Advanced Scientific Programming in Python

A Winter School by the G-Node and University of Warsaw
Warsaw, Poland, February 8 → February 12, 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 March 1st, 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	
	34		106	
Different nationalities	18		26	
States of affiliation	14		17	
Female	6	18%	16	15%
Undergraduates	1	3%	12	11%
PhD Students	18	53%	71	67%
Post-Docs	10	29%	16	15%
Professors	2	6%	3	3%
Others	3	9%	4	4%
Python crash course	28	82%		
Completed surveys	31	91%		

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	0.97	1.00	0.73	0.90
Zbigniew Jędrzejewski-Szmek	0.64	0.44	0.64	0.57
Valentin Haenel	0.87	0.94	0.81	0.87
Rike Schuppner	0.77	0.60	0.70	0.69
Niko Wilbert	1.00	0.87	0.67	0.84
Bartosz Teleńczuk	0.81	0.84	0.48	0.71
Eilif Muller	0.84	0.71	0.47	0.67
Francesc Alted	0.77	0.87	0.42	0.69
Tiziano Zito	0.74	0.77	0.21	0.57

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?

1. Some advanced stuff in the introduction was just too much for the beginner's first day (e.g. decorators,special exceptions) The introduction shouldn't introduce more than absolutely indispensable for the rest of the course
2. the starving CPUs problem
3. intro to subversion not necessary in my view

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

1. Regular experssions and parsing large amounts of data
2. Some brief mention of more tools (IDEs/editors, third-party libraries, web resources). I see that there's some of that on the wiki, but a bit more could have been presented in the lectures
3. Team Working and software development, which was already in the schedule
4. a little more on 'good' graphics with python
5. The lectures covered most of significant topics. Maybe the numpy/scipy part could be extended little bit
6. It would have been interesting to use some software written in python with specific application. Pacman is a basic example, but I guess there are a plenty others more sophisticated with possible scientific interest. Well may there was not enough time for this
7. Maybe some best practice about organizing programming in a team. That was covered in a kind of learning-by-doing style for the project and worked well for my group, but same other groups seemed to have issues. Each group could be given a tutor who could help in the process of how to organize group programming work, maybe?
8. More about databases
9. Techniques to include existing codes in other languages like C or fortran into python programs

10. 3D plotting. Scipy overview. Parallel Python library for trivial parallelization (vs. more involved multiprocessing and mpi modules). Interface to C/Fortran codes tutorial. Specific speed optimization modules as Cython, weave, etc
11. i would have liked more depth on the numpy etc, material (day 2) and more interesting exercises on this material too. in particular, python best practice for the working scientist to integrate data, analysis, and display
12. Software development methodologies were explained only very briefly. They should be explained in greater detail to be more useful. Common tools like bug repository software and concepts like release cycles and release planning can be covered
13. a better coverage of SciPy; as most of us where from neuroscience or machine learning, maybe something specific on libraries useful there
14. The course was rather consistent but if I were to extend it, I would add some mention about usable R interface
15. More about data manipulation and advanced numpy/scipy features
16. There could be more about writing faster code
17. highly parallel numerical calculations, web graphical programming, web servers

Q: If you attended the Day0 introduction to Python lecture, do you think it should be changed?

It was OK	45%
It should be extended with more time for exercises	20%
It should become an introduction to "advanced" techniques only (e.g. exception handling, decorators, metaclasses, etc)	30%
It should be skipped completely and its time slot should be used for the other lectures	0%
Other	5%

Other:
1. Includes more basic topics from other days, like OO

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

Yes, I have learned from my partner / I have helped my partner	61%
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.	26%
Other	13%

Other:
1. yes but only if the level of my partner was not too different from mine
2. This was VERY useful for both sides!
3. I think it is a good exercise, but more active input from the instructors could've been valuable
4. I really liked this approach. I think it was fun and really facilitated learining by itself but I cannot say I learned much from my partners

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	6%
Lectures were too short, there should be more time for lectures	10%
The time dedicated to lectures and exercises was correct	81%
Other	3%

Other:

1. I think the time was fine, but I think a more 'guided' excercise session could be valuable. Example could be taken from how Francesc led his excercise session.

Q: Any further comments on lectures and exercises?

1. The only problem was catering: - the place where the whole group went the first day (following one of the organizers) didn't serve good food
2. I don't think I've learned so much in such a short time in quite a while! The exercises were really well designed and helped me understand the lectures better. The speakers really tried to make us understand and I really liked that there was always time for questions! (I really got the impression that people felt it was ok to ask questions even if we were running late which was fab!)
3. 1,5 h for a lecture is quite a long time to concentrate, especially when having to understand code, even though the lecturers did a great job at keeping things interesting. Maybe it would be a good idea to have a short break every 45 min. Also, the time for most exercises was too short, which frequently resulted in us omitting all the essential techniques we had learnt during the past days, such as unit testing. Although this might be impossible to realize in 4 days, it would have also been nice to have more time for the team project. Maybe the 4-day workshop could be turned into a 2-week school?
4. The advanced lectures (Eilif's Muller and Francesc Alted) session could be more extensive.
5. For me personally it would have been nice to have more of the "programming patterns" and less of the basic OOP stuff (wasn't this presented on day 0 already ?). Some of the exercises took a rather long time to "set up" until one got to the point that was meant to be learnt (e.g. path algorithm, virtual memory usage with PyTables). It is a good thing to have to think independently about the exercises, but it would be also nice to actually train the things that had just been presented, with the tutors around. The SVN exercises were a good example of being exactly to the point of what had been presented. Pair programming was not always as efficient as working alone would have been, but I still think it was a good idea to train this in preparation for larger projects.
6. Altogether, participants with little or no prior experience with Python have a somewhat hard time during the exercises, although they will still benefit tremendously. If possible it would be nice to have more tutoring available for the exercise sessions.
7. I found the balance between lecture and exercises to be very good. I know it is hard to estimate the time for a talk, given that the lecturer wants to answer questions of the audience (which is good!). However, to grant the exercises the necessary time needed, it would be good to have a slightly better time management and cut the talk off, if necessary (or discuss with participants whether they prefer to get the complete talk or would rather work on the exercises already).
8. As is the nature with 'surveys' it's easier to be critical than complementary. However, please take my criticism as honest 'critique'. Overall, I was highly impressed with the instructors, material, and the course in general. One thing I would have found helpful is for the exercises to be broken down into smaller... well, 'units' ;) I think they could've been more 'guided', where we conducted a small task, with an instructor working at the front of the class with a prompt for us to follow along on. This could've been good for a few introductory portions, then once people were more comfortable with the concepts of the exercise, we could've been left more alone.
9. Perhaps the exercises should be more clearly marked/designed as independent (within a session of course).
10. I think next school should be intended to programmers who know how to program well, but not in Python / numpy and such. I wished the more advanced topics which were covered very partially would be covered more deeply (gpu for example)
11. i think too much material was covered, and as a result not enough time for exercises. (i.e lectures weren't too long per topic, there was just too many topics)
12. I think the course was very nice and comprehensive. Of course always there could be something more to say, but I understand that the time was limited. Thank you!

13. Overall, lectures and exercises were mostly well prepared, interesting and informative. Thank you.
14. would be nice to see elegant solutions to the execises explained/presented at the end of the day.
15. The part on numpy and matplotlib could have been more extensive. The schedule should have been kept tight.

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	
0.61	0.65	0.87	0.42	0.64

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: 74%
No: 26%

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

Yes: 17%
No: 83%

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

Yes: 24%
No: 76%

Q: Any further comments on the programming project?

1. Video games *are* real-world scientific problems, at least in my field ;) The provided game code was the weak point. I don't buy into the "it's getting you used to working with other people's messy code"-excuse. Hoping Valentin will keep the motivation for the rewrite for next year. Ideally, the project code should be so good that it can be used in the lectures already, as example code to illustrate some concepts. But altogether: a great experience! Seeing that one can pull off something nice with random team-members, within 24 hours of coding...
2. it was great to be forced to work hard for the project, but on the other hand stress hindered me from applying all the different techniques that we just learned
3. I would have preferred to work on very smaller project, however, I would use most techniques I've learned during the school, working with different libraries, writing in parallel programmings. Anyhow, I would have preferred working on my own project, instead of a game that I had done as my first B.Sc. programming project while I couldn't use any technique I've learned, not much time regarding all changes that must have done on the code. I suggest dividing the students to two different groups, ones who would like to work on their own code, ones who prefer to do team working.
4. While the project itself was definitely not related to what I do it was really helpful to do something different and to do it in a group. I really got the impression that everyone in my group learned a lot from working as a group - not neccesarily programming-wise but from how others approached different problems. Having the time-limit was also really important to experience - for those in my group who work on their own project for their phd this was a completely novel and incredibly useful experience. Working on a game instead of on a real-world problem was good. It kept things on the light side....
5. Too little time. Much too little time...

6. We had a huge number of problems with the project, most of them social with regard to cooperation and division of work, some due to lacking programming skills. Still, I think the problems were very instructive, and I will remember much better the usefulness of SVN, OOP and unit-testing because they were/would have been crucial for success in this project.
7. Although it was great fun, I feel that the considerable time expense of the pacman project is questionable. Maybe there should be some alternative offer for using this time for other Python-related training. Participants with little prior experience with Python could use this time to understand exercises better or to work on additional exercises.
8. I think the programming project needs to be about a fun topic, even if that it is not necessarily what we encounter in our everyday programming life unless we're in game theory or AI. I think, the experience of programming in a group is valuable, no matter which topic is being worked on. And also, it is hard to find a topic other than a game situation which can be worked on in just a few days. Personally, I had lots of fun and learned quite a bit! As I mentioned before, a tutor for each group would be great!
9. I imagine the results from this component of the survey may vary significantly. The point is, if you had a good group, your experience will be very different. I have mixed feelings about the project. I think the idea of having a group programming project is good, and perhaps it's fine to use the video game. The challenge comes in being sure that all members of the group have opportunities to contribute equally. All in all, I think a project that requires a little more brainstorming / paperwork in how to solve the problem perhaps may be better. Even something that requires going through some mathematics and converting it to code.
10. It was a great project. What I enjoyed the most was there was no "right" solution, and every group had to thinks on strategies.
11. it was great. it had the right balance of preset code vs team-made code to get us up and running quickly, the right amount of workshop time, of scope for originality, the right amount of frustration, chaos & unpredictability and scope for team interaction experinces, both good and not so good
12. I think that time limit and competitive environment forced most groups to employ bad software design due to rush. A tutor assigned to guide each group only in terms of object oriented design and good practices would be nice.
13. It was really fun, especially at the end and a nice experience of team work, it also provoked thinking about problems outside my field of study, I realized that some of them were not trivial, what I had tended to oversee. But for programming in Python, it didn't bring me much, more experienced members of the team could do the programming work much faster and better, and as the time was pretty constrained and adrenaline high, I did't do much on the programming side, and definitely haven't discovered anything new in Python during the programming project, as simple means were sufficient for most of the time. Working on a less constrained problem, which demands for more creativity and decisions on the software design level would be more interesting for me, I think. Designing a really simple packman-like game, or an ALife-like agents' world from scratch or form a simple template, instead of developing strategies for a given game could be an example...
14. My impression is that most scientific programmers work alone (al least I do) therefore I think the experience of team programming is not directly useful for them. On the other hand I consider it a very important and clear example showing why the proper programming practice is actually paying off. Although I work on my own I had encountered troubles in understanding my own code when got back on it after some time. That is the reason why I feel uneasy when it gets to publishing the code and I am afraid that quite large part of my work will be completely unusable to my possible successor.
15. The project was a lot of fun to do and very motivational, but i feel that the time was a little too short to be able to incorporate all the methodologies that were given in the lessons (e.g. unit testing, design patterns, ...) but perhaps this was not entirely the goal. It was more an exercise in python programming and teamwork than real software design.
16. The project could be more scientific, however it is difficult to find the topic that would fit the most of audience. I did once with students the programing project which was writing protein structure viewer with simple python graphics library - vpython. It was ok, but probably you have more fun with pacman.

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: 10%
Right: 76%
Too basic: 14%

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: 83%
Too basic: 13%

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

Yes: 89%
No: 11%

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

Good: 82%
Neutral: 11%
Bad: 7%

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: 3
Colleague/Friend: 14
Website/Mailing list: 12
  of which:
    connectionists/comp-neuro: 7
    python: 2
    G-Node: 2
    echos: 1

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: 93%
No: 7%

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

OK: 76%
Too high: 24%
May be higher!: 0%

Q: Any further comments or suggestions?

1. 150 euro is OK if my company pays, but not for myself.
2. Keep up the great work!
3. Of course, if there was tuition fee for the course, I would have hesitated about the participation. In general, if the fee is less than 80 euro, I think of the attendance, because of the limited budget for attendance in the schools.
4. To make it more fair, you could introduce a fee to cover running costs (IMHO 100-200 euros is ok, as it compares well to the other costs of accommodation and travel, and is usually paid by the university anyways). In addition I suggest you introduce a few stipends to cover part of the costs for students who could otherwise

not afford participating.

5. The organizers did an excellent job. In the little time we had, we were offered a very well-balanced mix of lectures and exercises. As mentioned before, I would have wished for more time for lectures, exercises and team project alike. A two-week event may be a solution. Although there are a couple of schools which offer a 2-week program free of charge, charging a certain fee would be absolutely ok.
6. If Python becomes more wide-spread among scientists, or if there were a "basic" Python course for scientists, the school could concentrate entirely on "advanced" skills, that is version control, team programming, patterns and optimization. I think libraries like NumPy can also be worked on using a tutorial at home.
7. I really enjoyed this event. The social atmosphere was wonderful, and the group of participants was well-balanced. Although I had no prior knowledge of Python, I did learn much more than I could possibly have learned from just reading books and tutorials. It is not easy to find a compromise between too basic and too advanced contents of lectures, but altogether the speakers gave excellent talks. The enthusiasm of the organisers and speakers deserves particular praise. They have convinced me that I should study thoroughly how to use Python.
8. Great job, guys! Keep up the great work! (or convince others to do so ;-))
9. Ok, this is where it gets tricky. First, I like seeing that you broke out 'expectations' from what was 'advertised'. For me, the school was very different from my expectations. Hence, my blank answer. It wasn't too advanced, nor too basic, but simply much different. Overall I would say the use of 'Advanced' as it was used, is not appropriate in the title. The course overall is an introductory course, however, covering rather advanced topics. I was disappointed with the coverage of numpy / scipy, for example, finding it entirely too cursory and limited. Certainly, not 'advanced'. However, the coverage of tables, MPI, and some other concepts were excellent (though still introductory). I realize finding the balance is really a challenge, and overall I think you've all done an excellent job. Perhaps a title more representative would be: "An introduction to advanced programming techniques in Python" Thanks again for the opportunity to participate!
10. Good job, thanks!