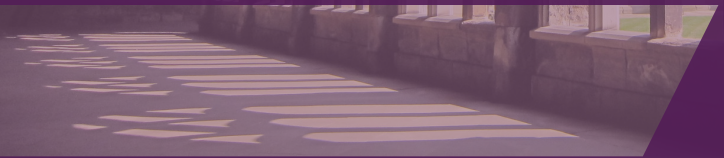
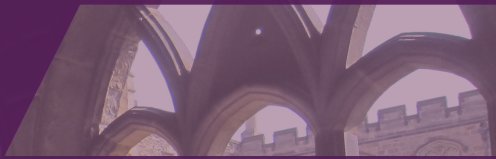
Session 2

Practical Bayesian Inference using MCMC





Anne Reinarz

Durham University

Overview

- Session 1: Introduction
- **Session 2:** Markov Chain Monte-Carlo
  - Markov Chains
  - Metropolis Hastings MCMC
  - Motivating Example
  - Hands-on exercise
- Session 3: Scaling up and hierarchical models

Inverse problems

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

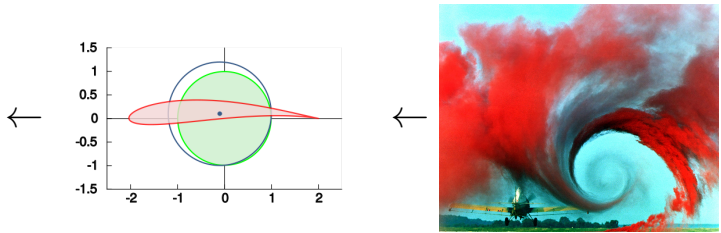
$\leftarrow \underbrace{y}_{\text{Observation distribution}}$

$\leftarrow$ 

Inverse problems

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

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



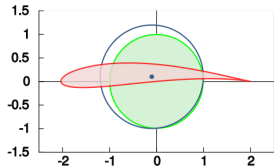
Inverse problems

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

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

?

$\leftarrow$



$\leftarrow$

The Bayesian inverse problem

$$\mathcal{P}(\theta|\mathbf{y}) := \frac{\mathcal{L}(\mathbf{y}|\theta)\pi_{\mathbf{0}}(\theta)}{\mathcal{P}(\mathbf{y})} \propto \mathcal{L}(\mathbf{y}|\theta)\pi_{\mathbf{0}}(\theta).$$

The Bayesian inverse problem

$$\mathcal{P}(\theta|y) := \frac{\mathcal{L}(y|\theta)\pi_0(\theta)}{\mathcal{P}(y)} \propto \mathcal{L}(y|\theta)\pi_0(\theta).$$

- The likelihood $\mathcal{L}(y|\theta)$ is the probability of observing the given measurements if θ were the true parameter

The Bayesian inverse problem

$$\mathcal{P}(\theta|y) := \frac{\mathcal{L}(y|\theta)\pi_0(\theta)}{\mathcal{P}(y)} \propto \mathcal{L}(y|\theta)\pi_0(\theta).$$

- The likelihood $\mathcal{L}(y|\theta)$ is the probability of observing the given measurements if θ were the true parameter
- $\pi_0(\theta)$ is the prior distribution encoding a-priori knowledge about the inferred parameters

The Bayesian inverse problem

$$\mathcal{P}(\theta|y) := \frac{\mathcal{L}(y|\theta)\pi_0(\theta)}{\mathcal{P}(y)} \propto \mathcal{L}(y|\theta)\pi_0(\theta).$$

- The likelihood $\mathcal{L}(y|\theta)$ is the probability of observing the given measurements if θ were the true parameter
- $\pi_0(\theta)$ is the prior distribution encoding a-priori knowledge about the inferred parameters
- the distribution of measurements $\mathcal{P}(y)$ is treated as an unobtainable scaling factor

Markov Chains

Markov Chains: Definition

Let

$$\mathcal{M} = \{X_n\}_{n \geq 0} \quad (\text{or } n \geq -1)$$

be a sequence of correlated random variables.

- Each X_n takes values in a discrete state space $\mathcal{S}$.
- States are labeled by integers.

The process $\mathcal{M}$ is called a *Markov chain* if it satisfies the Markov condition.

$$\Pr(X_{n+1} = j \mid X_n = i, X_{n-1} = x_{n-1}, \ldots, X_0 = x_0) = \Pr(X_{n+1} = j \mid X_n = i).$$

More details check out Colin Fox's very nice lecture notes.

Specification of a Markov Chain

A Markov chain is completely determined by:

- An initial distribution $\Pr(X_0 = i)$
- Transition probabilities $\Pr(X_{n+1} = j \mid X_n = i)$

If the transition probabilities do not depend on n, i.e.,

$$\Pr(X_{n+m+1} = j \mid X_{n+m} = i) = \Pr(X_{n+1} = j \mid X_n = i) \quad \forall m \in \mathbb{Z},$$

the chain is called *homogeneous*.

Stationary Distributions

A distribution π is stationary if:

$$\pi = \pi P.$$

- Here P is the transition matrix

$$P_{ij} := \Pr(X_{n+1} = j \mid X_n = i).$$

  and it is stochastic: $\sum_{j \in \mathcal{S}} P_{ij} = 1$.
- Stationary distributions remain unchanged after transitions.
- The chain reaches equilibrium if π is stationary.

Equilibrium Questions

Fundamental questions:

- Does a stationary distribution exist?
- Is it unique?
- Does $\pi_n \to \pi$ for any initial distribution π_0?

Reversibility and Detailed Balance

Informally, a Markov chain is *reversible* if its probabilistic behavior is unchanged when time is reversed.

A homogeneous Markov chain with transition matrix P and unique stationary distribution π and with $X_n \sim \pi$ for all n. is reversible if and only if

$$\pi_i P_{ij} = \pi_j P_{ji} \quad \forall i, j \in \mathcal{S}.$$

This condition is called the *detailed balance condition*.

Metropolis Hastings MCMC

Markov Chains in Practice

In the previous section, we considered finding the equilibrium distribution π of a Markov chain with a given transition matrix P.

We now consider the *reverse problem*:

> *Given a probability distribution π, how do we construct a transition matrix P such that π is the equilibrium distribution of the Markov chain?*

Markov Chains in Practice

A Markov chain may be specified either by:

- An explicit transition matrix P, or
- A *microscopic dynamics*: an algorithm which generates X_{n+1} from X_n.

In realistic problems, P is rarely given explicitly. Instead the idea is to specify an algorithm and prove that it induces a Markov chain with equilibrium distribution π. (We're not going to do this, look at Colin Fox's lecture notes.)

Metropolis-Hastings MCMC

Given $X_n = i$:

1. **Generation step:** Generate a candidate j from a proposal distribution $g(j \mid i)$, where
   - $g(j \mid i) = 0 \Rightarrow g(i \mid j) = 0$
   - g is irreducible on the state space
2. **Acceptance step:** Accept j with probability

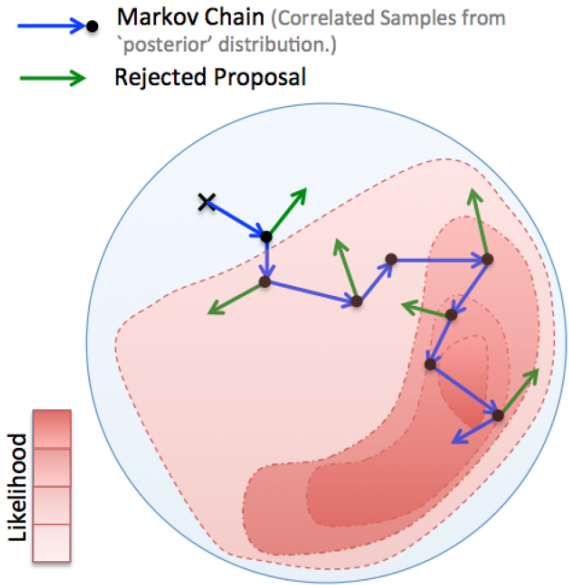
$$\alpha(j \mid i) = \min\left\{1, \frac{\pi_j}{\pi_i}\frac{g(i \mid j)}{g(j \mid i)}\right\}.$$

Summary of MCMC Process

- Start with an initial state X_0.
- Propose new states based on $g(j \mid i)$.
- Accept the proposal with probability $\alpha(j \mid i)$.
- Repeat until convergence to target distribution π.

- MCMC allows sampling from complex distributions by simulating Markov chains with known equilibrium distributions.

Metropolis-Hastings MCMC



Constructing the Proposal Distribution

The proposal distribution $g(j \mid i)$ specifies the *local dynamics* of the Markov chain.

Typical design principles:

- **Local moves:** Propose j by making a small random modification to the current state i.
- **Symmetry (when possible):** Choose $g(j \mid i) = g(i \mid j)$, which simplifies the acceptance probability.

The proposal controls the efficiency of the algorithm, while the acceptance step enforces the target distribution π.

Random-Walk Proposal

A common choice is the *random-walk proposal*, which proposes a local perturbation of the current state.

Given $X_n = i$, propose

$$j = i + \xi,$$

where ξ is drawn from a symmetric distribution.

Typical example:

- $\xi \sim \mathcal{N}(0, \sigma^2 I)$ on $\mathbb{R}^d$

Since $g(j \mid i) = g(i \mid j)$, the acceptance probability reduces to

$$\alpha(j \mid i) = \min\left\{1, \frac{\pi_j}{\pi_i}\right\}.$$

Preconditioned Crank–Nicolson (pCN) Proposal

The pCN proposal is designed for sampling on high-dimensional spaces
Given $X_n = i$, propose

$$j = \sqrt{1 - \beta^2}\, i + \beta\, \xi, \quad \xi \sim \mathcal{N}(0, C),$$

where $0 < \beta < 1$ and C is a covariance operator.

Preconditioned Crank–Nicolson (pCN) Proposal

Assume the posterior satisfies

$$\pi_i \propto L(i)\,\pi_0(i),$$

where π_0 is a Gaussian prior.

The pCN proposal is reversible with respect to π_0, so the prior terms cancel in the Metropolis–Hastings ratio, giving

$$\alpha(j \mid i) = \min\left\{1,\ \frac{L(j)}{L(i)}\right\}.$$

More MCMC methods

A few methods (very much not exhaustive)

- **Adaptive Metropolis Hastings**
  - Improves mixing without gradients
- **Gradient-based MCMC (MALA)**
  - Faster mixing via local geometry
- **Hamiltonian Monte Carlo (HMC)**
  - Exploits gradients to make long-distance proposals
- **Delayed Acceptance MCMC**
  - Uses a cheap surrogate to screen proposals before full evaluation

Hands-on

PyMC with UM-Bridge: Quick Example

- Probabilistic programming in Python
- Supports Bayesian inference via MCMC
- Integrates with black-box models through UM-Bridge
- Define priors, likelihood, and run sampling in a few lines

```
with pm.Model() as model:
        # UM-Bridge models with a single 1D
        ↪ output implementing a PDF
        # may be used as a PyMC density that in
        ↪ turn may be sampled
        posterior =
        ↪ pm.DensityDist('posterior',logp=op,
                         shape=input_dim)

        map_estimate = pm.find_MAP()
        print(f"MAP estimate of posterior is
        ↪ {map_estimate['posterior']}")

        inferencedata =
        ↪ pm.sample(tune=100,draws=400,cores=1,
    step="metropolis")
```

Over to you

Tutorial section 3 Bayesian inverse problems

`https://um-bridge-benchmarks.readthedocs.io/en/docs/tutorial.html`

- Talk to each other
- Ask questions

Some practical considerations

Practical Considerations for MH-MCMC

- How do I choose the proposal?
- Burn-in: discard initial samples before stationarity
- Thinning: subsample to reduce correlation

Practical Considerations for MH-MCMC

- How do I choose the proposal?
- Burn-in: discard initial samples before stationarity
- Thinning: subsample to reduce correlation
- Computational bottleneck: evaluating $F(\theta)$ in $\mathcal{L}(\theta|y)$
- Parallelization strategies: multiple chains, surrogate models, or reduced-order models (session 3)

Optimal Acceptance Rate for MH-MCMC

- Trade-off in Random Walk Metropolis:
  - Too small steps $\rightarrow$ high acceptance but slow exploration
  - Too large steps $\rightarrow$ low acceptance, many rejections

Optimal Acceptance Rate for MH-MCMC

- Trade-off in Random Walk Metropolis:
  - Too small steps $\rightarrow$ high acceptance but slow exploration
  - Too large steps $\rightarrow$ low acceptance, many rejections
- As $d \rightarrow \infty$,

$$\alpha_{\text{opt}} \approx 0.234$$

Optimal Acceptance Rate for MH-MCMC

- Trade-off in Random Walk Metropolis:
  - Too small steps $\rightarrow$ high acceptance but slow exploration
  - Too large steps $\rightarrow$ low acceptance, many rejections
- As $d \rightarrow \infty$,

$$\alpha_{\text{opt}} \approx 0.234$$

Weak convergence and optimal scaling of random walk Metropolis algorithms, G. O. Roberts, A. Gelman, W. R. Gilks, 1997. DOI: 10.1214/aoap/1034625254

Burn-in in MCMC (PyMC example)

```
import pymc as pm

with pm.Model() as model:
    trace = pm.sample(2000, tune=500)  # 'tune' = burn-in
print(trace)
```

- PyMC discards the first `tune` samples automatically

Subsampling (PyMC example)

- Reduce autocorrelation by keeping every k-th sample

```
with pm.Model() as model:
    trace = pm.sample(2000, tune=500, thin=5)  # keep every 5th
    ↪ sample
print(trace)
```

- `thin=k` keeps every k-th sample

Inspecting MCMC Output

In python a good tool to visually inspect MCMC output is the arviz package, look at:

- number of chains and draws
- basic summary statistics
- initial convergence indicators

This provides a quick check for obvious failures before detailed visualization.

```
import arviz as az

az.summary(idata)
```

Inspecting MCMC Output

```
az.summary(idata)
```

	mean	sd	hdi_3%	hdi_97%	mcse_mean	mcse_sd	ess_bulk	ess_tail	r_hat
theta_0	0.301	0.871	-1.292	1.867	0.046	0.032	371.0	683.0	1.01
theta_1	0.326	0.866	-1.252	1.879	0.045	0.032	370.0	681.0	1.01
theta_2	0.351	0.860	-1.213	1.899	0.045	0.032	368.0	693.0	1.01
theta_3	0.377	0.852	-1.156	1.918	0.045	0.032	366.0	711.0	1.00
theta_4	0.402	0.844	-1.119	1.914	0.044	0.031	364.0	736.0	1.00
theta_5	0.427	0.833	-1.061	1.935	0.044	0.031	362.0	757.0	1.00
theta_6	0.451	0.821	-1.015	1.938	0.043	0.031	361.0	767.0	1.00
theta_7	0.476	0.806	-0.967	1.930	0.043	0.030	359.0	776.0	1.00
theta_8	0.500	0.789	-0.875	1.955	0.042	0.030	358.0	768.0	1.00
theta_9	0.523	0.769	-0.818	1.941	0.041	0.029	357.0	773.0	1.00

Interpreting the `arviz` Summary Table

- **mean, sd:** empirical posterior mean and standard deviation
- **hdi_3%, hdi_97%:** 94% highest-density credible interval
- **mcse_mean, mcse_sd:** Monte Carlo standard errors for estimated moments
- **ess_bulk, ess_tail:** effective sample sizes for central mass and tails
- **r_hat:** between- vs within-chain variance diagnostic

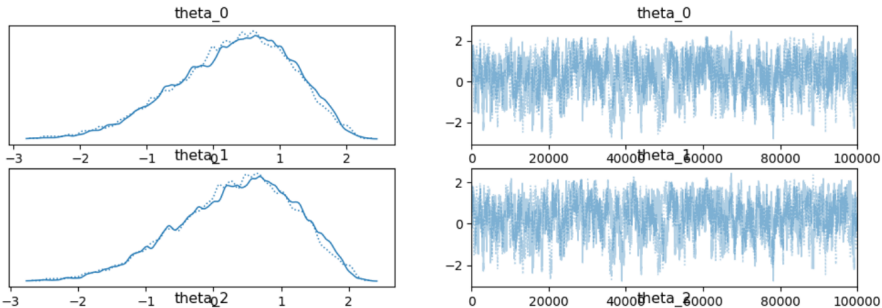
Interpreting the `arviz` Summary Table

- **Small MCSE:** Monte Carlo error is negligible relative to posterior uncertainty
- **Large ESS:** estimates behave like many independent samples
- $\hat{R} \approx 1$**:** chains have mixed and converged

If MCSE is large or $\hat{R} > 1.01$, the numerical estimates are not yet reliable.

Arviz traces

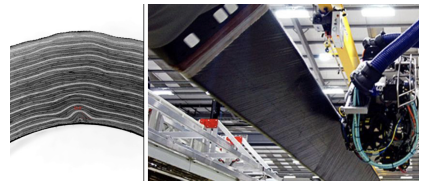
```
# plot posterior kernel densities and traces.
az.plot_trace(idata)
plt.show()
```

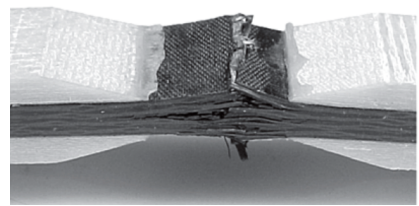


(Hopefully) motivating example

Defects in Composite Materials

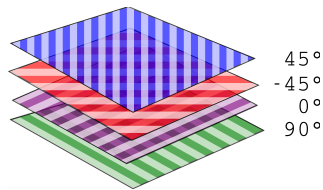
- **Defects** can form when manufacturing complex aerospace components.
- Reduce testing by using numerical simulations
  - Requires good mathematical / mechanical models for composite failure
  - Efficient stochastic algorithms to calculate the probability distribution of failure





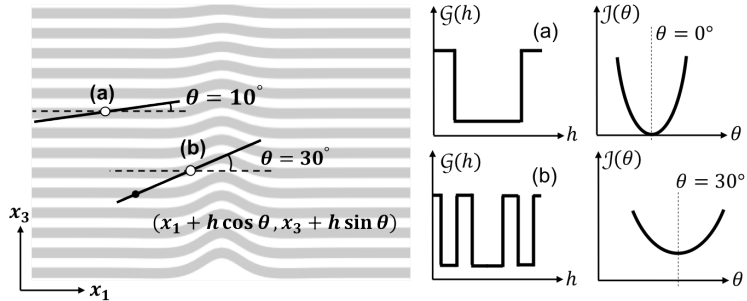
Modelling Challenges for Composite Materials

- Full component parts $>$ 10m scale
  $\rightarrow$ **massive systems of equations**
- **High Contrast** Fibre to Resin ($\sim 10:1$)
  $\rightarrow$ **very ill-conditioned equations**
- **Strongly Anisotropic** (cannot be grid aligned)
  $\rightarrow$ **non-local coupling in A**
  $\rightarrow$ Difficult for AMG solvers



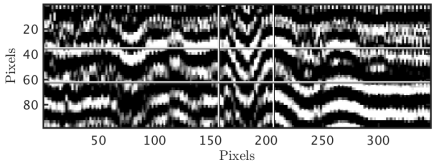
- Reinarz, et al. *High-performance dune modules for solving large-scale, strongly anisotropic elliptic problems with applications to aerospace composites*. 2019.

Characterising Wrinkles



Estimating alignment at a point by minimizing the integral of the gray scale over the trial fibre using the multiple field image analysis method (MFIA).

Wrinkle parameterisation



Karhunen-Loéve expansion:

$$W(\mathbf{x}) = \sum_{i=1}^{N_{KL}} a_i \underbrace{\Psi_i(\lambda, x_1)}_{\text{KL modes}} \underbrace{F(\mathbf{x})}_{\text{Decay fct}} ,$$

where the decay functions

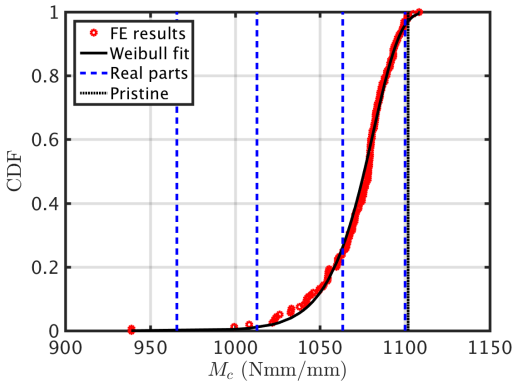
$$F(\mathbf{x}) = \prod_{j=1}^{3} \exp\left[- \frac{(x_j - X_j)^n}{\lambda_D} \right]$$

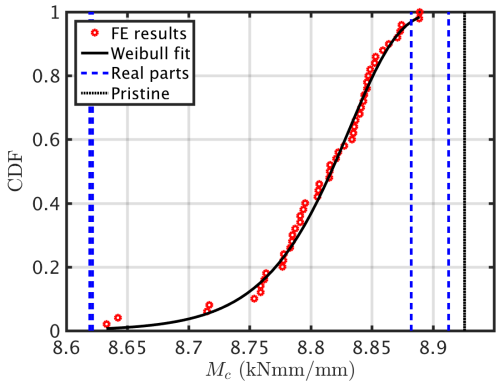
account for the localized nature of the wrinkles.

- Sandhu, et al. *A Bayesian Framework for Assessing the Strength Distribution of Composite Structures with Random Defects*. 2018.

Probability of failure

Left: distributions obtained by sampling using the Markov chain
Right: coefficients drawn at random from a normal distribution centered around the observed mean





Hands-on

Over to you

Consider the analytical benchmarks from the UM-Bridge repository.

- `https://um-bridge-benchmarks.readthedocs.io/en/docs/inverse-benchmarks/analytic-funnel.html`
- `https://um-bridge-benchmarks.readthedocs.io/en/docs/inverse-benchmarks/analytic-donut.html`
- `https://um-bridge-benchmarks.readthedocs.io/en/docs/inverse-benchmarks/analytic-banana.html`
- `https://um-bridge-benchmarks.readthedocs.io/en/docs/inverse-benchmarks/analytic-gaussian-mixture.html`

Run different MCMC approaches for them, what do you see?

PyMC

Use the PyMC documentation:
`https://www.pymc.io/projects/docs/en/v5.10.4/api/generated/pymc.sampling.mcmc.sample.html`

Options for samplers:

- nuts, hmc, metropolis, binary_metropolis, binary_gibbs_metropolis, categorical_gibbs_metropolis, DEMetropolis, DEMetropolisZ, slice

Some more resources

A nice visualisation of this can be found at:
`https://chi-feng.github.io/mcmc-demo/app.html`

There are notebooks to look at for different MCMC methods
`https://github.com/mikkelbue/tinyDA/tree/main/examples`

References

- L. Seelinger, V. Cheng-Seelinger, A. Davis, M. Parno, A. Reinarz. UM-Bridge: Uncertainty quantification and modeling bridge. *Journal of Open Source Software*, 8(83), 2023.
- A. Sandhu, A. Reinarz, and T. Dodwell, A Bayesian framework for assessing the strength distribution of composite structures with random defects, *Composite Structures*, 202 (2018), 1116–1129, DOI:10.1016/j.compstruct.2018.08.074