Intro to Uncertainty Quantification

Inverse UQ, Markov chain Monte Carlo, High Performance Computing

Linus Seelinger

Institute for Mathematics, Heidelberg.

Inverse Problems

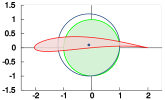
Given observations, how likely is a parameter? And how likely is a resulting quantity of interest?

$$\underbrace{y}_{\text{Observation}}$$

Inverse Problems

Given observations, how likely is a parameter? And how likely is a resulting quantity of interest?

$$\underbrace{F^{-1}(y)}_{\text{Inverse model}} \leftarrow \underbrace{y}_{\text{Observation}}$$

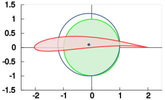


$\leftarrow$

Inverse Problems

Given observations, how likely is a parameter? And how likely is a resulting quantity of interest?

$$\underbrace{Q(F^{-1}(y))}_{\text{Quantity of interest}} \leftarrow \underbrace{F^{-1}(y)}_{\text{Inverse model}} \leftarrow \underbrace{y}_{\text{Observation}}$$

 ? $\leftarrow$ 

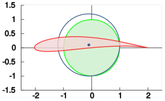
$\leftarrow$ 

Inverse Problems

Given observations, how likely is a parameter? And how likely is a resulting quantity of interest?

$$\underbrace{Q(F^{-1}(y))}_{\text{Quantity of interest}} \leftarrow \underbrace{F^{-1}(y)}_{\text{Inverse model}} \leftarrow \underbrace{y}_{\text{Observation}}$$

? $\leftarrow$



$\leftarrow$

Cannot invert F, therefore Bayesian problem!

Basic MCMC

Algorithm 1: Metropolis-Hastings

Initialize the Markov chain with a starting point $\theta^0 \in \mathbb{R}^m$.
for *Sample n = 0, ..., N-1* **do**
  Draw a correlated proposal θ' from proposal distribution $q(\theta'|\theta^n)$.

$$\alpha(\theta'|\theta^n) := \min\left\{1, \frac{\pi(\theta')q(\theta^n|\theta')}{\pi(\theta^n)q(\theta'|\theta^n)}\right\}.$$

  Draw a random number $r \in [0, 1]$.
  if $r < \alpha(\theta'_k, \theta^j_k)$ **then**
    Accept proposal: $\theta^{n+1} := \theta'$
  else
    Reject proposal: $\theta^{n+1} := \theta^n$

- Evaluate target density at finite num. of points
- Result: Markov chain with stat. distribution π

$\rightarrow$ Online Demo: MCMC

Exercise

Try online demo: `https://chi-feng.github.io/mcmc-demo/app.html`

Exercise

UM-Bridge docs, Tutorial 3: Solving UQ problems

`https://um-bridge-benchmarks.readthedocs.io`

Next: Switch to Metropolis-Hastings sampler, then try out shorter/longer chains etc.

Define a custom density to sample from (i.e. UM-Bridge model with 2D input and single output. Return log density!).

Exercise

Implement Bayesian posterior for space model:

Create artificial data: Choose some "true" $\hat{\theta}$, then draw one sample y from $F_5(\hat{\theta}) + \mathcal{N}(0, \Sigma_{\text{obs}})$.

Then define Bayesian posterior

$$\pi(\theta|y) \propto \mathcal{N}(\theta; \mu_{\text{prior}}, \Sigma_{\text{prior}}) \cdot \mathcal{N}(F_5(\theta); y, \Sigma_{\text{obs}})$$

and implement it as UM-Bridge model with 2D input, 1D output (return $log \circ \pi$!).

Solve with PyMC.

Alternative: Implement custom Metropolis-Hastings MCMC as UM-Bridge client

Advanced MCMC Methods

Basic Idea of Multilevel Methods

A380: Roger Green, A319: A.Savin, WikiCommons

Basic Idea of Multilevel Methods

$>$

$>$

$>$

Do most of the work on cheap "coarse" models!

Links to multilevel PDE solvers, multiscale methods...

A380: Roger Green, A319: A.Savin, WikiCommons

Multilevel MCMC Overview

Telescoping sum of QOI like MLMC:

$$\mathbb{E}_{\nu^L}[Q_L] = \underbrace{\mathbb{E}_{\nu^0}[Q_0]}_{\text{Coarse approx.}} + \sum_{l=1}^{L} \underbrace{\left(\mathbb{E}_{\nu^l}[Q_l] - \mathbb{E}_{\nu^{l-1}}[Q_{l-1}]\right)}_{\text{Corrections}}.$$

How to sample?

High acceptance rates due to samples from coarser levels

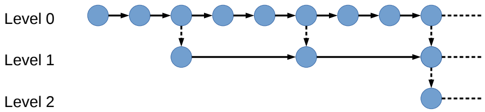
Coarser models cheap, low variance in finer corrections!

Multilevel MCMC Overview

Telescoping sum of QOI like MLMC:

$$\mathbb{E}_{\nu^L}[Q_L] = \underbrace{\mathbb{E}_{\nu^0}[Q_0]}_{\text{Coarse approx.}} + \sum_{l=1}^{L} \underbrace{\left(\mathbb{E}_{\nu^l}[Q_l] - \mathbb{E}_{\nu^{l-1}}[Q_{l-1}]\right)}_{\text{Corrections}}.$$

How to sample?



High acceptance rates due to samples from coarser levels

Coarser models cheap, low variance in finer corrections!

MLMCMC Algorithm

Algorithm 2: MLMCMC

On level 0, run a conventional MCMC chain, delivering samples θ_0^i.

for *level* $k = 1, \ldots, L-1$ **do**

Choose starting point θ_k^0 with coarse component from next coarser starting point θ_{k-1}^0.

for *sample* $j = 1, \ldots, N_k$ **do**

Given θ_k^j, generate proposal $\theta_k' = \left\{ \begin{array}{c} \theta_{k,C}' \\ \theta_{k,F} \end{array} \right\}$ where

- $\theta_{k,C}'$ is drawn from a coarser chain and
- $\theta_{k,F}'$ from proposal density $q_k(\theta_{k,F}' | \theta_{k,F}^j)$.

Compute acceptance probability

$$\alpha(\theta_k', \theta_k^j) = \min \left\{ 1, \frac{\pi_k(\theta_k') q_k(\theta_k^j | \theta_k')}{\pi_k(\theta_k^j) q_k(\theta_k' | \theta_k^j)} \cdot \frac{\pi_{k-1}(\theta_{k,C}^j)}{\pi_{k-1}(\theta_{k,C}')} \right\}.$$

Draw a random number $r \in [0, 1]$.

if $r < \alpha(\theta_k', \theta_k^j)$ **then**

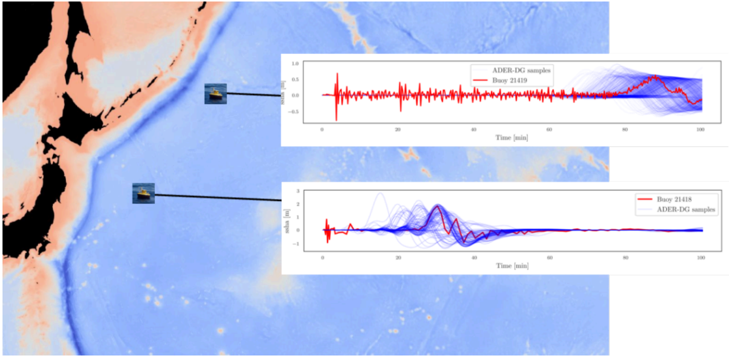
Accept proposal: θ_k' as θ_k^{j+1}

else

Reject proposal: $\theta_k^{j+1} = \theta_k^j$

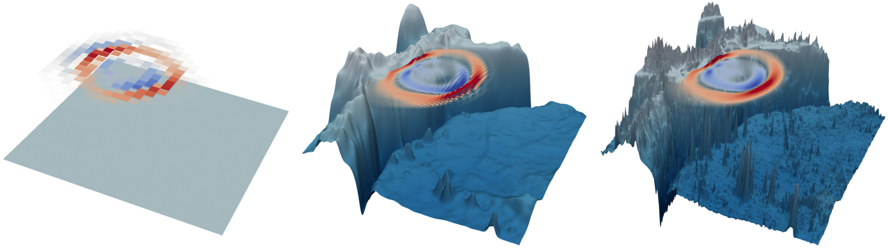
- Use hierarchy of posteriors (i.e. models)
- Gains: Cheap coarse models, variance reduction between levels
- Source: Dodwell et al., 2015/2019

Tsunami Problem



- Modelling Tohoku event (2011) using Shallow Water eq. and real bathymetry
- Forward model built on ExaHyPE PDE engine
- Data: Buoy measurements. Parameter: Tsunami source

Model Hierarchy



Across levels, we adapt

- mesh size
- bathymetry smoothness (specific to hyperbolic solvers!)

Parallel ML / MI MC / MCMC Processor Layout

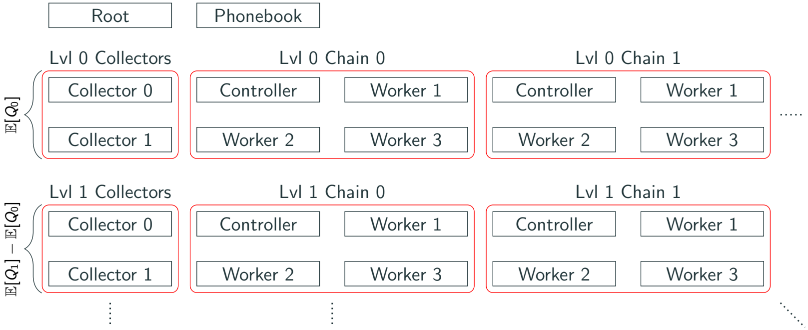
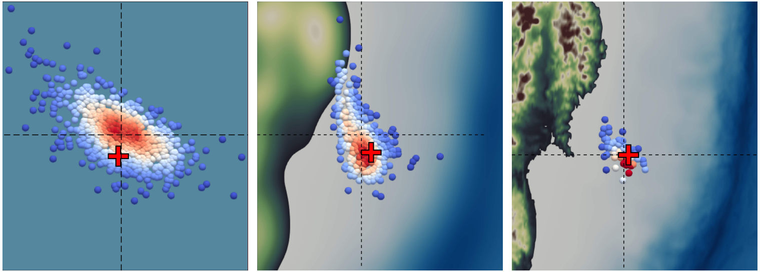


Figure 1: Parallel layout (each box is a processor / MPI rank)

Tested on 2048 parallel chains, more is possible

Results



lvl l	t_l[s]	ρ_l	$\mathbb{V}[Q_0]$ or $\mathbb{V}[Q_l - Q_{l-1}]$		$\mathbb{E}[Q_0]+$ $\sum_{k=1}^{l} \mathbb{E}[Q_k - Q_{k-1}]$	
0	7.38	25	1984.09	1337.42	3.61	27.96
1	97.3	5	1592.17	1523.18	-12.29	23.39
2	438.1	0	340.56	938.53	-5.46	0.12

Run on 3456 cores (72 nodes of 48 cores)

Cloud and High Performance Computing

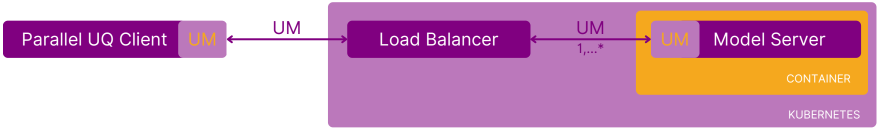
Overview: Cloud Infrastructure

Infrastructure hierarchy	Google Cloud Platform
Application	Cloud Functions
Container	Kubernetes Engine
Virtual machine	Compute engine
Server hardware	
Network	

Servers for rent, (very) different levels of abstraction possible

Kubernetes: "Container orchestration" - fully reproducible HPC setups

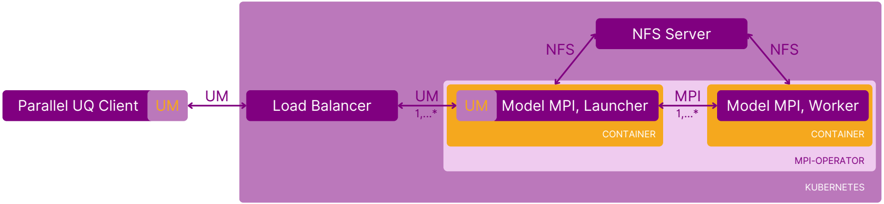
Kubernetes Configuration - Sequential Model



Pre-built configuration, simply plug in your own model container

UQ client only sees an UM-Bridge server. But may make parallel requests!

Kubernetes Setup - MPI Parallel Model



Pre-built reference setup

Shared filesystem between nodes via NFS

Support for OpenMPI, Intel MPI, possibly MPICH in the future

Minor restrictions on model container