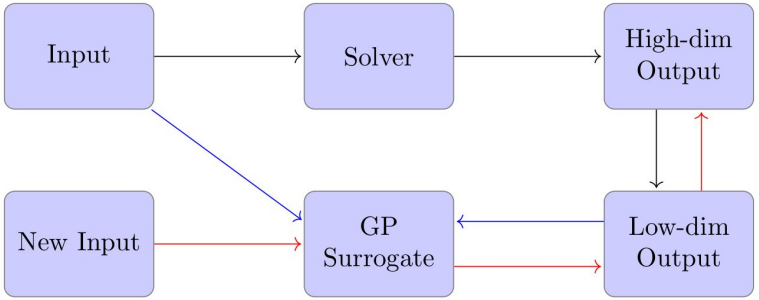
Advanced Quantification of Uncertainties In Fusion modelling at the Exascale with model order Reduction
(**AQUIFER**)
UCL

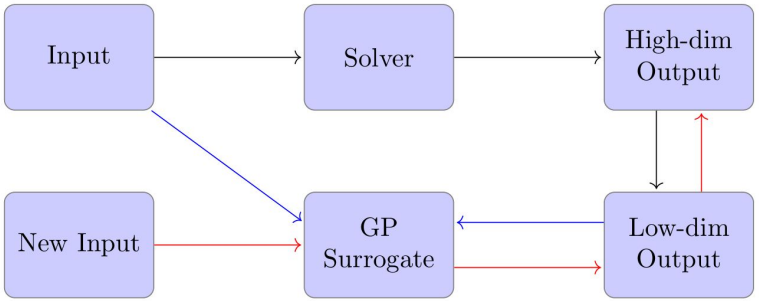
Future steps

Surrogates plus Model Order Reduction



- The OPE could be used as surrogates for further tasks like sensitivity analysis and uncertainty quantification
- Develop OPEs to have a linked structures for describing some complex coupled systems.
- Generalize the OPE to emulate the physical profiles over some manifold like tori by designing the regressors and GPs.

Surrogates plus Model Order Reduction



- MOR approaches to be tailored to the computational implementation to further reduce data flows, as well as frequencies and latencies of I/O exchanges across sub-models.
- MOR for Surrogate models of feed-forward suites of models, which have analytical expressions

  Ming, D., & Guillas, S. (SIAM/ASA JUQ 2021, code https://github.com/mingdeyu/DGP).
- Surrogates for PIC
- Physics-aware and Deep GPs

Bayesian calibration against observations

Leeds/UCL collaboration: velocity fields

- Informed discrepancy
- Only velocity (in water) for now, but what about combination of modelling with plasma physics, etc.
- Value of experiments?

- What variables?
  - Statistics of turbulence flow.
  - Energy,
  - heat transport (order 1 question!).
  - TKE?
  - Velocity?

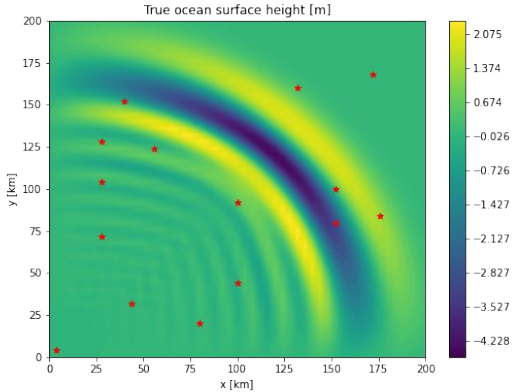
- What science/engineering design question?

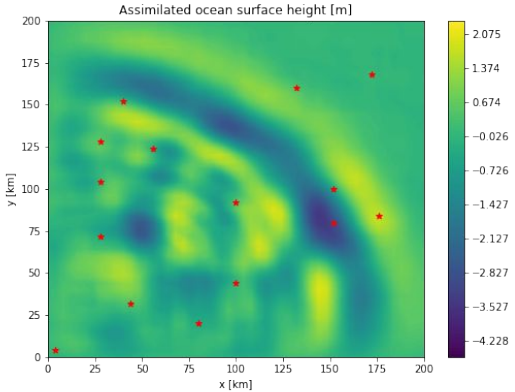
Data Assimilation

1. Particle Filters:

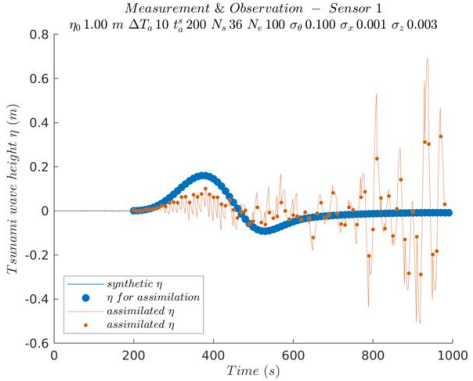
ParticleDA.jl

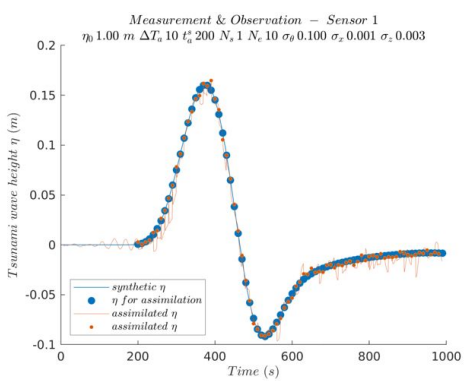
https://github.com/Team-RADDISH





2. Geometrically driven EnKF for joint parameter and state estimation





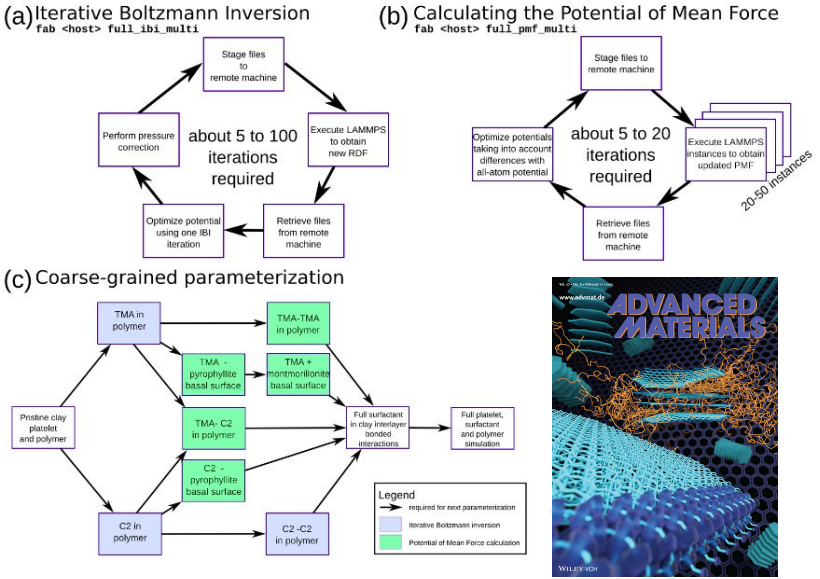
Software implementation. HPC deployment

- Perform UQ on NEPTUNE single-scale models (BOUT++, Nektar++ and EPOCH)
- Integrated approaches: multi-physics calibration/emulation/DA and semi-intrusive UQ

- Multi-Output Gaussian Process Emulator (MOGP) deployment at scale
- Data Assimilation (DA) that will specifically address the needs of the fusion modelling community. These are mainly around scalability, nonlinearity and non-Gaussianity. Number of particles/precision/frequency.
- Use of FPGAs for UQ? STFC FPGA test-bed
- UQ toolkit scaling to production simulations at the exascale.

Facilitate Remote Job Management

- Submit remote jobs and workflows using a one-liner command.
  - Single jobs.
  - Replicated jobs (job array with identical inputs).
  - Ensemble jobs (job array with different inputs).
  - Ensemble & Replicated jobs.
- Cancel jobs or wait until their completion
- Commands can be combined to create iterative and/or dynamic workflows.
  - Remote executions can be combined with local processing activities.



Suter, J.L., Groen, D. and Coveney, P.V., 2015. *Advanced Materials*, *27*(6), pp.966-984.

Facilitate Remote Job Management

FabSim3 supports three mechanisms:

- Manage jobs through QCG-PilotJob.
- Interact with scheduler directly.
- Run interactively on the remote node (localhost).

FabSim3/fabsim/deploy/machines.yml

```
eagle_vecma: # machine name
  remote: "eagle.man.poznan.pl"
  home_path_template: "/home/$username"
  budget: "vecma2020"
  corespernode: 28
  partition_name: "standard"
  job_dispatch: "sbatch"
  cancel_job_command: "scancel $jobID"
  job_info_command : "squeue --jobs $jobID"
  stat: "squeue -u $username"
  ...
```

(a) machines.yml

FabSim3/fabsim/deploy/machines_user.yml

```
eagle_vecma: # machine name
  username: "test"
  # format days-hours:minutes:seconds
  job_wall_time : "0-1:40:00"
  partition_name: "fast"
  ...
```

(b) machines_user.yml

FabSim3/fabsim/deploy/templates/slurm-eagle

```
#!/bin/bash

#SBATCH --nodes=$nodes
#SBATCH --tasks-per-node=$corespernode
#SBATCH --time=$job_wall_time
#SBATCH --account=$budget
#SBATCH --output=$job_results/JobID.output
#SBATCH --error=$job_results/JobID.error
#SBATCH --mem=$memory
#SBATCH --partition=$partition_name
```

(c) Sample SLURM based scheduler job script

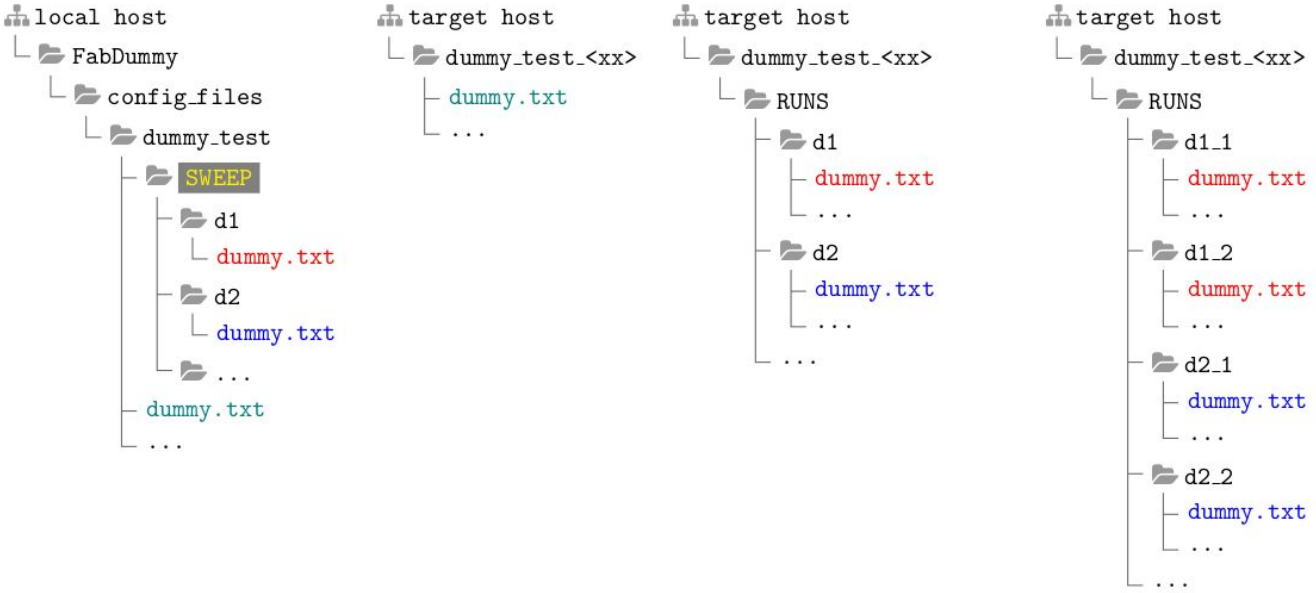
submitted job script

```
#!/bin/bash

#SBATCH --nodes=1
#SBATCH --tasks-per-node=28
#SBATCH --time=0-1:40:00
#SBATCH --account=vecma2020
#SBATCH --output=/home/test/JobID.output
#SBATCH --error=/home/test/JobID.error
#SBATCH --mem=6000
#SBATCH --partition=fast
```

(d) generated output script

Curate local and remote environment



(a)
plugin configuration

(b)
```
>_ fabsim <target host>
dummy:dummy_test
```

(c)
```
>_ fabsim <target host>
dummy_ensemble:dummy_test
```

(d)
```
>_ fabsim <target host>
dummy_ensemble:dummy_test
,replicas=2
```

Curate local and remote environment: Results directory

- `<config>_<machine>_<cores>.sh`
  - Contains the exact runscript executed remotely
  - Good for finding wrongly defined or undefined FabSim3 variables.
- `env.log`
  - Contains all environment variables on the remote machine when the job started on the worker node.
  - Good for finding deployment issues / definition issues in the machines yml files.
- `env.yml`
  - Contains the full internal variable state of FabSim3 when the job was submitted.
  - Good for finding more subtle variable issues that did not directly manifest above.
- `stdout / stderr files`
  - Good for checking any issues arising during code execution (e.g. with input files).
- + all input files used.

Enable to use of pattern-based VVUQ (+SA)

- FabSim3 supports non-intrusive UQ+SA in two ways:
  - By linking with EasyVVUQ.
  - Using ensembles directly.
- It can also be used with MOGP and EasySurrogate to train

FabSim3 supports 3 Verification and Validation Patterns:

- Stable Intermediate Form
- Level of Refinement
- Ensemble Output Validation