Brunel University London

HiDALGO

Local COVID-19 Simulations — expansion to national

Derek Groen, David Bell, Arindam Saha, Anastasia Anagnostou, Imran Mahmood and many others.
Department of Computer Science

ECMA

SEAVEA

STAMINA

Before I start: about the tutorial

For the JupyterLab users: the solution was in a small patch in FabFlee.

→simply reinstall the FabFlee plugin and the issue will be fixed.

```
590
591 def vvp_validate_results(output_dir="", **kwargs):
592     """ Extract validation results (no dependencies on FabSim env). """
593
594     flee_location_local = user_config["localhost"].get(
595         "flee_location", user_config["default"].get("flee_location"))
596
597     local("export PYTHONPATH=%s:${PYTHONPATH}; export FLEE_TYPE_CHECK=False; python3 %s/flee/postprocessing/extract-validation-results.py %s "
598           "> %s/validation_results.yml"
599           % (flee_location_local, flee_location_local, output_dir, output_dir))
600
```

```
[local] export PYTHONPATH=/home/jovyan/tutorials/FabSim3/flee:${PYTHONPATH}; export FLEE_TYPE_CHECK=False; python3 /home/jovyan/tutorials/FabSim3/flee/flee/postpr
ocessing/extract-validation-results.py /home/jovyan/tutorials/FabSim3/FabSim3/results/validation_localhost_4/RUNS/syria2013 > /home/jovyan/tutorials/FabSim3/FabSi
m3/results/validation_localhost_4/RUNS/syria2013/validation_results.yml
Validation syria2013: 0.2754147407630768
/home/jovyan/tutorials/FabSim3/FabSim3/results/validation_localhost_4/RUNS/car2013
[local] export PYTHONPATH=/home/jovyan/tutorials/FabSim3/flee:${PYTHONPATH}; export FLEE_TYPE_CHECK=False; python3 /home/jovyan/tutorials/FabSim3/flee/flee/postpr
ocessing/extract-validation-results.py /home/jovyan/tutorials/FabSim3/FabSim3/results/validation_localhost_4/RUNS/car2013 > /home/jovyan/tutorials/FabSim3/FabSim
3/results/validation_localhost_4/RUNS/car2013/validation_results.yml
Validation car2013: 0.3306290601316034
Mean score: 0.39061343115424124
```

Origin story

During the initial lockdown, I tried to help the community by *reviewing* COVID models on my personal blog.

However, a few weeks in I was approached by a colleague of mine with a rather urgent request.

https://theconversation.com/six-unknown-factors-in-coronavirus-models-and-how-they-could-affect-predictions-134855
https://theconversation.com/coronavirus-techniques-from-physics-promise-better-covid-19-models-can-they-deliver-139925

COVID-19 in the local context

In March/April 2020 there was a good range of COVID models.

But almost all of them were national level.

- These models were not so helpful to answer questions for individual hospitals, e.g.
  - How many beds need to be reserved during the next pandemic wave.
  - When can non-urgent care be scaled back up again?
  - How will these circumstances change given different measures and other factors?

Flu and Coronavirus Simulator (FACS)

- Decision-makers need reliable and reproducible forecasts of the spread of COVID-19 in local regions.

So we established a local collaboration, with volunteers in the Department combined with some redirected effort from the HiDALGO project.

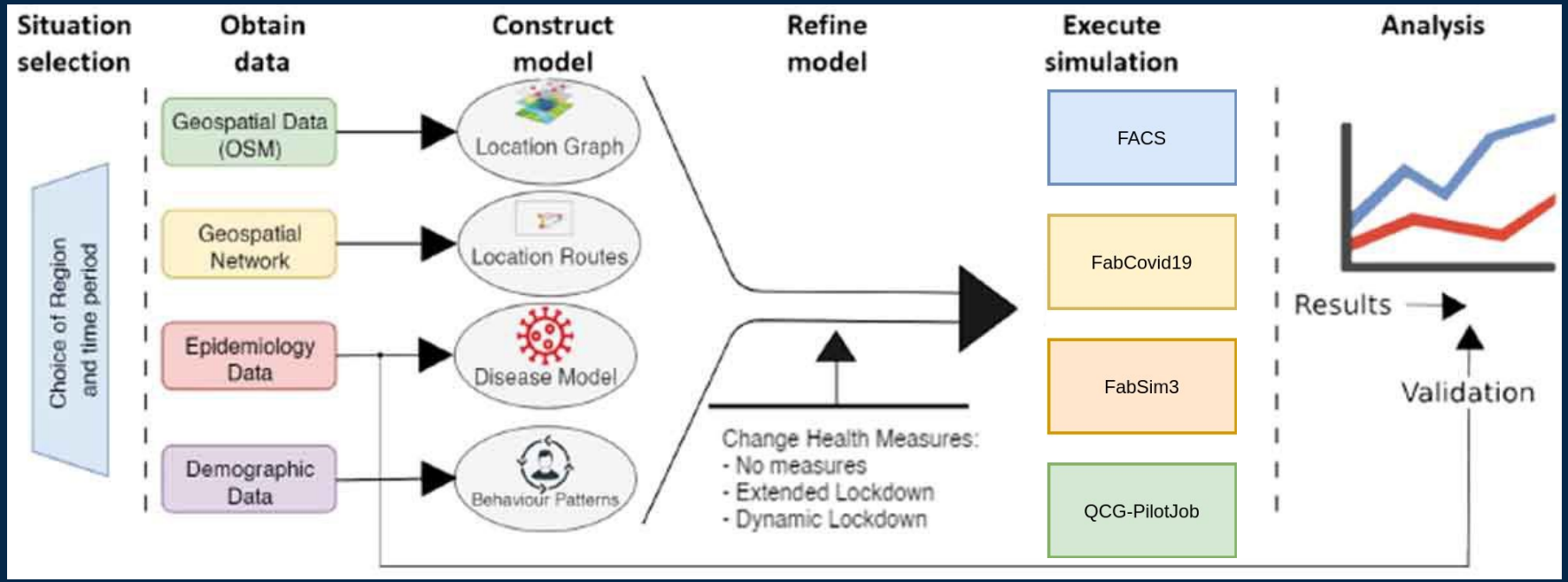
Our aim is to support:

- different lockdowns and other interventions,
- different viral strains,
- and later: different vaccination efficacies and strategies.

… on the hyperlocal sub-national level.

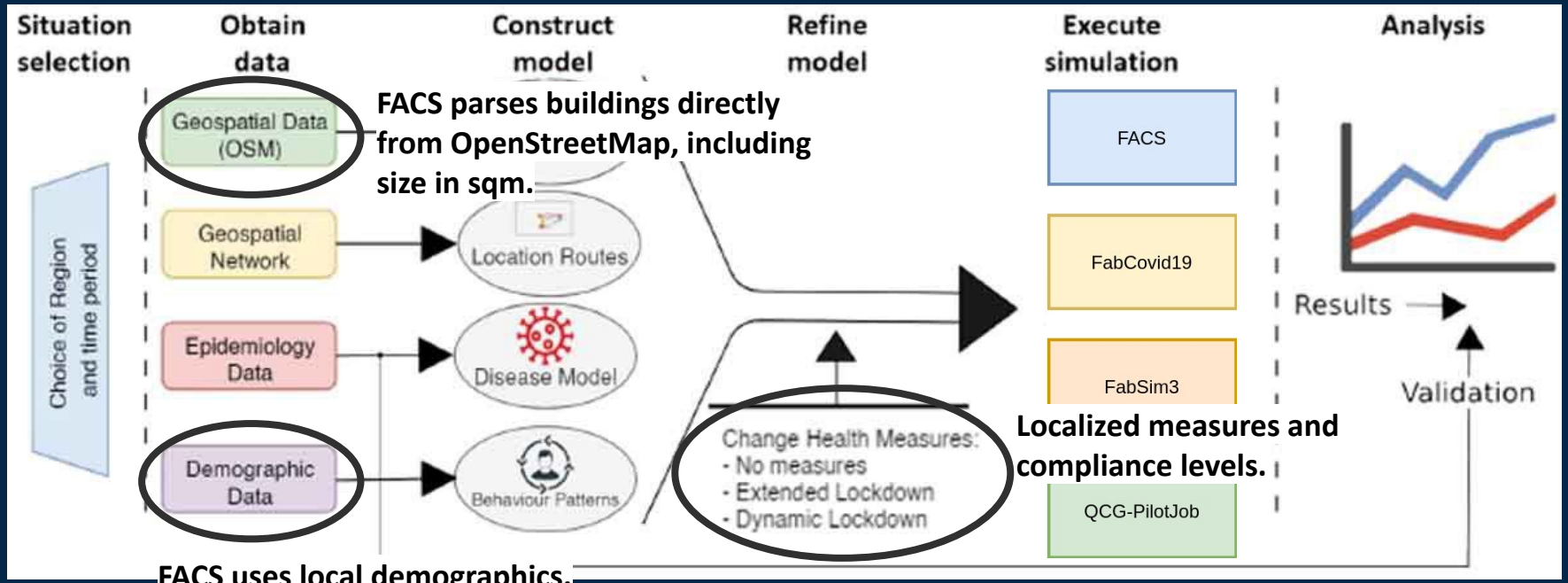
https://facs.readthedocs.io

FACS: Simulation Development Approach



Mahmood et al., Journal of Simulation, 2020.
https://doi.org/10.1080/17477778.2020.1800422

A *Localized* Simulation Development Approach

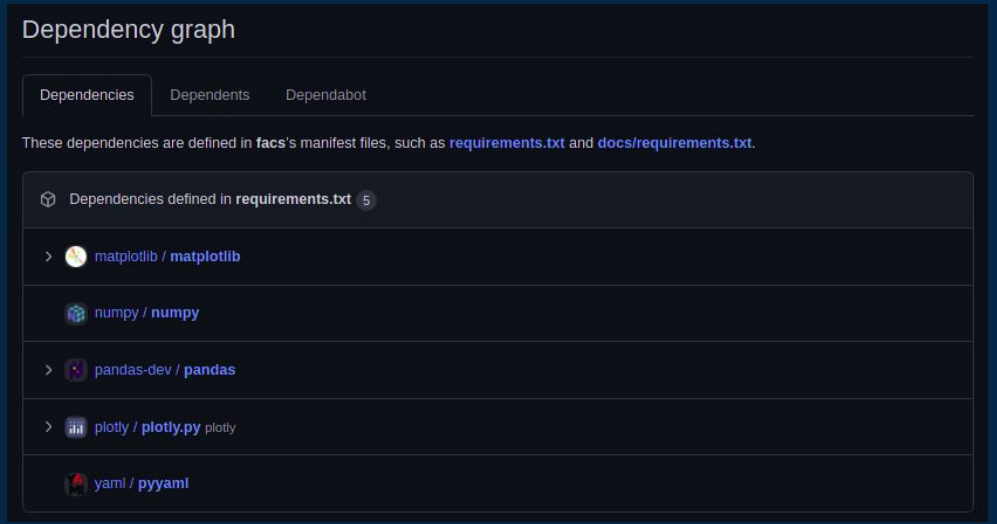


Mahmood et al., Journal of Simulation, 2020.
https://doi.org/10.1080/17477778.2020.1800422

FACS Overview

- Written in Python 3.
- Commonly used with 100-300k households.
- Runtime: 1-5 hours per sim on one core.
- Parallel FACS now in testing.

- Applications
  - W. London NHS Trusts
    (13 forecasting reports)
  - Trials in 3 other countries.
  - UK Trial with Health Security Agency
    - 2M households per run…

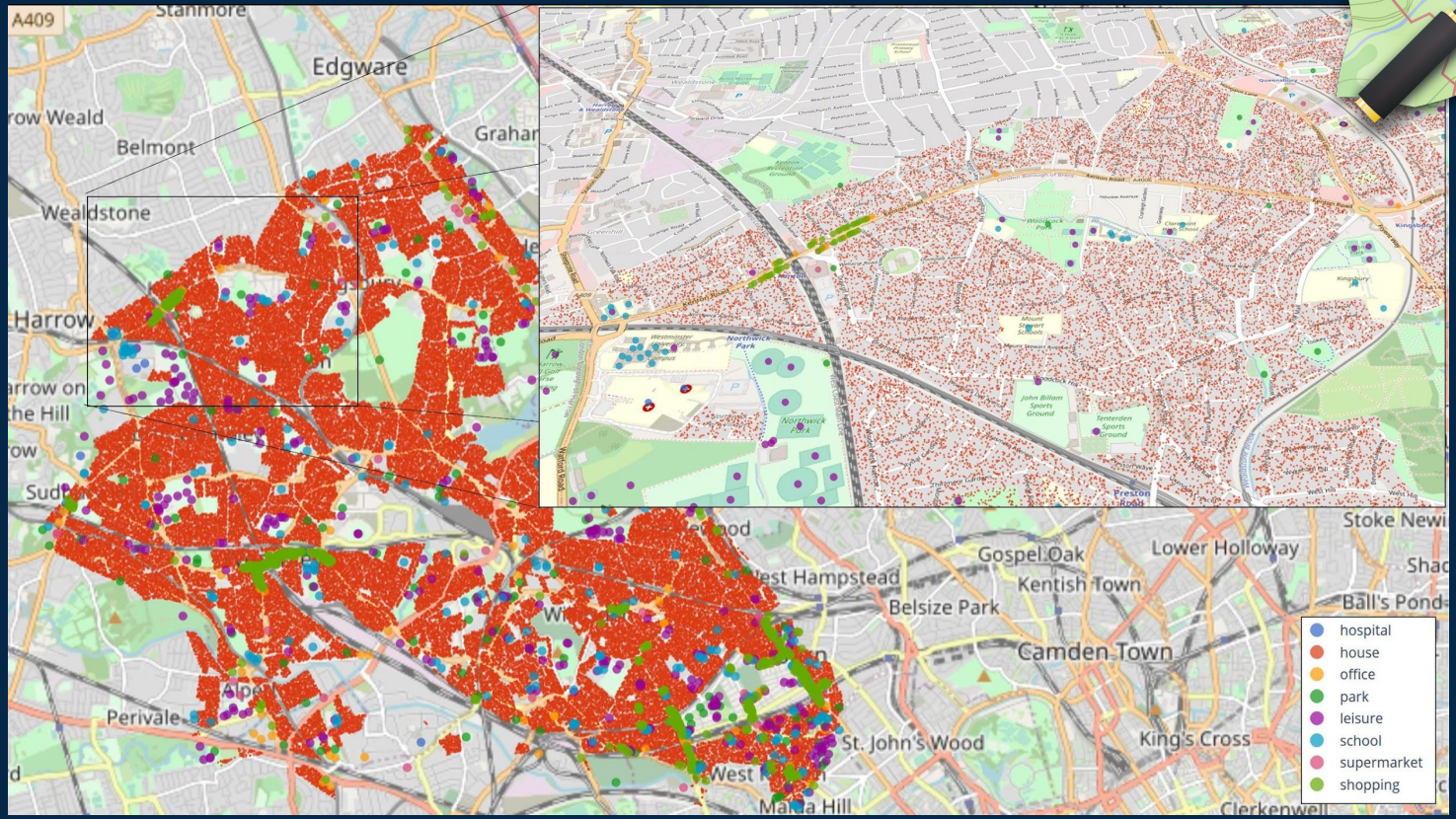


Parallelisation approach [in progress]

1. Houses are distributed across processes.
2. All other locations types are duplicated.
3. During evolution:
    a. Each house/household books the visits to the other locations, and calculates # of infected minutes per location on that rank.
    b. For each location, the # of infected minutes are aggregated across all ranks (MPI_Allreduce).
        i. 1 combined call across all locations.
    c. One smaller MPI_Allreduce is performed per timestep to aggregate the diagnostic information (e.g. # of infectious agents).

Current status: the parallel version runs and gives a speedup, but the accuracy still differs from the sequential version.

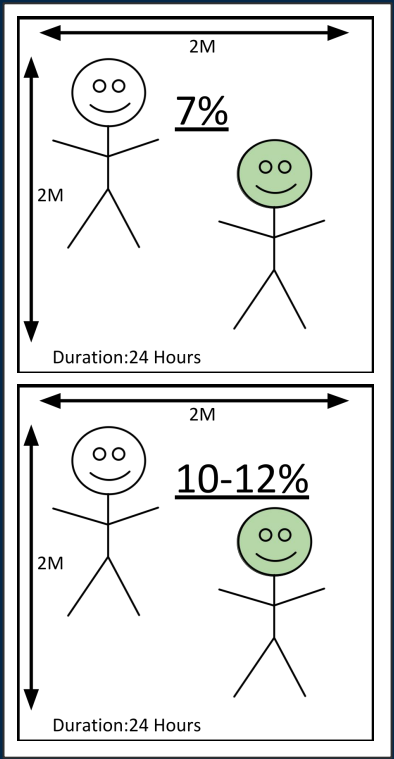
Example input: location graph



Example input: public health interventions

```yaml
16/3/2020: # ramp up of measure uptake between 11-21 March
  partial_closure:
    leisure: 0.5
  work_from_home: 0.325 # representing average of 0 at 16-3 and 0.65 at 21-3
  social_distance: 0.375 # ditto (0.75)
  mask_uptake: 0.025 # ditto (0.05)
21/3/2020: # ramp up of measure uptake between 11-21 March
  partial_closure:
    leisure: 0.5
  work_from_home: 0.65
  social_distance: 0.75
  mask_uptake: 0.05
23/3/2020:
  partial_closure:
    school: 1.0
    shopping: 0.7 #indicates full school closure, but partial_closure is used to allow keyworkers to be added.
  closure: ["leisure"]
  social_distance: 0.75
  mask_uptake: 0.05
  mask_uptake_shopping: 0.1
  work_from_home: 0.96
```

Modelling spread in local environments



P_{inf} =

Contact rate multiplier [dimensionless]
*
Infection rate [dimensionless] / airflow coefficient [dimensionless]
*
Duration of susceptible person visit [minutes] / 1 day [minutes]
*
(Ave. # infectious person visiting today [#] * area of 1 person [m^2]) /
(Area of space [m^2] * max number of persons that can fit within 4 m^2 [#])
*
Average infectious person visit duration [minutes] / minutes_opened [minutes]

FACS: sources of uncertainty

- Aleatoric uncertainty: the code is non-deterministic.
  - Plotted with error ranges, but true variability is large and non-Gaussian.
- Uncertainty in underlying assumptions, e.g. vaccination efficacy or uptake of lockdown measures.
  - Stratified in different scenarios.
- Uncertainty in expected future lockdown / release interventions.
  - Stratified in different scenarios.

FACS: Coping with uncertainty

- How do we cope with uncertainty in a world of hasty deadlines and shoestring funding?
  - **Find and adopt efficient but widely and easily applicable algorithms.**
    - Not much room for feasibility studies…
  - **Use HPC to save time spent computing.**
  - **Automation: save money spent on toiling researchers.**
- Aside from the three factors above I argue it's a spectrum between perfect UQ with an oversimplified model or Crap UQ with a hyper-realistic model?

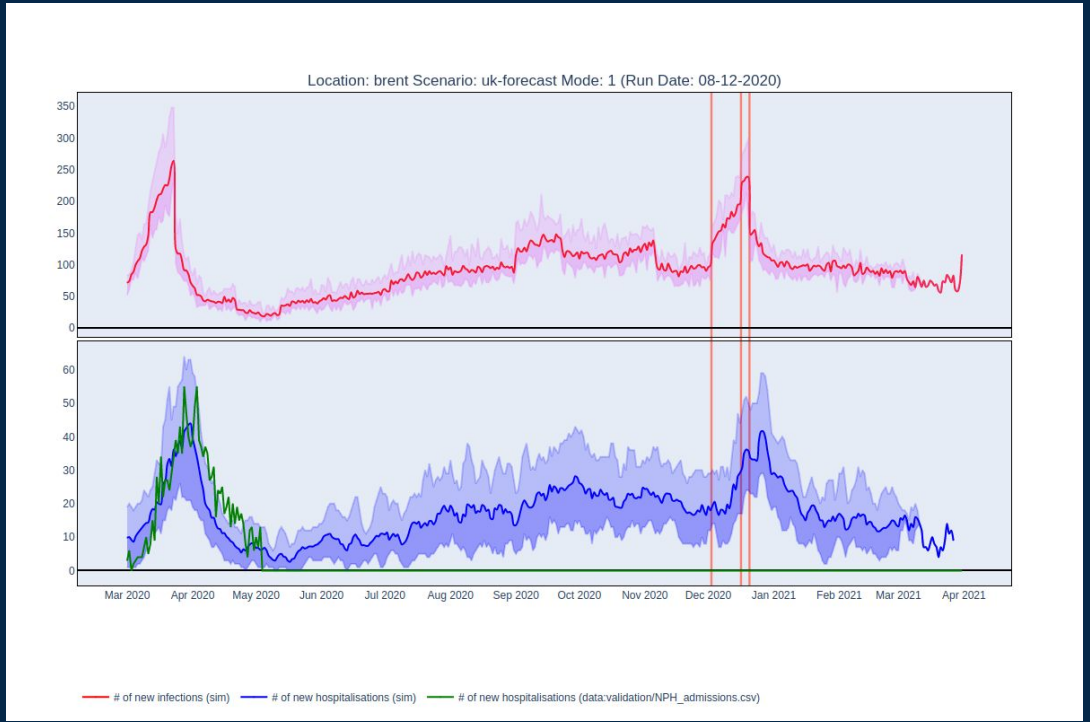
FACS: example result

Forecast for the Borough of Brent.

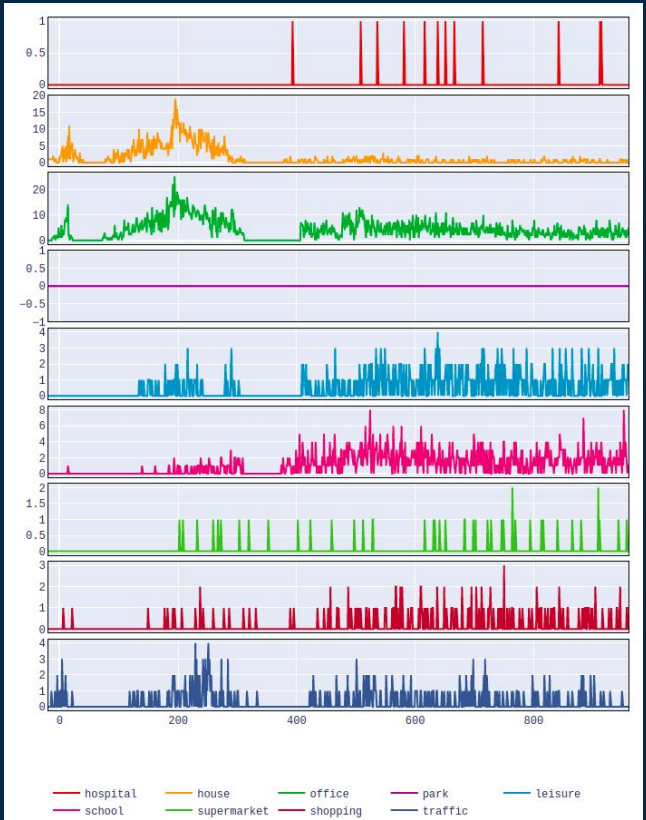
Done in December 2020.

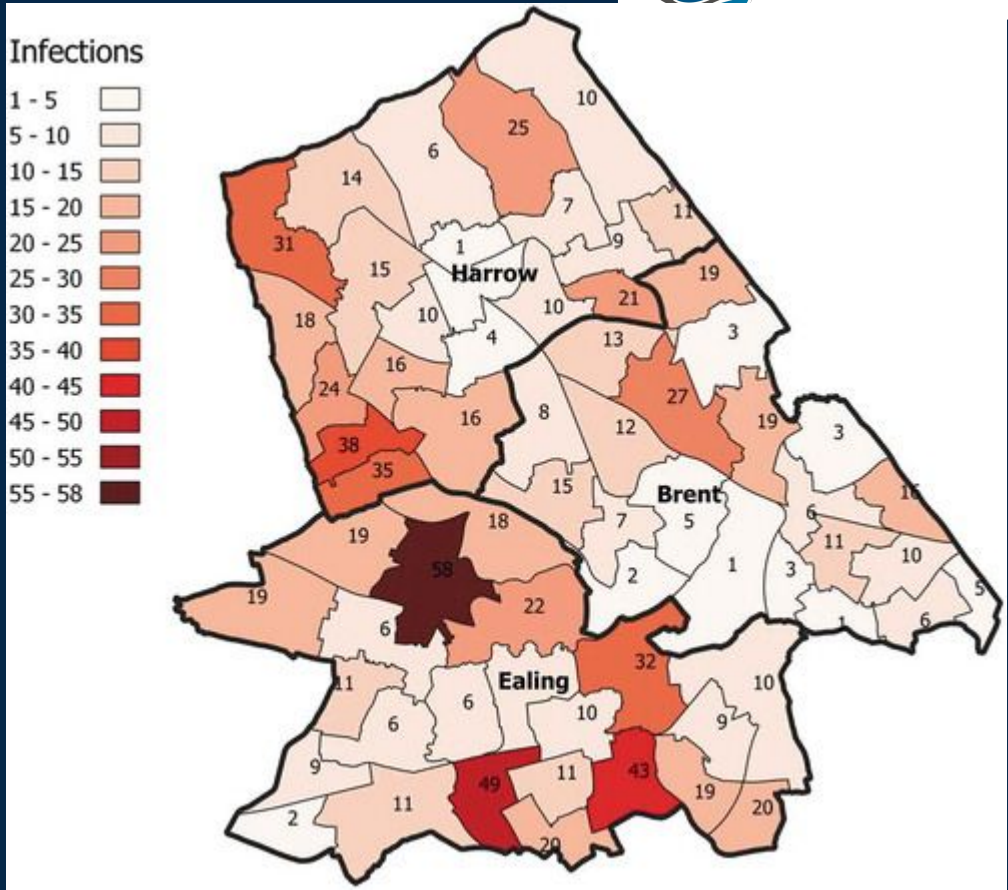
(green line is early validation data)

Vertical lines are times of key Lockdown interventions.



FACS: example results





Sensitivity Analysis

VECMAtk: a scalable verification, validation and uncertainty quantification toolkit for scientific simulations

D. Groen, H. Arabnejad, V. Jancauskas, W. N. Edeling, F. Jansson, R. A. Richardson, J. Lakhlili, L. Veen, B. Bosak, P. Kopta, D. W. Wright, N. Monnier, P. Karlshoefer, D. Suleimenova, R. Sinclair, M. Vassaux, A. Nikishova, M. Bieniek, ... See all authors

Published: 29 March 2021 https://doi.org/10.1098/rsta.2020.0221

HiDALGO

Appendix B:

FabSim3 + EasyVVUQ + QCG-PilotJob

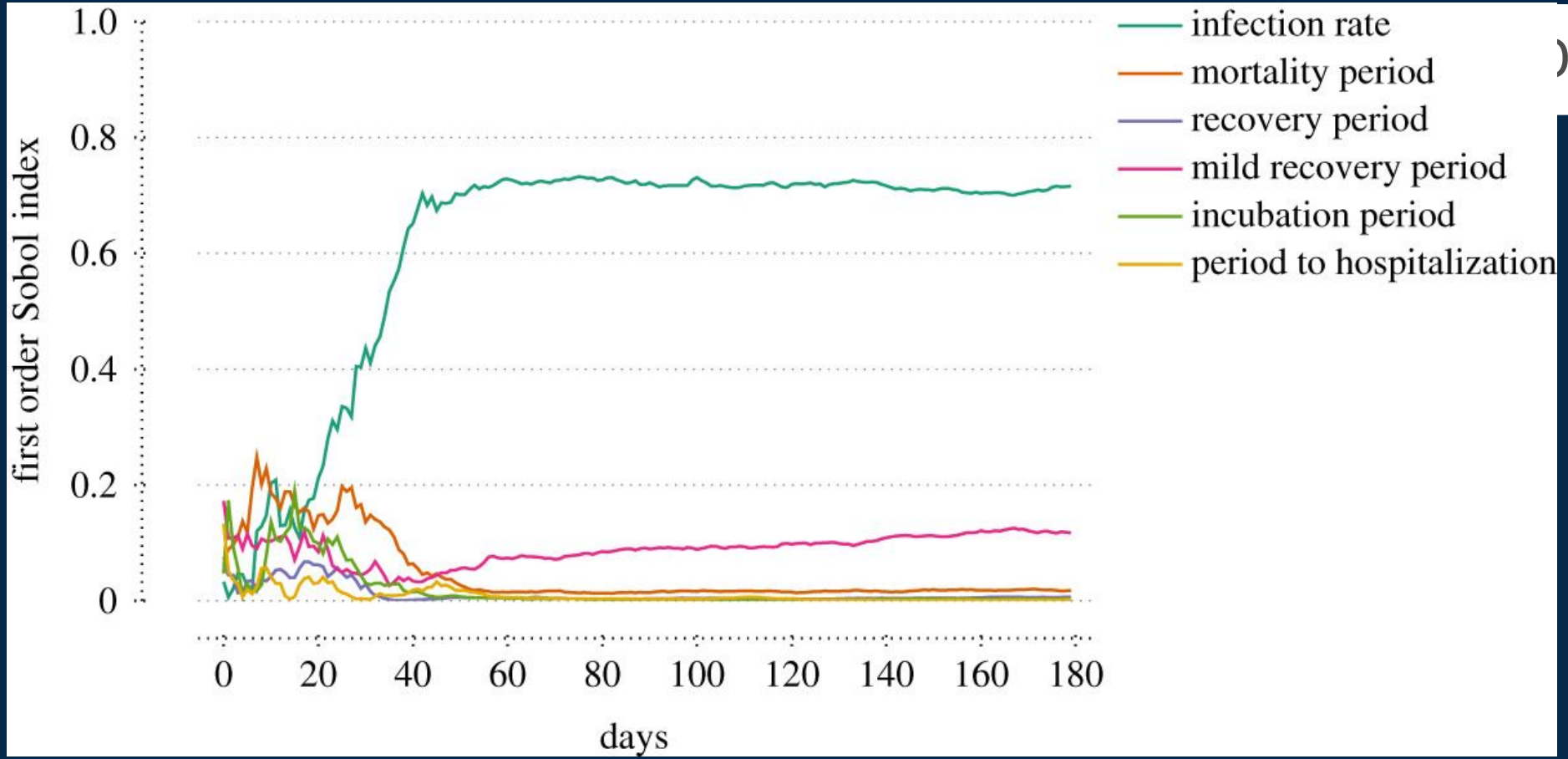
15,121 samples

Using Stochastic Collocation.

Run on the Eagle supercomputers (PSNC).

Table 3. Defining parameter space for the uncertain parameters of the Flu And Coronavirus Simulations (FACS). Collapse

parameters	type	default value	uniform range
infection rate	float	0.07	(0.0035, 0.14)
mortality period	float	8.0	(4.0, 16.0)
recovery period	float	8.0	(4.0, 16.0)
mild recovery period	float	8.05	(4.5, 12.5)
incubation period	float	3.0	(2.0, 6.0)
period to hospitalization	float	12.0	(8.0, 16.0)



The first-order Sobol indices for each of the uncertain parameters of the Flu And Coronavirus Simulations (FACS) for the London Borough of Brent.

Summary

FACS is partially validated and has been used to support strategic decisions for the NHS in Northwest London.

The code also serves a public good, as we provided occasional forecasts to the general public:
https://www.thesun.co.uk/news/12328802/second-coronavirus-wave-london-likely-almost-all-cases/
https://theconversation.com/coronavirus-how-school-closures-affect-infection-numbers-152707

Since the Journal of Simulation paper, we added:

- Non-permanent immunity.
- Vaccinations and mutations.
- Parallel execution.
- A user-friendly dashboard

We now focus on scaling up the parallel implementation, improving the VVUQ, and enabling country-wide runs!

INTERNATIONAL CONFERENCE ON COMPUTATIONAL SCIENCE

London, United Kingdom | 21-23 June, 2022

Home | Calls | Meeting Information | Registration | Organisers | Program | Previous ICCS | Help

New in ICCS

- Paper submission is open. Deadline extended to 18 February 2022 (final extension).
- Six new Keynote Speakers: Peter Coveney, Thomas Engels, Neil Ferguson, Giulia Galli, Alessandro Pandini and Rebecca Wade.
- 16 Thematic Tracks are now available.

https://www.iccs-meeting.org

Tracks include:

Wouter Edeling -> UnEQUIVOCAL

Diana Suleimenova & Derek -> Multiscale Modelling & Simulation

\+ Two SEAVEA events (one hack event & one workshop)

Thank you!

Contributors include:

David Bell, Nura Abubakar, Arindam Saha, Imran Mahmood, Hamid Arabnejad, Anastasia Anagnostou, Allan Serrano, Isabel Sassoon, Simon Taylor, Panos Louvieris, Alaa Marshan, Diana Suleimenova, Nura Abubakar, Arindam Saha.

https://facs.readthedocs.io

Mahmood et al., Journal of Simulation, 2020.
https://doi.org/10.1080/17477778.2020.1800422

Localisation: Pros and Cons

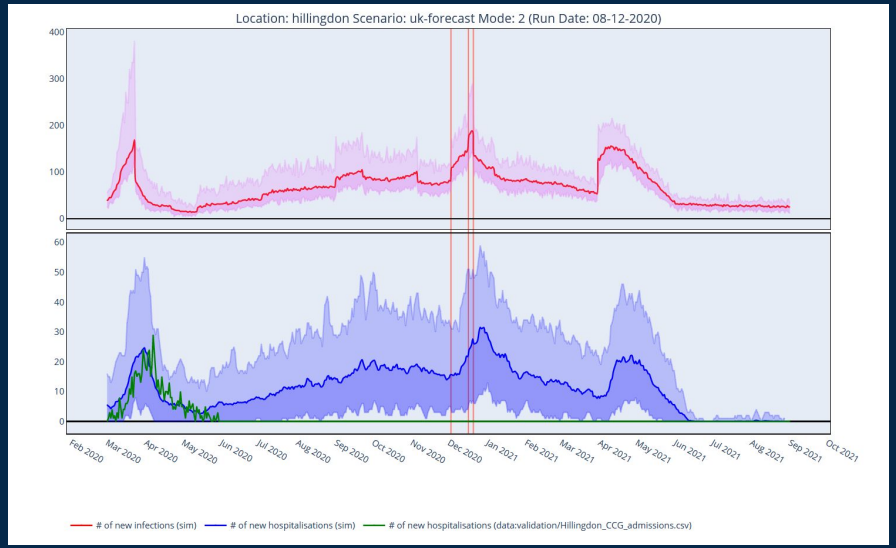
Pros:

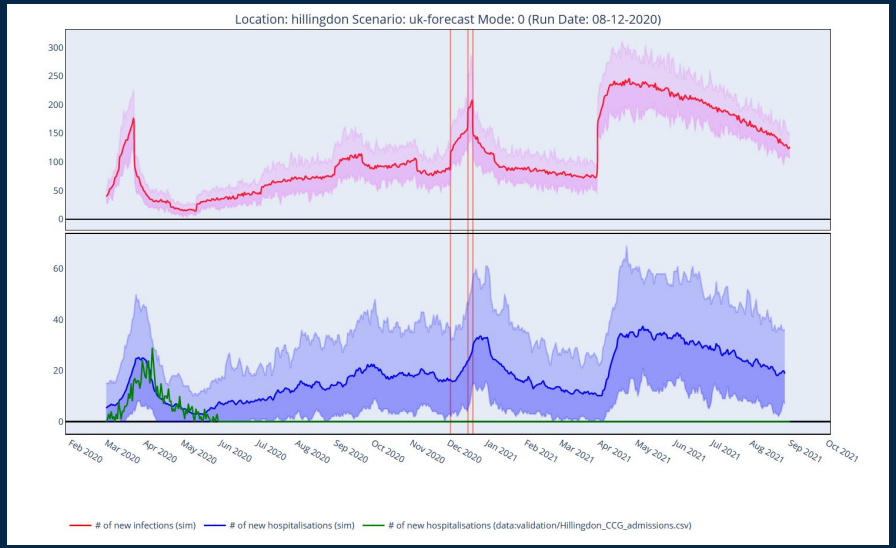
- Granularity on the building level.
  - Possible to base some interaction rules directly on observation studies (more later).
- Room for highly localized interventions.
- Easier to specifically model and validate a single hospital catchment area.
- Ability to provide highly localized and detailed outputs

Cons:

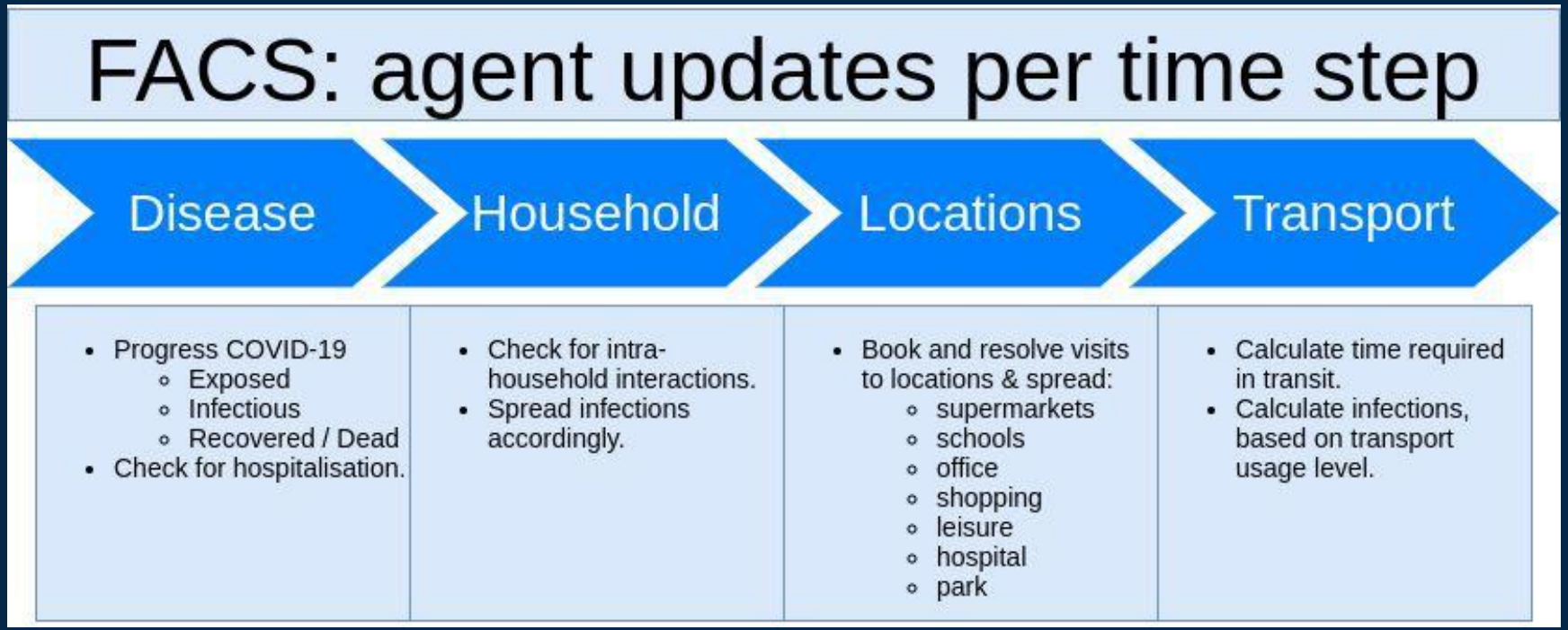
- Simulations can be more expensive.
- Some aspects are very challenging to model:
  - Interaction with outside regions.
  - Home-work traffic (which is typically less local).

FACS: example results





FACS: Key procedures per time step

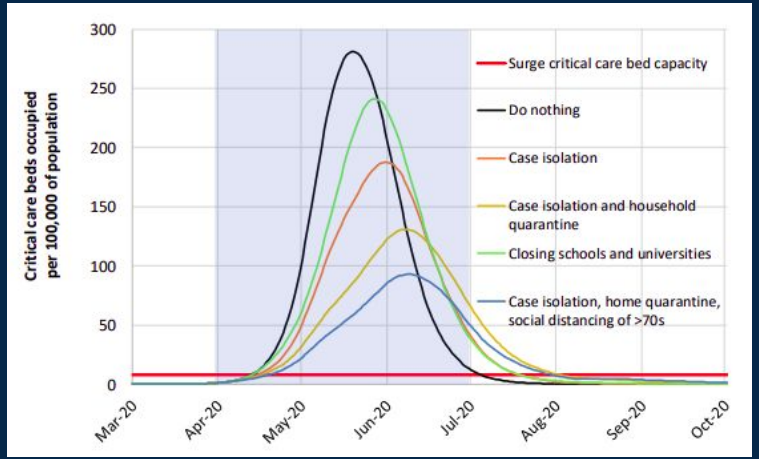


COVID-19 in the local context

The recent Covid-19 outbreak has had a tremendous impact on the world.

Modelling helps to prepare for the emergency response and mitigate the challenges involved in the front line defence against the COVID-19 outbreak.

By March 2020, national level models such as CovidSim were in the public eye:



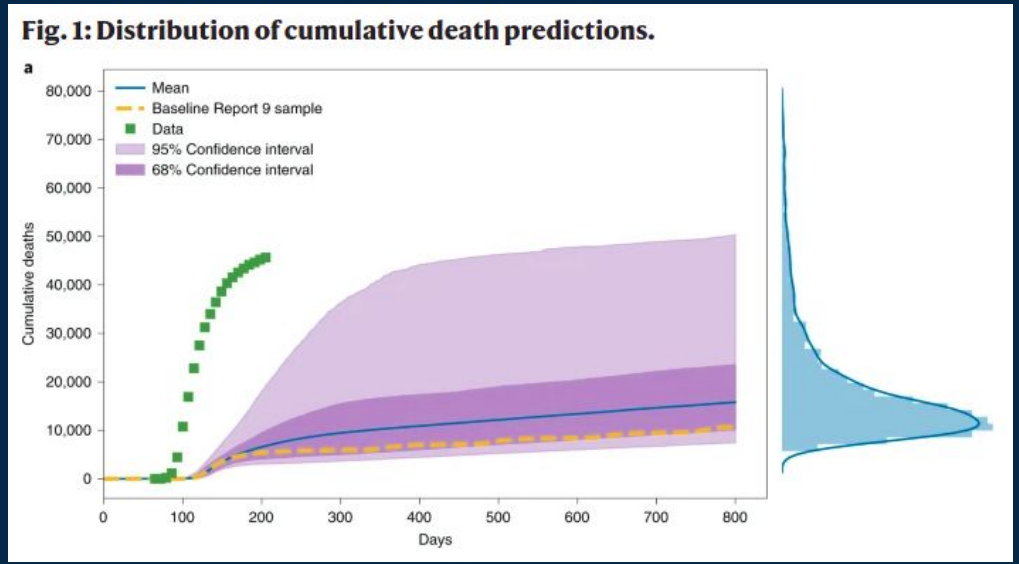
Source: Imperial Report 9 / CovidSim

Related models: CovidSim

Agent-based model with stylized location graph that approximates the UK.

Instrumental in lockdown-decision making in 2020.

Widely used for the national context, less informative locally.



Edeling, W., Arabnejad, H., Sinclair, R. *et al.* The impact of uncertainty on predictions of the CovidSim epidemiological code. *Nat Comput Sci* 1, 128–135 (2021). https://doi.org/10.1038/s43588-021-00028-9

Related models: CityGraph

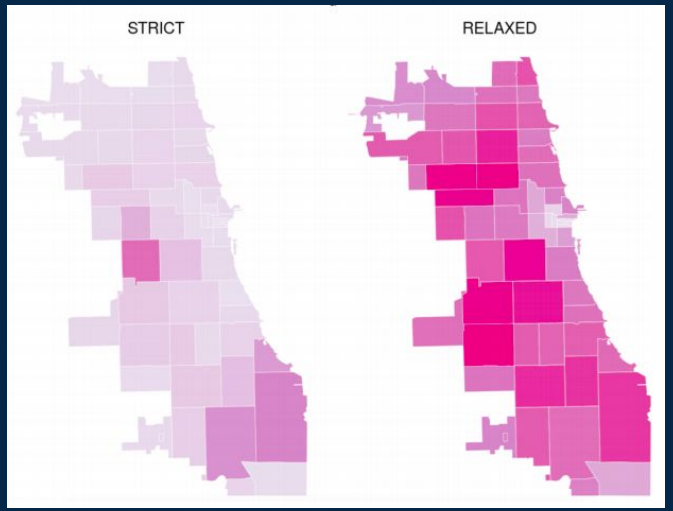
Developed since early 2020, but kept out of the public eye for an extended period.

Focus on very high detail, single city scope.

C++ code, parallelized.

Not yet clear how quickly new cities can be modelled.

SC2021 Gordon Bell Prize Preprint now available:
https://www.mcs.anl.gov/~wozniak/papers/GB_2021.pdf



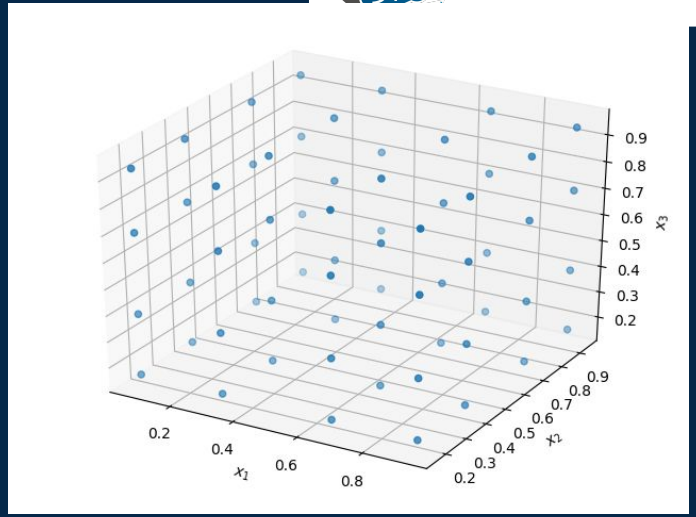
Sensitivity analysis: a few basics

- Goal: to assess how sensitivity key outputs are to changes in assumptions:
  - Or: how sensitive key *quantities of interest (QoIs)* are to modifications in specific input parameters.
- Method:
  - Run a representative sample of simulations with different input parameters.
    - E.g. using Monte-Carlo, Stochastic Collocation or Polynomial Chaos Expansion methods.
  - Use a method to calculate the sensitivity for each parameter, and combination of parameters.
    - E.g. using Sobol's sensitivity analysis.

Sobol', I.Y.M., 1990. On sensitivity estimation for nonlinear mathematical models. *Matematicheskoe modelirovanie*, 2(1), pp.112-118.

Automated Sensitivity Analysis

- We use the VECMA toolkit (http://www.vecma-toolkit.eu), and Sobol's sensitivity analysis to determine how much each assumption matters to the final result.
- This needs a lot of simulations, so we use supercomputers (e.g., Eagle/Altair [Poznan], Hazel Hen/Hawk [Stuttgart], SuperMUC(-NG) [Garching]).



Relation (Mali run)	Value
camp move chance (def: 0.001)	0.88
conflict move chance (def: 1.0)	0.60
default move chance (def:0.3)	0.45
Combinations of assumptions	<0.004

Sensitivity-driven simulation development (SD2)

- Automatically analysed the sensitivity of FLEE's output metrics to the input parameters/assumptions across four African conflict simulations.
- Draw samples using stochastic collocation and Sobol's method.
- Identified pivotal assumptions that dominate the validation results.
- Refined FLEE based on our analysis, released a new algorithm rule set FLEE 2.0 and analysed the sensitivity of input parameters across four African countries.

		FLEE				FLEE 2.0			
		Mali	Burundi	South Sudan	CAR	Mali	Burundi	South Sudan	CAR
Input parameters	max_move_speed (0)	**0.2249**	0.0485	**0.1788**	**0.1617**	**0.1367**	**0.0556**	**0.1326**	**0.0837**
	camp_move_chance (1)	**0.3427**	**0.4404**	**0.0766**	**0.1945**	**0.3356**	**0.7242**	0.0261	**0.1524**
	conflict_move_chance (2)	**0.0739**	0.0233	0.0477	**0.1869**	**0.2133**	0.0482	**0.1968**	**0.4591**
	default_move_chance (3)	**0.2273**	**0.4121**	**0.4891**	**0.2503**	**0.0929**	0.0447	**0.1619**	0.0476
	camp_weight (4)	0.0396	**0.0526**	**0.0835**	**0.0758**	**0.0667**	**0.0829**	**0.1558**	**0.0811**
	conflict_weight (5)	0.0288	0.0022	0.005	0.0315	**0.0835**	0.0071	0.0066	0.0444

Suleimenova, D., Arabnejad, H., Edeling, W.N. and Groen, D., 2021. Sensitivity-driven simulation development: a case study in forced migration. Philosophical Transactions of the Royal Society A, 379(2197), p.20200077.

Verification, Validation & Uncertainty Quantification

It is critical that we can trust the results of supercomputer simulations, because simulations are increasingly relied on for important matters that affect our lives.

It is therefore important that:

- any simulation adheres to its underlying conceptual (or mathematical) model description [verification].
- any simulation produces results that closely match with observations in the real world [validation].
- any simulation result has a clearly and systematically quantified range of uncertainty [uncertainty quantification].

Verification and Validation Patterns

VVP #1: Stable Intermediate Forms [verification]

VVP #2: Level of Refinement [verification]

VVP #3: Ensemble Output Validation [validation]

VVP #4: QoI (Quantities of Interest) Distribution Comparison [validation]

VVP #3: Ensemble Output Validation

- The aim of Flee is to be rapidly re-applicable to a wide range of conflicts.
- To evidence that this is possible, we need to validate the code concurrently against data from a range of different scenarios.
- Using VVP #3, we are able to flexibly build automated ensemble validation routines, and move from ad-hoc validation to systematic and recurring validation testing.
- Four scenarios currently, but we are planning to add 2 more (Ethiopia and Syria) in the coming month.

Code	Burundi	CAR	Mali	South Sudan	Average
1.0	0.258	0.254	0.46	0.523	0.38
2.0	0.257	0.236	0.285	0.525	0.326

VVP #3: Ensemble Output Validation

VVP is defined in Python3 code, but without any dependencies.

This allows the pattern to be easily inserted into other software components.

```python
@task
@load_plugin_env_vars("FabFlee")
def validate_flee_output(results_dir):
    """
    Goes through all the output directories and calculates the validation
    scores.
    """
    vvp.ensemble_vvp("{}/{}/RUNS".format(env.local_results, results_dir),
                     vvp_validate_results,
                     make_vvp_mean)
```

```python
def ensemble_vvp(results_dirs, sample_testing_function,
                 aggregation_function, **kwargs):
    """
    Goes through all the output directories and calculates the scores.
    Arguments:
    - results_dirs: list of result dirs to analyse.
    - sample_testing_function: analysis/validation/verification function to be
    performed on each subdirectory of the results_dirs.
    - aggregation_function: function to combine all results
    - **kwargs: custom parameters. The 'items' parameter can be used to give
    explicit ordering of the various subdirectories.

    return : ensemble_vvp_results (dict)

    Authors: Derek Groen, Wouter Edeling, and Hamid Arabnejad
    """

    ensemble_vvp_results = {}

    # if a single result_dir is specified, still add it to a list
    if type(results_dirs) == str:
        tmp = []
        tmp.append(results_dirs)
        results_dirs = tmp

    for results_dir in results_dirs:

        scores = []

        # use user-specified sample directories if specified,
        # otherwise look for uq results in all directories in results_dir
        if 'items' in kwargs:
            items = kwargs['items']
        else:
            items = os.listdir("{}".format(results_dir))

        for item in items:
            if os.path.isdir(os.path.join(results_dir, item)):
                print(os.path.join(results_dir, item))
                scores.append(sample_testing_function(
                    os.path.join(results_dir, item), **kwargs))

        scores_aggregation = aggregation_function(scores, **kwargs)

        # update return results dict
        ensemble_vvp_results.update({results_dir: {}})

        ensemble_vvp_results[results_dir].update({
            'scores': scores,
            'scores_aggregation': scores_aggregation
        })

    return ensemble_vvp_results
```

Verified Exascale Computing for Multiscale Applications

- Horizon 2020 EU project (FET-HPC)
  - http://www.vecma.eu
- Aim: Establish a toolkit for VVUQ for (multiscale) applications that require large scale supercomputers.
  - -> The VECMA Toolkit.
- Budget: €4M
- Duration: June 2018 - December 2021.
- Next major toolkit release date: June 2021

HiDALGO

UCL

Brunel University London

CWI Centrum Wiskunde & Informatica

THE STATE UNIVERSITY OF NEW JERSEY RUTGERS

Bull atos technologies

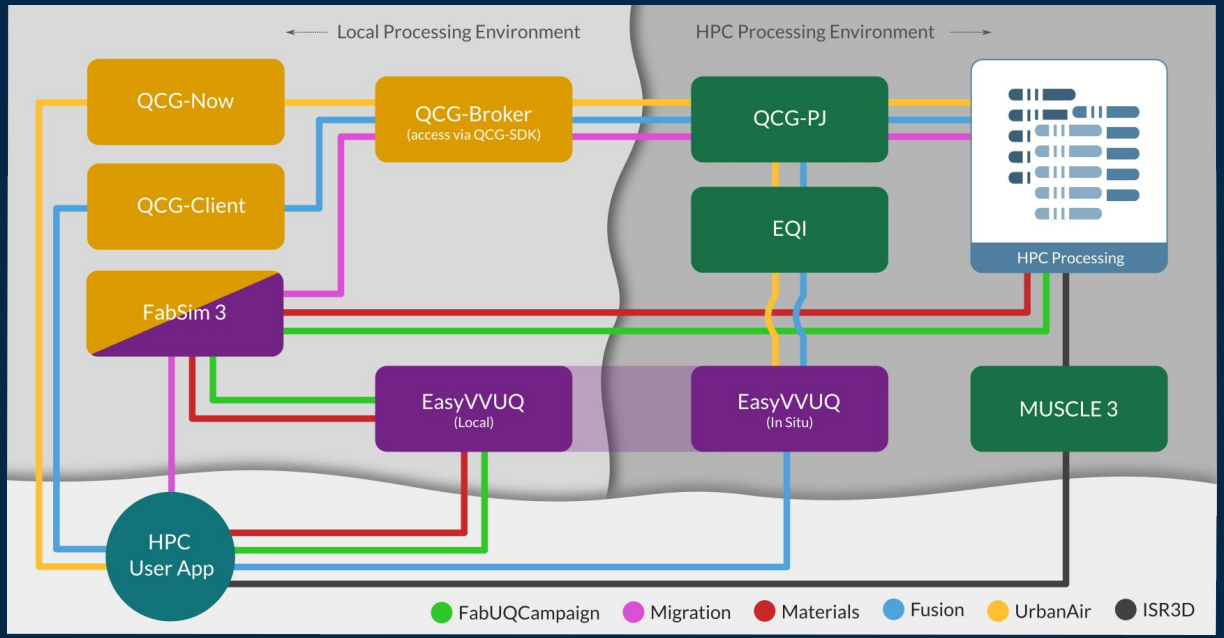
CBK

IPP

lrz Leibniz-Rechenzentrum der Bayerischen Akademie der Wissenschaften

PSNC

The VECMA toolkit



1. Support the creation [illegible] for VVUQ and sen[illegible]
2. Enable *UQP/VVP*-based application execution using a variety of infrastructures.
3. Facilitate curation and automation of complex applications.
4. Provide the ability to flexible schedule large numbers of jobs on remote resources.
5. Be *used*.

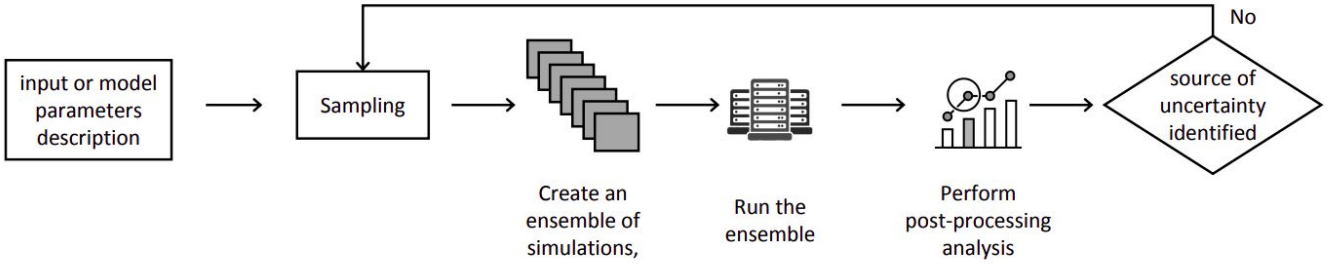
HiDALGO

- **EasyVVUQ**
  - “I need error bars.”
- **MUSCLE3**
  - “I want to couple codes, and propagate uncertainty through the coupling.”
- **FabSim3**
  - “I don’t want to do anything tedious to get all this complicated stuff working.”
- **QCG-Client & Broker**
  - “I want to manage different job types, including complex workflows using supercomputers.”
- **QCG-PilotJob & EQI**
  - “I need to run more jobs...say, *perhaps a hundred million*?”
- **QCG-Now**
  - “I want to use a supercomputer...now!”
- **EasySurrogate**
  - “I want to run my models much more cheaply.”

EasyVVUQ introduction

- EasyVVUQ aims to make it as easy as possible to implement advanced techniques for uncertainty quantification for existing application codes (or workflows).
- Primary focus on non-intrusive UQ techniques, where many model instances are run to quantify uncertainties and analyse parameter sensitivities.

https://easyvvuq.readthedocs.io



FabSim3

HiDALGO

- User-developer oriented automation/curation toolkit
  - Aim: reduce human effort required to create/modify/repurpose/salvage complex computational workflows.
- Uses generic pattern code (reusable in other tools) for UQ and V&V.

- Supports range of backends:
  - Localhost.
  - Range of supercomputers.
  - QCG broker & Pilot Jobs.
- FabSim3 is generic, but plugins provide application-specific advantages.

FabSim3 relies on bash one-liners, e.g.:

```
fabsim localhost run_amazingly_complex_app
fabsim eagle validate_flee:mali,cores=24,replicas=5
fabsim localhost install_plugin:FabUQCampaign
```

https://fabsim3.readthedocs.io

Groen et al., *Comp. Phys. Comm. (2016) vol. 207, pp. 375--385.*

QCG PILOT JOB

- Dynamic scheduling and running of tasks inside a single queuing system allocation (e.g. SLURM)
- Support for task dependencies and iterations
- Different modes of usage:
  - static (description in JSON file)
  - dynamic (python API and remote API)
- Resuming mechanism
- Good scalability thanks to partitioning mechanism.

https://qcg-pilotjob.readthedocs.io/

MUSCLE 3

- Aim: make creating coupled multiscale simulations easy, and to then enable efficient uncertainty quantification of such models using advanced semi-intrusive algorithms.
- Development is funded through the e-MUSC project.

https://muscle3.readthedocs.io