest using more of the

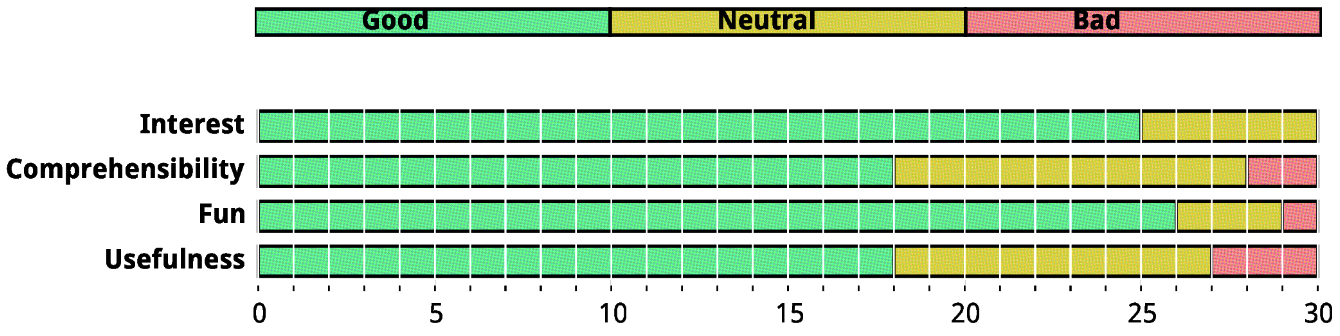
notebooks, because I found it was useful to have a blank page in front of me from time to time and be forced to think, but I think the ratio of notebooks to free-form exercises was good. An aside: I think it'd be good to strongly discourage people from holding conversations above a whisper while the lecturer is speaking. I found this very distracting and somewhat rude. It was odd that the lecturers themselves were the most frequent culprits.

6. The lecture about generators and decorators and context managers was badly organized. The topic seemed so interesting and those features seem so powerful but sadly, the lecturer didnt explain the generators that well at all and he ran out of time early so he didnt even talked about decorators and context managers.
7. This school is an ADVANCED course on scientific Python and it means it requires a basic knowledge on Python. Anyway, a brief review of the basics at the beginning of the week could have been very useful. I can suggest a introductory lecture (2 hours) with the main basic concepts which will be used during the following lectures.
8. Some lectures would integrate slides-presentation and short exercises. This format was also very useful, as it helps to consolidate the concepts just learned. For programming, learning by practice is the only thing that works!
9. The lecturers were great, and the exercises were very useful. I think the whole package of slides, exercises etc. should be one git repository instead of ~25 different files that have to be downloaded individually.
10. I think that lectures in some cases should not go into such detail and thus leave more room for exercise. Interested students can always talk to the mentor after the lecture or mentor can leave some useful links and resources for those who want to know more.
11. - git lecture: very good introduction to using git on your own, but the part of how to contribute to open source was not so helpful because it was too fast. There was a part missing explaining how to use git for a small project (like our pelita player), i.e. 1.) use git on your own, 2.) use git for a small project, 3.) contribute to big open source projects using git ; - cython/parallelisation: I was hoping for fewer and clearer recipes on how and when to use both, maybe this could be exemplified with better exercises
12. Having hands on and lecture at the same time is in my view the best method of learning
13. The group programming exercise (Pelita) was extremely valuable. The tutors must have spent a great amount of time preparing that exercise: you have my deepest gratitude.
14. I had comparatively little experience. Therefore, I was often paired with a more experienced programmer and thus watching during exercise and also the tournament. This way I learned things theoretically but could not find out where my personal difficulties are and ask the questions necessary for me. Therefore, I would suggest for the future to categorise people visibly into experienced and less experienced programmer, so that you can chose who you want to work with. I found the post it signalling for interactions with the tutor very useful. Thank you for this very enriching summer school.
15. Few suggestions and thoughts: According to my opinion, the schools should be a bit more balanced on "scientific" part and less on "programming". Also, there is a lot about the quest for speed, and scientists programmers are usually more worried about how to implement and test their ideas and they are less caring about code efficiency and memory usage. More specifically, a lecture about Scipy, or something like Theano would be maybe more useful for scientists.
16. I really liked the first few lectures where the exercises where interwoven with the lectures. That would have been helpful for all lectures.

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: 77% (23)
No: 23% (7)

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

Yes: 20% (6)
No: 80% (24)

Q: Any further comments about the programming project?

1. It is really a very good idea. Finally we are working on something for real, not just a toy example. As I mentioned before, I would have liked to tackle a real-world scientific problem for a programming project, but I must admit that the video game is way more fun and gives extra motivation, with the competition and tournament.
2. I think it would be more useful to have an overview of other scientific libraries out there or maybe use the time to do more exercises related to the lectures.
3. I did not give such a good evaluation because I felt almost useless during the programming project. I would have preferred to have a simple task for myself but we were working in pairs and I did not want to delay the process. It would be good if the winning team presents the features of their code. As well as the losing team.
4. In my opinion a real-world scientific problem would be very interesting to work with, but I definitely think working in a problem which is not directly related to science is a great way to give a taste to non-computer scientists of how working in a programming team is. In my case, all the members of the group had a level much more advanced than mine, and I felt in some cases there was no real interest from them in helping me learn (which is a pity for me because I could have learnt a lot from their experience), maybe because of the competitive feeling. This is not a criticism to the game, though, since I know than in other groups, where the level was a bit more homogeneous (and in average more basic) that problem didn't appear.
5. Documentation should be slightly upgraded (some data was wrong). Also in final tournament best of 3 for all matches should be practiced. Also I would like to see more mentors participate in the tournament.
6. I think the game was fun! But, as I said before, it might be useful to get a lecture about team-work theory :-)
7. I think it's a very useful crash-course in the pitfalls of team programming, and the competition provides motivation to work. It was great fun. My only complaint is that most of the documentation is written so that the function comments are just the name of the function expanded into a sentence:

   ```
   def get_bot():
       """Gets the bot."""
       ...
   ```

   This is really, really annoying. I know it's not common in Python documentation but I also want to know the object type returned by functions (preferably with a link to the doc page) so I know what to do with it. As it was, I had to call the function, stop with the debugger and examine the object myself.

8. I think that the project should be about creating something (small) from scratch. Even though Pelita was fun, I believe people spent most of the time on getting to know the game API and reading the documentation.
9. The project was a lot of fun!! We didn't really have the opportunity to utilise a lot of the things we've learned in the lectures, but I think if it was instead about some scientific problem a) maybe not everyone had the right background to work on it and b) I probably would have gone to the beach instead. I think the groups should be balanced better though in terms of programming experience. Personally I don't really like working in teams that much, but it really depends on the team; if in my group there would have been another good programmer, I could have work with him or her, but since this is a time critical task, in my opinion it just slows a good person down if they have to explain everything the do to a beginner. For the beginners though I think it helped a lot that they could work in pairs and figure out easier parts of the problem on their own.
10. I think it would be difficult to have a real-world scientific problem, since people come from such varied scientific backgrounds. The video game is a great exercise because there are so many possible ways to solve it. This makes it necessary to spend much of the time discussing what to implement, not just how to implement it. This factor makes it much more valuable as a group exercise. Not to mention that the tournament was fun!
11. I do not remember the last time I had so much fun doing programming work!
12. While the project mightn't have been as useful as other exercises given the amount of time spent on it, it was still fun and a nice part of the summer school. I was on a fairly inexperienced team and there were times when we could have been much more efficient with outside help, but we never really got in the habit of asking for it. Maybe tutors could be attached to 1-2 teams each and discuss documentation/implementation, and help out some of the less experienced people in the team.
13. I liked the project, but I think it takes up too much time. I would rather have more time for the exercises following the lectures than doing the project for 2 days. But I think it is a valuable experience to write code in a group.
14. One of the interesting parts of the projects was the coordination as a team, that was surely helpful. It was also well-suited to do agile programming. I also liked that it was so easy to get something running - the learning curve seemed to be good. I was looking forward to do some test-driven development in the project. However, as always, the time pressure destroyed that plan very quickly.
15. i think it was excellent to learn that, apart from programming, there is much more to worry about when working on a software project in a team with a hard deadline (organizing, distribution of work, making the most out of the individual strengths, planning the time course until the deadline, NOT creating conflicts just before the deadline ;))
16. It was a lot of fun, but maybe a little more help for getting started could have been useful, we were quite confused at the beginning. Also, I really would have liked to hear the lecture about programming in teams.

The School in General

Q: How do you overall evaluate the school?

Good: 97% (29)
Neutral: 3% (1)
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: 3% (1)
Just Right: 93% (28)
Too basic: 3% (1)

***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: 7% (2)
Just Right: 93% (28)
Too basic: 0% (0)

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

Yes: 97% (29)
No: 3% (1)

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

Good: 97% (29)
Neutral: 3% (1)
Bad: 0% (0)

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

Yes: 100% (30)
No: 0% (0)

Q: How did you hear about the school?

Google Search: 3
Professor/Tutor/Supervisor: 5
Colleague/Friend: 13
Website/Mailing list: 11
  of which:
    scipy: 2
    connectionists/comp-neuro: 5
    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: 87% (26)
No: 13% (4)

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

OK: 81% (21)
Too high: 15% (4)
May be higher!: 4% (1)

Q: Any further comments or suggestions?

1. I think a fee would definitely be appropriate, but maybe you can make this sort of on a voluntary basis, i.e. if the attendee is a phd student that gets the trip payed for by his or her professor anyways, a fee is ok, but for students that pay for the summer school themselves, the fee is dropped?
2. The social part of the school was amazing. Thanks again to the local team, you were the best. In some of the topics, I had the feeling that the level was a tiny bit too low. List: - numpy: could contain more advanced topics, could cover parts of scipy (but exercises were fun, especially the 50 extras); - git: could speak about more features; - multiprocessing: could include exercises where users actually work with Values, Queues, Processes and so on, instead of just running given code. But this is just my personal opinion!
3. I would have still attended because I am in a well-funded lab. However, I know of some students who probably would not have had the chance to attend with a fee. Maybe introduce scholarships for students without sufficient financial support?
4. Yes, I think that a fee of 200 would be OK if a fee waiver for students that are paying themselves (like I did) is included.
5. Overwhelming majority of the crowd were from natural sciences background. I was the only economist among them. And I think it is a pity. Economists write as much code as anybody else in science. But at the same time the culture of writing, sharing, maintaining, etc. of the software side of any research project is quite low. Not many people know much of version control, of free programming languages. Which is even more surprising since economics should be about efficient allocation of limited resources! My suggestion is that you can easily attract even more people (if you need that, of course) by wider advertising. Many more people whould benefit besides neurobiologists.
6. BTW, it's me. Marcin :-) I was to late to join you for first 2.5 day so please don't count me fake answers for scoring lectures that I have not been participating. My real scoring goes > Stéfan van der Walt Cython Day2
7. Thank you for organizing the school! It was a great week and I learned a lot from it, that I hope to use in upcoming projects.
8. The whole experience was absolutely fantastic. I felt guilty at the end because I had such a good time in both the social activities AND the lectures that I felt like I'd been on a paid holiday. And I think that the course will make a material improvement in my science and that of my colleagues. I can't praise the organisers highly enough.
9. I think 50 to 100EUR would be appropriate.
10. i think a longer lunch break would have been nice (about 2.5h), at the expense of working later in the evenings. it would have been great to have access to the laptops also in the evening after the official part was over, to finish excercises and talk with other people about the stuff that was done during the day. i was impressed by the quality of the lectures and the faculty in general, especially in making the links to our everyday life in science obvious and giving the right hints on how to improve my workflow
11. I put 200€ as too high since I came to the summer school on my own, without being affiliated to any research group yet which could sponsor my participation. Obviously, I'm aware I'm the exception and not the norm, and probably 200€ is not too high of a fee if the home university is paying for it. In my particular case, though, I would have not been able to attend the school.
12. Thank you very much, the school was great!
13. This was really a great course where I learned a lot and had a lot of fun.
14. Thank you, again. free time and work time were very well balanced. I learned so much and my programming abilities have made a leap after this school.