Advanced Scientific Programming in Python

an Autumn School by the G-Node, the Center for
Mind/Brain Sciences and the Fondazione Bruno Kessler
Trento, Italy, October 4 → October 8, 2010

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 within October 20, 2010.
Values have been computed as average of the available responses. No answer was mandatory.
The free-text field replies have not been edited and are presented in their original form, including typos.

Attendants and Applicants Statistics

	Attendants		Applicants	
	30		80	
Different nationalities	19		28	
States of affiliation	20		20	
Female	8	27%	11	14%
Undergraduates	2	7%	12	15%
PhD Students	18	60%	40	50%
Post-Docs	3	10%	15	19%
Professors	1	3%	1	1%
Others	6	20%	12	15%
Completed surveys	30	100%		

Lectures & Exercises

Q: Please grade the lectures and all lecturers according to the provided criteria.
[Grades: "Bad"=−1, "Neutral"=0, "Good"=+1]
The criteria are defined as follows:
Interest: How interesting was the subject of these lectures?
Comprehensibility: How clear and comprehensible was the material presented? Was it easy to follow the lecturer? Did he take time to answer students' questions?
Material&Exercises: Evaluate the quality of the teaching material provided by the lecturer, e.g. the clarity of the slides, references given, scripts, exercises etc.

	Interest	Comprehensibility	Material&Exercises	
Overall	1.00	0.97	1.00	0.99
Zbigniew Jędrzejewski-Szmek Advanced Python	0.62	0.07	0.72	0.47
Bartosz Teleńczuk OO Programming	0.76	0.93	0.79	0.83
Valentin Haenel Best Practices	0.90	0.97	0.72	0.86
Emanuele Olivetti Git	0.83	0.90	1.00	0.91
Pietro Berkes Software Carpentry	1.00	0.93	0.81	0.91
Tiziano Zito Programming in Teams	0.90	0.93	0.57	0.80
Stéfan van der Walt Advanced NumPy	1.00	0.97	0.90	0.96
Bartosz Teleńczuk Data Visualization	0.68	0.47	0.16	0.44
Rike Schuppner Programming Project Intro	0.73	0.23	0.41	0.46
Eilif Muller Parallel Applications	0.70	0.33	0.23	0.42
Francesc Alted The Starving CPUs	0.86	0.86	0.82	0.85
Francesc Alted Data Serialization	0.59	0.76	0.67	0.67

Note: values in gray are computed as an average over the grades in all criteria.

Q: Are in your opinion some of the topics presented in the lectures not relevant for a programming scientist? Leave blank if you think all topics were relevant.

1. Maybe a little less time explaining "Advanced Python" techniques, as the most useful one (exceptions) should be a BASIC topic. Decorators are nice stuff, but lots of people are not going to use them... Understanding context managers is nice too, but I doubt most of the people attending the course is going to program their own. Most of these topics fit better in a generic "Advanced Python", instead of a "Advanced Programming for Scientists", where one would expect a focus on more practical aspects (from the point of view of a scientist!)Also, take into account that going too deep into certain subjects during this general talks (like the second half of Eilif's on GPUs) may be boring or directly uninteresting to part of the audience. Going into detail about low-level stuff is something that has to be carefully balanced. Compare Francesc's "Starving CPUs" talk: he showed the needed detail to understand why this matters and then went on to common solutions to the problems and how certain tools help you overcome the problems, etc. In the case of Eilif's talk, you could easily cut 15-20 minutes, and it would be perfectly valid, which helps staying on schedule, too. The more detailed stuff can be discussed over some coffee/beers, or presented as extra material (handouts, links, etc).
2. I thought the topics were overall very good. I would have liked a little more on IDEs for Python, on debugging and on profiling. I would have liked to attend the MatPlotLib lecture, but I also wanted to attend the Cython lecture, so running those lectures in parallel did not suit me.
3. I think the first lecture was a little bit hard and unnecessary for programming scientists. Maybe it was because it was the first day and came like a shock to everybody not prepared.
4. The lecture on parallel programming was not relevant for my research but I can very well imagine a setting where this is interesting. However, if in the future I will ever need to program parallely, I can always refer to the slides, which is great.

Q: Are there further topics relevant to the programming scientist that could have been presented?

1. Presenting extra stuff would maybe need a focus on specific fields and the solutions they need, which is not practical. I think refining the current topics is the way to go. Maybe you could take some of the more advanced stuff, if it's perceived as something that will be rarely ever used.
2. A lecture concerning symbolic calculus (sympy for ex) should have been great and/or something for advanced statistical analysis.
3. GUI construction, so perhaps not as relevant.
4. track
5. Social stuff: how to convince others to use what is a known good practice. Activism: how to get this to the journals
6. More on data visualizationHow to efficiently store data How to efficiently make databases of research data
7. Scientific analysis of real world data!
8. What IDE exists how to select one or more and which for what use. This could have been a part of best practice tools.String manipulation and text parsing is a task that is needed in some scientific domains. This topic was not touched at all in the course.Apart to numpy, numexpr, matplotlib ... there are other extensions that exists like SciKits collection that may be interesting not only for domain specific tutorials.
9. Python bindings to other common scientific programming languages, other than C; how to make best use of existing code - most lectures showed how to write totally new code and it's often not the case
10. Maybe some short talk about editor recommendations, the different python environments perhaps.
11. There could have been use cases of how all this fits together. There must be numerous case studies that could be presented showing how all this helped a lab produce experiments and papers. How MDP was built would be interesting.
12. Would be nice to see more numpy examples and scientific calculations, but it's too complicated too cover all within a week
13. It's not possible to cover all topics but it could be useful to have a parallel session where every tutor show how they use Python in their research field. So there can be a session on Machine Learning, Image Processing, etc.
14. I would have liked more on visual presentation of data, especially animation, which I find useful in determining of my code (e.g. simulations) is working as I expect it to work.

Q: What do you think of the level of the "Advanced Python" lecture?

It was OK	47%
It should be extended with more time for exercises	20%
It should become an introduction to Python and advanced topics should be skipped	3%
It should be skipped completely and its time slot should be used for the other lectures	0%
Other	30%

Other:

1. it should be after oop and best practises
2. The topic was interesting and shall be kept. It was just a bit of abrupt dive into the subject, especially for newbies.
3. It was a tough start to the course, but I have already used some of what I learned there in my own programming.
4. This material is interesting but the presentation was no so great. I want to know what exactly you expect us to use it. It's not helpful to be told in abstract terms all the things it could be useful for. What do you expect us to use if for?
5. See above, on my opinion about the topics.
6. it was great
7. It was advance! but it was in an advanced course. I didn't understand a thing, but that doesn't mean that I've not learnt anything. I maybe would warn the people in the intro that what we were going to see was very advanced.
8. It gone to deep into some aspects that has been hard to understand for many. Givent that many concepts have not been used again in the course, i would have devoted more time to try to bring all the student to the same python knowledge level.

9. It was a jump in the cold water

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

Yes, I have learned from my partner / I have helped my partner	83%
No, it was a waste of time for both me and my partner	0%
Neutral. It was OK, but I could have worked by myself as well.	10%
Other	7%

Other:

1. It was extremely useful, especially switching pairs that allowed to test a lot of situations.
2. it would be much better if we were divided on pairsbased on our experience in python; it happend that 2 people with no experience were working together and then programming in pairs fail

Q: What do you think of the balance between lectures and exercises? Note that the exercise sheets were not meant to be finished in a single day: they were meant to offer a diverse set of tasks, among which every student may choose the ones they prefer. When answering, please keep in mind that the overall time is limited ;-)

Lectures were too long, there should be more time for exercises	17%
Lectures were too short, there should be more time for lectures	0%
The time dedicated to lectures and exercises was correct	73%
Other	10%

Other:

1. The time dedicated to lectures was correct, there should be more time for exercises. But I don't keep in mind that the overall time is limited ;-)
2. It would have been nice to have more time for exercises.
3. I would have liked a little more time for exercises, but overall I thought the balance was good

Q: Any further comments on lectures and exercises?

1. They were overall of high quality and well harmonised, so that there were no duplicated topics. I appreciated that the lecturers were able to take account of students' feedback and refine the presentations on the fly.I would have liked more discussion on programming in teams, because I find it crucial in programming, and even more in scientific world that is often fragmented. I know it highly depends on the students' pool, but maybe a less formal disposition than the classroom could have helped shy people to express themselves. It's just a guess, I'm not sure myself on how to improve the discussion.I appreciated the importance given to best practices. It can be useful to add a small exercise session that asks to improve code badly written, because it helps spotting uglinesses in existing code (I think of Martin Fowler's "Refactoring" book [0], a collection of bad examples and their improvement).[0] Fowler, Beck, Brant, Opdyke, Roberts, 1999. "Refactoring. Improving the design of existing code" (ISBN-10: 0201485672; ISBN-13: 978-0201485677)
2. Perhaps make it clear on the first day of the course--or, even better, before the course starts--that it will be intense, that there will be many exercises, and that we will not be able to complete them all. I think it was good that we got exposed to so much material. I feel like I will still be learning from it for weeks to come by going through the slides, exercises and my own notes again in my own time.You all did an amazing job and I'm really happy that I attended the course.
3. great wiki and materials :-)
4. It was impossible to work on exercises after the pacman project started, time was limited and the brain occupied.
5. Maybe you can rearrange the “Advanced Python” and “Object-oriented programming” lectures and start the Day0 with "Object-oriented programming" because it's like "an introduction to advanced programming" which help people understand the main concepts and “Advanced Python” would be like natural extension of this “introduction”.The idea to have two lectures (parallel applications& optimization problem) and then split them into two parallel sections of exercises was good. It would be better don't split the practice and lecture in two parallel sections, which we had (Cython exercises and "Data Visualisation" lecture). I missed the exercises for

Data Visualisation as well. Otherwise the school was excellent! I enjoy it very much! Many thanks!

6. maybe it's better to swap "Advanced Python" and "Object-oriented Programming" lectures (the "Object-oriented Programming" must go first). Some short introduction to the course in the morning of day0 would be also nice.
7. There was to little time for exercises (though I understand, that lots of topics were covered and the schedule was pretty tight).I really loved taking part at the school!
8. The topics were selected well. Everybody was supposed to know basic Python, so I think it was a good idea to start with advanced Python right away. Also, it had "advanced" in the title, so nobody should expect a beginner's Python tutorial.
9. Maybe tutor hours?
10. People which have Python plus the relevant modules working on personal computers should be "allowed" to use them for the exercises and project, without having to deal with the virtual machine all the time. Maybe one could send out a list of required modules and such for people interested in using their own equipment. Just a thought though.
11. I really like most of the exercises, thought the one from Parallel programming we didn't manage to make it worked.The pair-programming thing was a great idea, but even better the being moving. It reminded me to my dance classes where you are in couples and change all the time. So you know people, you learn from them and you don't get unlucky of being all the time with someone that does all the job and doesn't help you much.The exercises looked great, and giving more that we can do incentive the people, or at least me, to keep trying the things ones the course has finished. In this aspect I would like to suggest to keep the mail-list active for a while, so we could ask open questions to everyone.
12. I really liked the organization of the course and the exercises after the lectures. I would suggest, though, that the first day could include a general overview about Python, rather than starting right away with the advanced topics. This would have been particularly useful for those of us who were less experienced.
13. This is an very interesting curs for scientists that have yet a programming and python background. I learned a lot and liked the setup of the curs. The ambiance of the curs was also very encouraging as most of the attendees had an "open-source" approach in there communication and interaction which is overall good base for interaction.I thank every body how helped to build up this school and encourage you to proceed in that way.
14. I think too many talkers used the exercises as examples during their talks. Repetition is OK as a learning method but, then again, you shouldn't expect the student to give a good thought to a "what do you expect as the result to this command" when you've just seen it verbatim 10 minutes before...
15. I really liked to style where lecturer demonstrated the stuff he was talking about by simple examples in ipython.
16. The school was very well organized and the selection of topics was very good.I especially liked the lectures on OOP, version control (and the introduction to git), unit testing/debugging, programming in teams (an eye opener!), advanced NumPy (and I thought I knew NumPy), Cython, the use of GPUs and parallelization.In places, I might have appreciated more of your opinions about things, even if not everyone agrees. For example, epydoc or sphinx, do you prefer one or the other? What are their relative merits? Your opinions are useful for students to hear, even if you all don't agree.The pacman project was very useful. It was an excellent testing ground for materials we covered in the course. If there was a problem with the teams, it had to do with the vast spread in programming experience amongst the team members. With this in mind, I think it would be helpful to give somewhat more explicit directions on how to proceed. What I would tell the groups is: first, run all the various supplied pacmen against each other several times to get a good feeling of how the game works, what the mazes look like, and most importantly so that you can decide on which of the supplied pacmen to use as the basis for your pacman. Second, analyze the complete class hierarchy of the pacman you choose making sure you understand all the functions (i.e. what they do). Do these first two things as a TEAM BEFORE any programming starts so that everyone understands what the already-written code does (it's ok not to do any programming the first day of the project!). Then, work out a strategy and start programming. In our group, we started programming too soon. Ultimately, it wasted time and left some of the team members a bit behind.Finally, I very much appreciated the helpful and cooperative spirit of the instructors. You managed to create a very convivial atmosphere that really facilitated learning. Bravo! And thanks.

Programming Project

Q: Please evaluate the programming project.

[Grades: "Bad"=−1, "Neutral"=0, "Good"=+1]
The criteria are defined as follows:
Interest: How interesting was the programming project?
Comprehensibility: How clear and comprehensible was the code presented? 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?

Interest	Comprehensibility	Fun	Usefulness	
0.90	0.60	0.83	0.70	0.76

Note: values in gray are computed as an average over the grades in all criteria.

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

Yes: 73%
No: 27%

Q: Do you think the inter-group competition inspired by the tournament had a detrimental influence on your experience at the school?

Yes: 10%
No: 90%

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

Yes: 14%
No: 86%

Q: Any further comments on the programming project?

1. I think that the choice of the programming project is very appropriate. It is a very nice way to present the technical and social complications of team programming and to prove the effectiveness of the best practices presented at the school. The competition inspired by the tournament introduces enough pressure to make the think more interesting. The fact that the project involves a video game makes it more fun and makes the different area of expertise of the students less relevant.
2. Maybe it too small amount of time dedicated to it. Maybe somehow more time should be asigned I guess that with just 1 extra afternoon it would be perfect. Maybe telling the groups before somehow or something similar.
3. I would have appreciated sessions at the end of days 2 and 3, with tutors and other groups, where we all tried to apply the material introduced in the lectures into the code that we had written on that day (e.g. writing decorators, generators into the code; using the debugger, profiling, unit testing, and Cython, also KIS and DRY). We used git a lot in our team, and this really helped to reinforce what I had learned about git in the exercises and lectures. In my team, instead of using the new tools covered in the lectures, a few of us spent most time just trying to read logical structures in codes of other team members. It would have been good to have an allocated hour or more where we all practiced using the new tools introduced in the lectures.
4. The setup with the ssh-key buried in the VM was a bit of a nuisance, until someone figured out a way to make it work on the host computers.
5. People could spend time explaining to others the way they had to break down tasks and the structure of their group. There wasn't too much pressure and only maybe 30 seconds of groups saying how their group worked. Students learn about working as a team implementing their project, but they would also learn from how other teams explaining their group dynamics.
6. The competition was GOOD! It was good-natured and really brought everyone together in nice way.
7. I think, in the situation where one defending agent is standing on the food it should be possible for the attacker to eat the food before dying, especially if the defender is not moving at all.
8. The inter-group competition was not detrimental at all, quite on the contrary. The common goal of knocking up something that'll work in the match made us get to know each other well, having dinner together and

doing late hours and all that. When I look back and think of which people I got into touch with most, it was mostly the people from the Pacman project. I think Pacman as a topic is perfect, because it's a game, which means that it's usually fun, and because it's a game, there's a competitive theme to it, which I could not imagine in a scientific problem, or it would be much harder to compare. If we do a project and don't compare things at the end, also there would be less pressure to do a good job. Bottom line: perfect idea! Can we have the pacman group shot?

9. The competition among groups has been an overall positive incentive, but not on the code quality. As the deadline came closer, we wrote less and less Pythonic code. To improve that, the pylint prize could have a minimum score threshold, as all of us had pretty low scoresI was in a good team, we were able to communicate a lot. In the beginning the most expert programmers worked together, leaving less experienced fighting with the code, and this was sub-optimal. Later we switched pairs and stayed on the experienced/newbie pair for all the time, switching partners often. This allowed to revise code and bring new ideas. The game has been fun, for sure; I would have also enjoyed a more scientific topic, so that we could reuse what we learned in the week.
10. I was not very enthusiastic at the beginning regarding the programming project, but in the end it was fun and very instructive: it was a good synthesis of what we learned during the school!
11. Regarding last question.... it's a bit tricky. I really loved the game, but maybe a scientific problem would be more helpful... but how engaging would it have been? the competition keeps everyone helping because everyone want to win! Probably if we would have more time we could do some excercise in any other scientific fields... but they won't be so interested to everyone and so much fun ;0)
12. Having a non-science and hence subject-neutral project is good.
13. Horrible tk dependency forced me to work on the usb OS and leave behind my beloved environment :)My distro doesn't package tkinter and won't (huge X dependency for python, which they have to package anyway)
14. It was a very nice and fun project and was good to work in teams. I think it should continue.
15. It was great.
16. In the beginning I was skeptical to make the video game project because I expected to make a project relevant to the real scientific problem. But my participation in the PacMan project completely changed my mind. The video game project is very good idea because you have a problem (create your game) then you have to do a real brain storm in your team how to solve this problem (create an algorithm, strategy, apply your skills&knowledges, test etc). It's like in a real scientific world. Moreover it's fun :). It seems this is true: before to start to solve a real serious scientific problem you should play.
17. 1. Real-world scientific problem would be very boring. 2. People may use a lot of scientific approaches to solve the Pac-man guidance task, and try to implement them in Python.
18. It would be a grate thing if the project could be accessible for others to learn and improve the agents. This is what open-source and in general also science is about. Even if I understand the reason for not opening the code from the source teacher point of view, it would be nice if for the python curs the tools and framework could be available. This could then also be used in later (local or other) courses.In general the use of git for managing the project code was a got idea but in my opinion the best way of using such a tool in a group collaboration should have been practised before in order to profit from it in the best way.
19. While I guess some people would appreciate a programming project related to a real-world scientific problem, I see a few problems with that, and you probably agree: 1) you'd need a problem easily understood by EVERYBODY attending the school 2) wouldn't be as FUN 3) the competition as it is puts extra pressure on the participants to use the team-programming techniques, and to value it from the point of view of someone that needs to deliver code NOW, while sharing it with othersOn the other hand, a more scientific-oriented problem would lean more on the second half of the talks (the more numerical-oriented)... but I guess you can't have everything :), and the first half is really what most scientists aren't USED to.
20. It actually showed that machine learning is not the ultimate answer (group ballerinas) (actual scientific value). It is difficult to solve a real scientific project in 16 hours and by writing the agent, one could really learn a lot, not only group programming (which is the most important lesson for me).
21. In our group we didn't do that much actual programming by everyone. We sort of discussed about possibilities and everyone tried various things out. However, the actual implementation was in the end written mainly by two/three members. We were not able to distribute the workload evenly during the project.

The Winter School in General

Q: How do you overall evaluate the school?

Good: 100%
Neutral: 0%
Bad: 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: 7%
Right: 83%
Too basic: 10%

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: 3%
Right: 97%
Too basic: 0%

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

Yes: 100%
No: 0%

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

Good: 100%
Neutral: 0%
Bad: 0%

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

Yes: 100%
No: 0%

Q: How did you hear about the school?

Google Search: 2
Professor/Tutor/Supervisor: 7
Colleague/Friend: 18
Website/Mailing list: 10
of which:
connectionists/comp-neuro/ML-news: 5
python: 3
linux: 2

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: 86%
No: 14%

Q: If yes, do you think a fee of about 150 € would be appropriate?

OK: 78%
Too high: 13%
May be higher!: 9%

Q: Any further comments or suggestions?

1. I really learnt a lot during this autumn school - it would have taken me ages to learn from scratch on my own the topics covered in the lectures. I would then like to thank the organization for the course (and the

possibility of me taking part in it) and all the tutors for the great learning and social environment they managed to create.

2. Overall the school was really helpful, although some lectures were too advanced for me to follow. I think this was because of my poor knowledge of operating systems at lower-level and linux commands. I have been inspired by the school to learn much more about this!
3. Thank you guys very much!!! I learnt a lot, improved my skills, met people from different countries and fields, and I had fun :)
4. The course suited very well for me. The inspiration I got and ideas about future directions as well as the practical advice how to make my code better is something that is not easily available just on book and newsgroups. The practical arrangements of the course were very good. Overall, a great course!
5. I was wondering how such an awesome week which is totally beneficial for the students can be free. I am very thankful to the organizers and tutors for doing such a great job, taking holiday from work and spreading knowledge for the sake of it. This course is much better than books or websites, and it is much better than conferences due to the well-balanced mix of lectures, execises, and the programming project.
6. 150 euro is a high fee if the scientist is not funded by his institution. He also has to pay for the lodging and other expenses. If the scientist is funded, then the fee is OK.
7. I hope you are able to keep it going, maybe passing it on to others as the instructors move on.
8. I think I would have been able to get funding from my department for a 150 EUR workshop fee--seeing as they were already willing to pay for travel and acommodation (which came to about 350 EUR).
9. Good job!There's zero elitism, even though the level was very high.Maybe this could be captured in videos next time, for posterity.Also, expanding this so more people can enjoy it would be great, for example by making video series, like www.khanacademy.org.You could ask for donations to keep the teachers going.The more people in science get this course, the better for all of us scientist (who have to reuse code) and humanity (who has to believe the results and use the applications).It could be made into an EU grant to teach programming to scientists, and 'fix' the current status quo. Let me know if you need help for this.
10. You already had some parallel lectures this year. I think it is a good idea to expand.
11. Q: Did you learn more from attending the school than you would have learned from reading books and online tutorials alone?A: I learnt Python language alone, reading others' code, online tutorials, accepting corrections from more experienced developers. This school has given a higher level information: a wide overview of features and techniques I wouldn't have found myself, and that I will study deeper.Q: How do you evaluate social interactions and social activities at the school?A: They have been well planned. A further improvement can be the suggestion for a common accomodation.Q: Do you think a fee of about 150 € would be appropriate?A: Yes - can be also more, as the quality is high.
12. A note on the *price*... I would definitive have paid 150 € for the course, however I'm a post-doc and found a very cheap ticket to go there. The way it's now it's perfect because the people that it's there is not because they paid, it is because the have been selected over a group, and I'm sure make them being more attentive. I would try to keep offering on the same way. Let me know if you need collect comments or signatures to any funding body so they believe it's useful... from me I will advertise the course to everyone I can.Other comment, I think it would be helpful if the admitted people is announced earlier, so the flight tickets are cheaper. ;-)Finally: Well done guys!!!!:
13. I haven't answered the "Did you learn more from attending the school [...]" question because it's a bit difficult.On one hand, as a computer engineer, I'm *used* to learn by myself, and could probably have learnt most of the things in the course just by reading material on the Internet, but a course like this typically assumes a good deal of knowledge from the attendees and focus on more advanced topics and are based on the talker's experience, while getting the same results on your own would need browing a few hundred pages on a generalistic book, or scavenging the web for some advanced tutorial.On the other hand, as I wrote in the application, even as a experienced programmer I feel the need every now and then to get a "refresher", as learning just by oneself can lead to bad practices or just to overlook some material that could be interesting.And there's the beers, of course :)