Research software in mathematical biology

As a 2022 SSI Fellow, I recently had the opportunity to use some of my funding to attend RSECon22. While there, I gave a talk about some nifty Python tools I have come to rely on in my work to explore and analyse mathematical models in my remit of mathematical biology. In this blog post, I talk more generally about how research software and mathematical modelling go hand-in-hand, and my fellowship goal to promote software sustainability in this area of academic research.

Mathematical modelling has never been in the limelight more so than in the past two pandemic years, where leading epidemiologists became household names almost overnight. At RSECon, which took place last month from 6th-8th September in Newcastle, we got to hear first hand from keynote speaker Professor Neil Ferguson about the evolution of research questions, data streams, and the speed of reporting that unfolded in 2020 and is still ongoing. Moreover, we got to hear about the importance of the supporting RSE team led by Richard FitzJohn at Imperial to handle the volume of incoming data, scale up the software to run highly detailed models daily, and to promote sustainable and test-driven code development.

This sustainable approach to software development does not need to be confined to high-stakes epidemic modelling; though you are less likely to have quite the volume of resources afforded to the Imperial team, time spent investing in the sustainability of your software will pay off. The benefits of well-written, tested code are two-fold: you are less likely to have code-related setbacks in your own research, and you can participate in an acceleration of research in your domain by sharing your code in a reusable state. Creating an early habit of code sharing will naturally lead you to write more reproducible code, and pay dividends when it comes to writing up your work (what data did that plot come from again?). If you're not sure where to start, try the editorial Ten Simple Rules for Reproducible Computational Research.

In my talk, I sang the praises of a Python module 'Pyswarms', a handy tool that implements the particle swarm optimisation algorithm which can be used to find optimal parameter sets in a multi-parameter mathematical model. While a useful part of any model analysis pipeline, the tool itself originated from a PhD project - the code was shared, licensed, and is now maintained by an active community that recognises its value. Don't underestimate the use of your shared code - and remember to look for, and support, any existing tools that may save you a few days writing something new from scratch.

During my fellowship, I hope to create an open dialogue in my academic environment to dispel the myths surrounding good coding practice and code sharing. Writing good software is rarely on the curriculum, and new researchers often find themselves in the deep end. By supporting seminars and workshops, I am to create an inclusive community from research students to senior professors, that fosters knowledge exchange on a level playing field. It’s also common for senior researchers to have never had the opportunity to explore the principles of sustainable software - and they may be more willing to admit it than you’d think.