USING ACCELERATOR TOOLBOX IN A TEACHING ENVIRONMENT

S. Werin*, J. Björklund Svensson, F. Curbis, J. Lundquist, Lund University, Lund, Sweden

Abstract

Teaching accelerator physics is performed in many different environments, where we often think in terms of the major schools: CAS, USPAS or JUAS. The first encounter with accelerators though is most often on undergraduate level as a side-track in a course for particle physics or an introductory course on accelerator technology. Here the high-level tools for simulation often become too complex. For many years a tool suitable for teaching was available in WINAGILE. This gave us a user interface (GUI) and the possibility to easily model accelerators. The code is not supported anymore, and an alternative to be used in early education is highly needed. At the same time, it would be beneficial to have access to a code that is independent of platform (Windows, Linux, IOS) and that also connects to the scientific simulations. In this paper we describe how we use Accelerator Toolbox for *python* set-up in Google Colab with a basic GUI, but available also for Jupyter Notebooks (see Appendix), tuned to an application in introductory courses on Bachelor and Master level in Accelerator physics at Lund University, Sweden. The students look on aspects of the linear beam dynamics and as the simulation runs in the cloud it is independent of platform and can even run on a mobile phone. We welcome more development, good ideas and experiences from teaching situations.

INTRODUCTION/IDEA BEHIND THE APPROACH

For a number of years we have been teaching accelerator physics, accelerator technology, synchrotron radiation and free electron lasers (FEL) to students at bachelor and master level at Lund Univesity. The curriculum has changed over the years but typically included one introductory course on bachelor level and one advanced, continuation course on master level. The bachelor course has been open for students in science in general, with a connection to all fields using synchrotron radiation.

Early on simulations have been introduced in accelerator physics and modelling. The aim has been to introduce the tools and methods used in accelerator design and to enhance the learning of accelerator physics concepts. For many years the first encounter with simulation software for accelerators has been Winagile [1]. This code presented a graphical user interface and for the students no knowledge of programming was necessary. It was thus well possible to guide the students through the interface and they could study the basic optics of different systems. However a recurrent problem was students using different platforms (Windows, IoS or Linux), where only Windows was supported. Winagile was also stopped from development and getting harder to find download locations. Thus we realized that another solution was necessary.

With a new solution for early students we could pin down a requirement list. With these criteria we chose to use Accelerator Toolbox in *python* and, after some tests, use Colab for the cloud service.

- Simple
- Real simulation engine (Accelerator Toolbox)
- No installation
- No coding experience
- Customizable
- Can run on Cloud services
- Runs in web browser = platform independent

Other aspects that we came more or less for free were:

- Runs on mobile phone!
- Exportable as Jupyter-notebook

TEACHING AND HOW IT CONNECTS TO THE CURRICULUM AND FOLLOWING COURSES

We strongly believe that accelerator physics is in most scientific applications nowadays directly connected to simulations. It is thus important to introduce the students to "real codes" as early as possible. We also realize that many of these codes need excessive computer knowledge to install and run. They also need to connect to software that can read output data and present it in a comprehensible way.

One way to approach this problem is to use "basic tools", MATLab [4], *python* or even spreadsheets such as EXCEL, to make basic models through matrix calculations. The programming skills needed may stay on an acceptable level, but this neither introduces the real accelerator codes nor can be expanded to more complete models.

For a beginning bachelor student in science programming is not always a strong ability yet. While they would be able to do basic programming, it is often not on the level to help understanding of other complex phenomena. The programming in itself will hide the opportunities for learning.

By putting an interface (Fig. 1) on an existing code package for accelerator physics and making it available on a general cloud server we could hopefully provide the students with a tool to focus on learning accelerator physics. As we later realized, some students moved the code to their own environment and some even use samples of the code and built their own simulations. The latter is of course exactly what we hoped could happen!

The course at Lund University for which this solution was developed, *Photon and neutron production for science*, has four components out of which one is accelerator physics/beam dynamics. The basic concepts of matrix

* sverker.werin@maxiv.lu.se

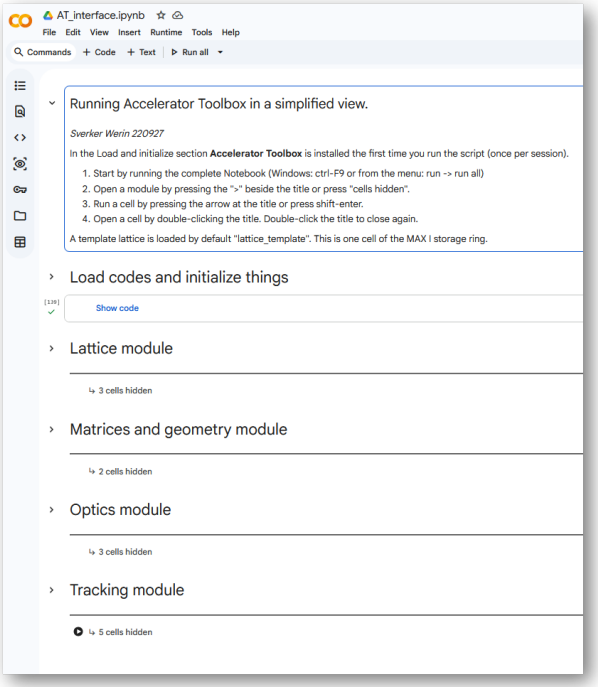

Figure 1: Starting interface as in Colab.

representations, magnet elements, the equations of motion, betatron oscillations, beta functions, tunes and emittance are discussed. To allow hands-on training a guided simulation session where the students work in groups lets the students try the concepts.

The students should assemble a given lattice with some remaining data reading a paper describing the design of the MAX IV 3 GeV ring. From this they should check the matrices of individual elements and the complete transfer matrix, manually calculate the beta function at the starting point and conclude a stable lattice. From this they move into looking at the optics. Calculating the tunes, beta functions and the dispersion. This could be followed up with looking at quadrupoles focusing in opposite directions and how/if the dispersion returns to zero in the straight sections between the achromats.

In the simulation the students can track individual particles down the accelerator and from this it is possible to discuss both tunes and the relation between individual particles and the beta function envelopes. With an energy offset on the particle the case can be connected to the dispersion and more advanced looking on change in focusing, origin of chromaticity.

In a last step a beam of particles with different properties is tracked and presented both in real space (x-y) and in phase space (x-x' / y-y') from which insights can be gained on emittance, rotation in phase space in focusing and defocusing.

All these aspects available through the code cannot be covered in a 2-3 h group simulation session, and thus a subset was used in the actual teaching exercise.

The exercise connects and acts as an introduction to a continuation course where programming is needed (also using pyAT) to create a more complete model of the MAX IV 1.5 GeV ring and then in a practical lab session operate the ring, measure tunes, look at kicks, chromaticity, frequency/energy changes.

STUDENT FEEDBACK

With limited statistics we can observe that students used PCs and most stayed in the Colab environment. The exercise based on the code gave some better understanding of accelerator physics (3.7/5) and how accelerator simulations are done (4.1/5). Some students would have preferred to do more coding themselves (approach in more advanced courses) Even though the code was straight forward, the students appreciated a Teaching Assistant to help

THE CODE

Choice of Package (pyAT)

Accelerator Toolbox is gaining interest in control of accelerators. It is not the main tool to study detailed and more advanced phenomena in beam dynamics, but it is increasingly used for the control and analysis of operation of accelerators. It is also easily used in connection to machine learning applications both related to operation and diagnostics. Accelerator Toolbox has initially been developed for MATLab, but the current focus is moving towards the version being based on *python*, pyAT. This is also in line with many teaching institutions which are turning towards *python* for their students. We also see a clear increasing knowledge of *python* among the students.

Choice of Platform (Colab)

One of the main objectives for developing a new tool was to ease the situation with students using different platforms. A cloud based system that can run in any web-browser is a significant simplification. Not only will it make life easy for our students at Lund University, but it also opens up for students anywhere in the world to easily get acquainted with accelerator simulations using a powerful tool. Google Colab was chosen partly because many students, also at Lund University, receives a Google account when receiving their student mail at the university and for many others through their Android phones. For the rest a registration is easy and without any financial cost. Colab works well in most browsers and hardware platforms, but more importantly is that it can install the Accelerator Toolbox package easily. Colab also allows for easy ways to "clean up" the interface in the Notebooks used for running the code.

However, we also realize that running the code on Google servers poses other concerns of ownership, integrity and supplying training data for GAI systems. Thus we have also assured that the code can run on other platforms, however this requires more computer and programming skills (traditional python and Jupyter Notebook installation) or some minor

complexity, in services away from the commercial providers, like EU science Cloud [5].

Layout and Approach

The interface should be as clean and simple as possible. Emphasising that coding knowledge is not necessary. At the same time it should not completely hide the need for code.

The interface is tuned partly to the duty the students of the course take and can be changed and extended. It is for example directly possible to add tune fitting. While the natural chromaticity is calculated, sextupoles are at the moment not included to make the lattice definitions easier. This can however also be expanded with slightly more effort. This comes for all other aspects of the pyAT package.

Starting the code in Colab gives a brief introduction and a set of headlines of modules: Lattice, Matrices and geometry, Optics and a Tracking module (Fig. 2). A "Load codes and initialize things" module is added to make the code work, but is not necessary to interact with. After running the code the different modules can be opened to investigate the different aspects. While the code has a template lattice (an achromat of the old MAX I 550 MeV accelerator) the option of changing to another lattice and writing one's own is available. To input or change a lattice is the only instance when the user needs to interact with some lines of python code. However, there is a template to follow and copy.

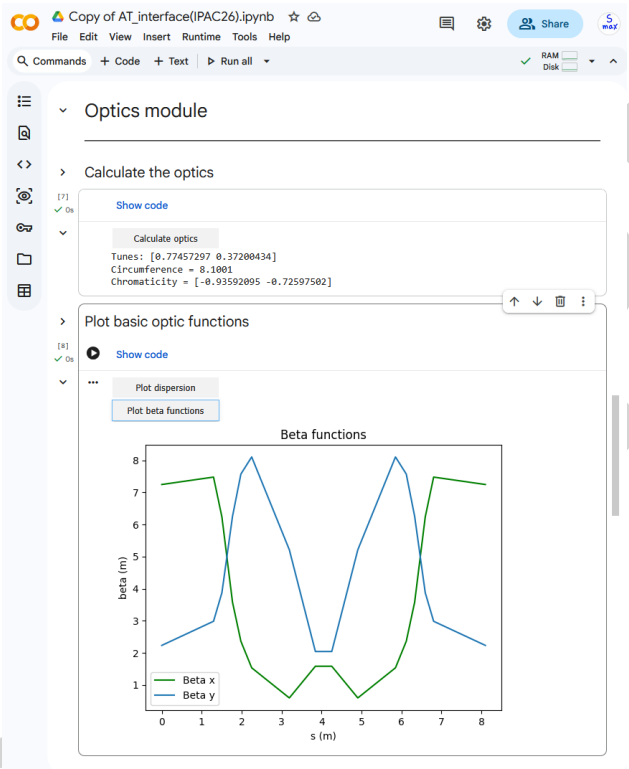

Figure 2: Sample screen showing parts of the Optics module.

CONCLUSION AND OUTLOOK

We have shown how students without the need for programming skills can start using an advanced accelerator simulation tool through a basic GUI. The system is easy enough to run platform independent, even on a mobile phone. This opens possibilities not only for students being newcomers to accelelerators but also students with limited access to computer resources anywhere worldwide.

APPENDIX - CODE

The code is available on GitHub both in a version suitable for Colab and for a general Jupyter Notebook environment: *https://github.com/werin99/pyAT_interface.git*

It can also be opened in Colab through a single link: *https://colab.research.google.com/github/werin99/ pyAT_interface/blob/master/pyAT_interface_Colab.ipynb*

This opens Colab, downloads from GitHub and starts the Notebook in one go.

REFERENCES

[1] P. J. Bryant, "AGILE, A Tool for Interactive Lattice Design", in *Proc. EPAC'00*, Vienna, Austria, Jun. 2000, paper TUP6B03, pp. 1357–1359.

[2] A. Terebilo, "AcceleratorToolbox for MATLAB", SLAC National Accelerator Laboratory, Menlo Park, CA, USA, Rep. SLAC-PUB-8732, May 2001. https://inspirehep.net/literature/557783

[3] Accelerator Toolbox (AT), https://github.com/atcollab/at

[4] V. Ziemann, "Hands-On Accelerator Physics Using MATLAB", CRC Press, 2019

[5] EU science Cloud, https://open-science-cloud.ec.europa.eu