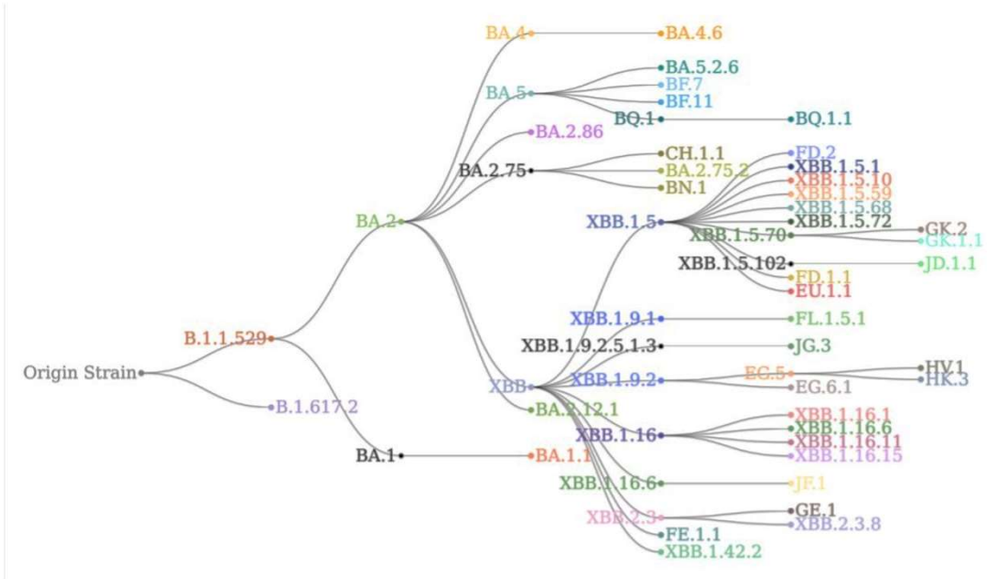
A Simulation Model and Dashboard For Predicting COVID-19 Bed Requirements

Yin-Chi Chan, Kaya Dreesbeimdiek, Ajith Kumar Parlikad, Tom Ridgman
Institute for Manufacturing, University of Cambridge

Nicholas J. Matheson, Ben Warne
Cambridge Institute for Therapeutic Immunology and Infectious Disease (CITIID) & Department of Medicine, University of Cambridge
Cambridge University Hospitals NHS Foundation Trust

Denise Franks
Cambridge University Hospitals NHS Foundation Trust

Background: COVID-19



COVID-19 continues to evolve

Image source: https://covid.cdc.gov/covid-data-tracker/#variant-proportions

COVID-19 in the UK:

- 22.2 million total cases (as of 19 May 2022)
- 232,837 deaths (as of 27 Oct 2023)
- Over 6% prevalence in March 2022 (1 in 16)

Sources:

- https://coronavirus.data.gov.uk/
- Science, 2022. https://doi.org/10.1126/science.abq4411

"...the world failed this test of preparedness and response. It remains to be determined whether the world will do better next time—because it is inevitable that there will be a next time."

Thomas R Frieden, Marine Buissonnière, Amanda McClelland; BMJ Global Health 66(3), 2021. https://doi.org/10.1136/bmjgh-2021-005184

Background: Healthcare in the UK

- National Health Service founded in 1948
  - "The new health service in all its branches will be free to all" [*A National Health Service – The White Paper Proposals in Brief*, HM Stationery Office (1944)]
  - In contrast: 16.6% of Texans were uninsured as of 2022 (worst in USA)
    KFF: https://www.kff.org/other/state-indicator/total-population/
- However, in recent years the NHS has come increasingly under strain
- Therefore:
  - Need to optimise the usage of limited medical & staff resources
  - Need contingency plans to deal with unexpected shocks in healthcare demand (e.g. due to pandemics)

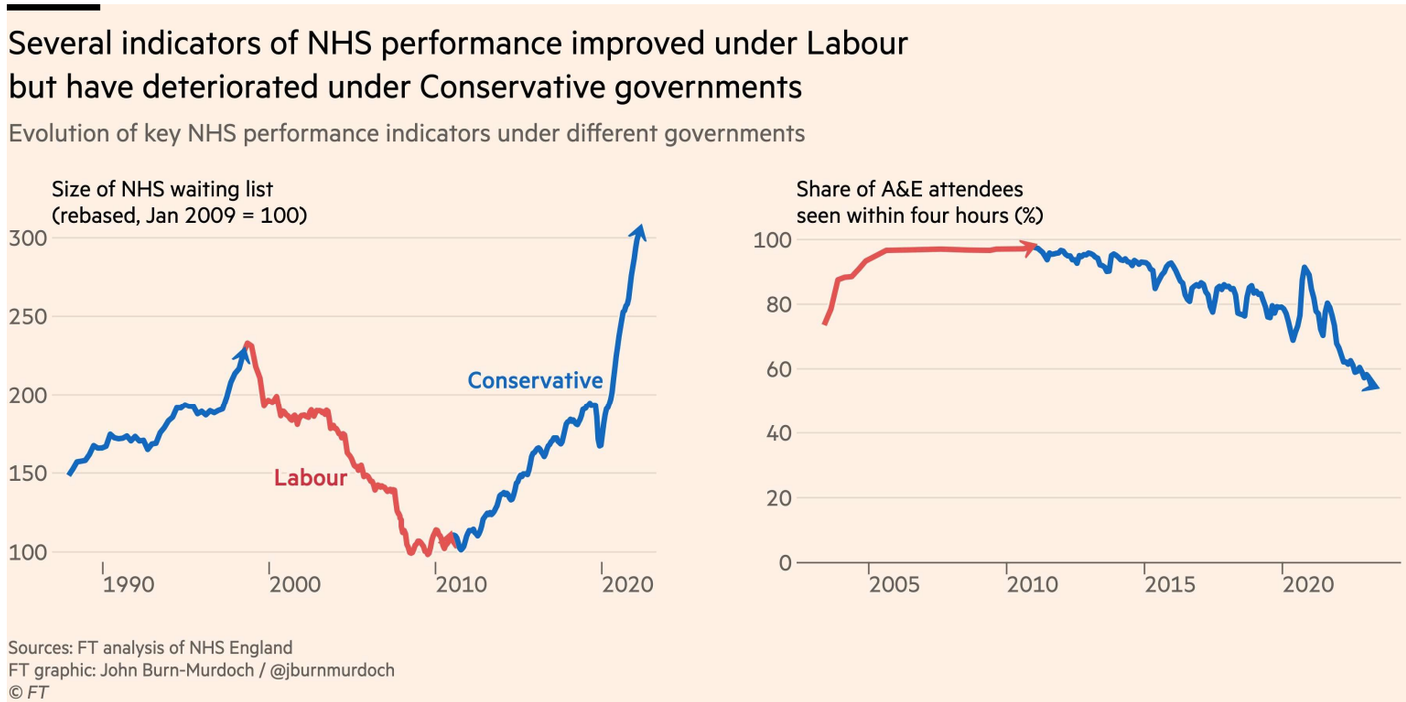


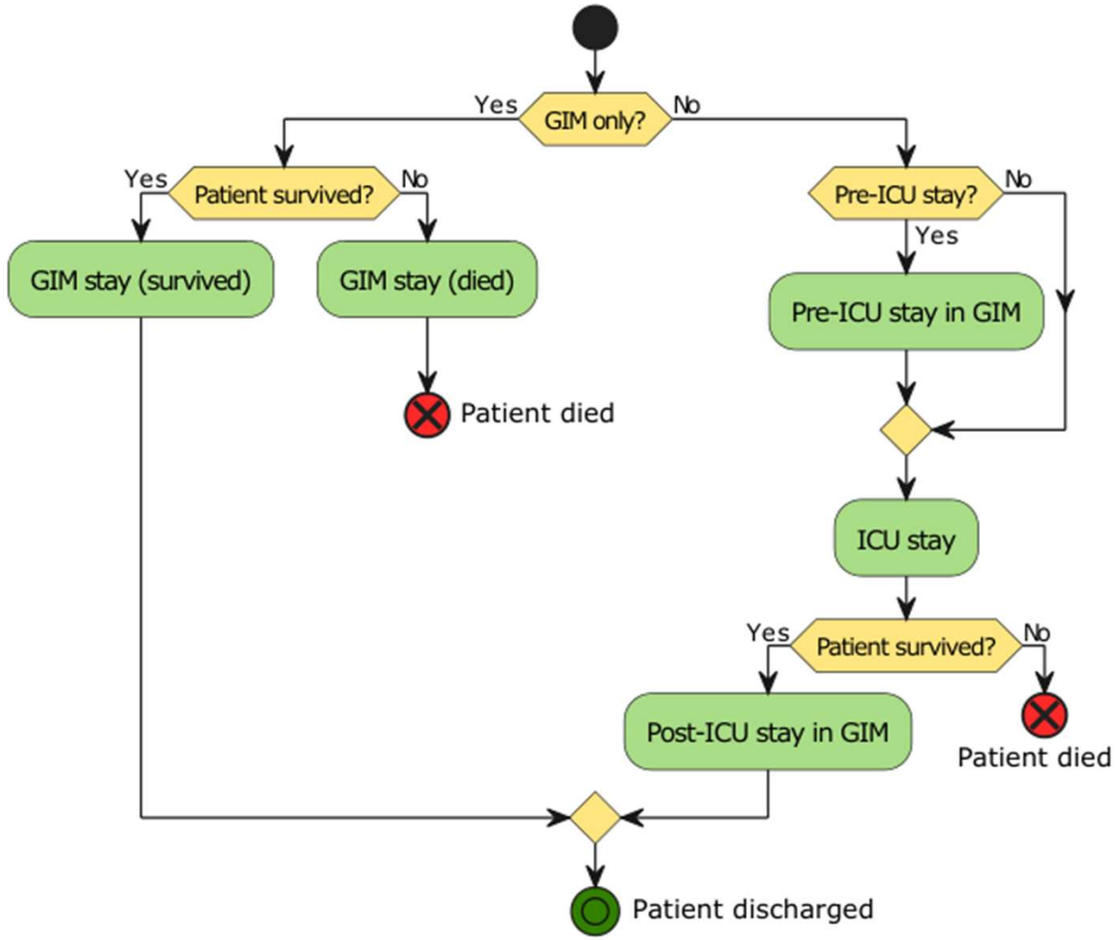
Image Source: John Burn-Murdoch / Financial Times
https://twitter.com/jburnmurdoch/status/1606223967903260673

Case study: Addenbrooke's Hospital, Cambridge, UK

- Part of Cambridge University Hospitals NHS Foundation Trust (CUH)
- Previously: patient flow model of COVID-19 patients
  - GJ Melman, AK Parlikad, EAB Cameron; Health Care Management Science 24(2), pp. 356-374, 2021. https://doi.org/10.1007/s10729-021-09548-2
- This work:
  - Analysis of General Internal Medicine (GIM) and Intensive Care Unit (ICU) bed demand, patient lengths-of stay (LoS)
  - Discrete-event simulation model of patient flow (Python)
    - Unite data preprocessing, modelling and simulation under a single framework
    - Web dashboard implemented using Plotly Dash
  - Simulation results used for "red/green" bed allocation planning at Addenbrooke's

Patient flow model

- Five types of length-of-stay (green boxes):
  - GIM-only, patient survived
  - GIM-only, patient died
  - Pre-ICU stay in GIM
  - ICU stay
  - Post-ICU stay in GIM



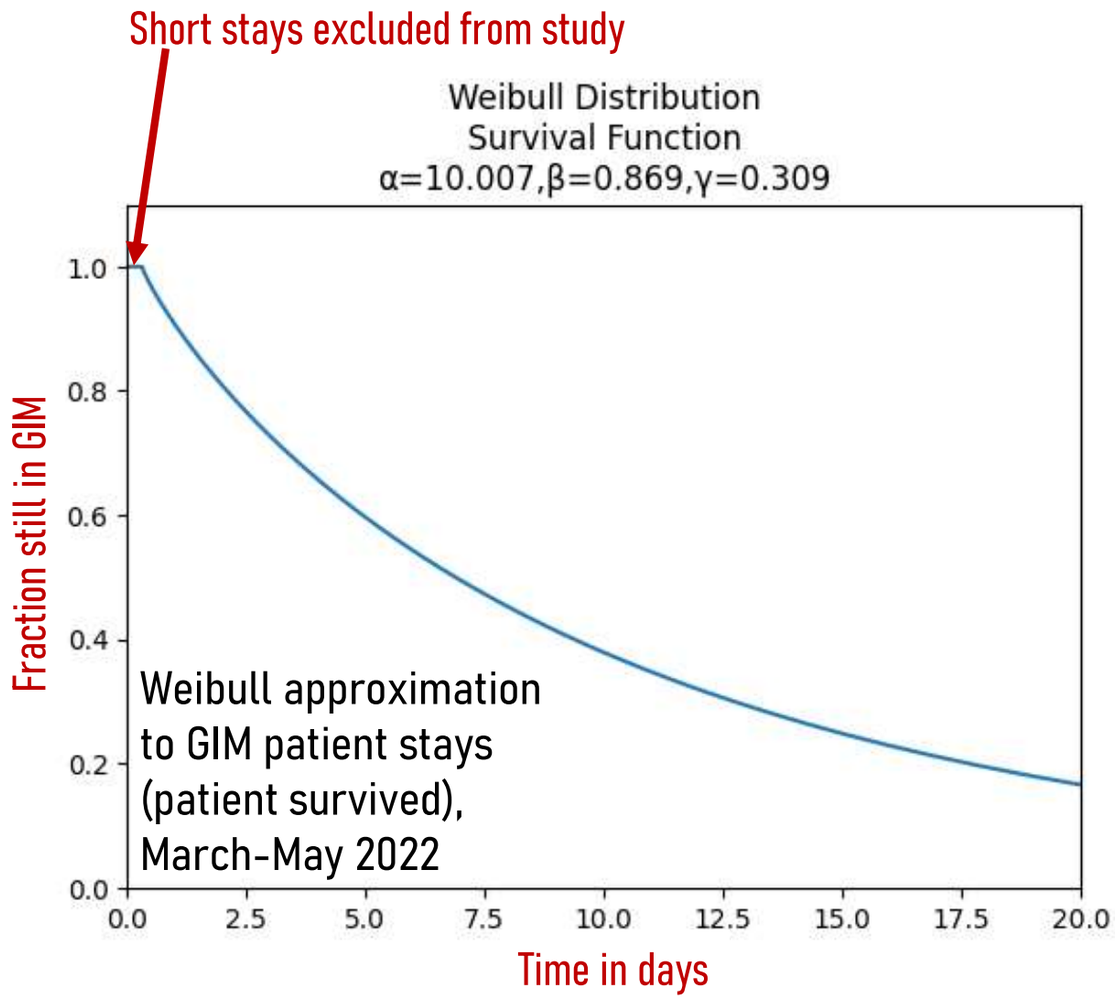
Data Collection & Preprocessing

- Patient data mined from CUH's EPIC electronic health record system
  - Demographic data
  - Admission/discharge times for general and/or ICU ward
  - Readmission data if applicable
    - Only the first readmission was tracked in this study
  - Infection source (community / hospital)
- Preprocessing:
  - Remove short stays
  - Remove patient history prior to infection (if hospital-acquired)
  - Combine stays if gap between first admission and readmission is small

Length-of-Stay Analysis (1)

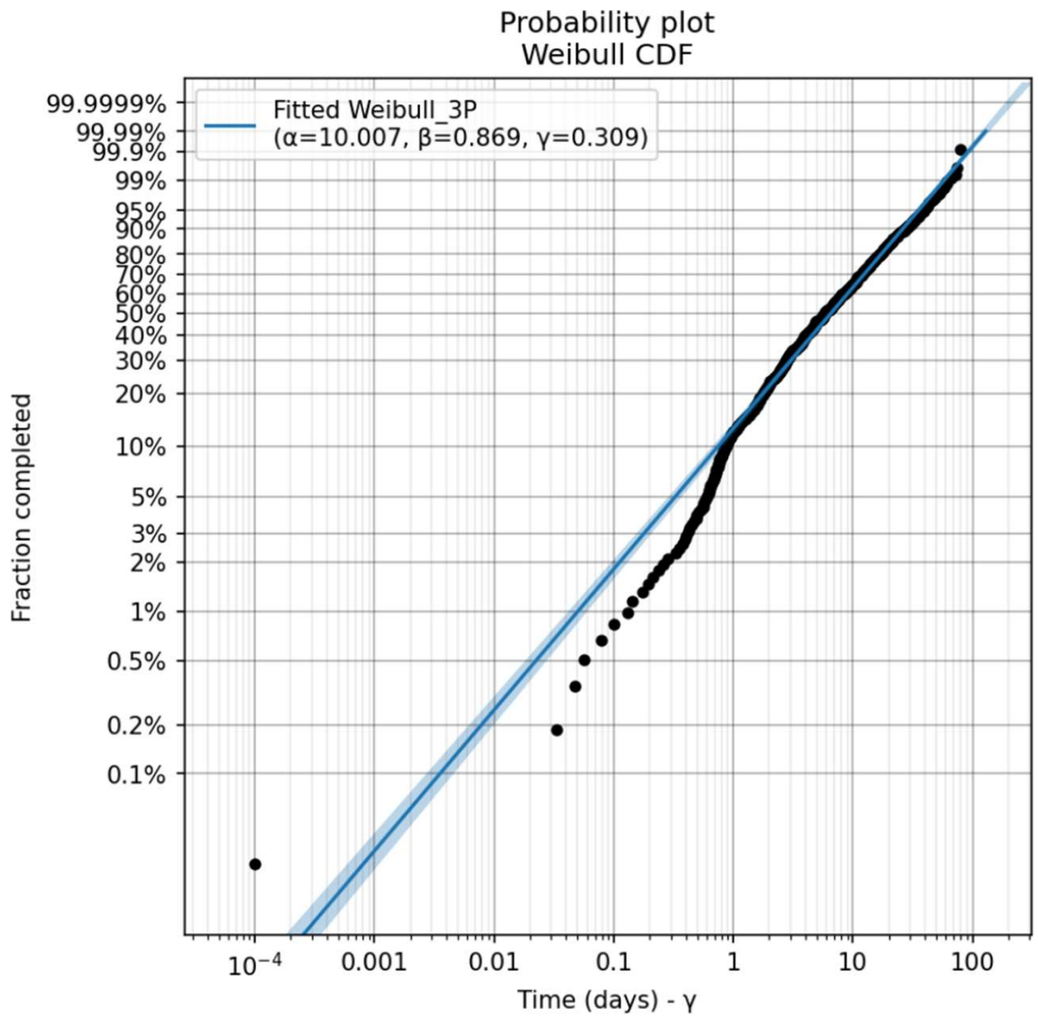
- The reliability Python package was used for distribution fitting
- A shifted Weibull distribution (`Weibull_3P`) was found to fit all five LoS types well
  - Proportion of patients remaining after t days:

$$\bar{F}(t) = \exp\left[-\left(\frac{t-\gamma}{\alpha}\right)^{\beta}\right] \qquad (t \geq \gamma)$$



Length-of-Stay Analysis (2)

- Probability plots were used to visualise the goodness of fit
  - In general, fit is poorer for shorter stays, but such stays have less effect on overall bed usage
  - Shifted Lognormal and Gamma distributions were also considered, with similar results

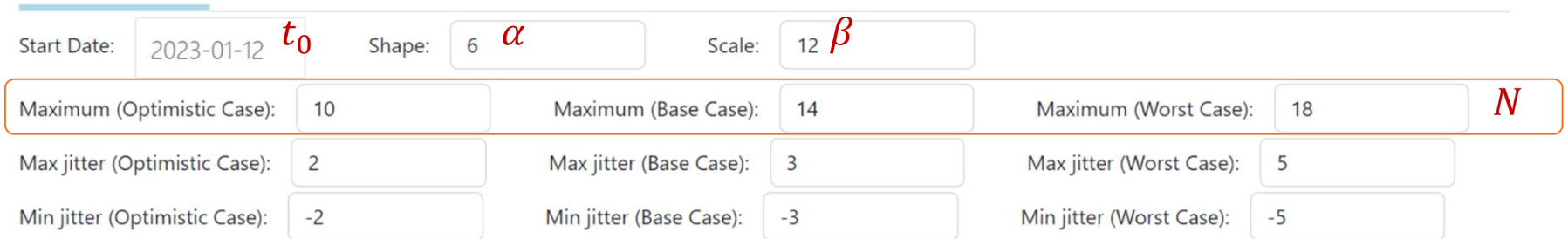


Step-down policy

- Long-stay patients generally are not infectious after a certain amount of time
  - Typically admitted for other reasons
  - "With" vs. "for" COVID-19
- Step-down policy introduced to reflect change in patient status
  - GIM patients removed from "red bed" status after N days (if still in hospital)
  - In reality, patients are only removed after negative test results
    - Would require data on inpatient testing results – not done in this study
- Our simulation model considers both total and "red" bed usage by patients admitted with COVID-19

COVID-19 Patient Arrival Modelling (1)

- Gamma functions were used to model daily patient arrivals
  - Parameters α, β, N, t_0
  - Maximum value at $t = t_0 + (\alpha - 1)\beta$ such that $f(t) = N$
- For prediction, we would create 3 scenarios, varying the maximum value N
- A visual tool was developed for parameter selection



COVID-19 Patient Arrival Modelling (2)

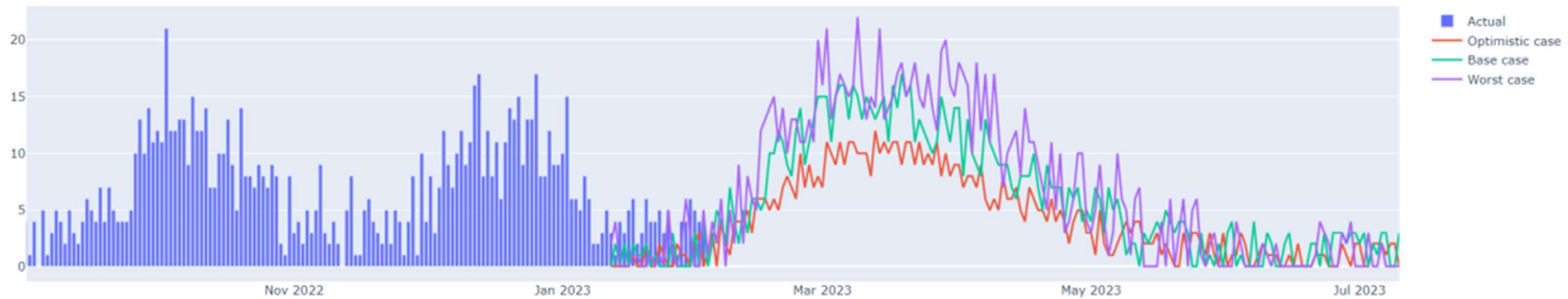
Modelling Tool | Info

Start Date: 2023-01-12 Shape: 6 Scale: 12

Maximum (Optimistic Case): 10 Maximum (Base Case): 14 Maximum (Worst Case): 18

Max jitter (Optimistic Case): 2 Max jitter (Base Case): 3 Max jitter (Worst Case): 5

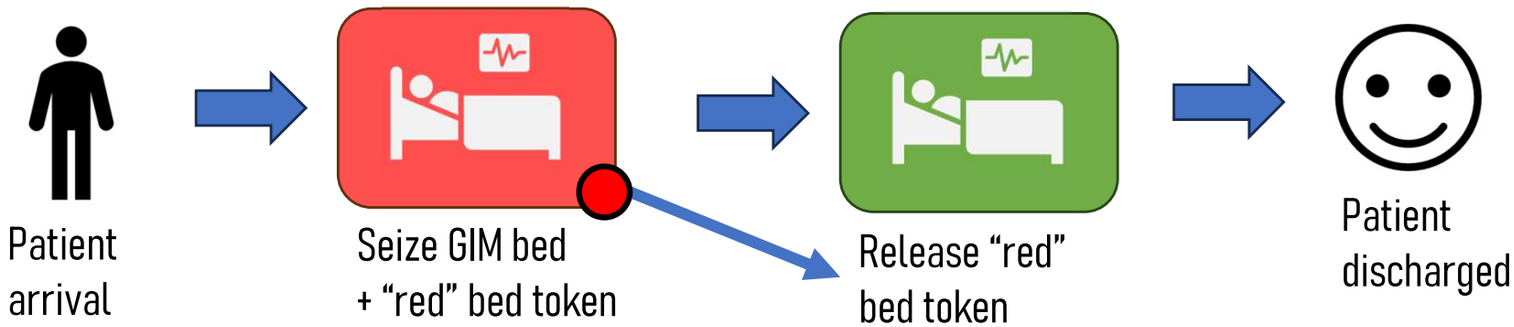
Min jitter (Optimistic Case): -2 Min jitter (Base Case): -3 Min jitter (Worst Case): -5



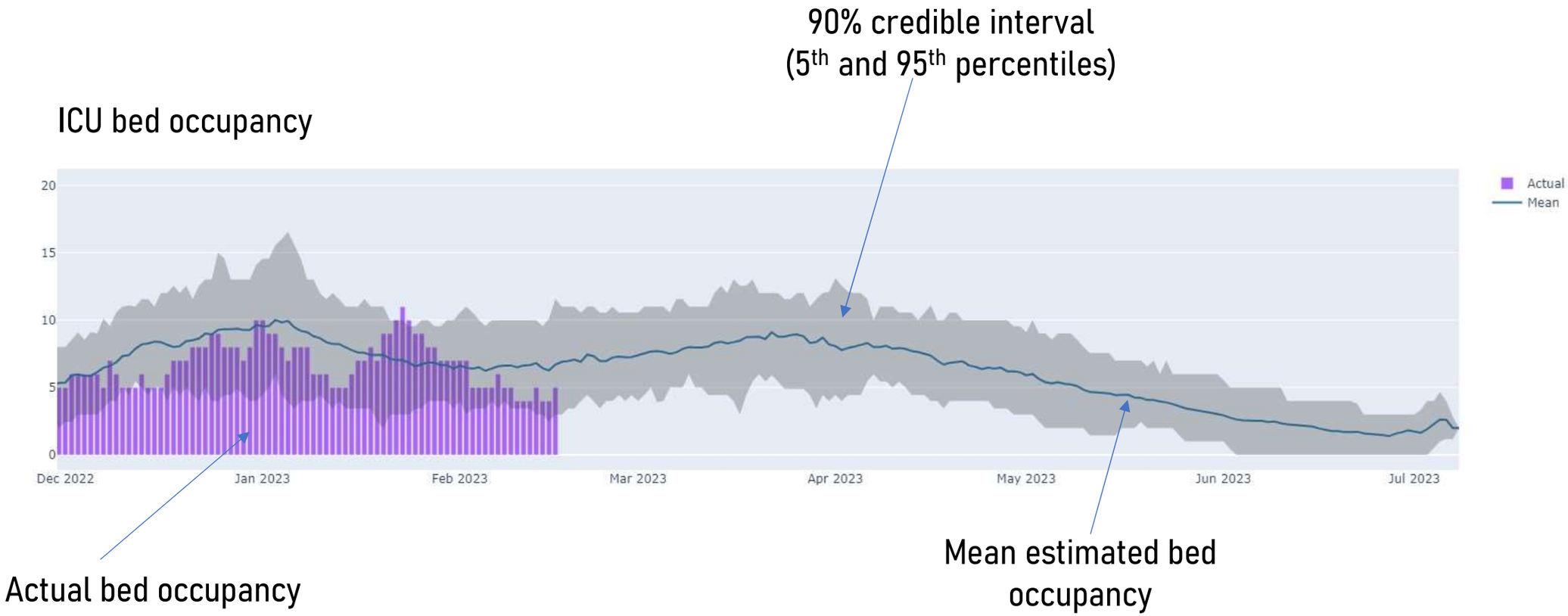
Scenario Start Date: 2022-09-01 Download file

Python Simulation

- Discrete event simulation using simpy
  - Based on Python generators
- Bed occupancy logged using pandas dataframes
- Trick to track “red” bed status:
  - Patients in GIM seize both an actual GIM bed and a “red bed” token
  - Releasing the token = transfer to “green” bed status (assuming patient is not immediately discharged)

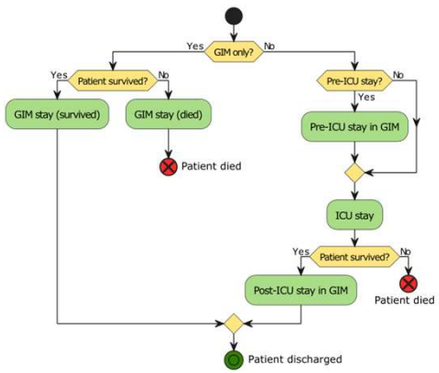


Example simulation output



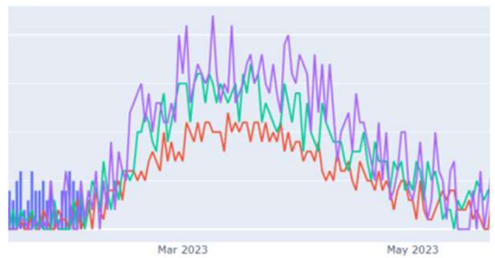
Dashboards

- Plotly Dash used to create dashboards to simplify modelling and simulation
  - Hybrid open-source/commercial product – we only use open-source components
- Three dashboards



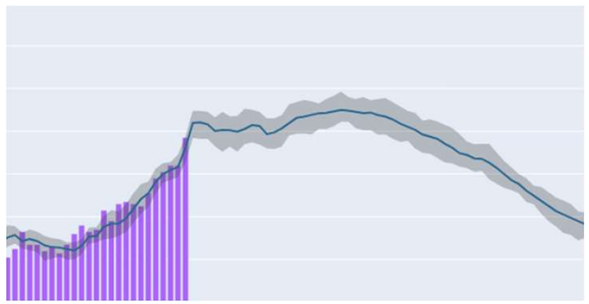
Patient analysis

Fit length-of-stay distributions and compute probability of each patient pathway



Scenario creation

Generate number of daily patient arrivals to the simulation model



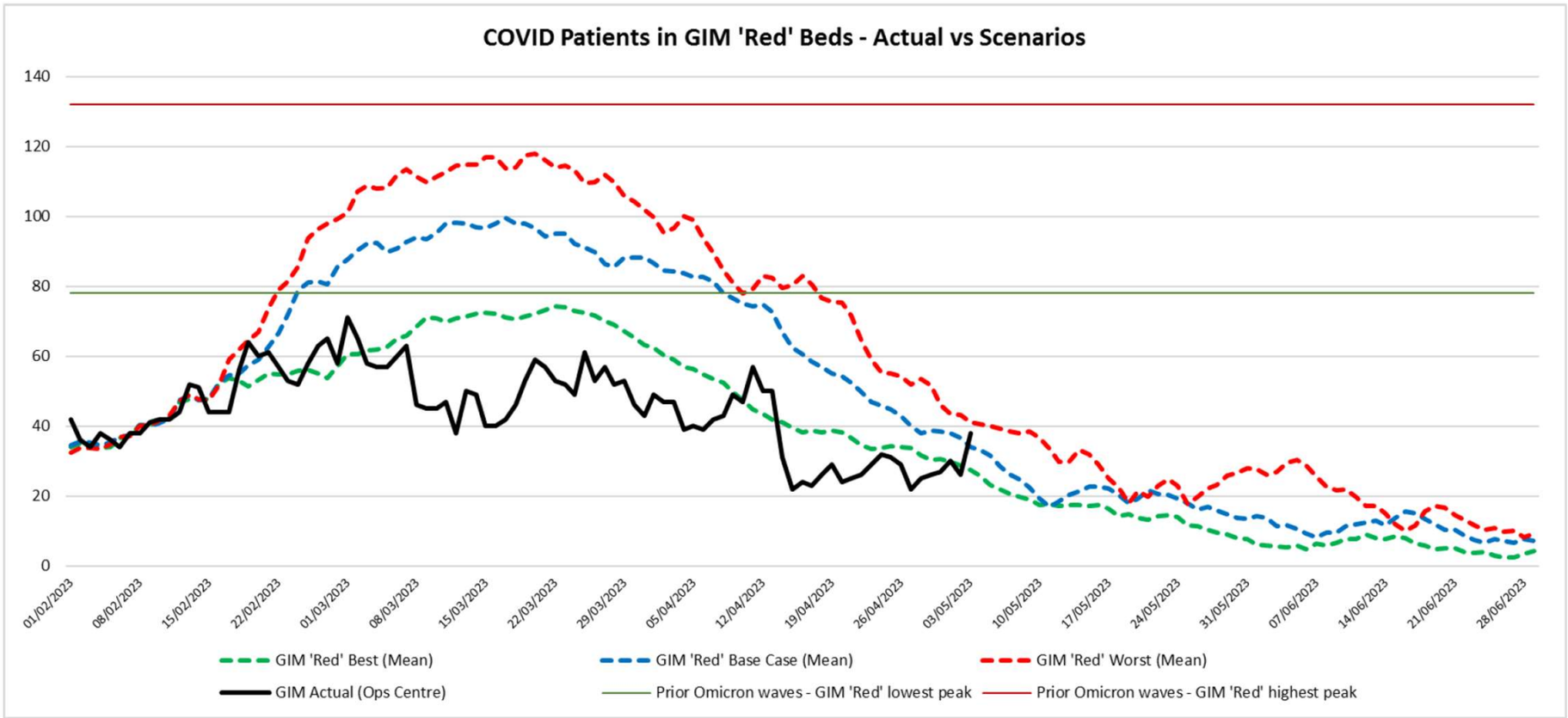
Simulation

Simulate the model with the given parameters and plot the results

Dashboard Usage

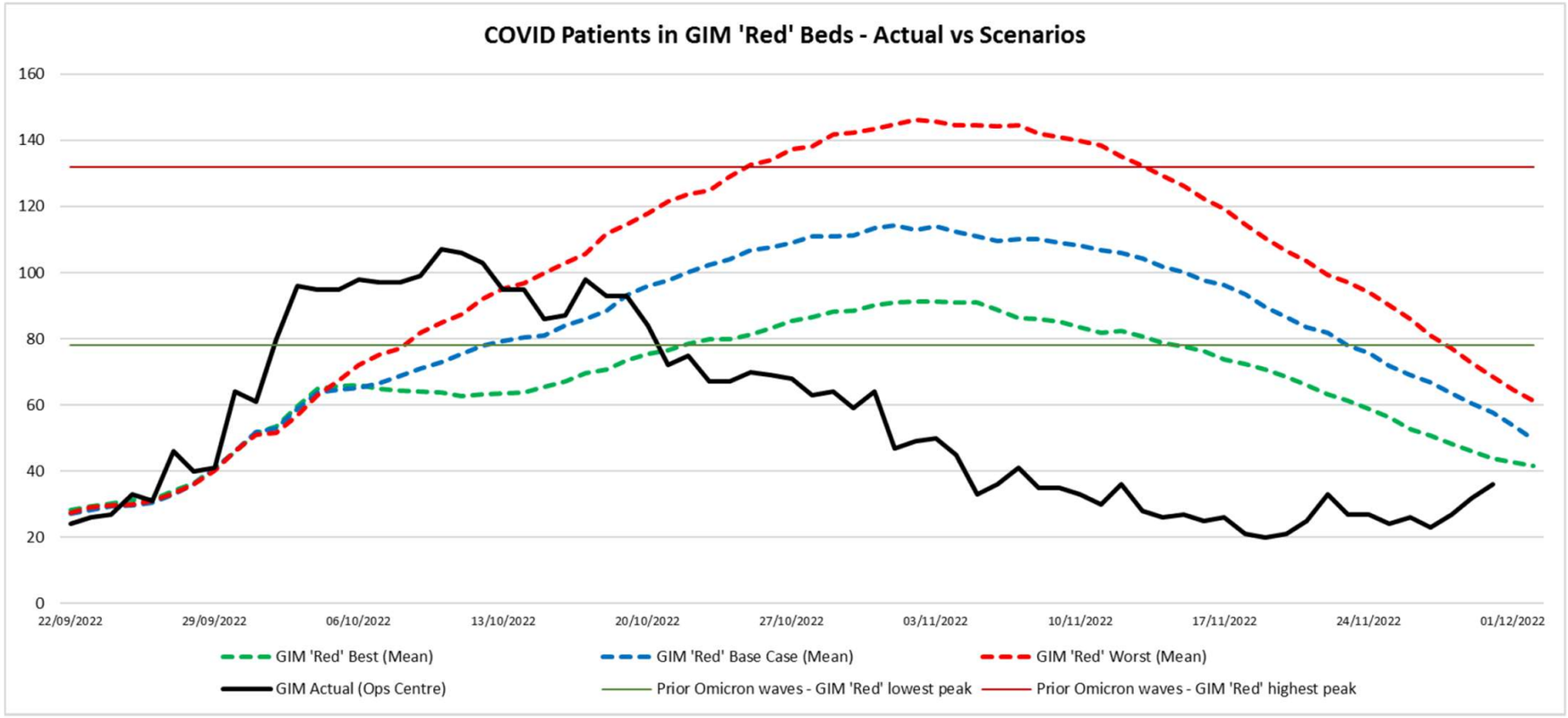
- Used to make medium-term (around 3 months) predictions on bed demand at Addenbrooke's Hospital
  - Complimented by an Early Warning Tool, which focused on short-term forecasting (up to 3 weeks)
- Typically, three scenarios (optimistic, base, "worst-case") would be generated near the start of a COVID-19 wave
  - Predict peak daily admissions and wave duration
  - Patient length-of-stay and ICU admission rate/survivability estimated using the previous wave
- We would generally run 30 simulation replications to generate credible intervals, but only report the means (see next slides)

Example: COVID-19 wave starting around Feb 2023



- "Red" bed occupancy close to (but slightly below) projected "best-case" scenario

Example: COVID-19 wave starting around Sept 2022



- Bed occupancy peak turned out to be much earlier than projected, with quick decline
- Demonstrates difficultly of making accurate predictions, especially early in a wave
- Long-term predictions should be complimented with short-term "nowcasting"

Looking Forward

- Currently examining applying the model to flu/RSV/other respiratory illnesses
- Future Technical improvements?
  - Automatic curve fitting for previous disease waves
  - Random noise in daily arrivals should be generated separately for each simulation replication (currently all replications use the same daily arrival numbers)
  - Ability to add baseline levels of patient admissions (more flexibility in scenario modelling)
    - Statistics from recent months show consistent low level of COVID-19 admissions

Summary

- A Python simulation program was used to estimate bed occupancy by COVID-19 patients at Addenbrooke's Hospital, Cambridge, UK
- Estimates were used to plan "red/green" bed allocation (red = infection control)
- A Python dashboard was created for easy data analysis + scenario creation + simulation
- Currently planning to use dashboard for flu/RSV analysis

Thank you!

- My website: yinchi.github.io
- Email: ycc39[at]cam[dot]ac[dot]uk
- IfM website: https://www.ifm.eng.cam.ac.uk/

Extra Content

The details...

Brief overview of SimPy

- Main classes used from SimPy:
  - Environment: the simulation state and event handling loop
    - Processes are added to the simulation using Environment.process()
  - Resource: seized/released by processes
    - Since we want to know hypothetical bed occupancy under no capacity limits, we set all our resources to have infinite capacity
- Processes are Python generators
  - yield keyword passes control back to the event handling loop – e.g. when waiting to seize a resource
  - Can nest generators using yield from → create subprocesses

Main patient arrival generator

```
def arrival_generator(env: Environment):
    """Generates patients for the simulation `Environment`.

    Args:
        env (Environment): The simulation `Environment`.

    Yields:
        simpy.events.Process: Each new patient's `Process`.
    """

    # Note that new arrivals are generated at midnight each day.
    # The `new_arrival` process is responsible for setting the
    # actual admission time of each patient within the day.
    # All time units in the Environment are in days.

    for n_arrivals in env.daily_arrivals:  # for each day
        for _ in range(n_arrivals):  # for each arrival in the day
            env.process(new_arrival(env))
        yield env.timeout(1.)  # Wait until next day
```

- Number of arrivals for each day pre-generated in an Excel file
- All arrivals for each day are generated at midnight
  - The new_arrival generator holds each patient for a random amount of time
  - Arrivals are thus randomly distributed throughout the day

Launching each patient's process generator

```
def new_arrival(env: Environment):
    """Generator process for a new patient arrival.

    Args:
        env (Environment): The simulation environment.

    Yields:
        A simpy Event.
    """
    # Get the time of day of the arrival (as a random number between 0 and 1)
    arrival_offset = env.hourly_dist.rvs() / 24.0 + uniform(0.0, 1.0 / 24.0)
    yield env.timeout(arrival_offset)

    if random() < env.params.icu_rate:
        yield from icu_process(env)
    else:
        yield from gim_process(env)
```

Probability of a patient arriving within each hour is taken from historical data

Delays in SimPy are done by waiting for a timeout event

Chooses which subprocess to follow based on a probability

Brief overview of Dash

- Components are Python classes
  - Anything from a simple HTML element to an entire plot figure
  - Components can be assigned an ID (string), child components, and other properties
- Interactivity is added using callbacks

```
@app.callback(
  Input('component1', 'property1'),
  Input('component1', 'property2'),
  Output('component2', 'property3'),
  Output('component2', 'property4'),
  State('component3', 'property5')
)
def my_callback(prop1, prop2, prop5):
  #code goes here
  return prop3, prop4
```

In this example, changes to property1 or property2 trigger this callback, which changes the values of property3 and property4. The value of property5 is read, but changes to it do **not** trigger the callback.