The Bayesian Analysis Toolkit (BAT)

Oliver Schulz
on behalf of the BAT team

MAX-PLANCK-GESELLSCHAFT

Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)

BAT
BAYESIAN ANALYSIS TOOLKIT

oschulz@mpp.mpg.de

JuliaHEP2023, Erlangen, November 7th 2023

Introduction

- The Bayesian Analysis Toolkit (BAT):
  A software package for Bayesian inference
- Typical tasks: Given a set of data and prior knowledge
  - estimate parameters
  - compare models (Bayes factors)

  according to Bayes theorem

$$P(\vec{\lambda}|\vec{D}) = \frac{P(\vec{D}|\vec{\lambda})P_0(\vec{\lambda})}{\int P(\vec{D}|\vec{\lambda})P_0(\vec{\lambda})\, d\vec{\lambda}}$$

Introduction

- The Bayesian Analysis Toolkit (BAT):
  A software package for Bayesian inference
- Typical tasks: Given a set of data and prior knowledge
  - estimate parameters
  - compare models (Bayes factors)

  according to Bayes theorem

$$P(\vec{\lambda}|\vec{D}) = \frac{P(\vec{D}|\vec{\lambda})P_0(\vec{\lambda})}{\int P(\vec{D}|\vec{\lambda})P_0(\vec{\lambda})\,d\vec{\lambda}}$$

- Functionalities
  - Multi-method posterior space exploration
  - Integration of non-normalized posterior
    (i.e. evidence calculation)
  - User-friendly plotting and reporting

BAT.jl, the successor of BAT-C++

- Original: BAT-C++, developed at MPP
  [DOI: 10.1016/j.cpc.2009.06.026 (2009).]
  - Very successful over the years, > 250 citations (INSPIRE)
  - Written in C++, based on CERN ROOT
  - Gained wide user base, esp. HEP/nuclear/astro-physics
  - Had reached limit of original software design, needed a complete re-write.

BAT.jl, the successor of BAT-C++

- Original: BAT-C++, developed at MPP
  [DOI: 10.1016/j.cpc.2009.06.026 (2009).]
  - Very successful over the years, > 250 citations (INSPIRE)
  - Written in C++, based on CERN ROOT
  - Gained wide user base, esp. HEP/nuclear/astro-physics
  - Had reached limit of original software design, needed a complete re-write.
- Successor: BAT.jl, written in Julia.
  [DOI: 10.1007/s42979-021-00626-4 (2021).]
  - MPP (A. Caldwell): O. Schulz (lead), A. Butorev, M. Dudkowiak
  - TU-Dortmund (K. Kröninger): C. Grunwald, S. Lacagnina,
  - ORIGINS ODSL: F. Capel, P. Eller, J. Knollmüller
  - . . . and many contributions from past students (thank you!)

Design goals for BAT.jl

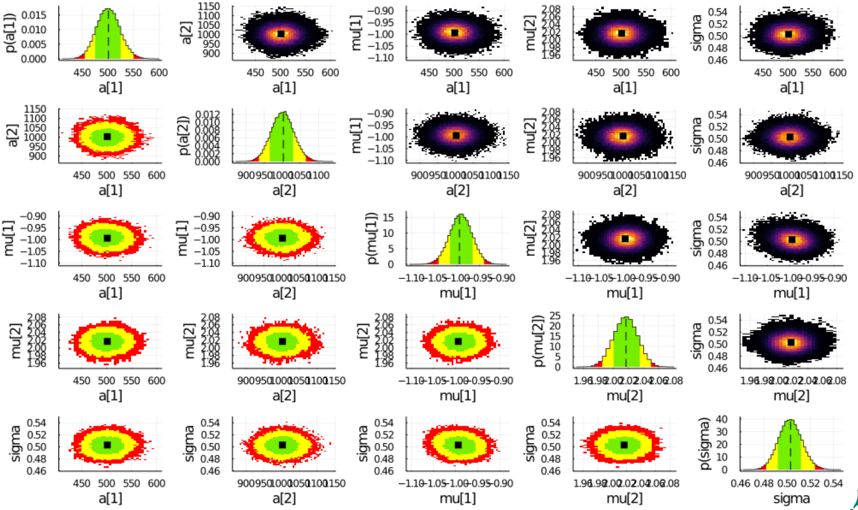
- Core philosophy: User provides forward model or likelihood BAT does the rest, no DSL required
- Easy to use with defaults, but allow for detailed fine-tuning
- Multiple sampling algorithms
- Support for parallel operation: Local (multiple threads) and distributed (compute clusters)
- Use auto-differentiation where gradients required
- Utilize Julia ecosystem (AdvancedHMC, etc.)

BAT.jl Features

- MCMC samlpling via Metropolis-Hastings, Hamiltonian Monte Carlo, Sobol and importance sampling
- Posterior integration with nested sampling, bridge sampling, AHMI (Caldwell et. al, MPP) or Cuba (T. Hahn, MPP)
- Automatic space transformations cast target density into space suitable for algorithm
- Julia brings auto-differentiation, excellent package management and unmatched code composability via multiple dispatch (and much more)
- Current version BAT.jl v3.1
- https://github.com/BAT/BAT.jl

Simple BAT.jl example: Histogram Fit

BAT.jl plotting: Posterior projections



Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)

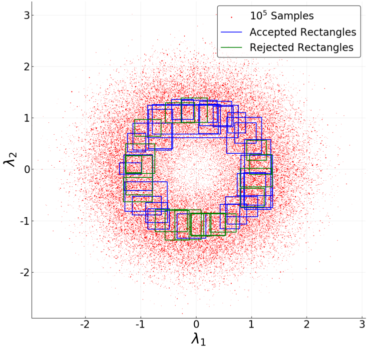
Adaptive Harmonic Mean Integration

- Given N samples Λ_i drawn from posterior, can estimate integral via harmonic mean:

$$\hat{I} \equiv \frac{NV}{\sum_{i=1}^{N} \frac{1}{f(\Lambda_i)}} \; .$$

- Problem: variance of estimator diverges in the general case
- AHMI algorithm:
  - Whiten sample distribution
  - Generate set of hyper-rectangles with bounded variance
  - Compute individual harmonic mean integrals
  - Estimate correlation of integrals
  - Combine integrals into robust overall integral with error estimate

Adapive Harmonic Mean Integration (AHMI)



- Computes posterior integral/evidence from samples via harmonic mean [Int.J.Mod.Phys.A 35 (2020) 24, 1950142]
- Operates in hyper-rectangles with limited posterior variance to control integral variance

Parameter space partitioning

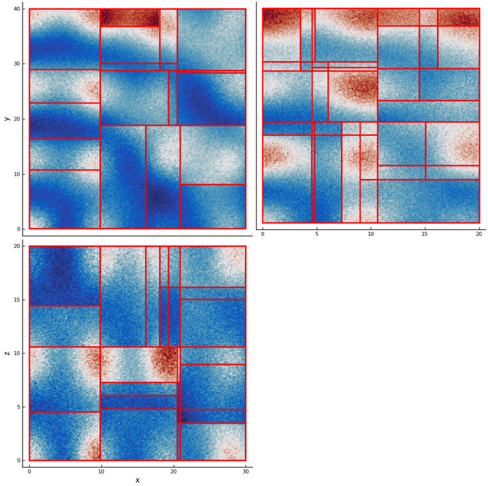
- MCMC expensive, need maximum parallelization
- Parallelization potential of likelihood often limited
- Increasing number of chains doesn't help (burn-in cost)

Parameter space partitioning

- MCMC expensive, need maximum parallelization
- Parallelization potential of likelihood often limited
- Increasing number of chains doesn't help (burn-in cost)
- Idea: partition parameter space
  run separate set of chains in each subspace
- Rationale: posterior in small subspaces simpler,
  fast burn-in
- Challenge: find good partitioning for given posterior,
  work in progress

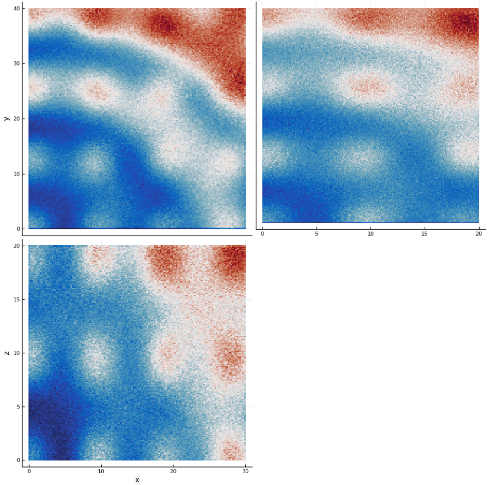
[DOI 10.1142/S0217751X20501420, IJMPA (2020)]

Parameter Space Partitioning, Raw



- Subspaces contains unequal probability mass: can't just stitch MCMC results together

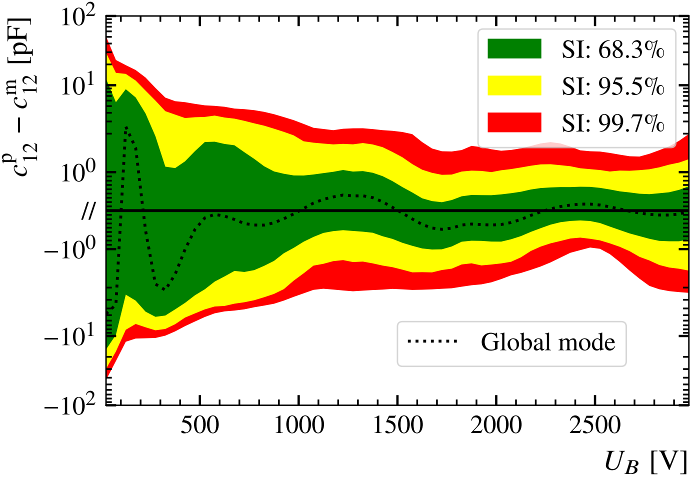
Parameter Space Partitioning, Reweighted



Integrate posterior in each subspace, then reweight by integral [Stat Comput 32, 56 (2022)]

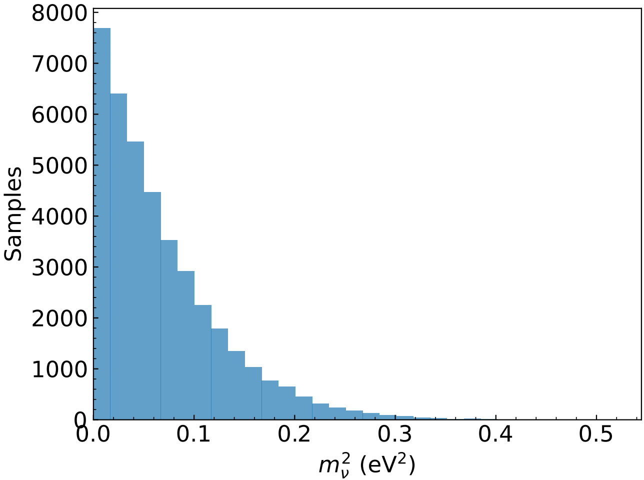
Some use cases . . .

HPGe-Detector impurity profile inference



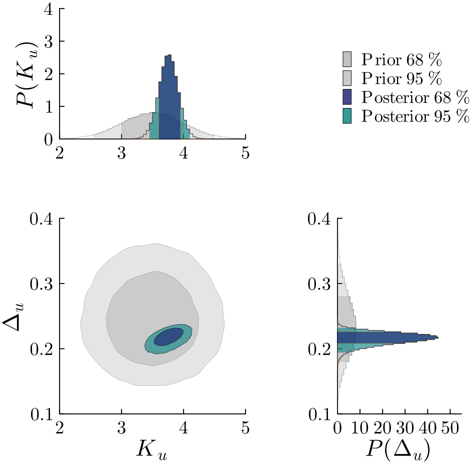
Cap./vol.-curves measured and simulated, ML surrogate, complex prior [Eur. Phys. J. C 83, 352 (2023)], Metropolis-Hastings

KATRIN m_ν^2 posterior, simulated data



NETRIUM DNN model [Eur. Phys. J. C 82, 439 (2022)] ported to Julia
Sampled with AdvandedHMC backend using Zygote-AD.

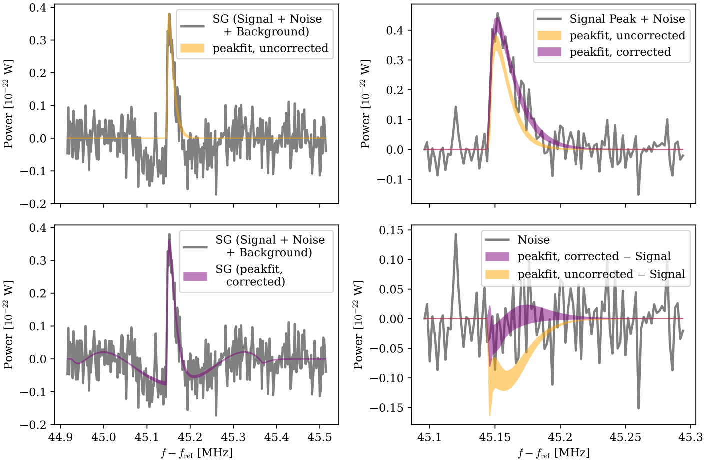
ZEUS ep-collision parton PDF fit



QCDNUM (Fortran) wrapped in Julia [PRL.130.141901]

Sampled with adaptive Metropolis-Hastings backend.

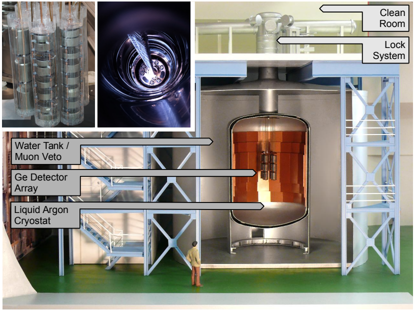
MADMAX simulated peak BG



Sampled with Ultranest backend

[arXiv 2306.17667]

Final Results of GERDA

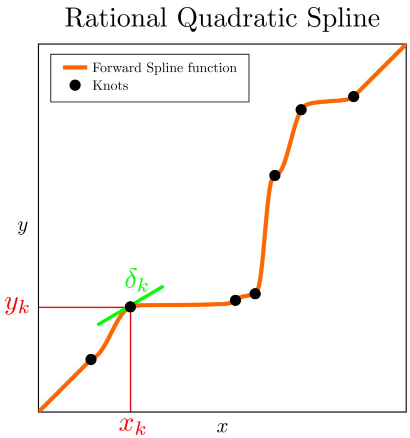


- $T^{0\nu}_{1/2} > 1.4 \times 10^{26}$ yr (90% CI) (equiprobable signal strengths)
- $T^{0\nu}_{1/2} > 2.3 \times 10^{26}$ yr (90% CI) (equiprobable Majorana neutrino masses)

Hierarchical prior,
sampled with adaptive Metropolis-Hastings backend.

[PRL 125, 252502 (2020)]

Monotone Rational-Quadratic Splines



K Segments

Characterized by

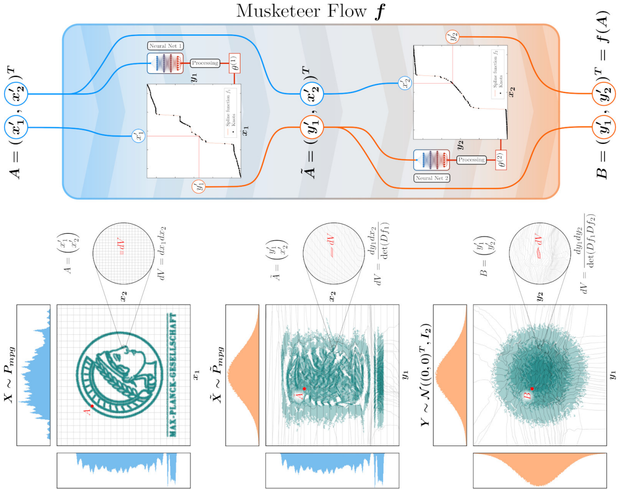
$\{\boldsymbol{x_k}, \boldsymbol{y_k}\}, \{\boldsymbol{\delta_k}\}$

[Conor Durkan et al. *Neural Spline Flows*]

MonoticSplines.jl: Based on "Neural Spline Flows" [NeurIPS 2019], high-performance CPU+GPU via KernelAbstractions.jl.

Max-Planck-Institut für Physik
(Werner-Heisenberg-Institut)

Spline flows for low-dim marginals



Could be a nice tool to pass marginal posteriors around (once trained, math is quite simple).

Conclusions and Outlook

- BAT concept: user brings domain knowledge and likelihood, BAT provides sampling, integration and visualization
- BAT.jl v3.x releases will gradually add more "measure language" in API:

$$\int_B \alpha_b(A)\,\mathrm{d}\,\bar{\beta} = P(A \times B) = \int_A \beta_a(B)\,\mathrm{d}\,\bar{\alpha}$$

$$\alpha_b(A) = \int_A \frac{\mathrm{d}\,\beta_a}{\mathrm{d}\,\bar{\beta}}(b)\mathrm{d}\,\bar{\alpha}(a)\,, \quad \bar{\beta}(B) = \int_{\mathcal{A}} \beta_a(B)\,\mathrm{d}\,\bar{\alpha}$$

- In progress: Switch from tuning MCMC proposals to tuning space transformations
- Next sampler (we hope): Dynamic space transformations via RQS normalizing flows during algorithm tuning
- ToDo: Add SciMAL optimization and integration algorithms