“What difference would another type of programming language make? It is a generation issue. The vast majority of this generation of students hates programming. You’re not going to change that by using a different programming language.” A number of lecturers and instructors of our BSc curriculum were in a meeting last year, discussing a potential solution to the problems we had with the first year course “Introduction to Programming”. There was fraud and I was shocked to hear a group of first year students explain to me how “they all hated programming”.

As a 15 year old I fell in love with programming after reading an article on BASIC in the KIX magazine. My first computers were a ZX81 and later a ZX Spectrum bought primarily to practice programming (“Commodore 64s were for gaming”). And I’ve programmed a lot during my study and for my research at NLR and I still use it, for solving or creating mathematical puzzles and right now for example to prepare a GPS navigation quest for my son’s birthday party.

How you can like math, physics and technology and not like programming is utterly incomprehensible for me. When you know how to program, you can model anything in your computer and watch what happens, especially if you add a nice real-time visualization. You’ve got a “sandbox universe” in your laptop. And you feel doped when you see your program finally working, making it highly addictive.

Since the beginning I’ve tried many programming languages: Z80 assembler, Pascal, Fortran, C, C++ and Visual C++ and for me was always about the fun I had. Fortran was always a lot of fun whilst Windows programming in Visual C++ was a nightmare with its dreadful ‘Hungarian notation’ and ‘lost handles’. And lately I discovered Python, a language which is as easy to learn as a beginners language, but turned out to be as powerful as C++. Then I discovered that it was used by MIT, Stanford, Cornell and many other universities for education and scientific purposes. And as I found out that next to the universities also NASA, NOAA, TNO and more research organisations use it, I realized that this language could be something special. On top of that, Python is a totally free, open source and multi-platform (Windows, Linux, Sun OS and Mac) language. And for me personally, Python brought back the fun of programming after my brief excursion to Visual C++, allowing me again to quickly make graphical games for my sons and nephews, as well as visualizations of simulations for myself.

Back to the meeting, there were disadvantages to changing the language: the change itself. Many people in the faculty now know how to use Matlab and are used to it (“But there is hardly any effort involved: it is easy and similar.” “Similar? Then why change?”) And isn’t it a hype? (No, Python was invented in the 90s). Etcetera etcetera.

Two advantages of switching to Python stood out (and still do) for me:

1. Python has many other applications besides from scientific computing, it is a general purpose language. It is used for websites (by YouTube and Google for instance), by game developers and for visual effects in movies. These attractive applications add to the motivation to try programming in different fields than merely creating plots.

2. Matlab is an expensive tool, while Python is free. As a university we get a discount on Matlab. So we teach our students to use it and when after graduation they work in the industry, they will ask their employer to buy this expensive program even for simple programming tasks.
Companies have already complained to me about this. In student projects, buying specific Matlab toolboxes is sometimes prohibitive for a student’s budget. Although it is a win for both sides, by using the discount for education, we have become a part of the sales department of Mathworks. Wouldn’t it be better to teach our students the skill for a tool like, for example, a Swiss army knife, a tool they can keep and use for the rest of their life and career? Anytime, anywhere with all toolboxes, for every application domain, for free?

Despite the exchange of these arguments, the meeting remained inconclusive. Discussing programming languages carries along the risk of becoming almost a religious discussion. Comparable to pilots who will always claim that they are currently flying the best airplane in the world and will claim this again a few years later when they’ve switched from Boeing to Airbus or the other way around.

The director of education therefore proposed a pilot: a random, representative group of first year students would get the AE1106 course on Python, whilst the others would still get it taught in Matlab. With the question at the beginning of this article in mind, I saw this as a challenge: to try to convey not just the programming skills, but also the love for it. Python with its easy syntax and its immense power would allow me to do that, I believed firmly.

There were some things I was not prepared for. In the first lectures I discovered that the book I had chosen was too hard. The learning curve of the assignments I had chosen (Project Euler) was also too steep. But after taking this initial hurdle, it became a lot of fun. The students discovered how to translate a problem into a program. They learned the importance of sketching when modeling and how to produce animated graphics on their screen, next to their plots. In the final part, we held a programming contest, resulting in the most beautiful games and simulations (figure 1).

So, the language, does it matter? Well, not for everyone. There are still people who had a hard time with programming and for whom the frustration level in the beginning exceeds the pleasure of the final success at the end. But in the final session, this group consisted of only three or four students out of the 65. Most indicated they liked programming and the creativity in the entries of the contest showed how much fun they had.

The results of the pilot were overall positive: a higher success rate, more participation, a higher level of fun and a higher level of programming was reached. After the evaluations, it was therefore decided by the director of education to follow the example of Stanford, MIT and other universities and use Python instead of Matlab for our programming and scientific computing course.

In figure 3 and 4 you can see the syntax of Python and also how the use of Scipy makes it very similar to Matlab, easing the transition. If you’re interested in trying it out yourself, try downloading Python(x,y)*; this packaged version of Python has many useful scientific modules already built in, saving the work of downloading them separately. Use Python 2.7 (or any other 2.x) as Python 3.x is still experimental and not yet fully adopted by the community. Then install the pygame module for visualizations, simulations and, yes, games.

Try it and join the large, growing community of Python enthusiasts!

---

*Apple users can download Python, Numpy, Scipy and Matplotlib separately. They can also use a packaged distribution from the Enthought site. They charge a fee for the download but not for universities so use your academic e-mail address. Pygame is also available for Apple.