LOCKDOWN SCEPTICS
STAY SCEPTICAL. CONTROL THE HYSTERIA. SAVE LIVES.

Second Analysis of Ferguson's Model

9 May 2020

by Sue Denim (not the author's real name)

I'd like to provide a followup to my first analysis. Firstly because new information has come to light, and secondly to address a few points of disagreement I noticed in a minority of responses.

The hidden history. Someone realised they could unexpectedly recover parts of the deleted history from GitHub, meaning we now have an audit log of changes dating back to April 1st. This is still not exactly the original code Ferguson ran, but it's significantly closer.

Sadly it shows that Imperial have been making some false statements.

- ICL staff claimed the released and original code are "*essentially the same functionally*", which is why they "*do not think it would be particularly helpful to release a second codebase which is functionally the same*".

  In fact the second change in the restored history is a fix for a critical error in the random number generator. Other changes fix data corruption bugs (another one), algorithmic errors, fixing the fact that someone on the team can't spell household, and whilst this was taking place other Imperial academics continued to add new features related to contact tracing apps.

  The released code at the end of this process was not merely reorganised but contained fixes for severe bugs that would corrupt the internal state of the calculations. That is very different from "*essentially the same functionally*".

- The stated justification for deleting the history was to make "*the repository rather easier to download*" because "*the history squash (erase) merged a number of changes we were making with large data files*". "*We do not think there is much benefit in trawling through our internal commit histories*".

  The entire repository is less than 100 megabytes. Given they recommend a computer with 20 gigabytes of memory to run the simulation for the UK, the cost of downloading the data files is immaterial. Fetching the additional history only took a few seconds on my home WiFi.

> Even if the files had been large, the tools make it easy to not download history if you don't want it, to solve this exact problem.

I don't quite know what to make of this. Originally I thought these claims were a result of the academics not understanding the tools they're working with, but the Microsoft employees helping them are actually employees of a recently acquired company: GitHub. GitHub is the service they're using to distribute the source code and files. To defend this I'd have to argue that GitHub employees don't understand how to use GitHub, which is implausible.

I don't think anyone involved here has any ill intent, but it seems via a chain of innocent yet compounding errors – likely trying to avoid exactly the kind of peer review they're now getting – they have ended up making false claims in public about their work.

Effect of the bug fixes. I was curious what effect the hidden bug fixes had on the model output, especially after seeing the change to the pseudo-random number generator constants (which means the prior RNG didn't work). I ran the latest code in single threaded mode for the baseline scenario a couple of times, to establish that it was producing the same results (*on my machine only*), which it did. Then I ran the version from the initial import against the latest data, to control for data changes.

The resulting output tables were radically different to the extent that they appear incomparable, e.g. the older code outputs data for negative days and a different set of columns. Comparing by row count for day 128 (7th May) gave 57,145,154 infected-but-recovered people for the initial code but only 42,436,996 for the latest code, a difference of about 34%.

I wondered if the format of the data files had changed without the program being able to detect that, so then I reran the initial import code with the initial data. This yielded 49,445,121 recoveries – yet another completely different number.

It's clear that the changes made over the past month and a half have radically altered the predictions of the model. It will probably never be possible to replicate the numbers in Report 9.

Political attention. I was glad to see the analysis was read by members of Parliament. In particular, via David Davis MP the work was seen by Steve Baker – one of the few British MPs who has been a working software engineer. Baker's assessment was similar to that of most programmers: "*David Davis is right. As a software engineer, I am appalled. Read this now*". Hopefully at some point the right questions will be asked in Parliament. They should focus on reforming how code is used in academia in general, as the issue is structural incentives rather than a single team. The next paragraph will demonstrate that.

Do the bugs matter? Some people don't seem to understand why these bugs are important (e.g. this computational biology student, or this cosmology lecturer at Queen Mary). A few people have claimed I don't understand models, as if Google has no experience with them.

Imagine you want to explore the effects of some policy, like compulsory mask wearing. You change the code and rerun the model with the same seed as before. The number of projected deaths goes up rather than down. Is that because:

- The simulation is telling you something important?
- You made a coding error?
- The operating system decided to check for updates at some critical moment, changing the thread scheduling, the consequent ordering of floating point additions and thus changing the results?

You have absolutely no idea what happened.

In a correctly written model this situation can't occur. A change in the outputs means something real and can be investigated. It's either intentional or a bug. Once you're satisfied you can explain the changes, you can then run the simulation more times with new seeds to estimate some uncertainty intervals.

In an uncontrollable model like ICL's you can't get repeatable results and if the expected size of the change is less than the arbitrary variations, you can't conclude anything from the model. And exactly because the variations are arbitrary, you don't actually know how large they can get, which means there's no way to conclude anything at all.

I ran the simulation three times with the code as of commit 030c350, with the default parameters, fixed seeds and configuration. A correct program would have yielded three identical outputs. For May 7th the max difference of the three runs was 46,266 deaths or around 1.5x the actual UK total so far. This level of variance may look "small" when compared to the enormous overall projections (which it seems are incorrect) but imagine trying to use these values for policymaking. The Nightingale hospitals added on the order of 10-15,000 places, so the uncontrolled differences due to bugs are larger than the NHS's entire crash expansion programme. How can any government use this to test policy?

An average of wrong is wrong. There appears to be a seriously concerning issue with how British universities are teaching programming to scientists. Some of them seem to think hardware-triggered variations don't matter if you average the outputs (they apparently call this an "ensemble model").

Averaging samples to eliminate random noise works only if the noise is actually random. The mishmash of iteratively accumulated floating point uncertainty, uninitialised reads, broken shuffles, broken random number generators and other issues in this model may yield *unexpected* output changes but they *are not truly random deviations, so* they can't just be averaged out. Taking the average of a lot of faulty

measurements doesn't give a correct measurement. And though it would be convenient for the computer industry if it were true, you can't fix data corruption by averaging.

I'd recommend all scientists writing code in C/C++ read this training material from Intel. It explains how code that works with fractional numbers (floating point) can look deterministic yet end up giving non-reproducible results. It also explains how to fix it.

Processes not people. This is important: the problem here is not really the individuals working on the model. The people in the Imperial team would quickly do a lot better if placed in the context of a well run software company. The problem is the lack of institutional controls and processes. All programmers have written buggy code they aren't proud of: the difference between ICL and the software industry is the latter has processes to detect and prevent mistakes.

For standards to improve academics must lose the mentality that the rules don't apply to them. In a formal petition to ICL to retract papers based on the model you can see comments "explaining" that scientists don't need to unit test their code, that criticising them will just cause them to avoid peer review in future, and other entirely unacceptable positions. Eventually a modeller from the private sector gives them a reality check. In particular academics shouldn't have to be convinced to open their code to scrutiny; it should be a mandatory part of grant funding.

The deeper question here is whether Imperial College administrators have any institutional awareness of how out of control this department has become, and whether they care. If not, why not? Does the title "Professor at Imperial" mean anything at all, or is the respect it currently garners just groupthink?

Insurance. Someone who works in reinsurance posted an excellent comment in which they claim:

- There are private sector epidemiological models that are more accurate than ICL's.
- Despite that they're still too inaccurate, so they don't use them.
- "*We always use 2 different internal models plus for major decisions an external, independent view normally from a broker. It's unbelievable that a decision of this magnitude was based off a single model*"

They conclude by saying "*I really wonder why these major multinational model vendors who bring in hundreds of millions in license fees from the insurance industry alone were not consulted during the course of this pandemic.*"

278

A few people criticised the suggestion for epidemiology to be taken over by the insurance industry. They had insults ("mad", "insane", "adding 1 and 1 to get 11,000" etc) but no arguments, so they lose that debate by default. Whilst it wouldn't work in the UK where health insurance hardly matters, in most of the world insurers play a key part in evaluating relative health risks.