Accelerating 21-cm Cosmological Inference for REACH with JAX/GPUs

Jacob Tutt[1,2]

[1]Cavendish Astrophysics, University of Cambridge, UK
[2]Kavli Institute for Cosmology, Cambridge, UK

Bayesian Inference for REACH

$$\mathcal{P}(\theta \mid D, M) = \frac{p(D \mid \theta, M)\, p(\theta \mid M)}{\int p(D \mid \theta, M)\, p(\theta \mid M)\, d\theta} = \frac{\mathcal{L}(D \mid \theta, M)\, \Pi(\theta \mid M)}{Z(M)}$$

Prior, $\Pi(\theta \mid M)$

Describes our knowledge/ assumptions about the parameters θ *prior* to any data.

Bayesian Inference for REACH

$$\mathcal{P}(\theta \mid D, M) = \frac{p(D \mid \theta, M)\, p(\theta \mid M)}{\int p(D \mid \theta, M)\, p(\theta \mid M)\, d\theta} = \frac{\mathcal{L}(D \mid \theta, M)\, \Pi(\theta \mid M)}{Z(M)}$$

Prior, $\Pi(\theta \mid M)$

Describes our knowledge/ assumptions about the parameters θ *prior* to any data.

Likelihood, $\mathcal{L}(D \mid \theta, M)$

Quantifies how well a parameter choice θ explains the observed data D.

Bayesian Inference for REACH

$$\mathcal{P}(\theta \mid D, M) = \frac{p(D \mid \theta, M)\, p(\theta \mid M)}{\int p(D \mid \theta, M)\, p(\theta \mid M)\, d\theta} = \frac{\mathcal{L}(D \mid \theta, M)\, \Pi(\theta \mid M)}{Z(M)}$$

Prior, $\Pi(\theta \mid M)$

Describes our knowledge/ assumptions about the parameters θ *prior* to any data.

Likelihood, $\mathcal{L}(D \mid \theta, M)$

Quantifies how well a parameter choice θ explains the observed data D.

Posterior, $\mathcal{P}(\theta \mid D, M)$

An updated state of belief about the parameters θ after incorporating the data.

Bayesian Inference for REACH

$$\mathcal{P}(\theta \mid D, M) = \frac{p(D \mid \theta, M)\, p(\theta \mid M)}{\int p(D \mid \theta, M)\, p(\theta \mid M)\, d\theta} = \frac{\mathcal{L}(D \mid \theta, M)\, \Pi(\theta \mid M)}{Z(M)}$$

Prior, $\Pi(\theta \mid M)$

Describes our knowledge/ assumptions about the parameters θ *prior* to any data.

Likelihood, $\mathcal{L}(D \mid \theta, M)$

Quantifies how well a parameter choice θ explains the observed data D.

Posterior, $\mathcal{P}(\theta \mid D, M)$

An updated state of belief about the parameters θ after incorporating the data.

Evidence, $Z(M)$

The total support the data provides for a model. Crucial for model comparison.

Bayesian Inference for REACH

$$\mathcal{P}(\theta \mid D, M) = \frac{p(D \mid \theta, M)\, p(\theta \mid M)}{\int p(D \mid \theta, M)\, p(\theta \mid M)\, d\theta} = \frac{\mathcal{L}(D \mid \theta, M)\, \Pi(\theta \mid M)}{Z(M)}$$

Prior, $\Pi(\theta \mid M)$

Describes our knowledge/ assumptions about the parameters θ *prior* to any data.

Likelihood, $\mathcal{L}(D \mid \theta, M)$

Quantifies how well a parameter choice θ explains the observed data D.

Posterior, $\mathcal{P}(\theta \mid D, M)$

An updated state of belief about the parameters θ after incorporating the data.

Evidence, $Z(M)$

The total support the data provides for a model. Crucial for model comparison

Bayesian Inference for REACH

$$\mathcal{P}(\theta \mid D, M) = \frac{p(D \mid \theta, M)\, p(\theta \mid M)}{\int p(D \mid \theta, M)\, p(\theta \mid M)\, d\theta} = \frac{\mathcal{L}(D \mid \theta, M)\, \Pi(\theta \mid M)}{Z(M)}$$

Prior, $\Pi(\theta \mid M)$

Describes our knowledge/ assumptions about the parameters θ *prior* to any data.

Likelihood, $\mathcal{L}(D \mid \theta, M)$

Quantifies how well a parameter choice θ explains the observed data D.

Posterior, $\mathcal{P}(\theta \mid D, M)$

An updated state of belief about the parameters θ after incorporating the data.

Evidence, $Z(M)$

The total support the data provides for a model. Crucial for model comparison.

Parametrised Forward Model

The Global 21 cm Signal

Our inference aims to constrain the astrophysical parameters θ governing the evolution of:

$$T_{21}(\nu \mid \theta)$$

GlobalEMU; Bevins et al. 2021

f_*	Star formation efficiency
V_c	Minimum virial circular velocity
f_X	X-ray efficiency
τ	CMB optical depth
α	X-ray SED power-law slope
$\nu_{\min}$	Low-energy cutoff of X-ray SED
R_{mfp}	Mean free path of ionising photons

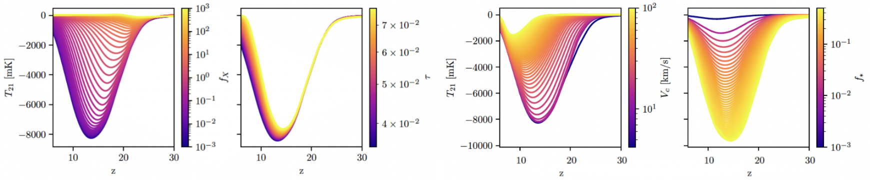


Figure from Bevins (2023).

Parametrised Forward Model

The Global 21 cm Signal

The astrophysical parameters - ≈ 7 params θ:

$$T_{21}(\nu \mid \theta)$$

The Galactic Foregrounds

Model for the diffuse Galactic emission by splitting the sky into N spectral-index regions, each parametised by β_i (Anstey et al 2021)

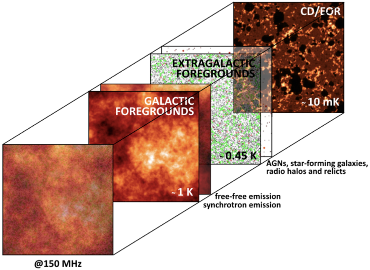


Figure from Chapman, Jelic (2019).

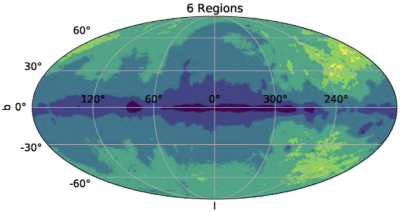


Figure from Anstey et al (2021).

Parametrised Forward Model

The Global 21 cm Signal

The astrophysical parameters - ≈ 7 params θ_{21}:

$$T_{21}(\nu \mid \theta)$$

The Galactic Foregrounds

Physics-motivated model for the diffuse Galactic emission $\approx 15 - 65$ params θ_{for}:

$$T_{sky}(\nu \mid \theta)$$

The Hot Horizon

Modeling an emissive and reflective horizon around the REACH telescope requiring parameterising soil temperature T_{soil} and reflection coeff $|\Gamma|$. (Pattison et al 2024)

Parametrised Forward Model

The Global 21 cm Signal

The astrophysical parameters - ≈ 7 params θ_{21}:

$$T_{21}(\nu \mid \theta)$$

The Galactic Foregrounds

Physics-motivated model for the diffuse Galactic emission $\approx 15 - 65$ regions / params θ_{for}:

$$T_{sky}(\nu \mid \theta)$$

The Hot Horizon

Modeling an emissive and reflective horizon around the REACH telescope requiring 2 extra params: T_{soil} and $|\Gamma|$.

Likelihood / Noise Structure

Different noise parameters (θ_{noise}).
(Scheutwinkel, 2023)

- **Gaussian:** $\theta_{\mathrm{noise}} = \{\sigma_L\}$
- **Generalised Normal:** $\theta_{\mathrm{noise}} = \{\beta_L, \sigma_L\}$
- **Radiometric:** $\theta_{\mathrm{noise}} = \{T_{\mathrm{rec}}, \eta, \sigma_{\mathrm{radio}}\}$

Parametrised Forward Model

The Global 21 cm Signal

The astrophysical parameters - ≈ 7 params θ_{21}:

The Galactic Foregrounds

Physics-motivated model for the diffuse Galactic emission $\approx 15 - 65$

And More:

- RFI Flagging (D Anstey and S Leeney, 2024)
- Extra-galactic Point Sources (S Mittal et al 2024)
- Forground Map Errors (M Pagona et al 2024)

telescope requiring 2 extra params: T_{soil} and $|\Gamma|$.

- **Gaussian:** $\theta_{\text{noise}} = \{\sigma_L\}$
- **Generalised Normal:** $\theta_{\text{noise}} = \{\beta_L, \sigma_L\}$
- **Radiometric:** $\theta_{\text{noise}} = \{T_{\text{rec}}, \eta, \sigma_{\text{radio}}\}$

High-Dimensional Inference

Typical analyses involve $30-80$ parameters:

- $\sim$ 7 astrophysical
- $\sim$ $15-65$ foreground
- $\sim$ 2 horizon
- $\sim$ $1-3$ noise

$\Rightarrow$ **millions of likelihood calls, $\sim$1-20 hours per run on CPUs.**

High-Dimensional Inference

Typical analyses involve 30 – 80 parameters:

- $\sim$ 7 astrophysical
- $\sim$ 15 – 65 foreground
- $\sim$ 2 horizon
- $\sim$ 1 – 3 noise

$\Rightarrow$ **millions of likelihood calls, ∼1-20 hours per run on CPUs.**

Using:
- **Chromatic functions** $K_{i,j,k}, R_{i,j,k}, J_i$: (encode beam + instrument response)
- **Neural network emulator** $T_S(\nu, \theta_S)$ (fast 21-cm signal generation)

$$\log \mathcal{L} = \sum_{i,j} \left[-\frac{1}{2} \log\left(2\pi\, \theta_\sigma^2\right) - \frac{1}{2} \left(\frac{T_D(\nu,t) - (T_F(\nu,t,\theta_F) + T_S(\nu,\theta_S))}{\theta_\sigma} \right)^2 \right]$$

$$T_{F_{i,j}} = \sum_k K_{i,j,k}\, F_i(\theta_{F_k}) + \sum_k R_{i,j,k}\, F_i(\theta_{F_k})\, |\Gamma|\alpha + J_i\, T_H\, (1 + |\Gamma|\alpha)$$

Credit: Pattison et al 2025

Inference reduces to **matrix multiplications + vector operations**.

$\Rightarrow$ Ideally suited to the **SIMD/SIMT architecture of modern GPUs**.

High-Dimensional Inference

Typical analyses involve 30 – 80 parameters:

- $\sim$ 7 astrophysical
- $\sim$ 15 – 65 foreground
- $\sim$ 2 horizon
- $\sim$ 1 – 3 noise

$\Rightarrow$ **millions of likelihood calls, ~1-20 hours per run on CPUs.**

Using:
- **Chromatic functions** $K_{i,j,k}$, $R_{i,j,k}$, J_i: (encode beam + instrument response)
- **Neural network emulator** $T_S(\nu, \theta_S)$ (fast 21-cm signal generation)

To allow wider time ranges and higher-dim models:

Our pipelines must move to GPU-accelerated inference.

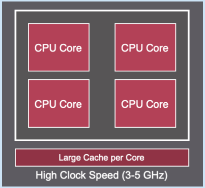
Inference reduces to **matrix multiplications + vector operations**.

$\Rightarrow$ Ideally suited to the **SIMD/SIMT architecture of modern GPUs**.

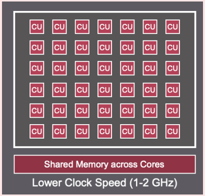
Modern Computational Architecture

CPU Architecture

- Few powerful cores (10s)
- Complex control logic
- Optimised for sequential workloads
- Large caches, low latency



GPU Architecture



- Thousands of lightweight cores
- Massive parallelism (SIMT)
- High memory bandwidth
- Excel at batched vectorised operations

Image Credit: AMD

An Introduction to JAX

High-performance numerical-computing and large-scale machine learning

High-performance numerical-computing and large-scale machine learning

Automatic Differentiation

$$(\nabla f)(x)_i = \frac{\partial f}{\partial x_i}(x) \quad \Longrightarrow \quad \texttt{jax.grad(f)(x)}$$

For more details on Autodiff/Dual Numbers see
JacobTutt/dual_autodiff_package

An Introduction to JAX

High-performance numerical-computing and large-scale machine learning

Automatic Differentiation

$$(\nabla f)(x)_i = \frac{\partial f}{\partial x_i}(x) \quad \Longrightarrow \quad \texttt{jax.grad(f)(x)}$$

For more details on Autodiff/Dual Numbers see
JacobTutt/dual_autodiff_package

XLA Compilation

- Transforms functions into optimised machine code
- Provides python flexibility alongside compiled language preformance

$$f(x) \quad \Longrightarrow \quad \texttt{jax.jit}(f)(x)$$

An Introduction to JAX

High-performance numerical-computing and large-scale machine learning

Automatic Differentiation

$$(\nabla f)(x)_i = \frac{\partial f}{\partial x_i}(x) \implies \texttt{jax.grad(f)(x)}$$

For more details on Autodiff/Dual Numbers see
JacobTutt/dual_autodiff_package

XLA Compilation

- Transforms functions into optimised machine code
- Provides python flexibility alongside compiled language preformance

$$f(x) \implies \texttt{jax.jit}(f)(x)$$

Automatic Vectorisation and Parallelisation

- `jax.vmap`: automatic vectorisation over batches of data
- `jax.pmap`: parallel execution across multiple XLA devices

Benchmarking Performance Increases

Data/ Chromaticity Function Generation

1. **Foreground model**
   Build $T_{\mathrm{sky}}(\Omega, \nu)$ with base + spectral index maps (power-law scaling around ν_0).
2. **Galactic Transforms**
   Transform maps from Galactic to Local (AltAz) frames
3. **Lunar / horizon environment**
   Inject lunar emission and horizon + soil emission/reflection model.
4. **Antenna convolution**
   Apply chromatic antenna pattern $A(\nu, \Omega)$, and average over the sky: $T_{\mathrm{ant}}(\nu, t)$.
5. **Time reduction**
   Optionally collapse time dimension (Averaged vs Separated).
6. **Noise model**
   Add instrumental noise using the selected model (Gaussian, radiometric, etc.).
7. **21-cm signal injection**
   Add the cosmological global signal model $T_{21}(\nu)$.

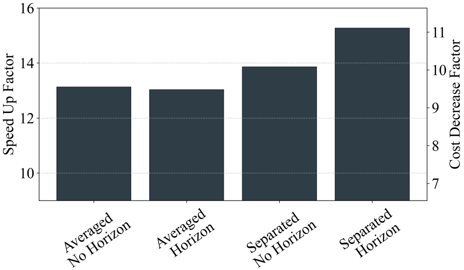
Benchmarking Performance Increases

Data Generation Pipeline

- Computed for 13 time intervals

Configuration	Old (s)	New (s)
Ave No Horizon	661	50
Ave Horizon	655	50
Sep No Horizon	725	52
Sep Horizon	813	53

** Old pipeline - 40-core CPU node (£0.40/hr)*
** New pipeline - NVIDIA A100 GPU(£0.55/hr)*



Key Takeaway: Speed up of up to **15X** (11X Financial Saving)

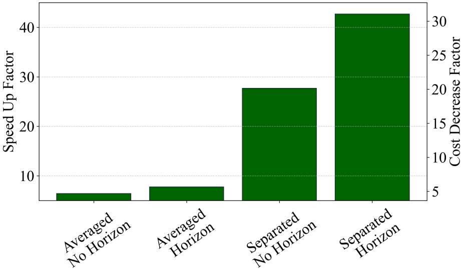
Benchmarking Performance Increases

Chromaticity Function Generation

- Computed for 13 time intervals and 35 regions

Configuration	Old (s)	New (s)
Ave No Horizon	328	51
Ave Horizon	406	52
Sep No Horizon	1474	53
Sep Horizon	2326	54

** Old pipeline - 40-core CPU node (£0.40/hr)*
** New pipeline - NVIDIA A100 GPU(£0.55/hr)*



Key Takeaway: Speed up of up to **43X** (31X Financial Saving)

Benchmarking Performance Increases

Data / Chromaticity Function Generation

Pipeline Stage	Run Time
Initialising/ Loading Data	$\sim \mathcal{O}(24s)$
Galactic Transforms (CPU)	$\sim \mathcal{O}(26s)$
Spectral Index Broadcasting	
Sky Map Interpolation	
Lunar / Horizon Injection	
Antenna Convolution	$\sim \mathcal{O}(0.75s)$
Time Averaging	
Noise Model	
21-cm Signal Injection	

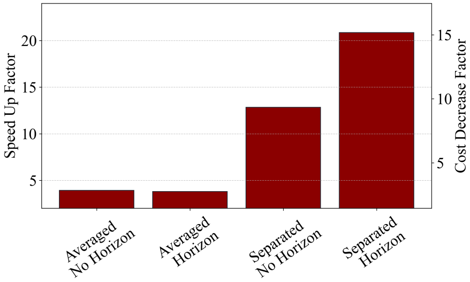
Benchmarking Performance Increases

Individual Likelihood Call

- Computed for 13 time intervals and 35 regions

Configuration	Old(ms)	New(ms)
Ave No Horizon	0.31	0.08
Ave Horizon	0.47	0.12
Sep No Horizon	1.05	0.08
Sep Horizon	2.80	0.13

** Averaged Over 1000 Likelihood Calls*
** Old pipeline - 40-core CPU node (£0.40/hr)*
** New pipeline - NVIDIA A100 GPU(£0.55/hr)*



Key Takeaway: Speed up of up to **21X**
(15X Financial Saving)

Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_{\Theta} \mathcal{L}(\theta)\, \pi(\theta)\, d\theta = \int_0^1 \mathcal{L}(\xi)\, d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\, d\theta \implies \text{inverse relation: } \mathcal{L}(\xi)$$

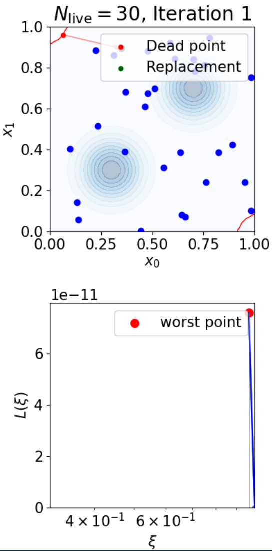
Shrinkage of the prior mass:

$$\xi_i = t \;\; \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log \;\; t] = -\frac{1}{N_{\text{live}}} \implies \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i \left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right) \right]$$



Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z \;=\; \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta \;=\; \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) \;=\; \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \qquad \Longrightarrow \qquad \text{inverse relation: } \mathcal{L}(\xi)$$

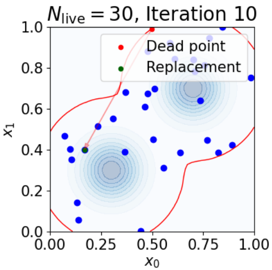
Shrinkage of the prior mass:

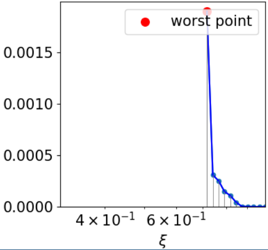
$$\xi_i = t\;\; \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\;\; t] = -\frac{1}{N_{\text{live}}} \qquad \Longrightarrow \qquad \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$\boldsymbol{Z \;\approx\; \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i\left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]}$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta = \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \quad \Longrightarrow \quad \text{inverse relation: } \mathcal{L}(\xi)$$

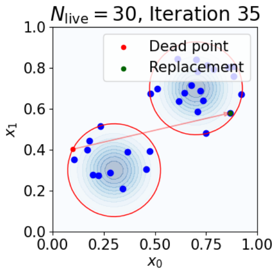
Shrinkage of the prior mass:

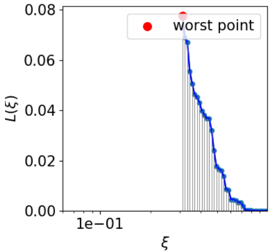
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \quad \Longrightarrow \quad \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i\left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta = \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \quad \Longrightarrow \quad \text{inverse relation: } \mathcal{L}(\xi)$$

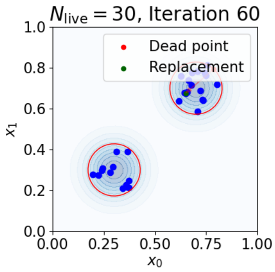
Shrinkage of the prior mass:

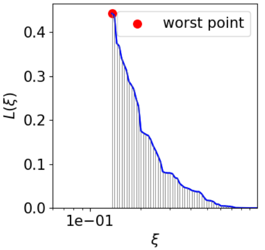
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \quad \Longrightarrow \quad \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i \left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta = \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \quad \Longrightarrow \quad \text{inverse relation: } \mathcal{L}(\xi)$$

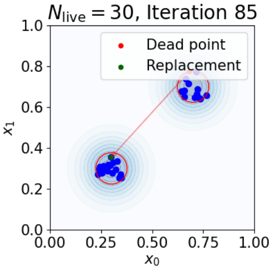
Shrinkage of the prior mass:

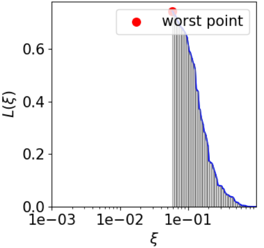
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \quad \Longrightarrow \quad \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$\boldsymbol{Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i \left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right) \right]}$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z \;=\; \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta \;=\; \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) \;=\; \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \qquad \Longrightarrow \qquad \text{inverse relation: } \mathcal{L}(\xi)$$

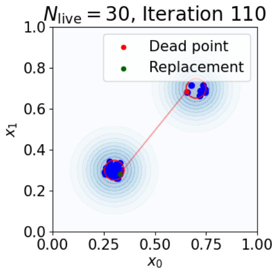
Shrinkage of the prior mass:

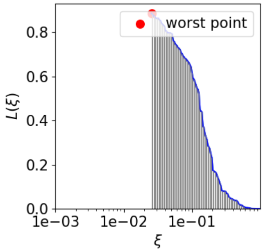
$$\xi_i = t \;\; \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log \;\; t] = -\frac{1}{N_{\text{live}}} \qquad \Longrightarrow \qquad \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$\boldsymbol{Z \;\approx\; \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i \left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right) \right]}$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_\Theta \mathcal{L}(\theta)\,\pi(\theta)\,d\theta = \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \quad \Longrightarrow \quad \text{inverse relation: } \mathcal{L}(\xi)$$

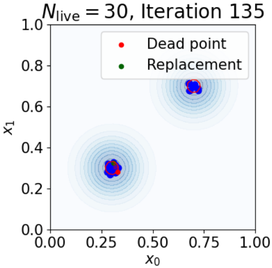
Shrinkage of the prior mass:

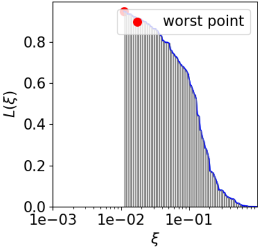
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \quad \Longrightarrow \quad \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i\left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta = \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \implies \text{inverse relation: } \mathcal{L}(\xi)$$

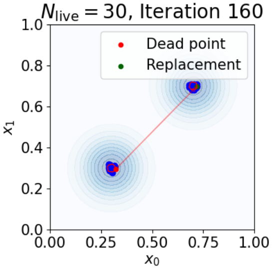
Shrinkage of the prior mass:

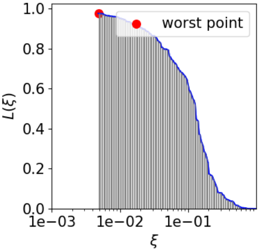
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \implies \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i\left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_{\Theta} \mathcal{L}(\theta)\,\pi(\theta)\,d\theta = \int_0^1 \mathcal{L}(\xi)\,d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\,d\theta \implies \text{inverse relation: } \mathcal{L}(\xi)$$

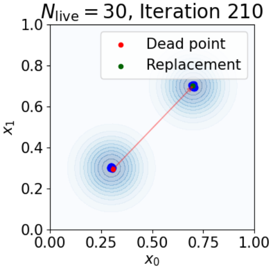
Shrinkage of the prior mass:

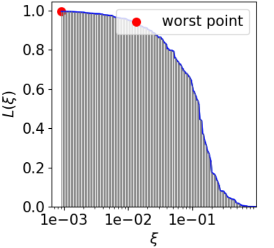
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \implies \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i\left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]$$





Traditional Nested Sampling

Nested Sampling provides:

- Posterior samples
- **The Bayesian Evidence**

Evidence integral:

$$Z = \int_\Theta \mathcal{L}(\theta)\, \pi(\theta)\, d\theta = \int_0^1 \mathcal{L}(\xi)\, d\xi.$$

Prior volume (mass):

$$\xi(\mathcal{L}) = \int_{\mathcal{L}(\theta)>\mathcal{L}} \pi(\theta)\, d\theta \implies \text{inverse relation: } \mathcal{L}(\xi)$$

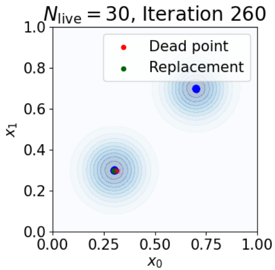
Shrinkage of the prior mass:

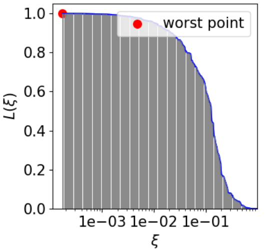
$$\xi_i = t\ \xi_{i-1},$$

Expected shrinkage:

$$\mathbb{E}[log\ t] = -\frac{1}{N_{\text{live}}} \implies \xi_i \approx \exp\left(-\frac{i}{N_{\text{live}}}\right)$$

$$Z \approx \sum_i \mathcal{L}_i(\xi_{i-1} - \xi_i) = \sum_i \mathcal{L}_i \left[\exp\left(-\frac{i-1}{N_{\text{live}}}\right) - \exp\left(-\frac{i+1}{N_{\text{live}}}\right)\right]$$





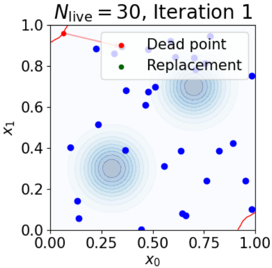
Accelerated Nested Sampling

Traditional nested sampling (serial):

- Remove one 'worst' live point each iteration.
- New point conditioned on:

$$\mathcal{L} > \mathcal{L}_{\text{min}}$$

- Inherently sequential MCMC slice samples
- `PolyChord - Handley et al 2025`

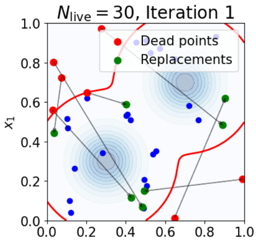


Accelerated nested sampling (parallel):

- Discard a batch of n_{del} 'worst' points.
- All replacements conditioned on:

$$\mathcal{L}(\theta) > \mathcal{L}_{\text{min}} \quad \mathcal{L}_{\text{min}} = \max\{\mathcal{L}_1, \ldots, \mathcal{L}_{n_{\text{del}}}\}$$

- Sample each replacement independently
  $\Rightarrow$ vectorised (vmap) across GPU
- `Blackjax`
  - `Cabezas at al 2024, Yallup et al 2025`



Accelerated Nested Sampling

Traditional nested sampling (serial):

- Remove one ‘worst’ live point each iteration.
- New point conditioned on:

$$\mathcal{L} > \mathcal{L}_{\text{min}}$$

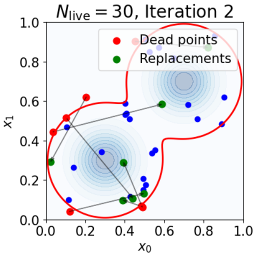
- Inherently sequential MCMC slice samples
- `PolyChord - Handley et al 2025`

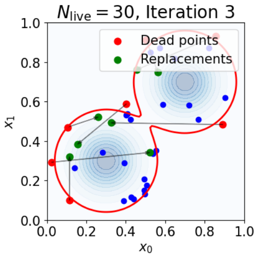
Accelerated nested sampling (parallel):

- Discard a batch of n_{del} ‘worst’ points.
- All replacements conditioned on:

$$\mathcal{L}(\theta) > \mathcal{L}_{\text{min}} \quad \mathcal{L}_{\text{min}} = \max\{\mathcal{L}_1, \ldots, \mathcal{L}_{n_{\text{del}}}\}$$

- Sample each replacement independently
  $\Rightarrow$ vectorised (vmap) across GPU
- `Blackjax`
  - `Cabezas at al 2024, Yallup et al 2025`





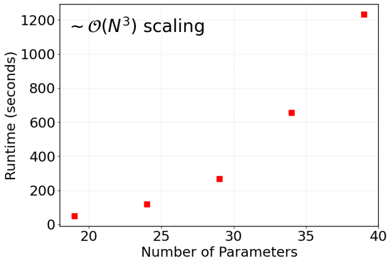
Benchmarking Nested Sampling

Time Separated Horizon Model Fits

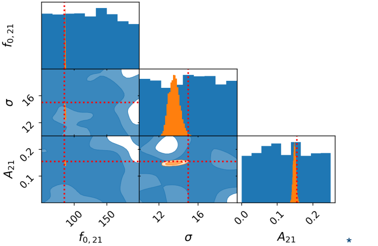
Configuration	Old Time (Hr:Min:Sec)	New Time (Min:Sec)	Speed Factor	Price Factor
15 Reg (19 Param)	12:00:00*	00:00:49	881*	641*
20 Reg (24 Param)	12:00:00*	00:02:00	360*	262*
35 Reg (39 Param)	12:00:00*	00:20:32	35*	26*

* CSD3 timed out prior to completion of fitting - thus lower limits

NS Parameter Scaling



Signal Recovery



Time-Separated, Gaussian Signal, 20 Region, No Horizon*

Future of REACH Pipeline?

Accelerated Forward Model:

$$T_{\mathrm{obs}}(\nu) = T_{21}(\nu) + T_{\mathrm{FG}}(\nu) + \sigma(\nu)$$

Configuration	New (ms)
Ave No Horizon	0.59
Ave Horizon	0.63
Sep No Horizon	0.61
Sep Horizon	0.59

** Averaged over 1000 forward model calls*
** Run on an NVIDIA A100 GPU (£0.55/hr)*
** Gaussian signal and noise structure*

Note: An order of magnitude faster without noise addition.

Future of REACH Pipeline?

Accelerated Forward Model:

$$T_{\text{obs}}(\nu) = T_{21}(\nu) + T_{\text{FG}}(\nu) + \sigma(\nu)$$

Configuration	New (ms)
Ave No Horizon	0.59
Ave Horizon	0.63
Sep No Horizon	0.61
Sep Horizon	0.59

** Averaged over 1000 forward model calls*
** Run on an NVIDIA A100 GPU (£0.55/hr)*
** Gaussian signal and noise structure*

Note: An order of magnitude faster without noise addition.

Future of REACH Simulations

- XLA compilation and full vectorisation / parallelisation allows:

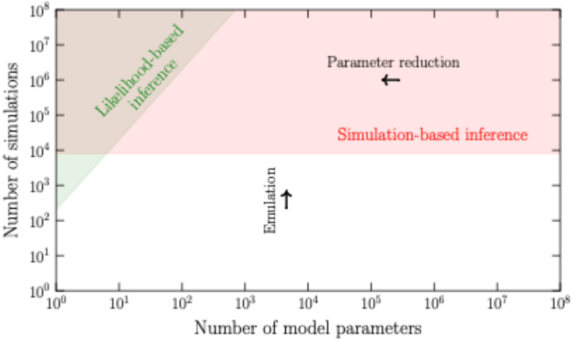
$\mathcal{O}(10^6)$ simulations per second

Future of REACH Pipeline?

Simulation Based Inference

As we begin to consider more parameterisations of our system

- eg. Parameterised/ Emulated Beam Pattern (+4/5)
- Saxena et al 2024 - TMNRE



Boddy et al 2022

Questions

Thank you for your attention!

Jacob Tutt
Department of Physics, University of Cambridge
jlt67@cam.ac.uk

REACH-telescope/REACH_data_analysis_gpu

An Introduction to JAX

High-performance numerical-computing and large-scale machine learning

Automatic Differentiation

Provides ‘free’ exact derivatives of numerical at machine precision by systematically applying the chain rule.

$$(\nabla f)(x)_i = \frac{\partial f}{\partial x_i}(x) \quad \Longrightarrow \quad \texttt{jax.grad(f)(x)}$$

For more details on Autodiff/Dual Numbers see
JacobTutt/dual_autodiff_package

Gradient-based Optimisation

JAX computes gradients of a loss function:

$$\nabla_\theta \mathcal{L}(\theta) = \texttt{jax.grad}(\mathcal{L})(\theta)$$

Exploited during ML training’s gradient descent loops using **Optax**’s optimisers (e.g. Adam):

$$\theta_{t+1} = \theta_t - \alpha \frac{m_t}{\sqrt{v_t} + \epsilon}$$

Hamiltonian Monte Carlo/ NUTS

Exploits gradients of the log-posterior to accelerate exploration of parameter space:

$$\nabla_\theta \log p(\theta \mid D)$$

BlackJAX enables efficient sampler implementations (eg via Leapfrog Integration).

$$(p, q) \longrightarrow (p', q')$$

An Introduction to JAX

High-performance numerical-computing and large-scale machine learning

XLA Compilation

Transforms functions into optimised machine code:

- Inputs are wrapped in *tracers*
- JAX operation mapped to a computation graph (intermediate).
- jaxpr then device-dependently XLA compiled

$$f(x) \implies \texttt{jax.jit}(f)(x)$$

Compiled Languages

- Ahead-of-time compilation to machine code
- Static typing and memory layout
- Highly optimised loops / vectorisation

Pros: low-level control, maximum hardware efficiency.

Python + Compiled Speed

- Python - slow interpreted language.
- Produces near-C++ speeds
- Supports GPUs with little code changes

JAX provides Python's flexibility with compiled-language performance.

An Introduction to JAX

High-performance numerical-computing and large-scale machine learning

Automatic Vectorisation and Parallelisation

- `jax.vmap`: automatic vectorisation over batches of data
  - Deals with batches inside primitive operations
- `jax.pmap`: parallel execution across multiple XLA devices
  - True hardware-level parallelism (SPMD)

$$f(x) \implies \mathtt{vmap}(f)(X) \qquad f(x) \implies \mathtt{pmap}(f)(X)$$

From Loops to Vectorisation

Old way (NumPy/Python):

```
outputs = []
for i in range(N):
    outputs.append(f(xs[i]))
```

New way (JAX):

$$\mathtt{vmap}(f)(x_{1:N})$$

CPU to GPU Parallelism

Old way (MPI / multi-CPU):

```
with ProcessPoolExecutor() as ex:
    results = list(ex.map(f, xs))
```

New way (JAX):

$$\mathtt{pmap}(f)(x_{\text{devices}})$$

UNIVERSITY OF CAMBRIDGE
Cavendish Laboratory
Department of Physics
REACH
JacobTutt