Status update C1:
Monte Carlo Benchmark Suite

Salvatore La Cagnina

March 13th

KISS Annual Meeting 2025

technische universität
dortmund

KISS

Quality Control for Monte Carlo Samplers

- Several sampling solutions exist to solve numerical problems
- Samplers often developed and tested for specific problems

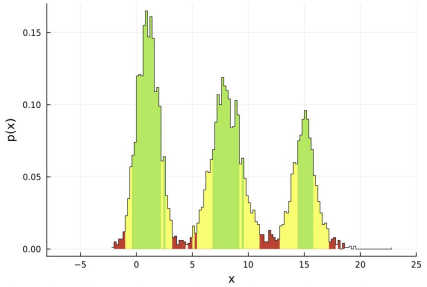
  → Challenge to find the correct sampler for different use cases

- Goal: Universal framework for quality tests of Monte Carlo samplers
  - Collect test functions/problems
  - Collect metrics and samplers
  - Establish a database for comparison/validation of sampling quality

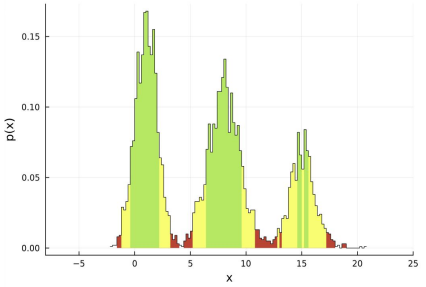
- How do we compare samples ?
- General idea: build test statistics for each metric and problem using IID/truth
  - Compare to metrics (or their distribution) of sampled distributions

Which sampler is “wrong”?

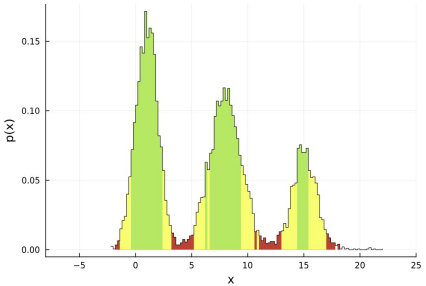
a)



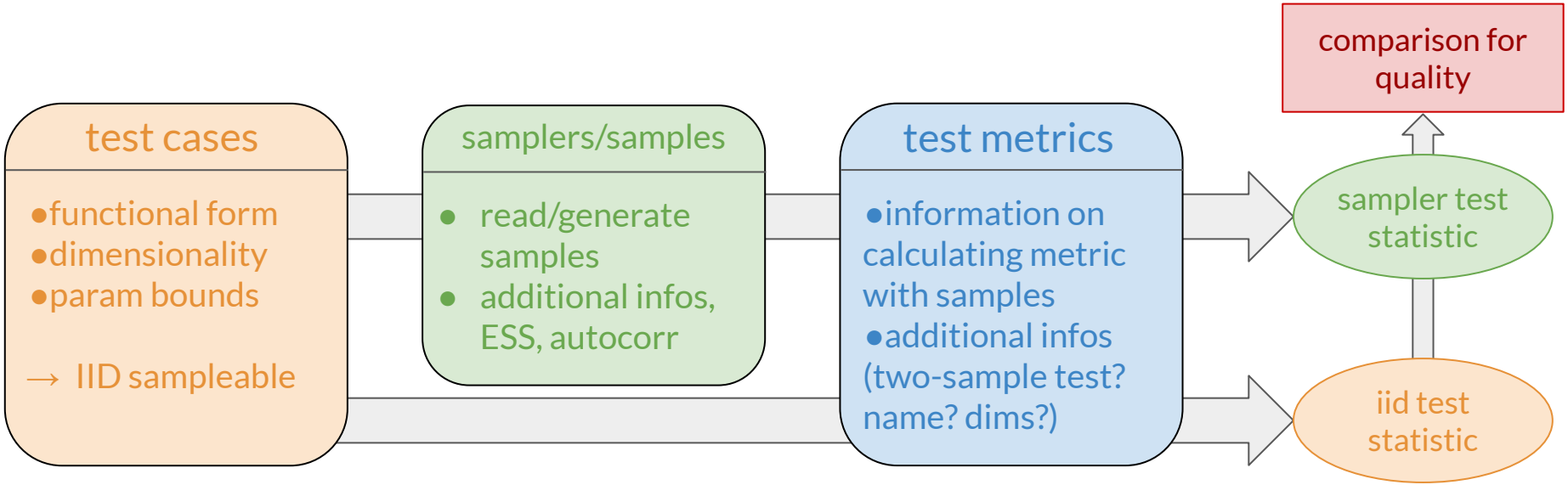
b)



c)



Design Concepts of Test Suite



- Main focus on easy expandability for samplers, test functions and test metrics
- IID sampling for comparisons of test metrics

MCBench - Quality Control for Monte Carlo Samplers

- Collection of Test Functions
- (easy expandable) collection of metrics (both one-sample and two-sample metrics)
- for more infos see the github documentation

Name	Equation	Properties
Standard Normal 1D	$f(x \mid \mu, \sigma) = \frac{1}{\sqrt{2\pi}\sigma} \cdot e^{-\frac{(x-\mu)^2}{2\sigma^2}}$	Unimodal
Standard Normal kD uncorrelated	$\mathcal{N}_k(\mu, \Sigma) = (2\pi)^{-k/2} \det(\Sigma)^{-1/2} \cdot \exp\left(-\frac{1}{2}(x-\mu)^\top \Sigma^{-1}(x-\mu)\right)$	Unimodal, $k = 2, 3, 10, 100$
Standard Normal kD weakly/strongly correlated	$\mathcal{N}_k(\mu, \Sigma)$ $\Sigma = r \cdot \mathbf{J} + (1-r) \cdot \mathbf{I}$	$r = 0.2, 0.9$ $k = 2, 10, 100$
Mixture Normal kD strongly correlated	$f(x \mid \mu, \Sigma) = 0.25\, \mathcal{N}_k(\mu, \Sigma) + 0.75\, \mathcal{N}_k(-\mu, \Sigma)$ $\Sigma = r \cdot \mathbf{J} + (1-r) \cdot \mathbf{I}$	Multimodal $r = 0.9, k = 3, 10$
Cauchy 1D	$f(x \mid x_0, \gamma) = \frac{1}{\pi\gamma} \cdot \frac{1}{1 + \left(\frac{x-x_0}{\gamma}\right)^2}$	Unimodal
Eight schools example	$L(\theta \mid y) = \prod_{i=1}^{8} \frac{1}{\sqrt{2\pi\sigma_i^2}} \exp\left(-\frac{(y_i - \theta_i)^2}{2\sigma_i^2}\right)$	For more details see Section 3

https://arxiv.org/abs/2501.03138

README MIT license

MCBench - Monte Carlo Sampling Benchmark Suite

Documentation dev license MIT CI passing

The **MCBench** benchmark suite is designed for quantitative comparisons of Monte Carlo (MC) samples. It offers a standardized method for evaluating MC sample quality and provides researchers and practitioners with a tool for validating, developing, and refining MC sampling algorithms.

Read more about MCBench at https://arxiv.org/abs/2501.03138

Installation

To use MCBench you need a Julia installation. We recommend to use Julia 1.10 or above. Install MCBench via the Julia package manager by running

```
using Pkg
pkg"add https://github.com/tudo-physik-e4/MCBench"
```

https://github.com/tudo-physik-e4/MCBench

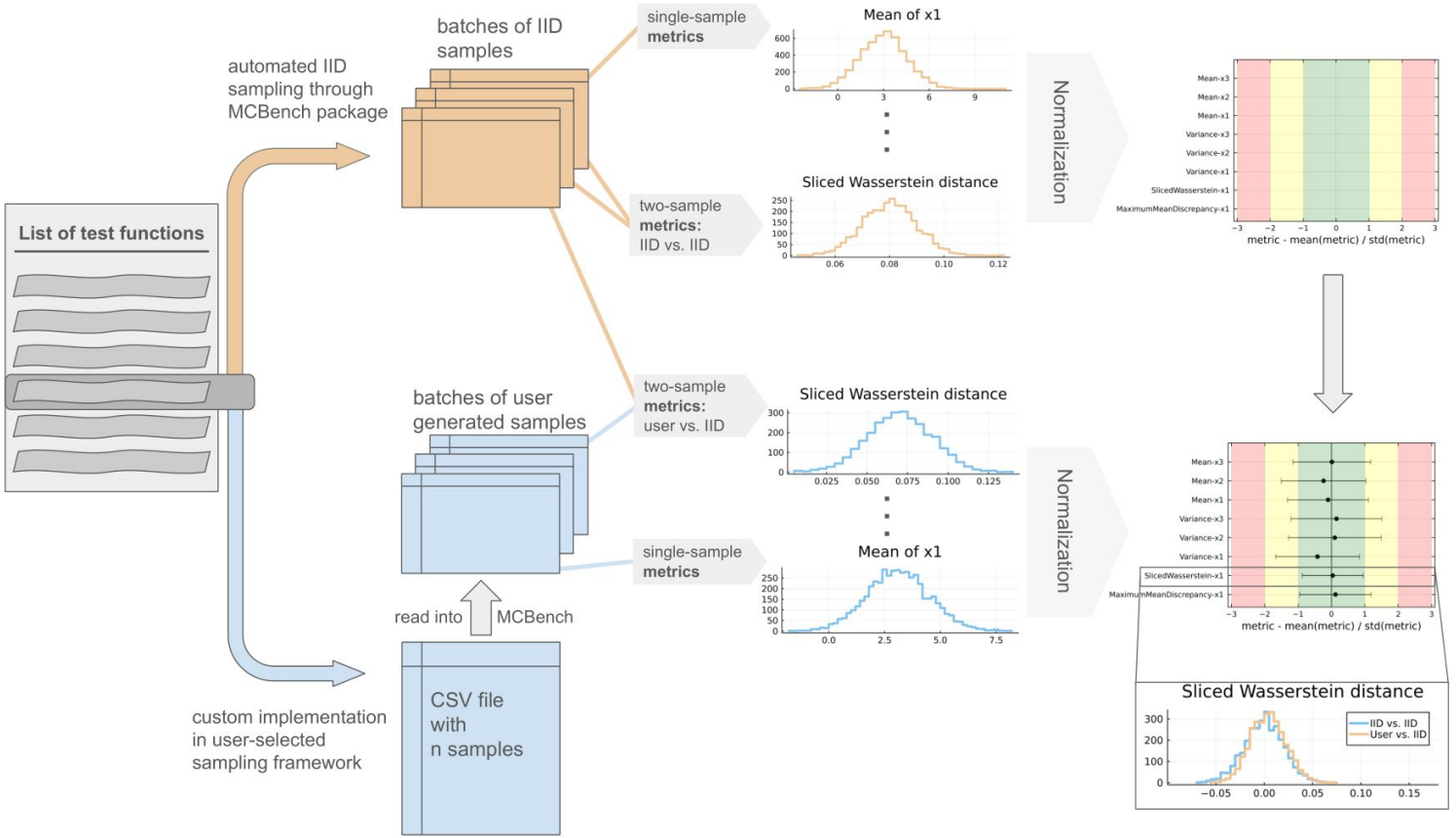
One-sample metrics

- Marginal mean: `marginal_mean()`
- Marginal variance: `marginal_variance()`
- Global mode: `global_mean()`
- Marginal mode: `marginal_mode()`
- Marginal skewness: `marginal_skewness()`
- Marginal kurtosis: `marginal_kurtosis()`

Two-sample metric

- Chi-squared: `chi_squared()`
- Sliced Wasserstein Distance: `sliced_wasserstein_distance()`
- Maximum Mean Discrepancy: `maximum_mean_discrepancy()`

Test Suite Workflow



Simple Example, 3D Unit Normal

- Defining Test case (using the Distributions.jl package):

```
f = MvNormal([0.0,0.0,0.0],I(3))
bounds = NamedTupleDist(x = fill(-10..10,3))
normaltestcase = testcases(f,bounds,3,"3DNormal")
```

[4] ✓ 0.7s Julia

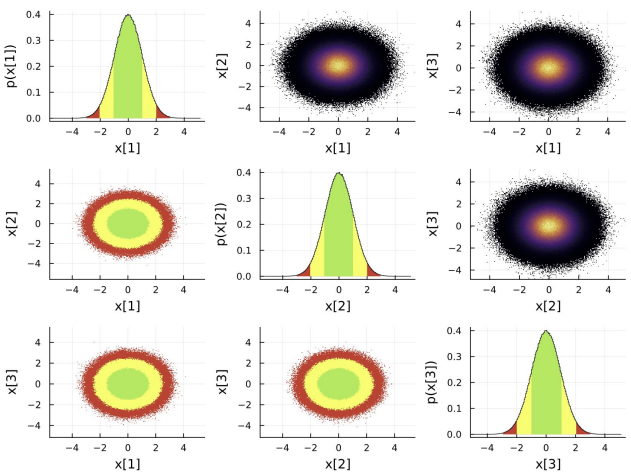
```
testcases{DiagNormal, NamedTupleDist{(:x,), Tuple{Product{Continuous, Uniform{Float64}, Vector{Uniform{Fl
dim: 3
μ: [0.0, 0.0, 0.0]
Σ: [1.0 0.0 0.0; 0.0 1.0 0.0; 0.0 0.0 1.0]
```

- Building test statistic for metrics with IID and BAT MH sampler

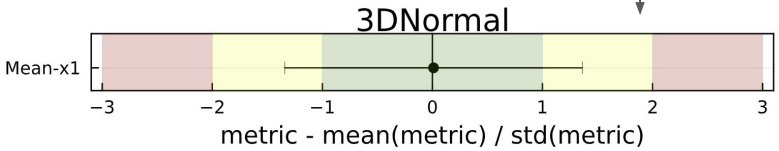
```
build_teststatistic(normaltestcase,marginal_mean())
build_teststatistic(normaltestcase,marginal_mean(),BATMH())
plot_metrics(normaltestcase,[marginal_mean()],BATMH())
```

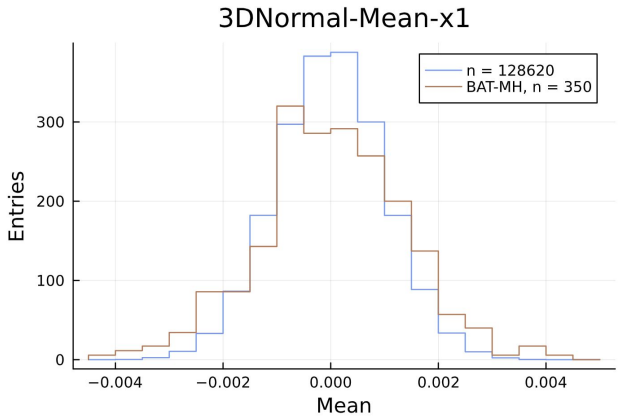
Julia

- Example: Mean of marginal distribution



- Normalize in terms of IID test statistic



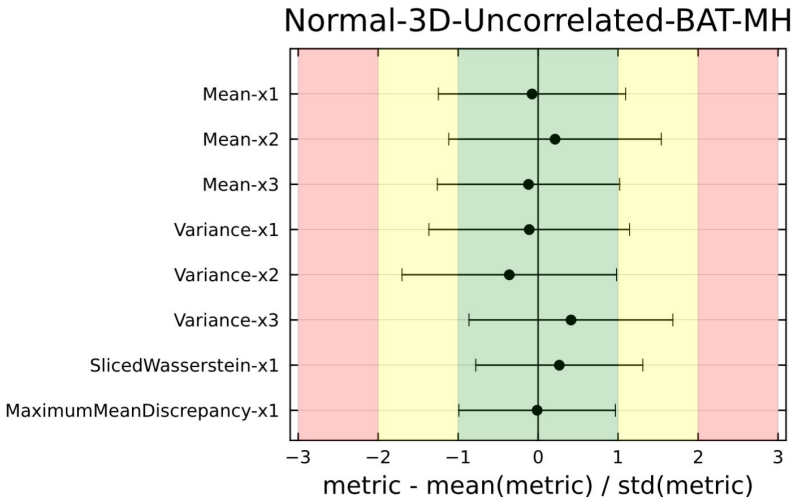


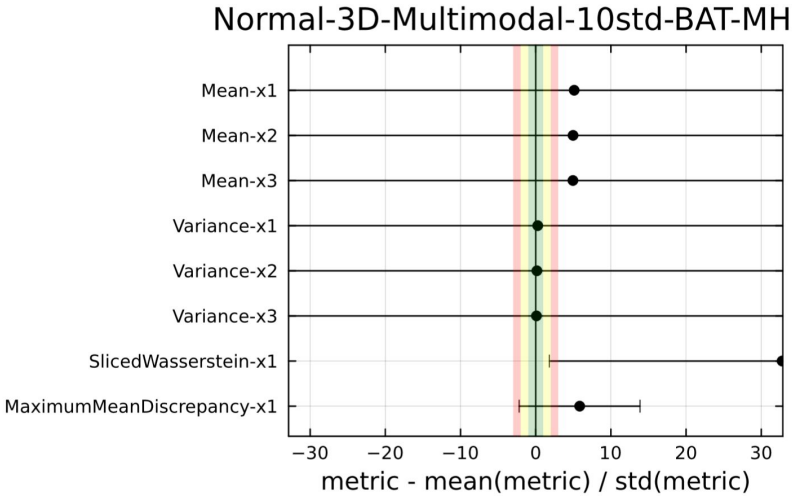
3D Normal vs mixture model

- Running testsuite using Metropolis-Hastings
- Two models in comparison
  - 3D standard normal
  - 3D mixture model with multiple modes far apart

```
r = 5
f1 = MvNormal(r*ones(10), ones(10,10)*0.9 + I(10)*0.1)
f2 = MvNormal(-r*ones(10), ones(10,10)*0.9 + I(10)*0.1)
f = MixtureModel([f1,f2], [0.25, 0.75])
bounds = NamedTupleDist(x = [-100..100 for i in 1:10])
normal_3d_multimodal_10std = Testcases(f,bounds,10,"Normal-3D-Multimodal-10std")
```

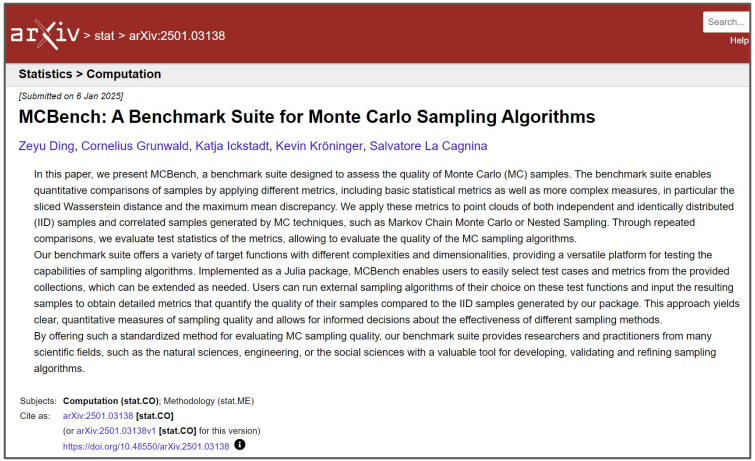
- Compare a set of metrics using both models
- Benchmark suite reflects difficulties in sampling multimodal distributions





Summary and Conclusions

- Developed a test suite (in the julia programming language
- Compare samplers to IID samples using metrics
- Provide a selection of (IID sampleable) test functions and (one and two-sample) metrics
- Visit our suite on github and paper on arxiv
  - https://github.com/tudo-physik-e4/MCBench
  - https://arxiv.org/abs/2501.03138



arXiv > stat > arXiv:2501.03138

Search... Help

Statistics > Computation

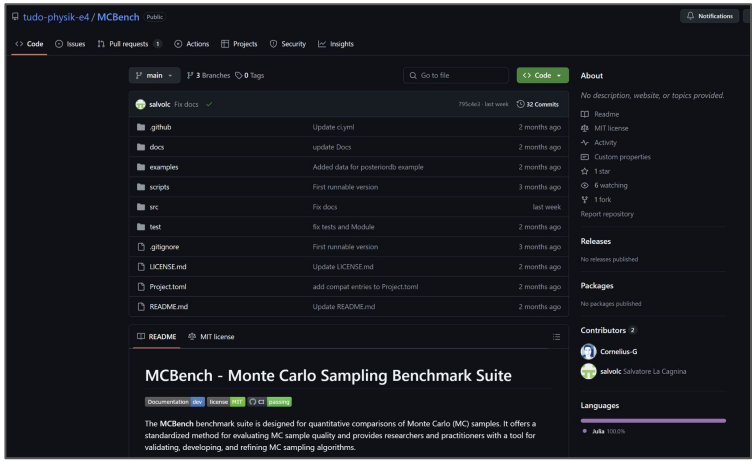
[Submitted on 6 Jan 2025]

MCBench: A Benchmark Suite for Monte Carlo Sampling Algorithms

Zeyu Ding, Cornelius Grunwald, Katja Ickstadt, Kevin Kröninger, Salvatore La Cagnina

In this paper, we present MCBench, a benchmark suite designed to assess the quality of Monte Carlo (MC) samples. The benchmark suite enables quantitative comparisons of samples by applying different metrics, including basic statistical metrics as well as more complex measures, in particular the sliced Wasserstein distance and the maximum mean discrepancy. We apply these metrics to point clouds of both independent and identically distributed (IID) samples and correlated samples generated by MC techniques, such as Markov Chain Monte Carlo or Nested Sampling. Through repeated comparisons, we evaluate test statistics of the metrics, allowing to evaluate the quality of the MC sampling algorithms.
Our benchmark suite offers a variety of target functions with different complexities and dimensionalities, providing a versatile platform for testing the capabilities of sampling algorithms. Implemented as a Julia package, MCBench enables users to easily select test cases and metrics from the provided collections, which can be extended as needed. Users can run external sampling algorithms of their choice on these test functions and input the resulting samples to obtain detailed metrics that quantify the quality of their samples compared to the IID samples generated by our package. This approach yields clear, quantitative measures of sampling quality and allows for informed decisions about the effectiveness of different sampling methods.
By offering such a standardized method for evaluating MC sampling quality, our benchmark suite provides researchers and practitioners from many scientific fields, such as the natural sciences, engineering, or the social sciences with a valuable tool for developing, validating and refining sampling algorithms.

Subjects: **Computation (stat.CO)**; Methodology (stat.ME)
Cite as: arXiv:2501.03138 **[stat.CO]**
(or arXiv:2501.03138v1 **[stat.CO]** for this version)
https://doi.org/10.48550/arXiv.2501.03138



tudo-physik-e4 / MCBench Public

Notifications

Code Issues Pull requests Actions Projects Security Insights

main 3 Branches 0 Tags Go to file Code

About

No description, website, or topics provided.

Readme
MIT license
Activity
Custom properties
1 star
6 watching
1 fork
Report repository

salvoic Fix docs 795cde3 · last week 32 Commits

.github	Update ci.yml	2 months ago
docs	update Docs	2 months ago
examples	Added data for posteriordb example	2 months ago
scripts	First runnable version	3 months ago
src	Fix docs	last week
test	fix tests and Module	2 months ago
.gitignore	First runnable version	3 months ago
LICENSE.md	Update LICENSE.md	2 months ago
Project.toml	add compat entries to Project.toml	2 months ago
README.md	Update README.md	2 months ago

Releases

No releases published

Packages

No packages published

Contributors 2

Cornelius-G
salvoic Salvatore La Cagnina

README MIT license

MCBench - Monte Carlo Sampling Benchmark Suite

Documentation dev license MIT passing

The **MCBench** benchmark suite is designed for quantitative comparisons of Monte Carlo (MC) samples. It offers a standardized method for evaluating MC sample quality and provides researchers and practitioners with a tool for validating, developing, and refining MC sampling algorithms.

Languages

Julia 100.0%

Next Steps:

- Add full test case support for different platforms (R, stan, pymc) including testpoints
- Lookout to include more complex test cases and applications