Status update C1: Monte Carlo Benchmark Suite

Salvatore La Cagnina

February 13th

KISS Annual Meeting 2024

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

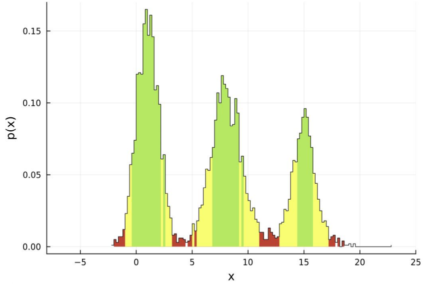
  → Challenge to find the correct sampler for different use cases

- 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

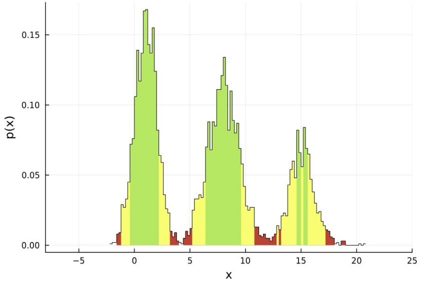
- How do we compare samples ?
- General idea: build test statistics for each metric and problem using IID/truth
  - Compare to metrics (or their distribution) of sampled distributions

Which sampler is “wrong”?

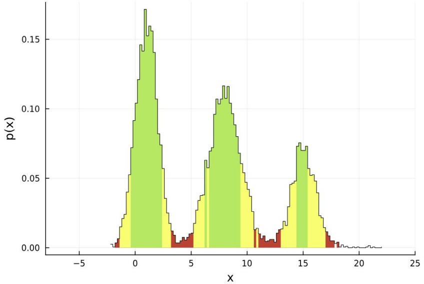
a)



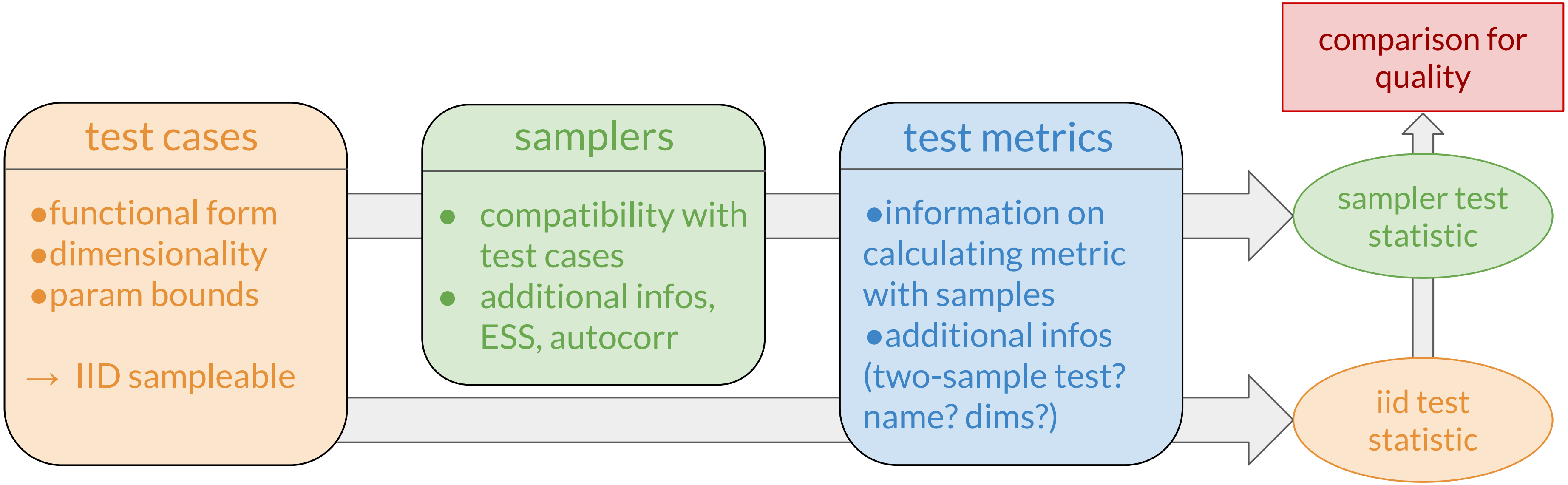
b)



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

Simple Example, 3D Unit Normal

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

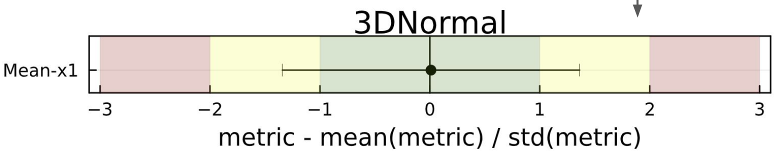
```
f = MvNormal([0.0,0.0,0.0],I(3))
bounds = NamedTupleDist(x = fill(-10..10,3))
normaltestcase = testcases(f,bounds,3,"3DNormal")
```

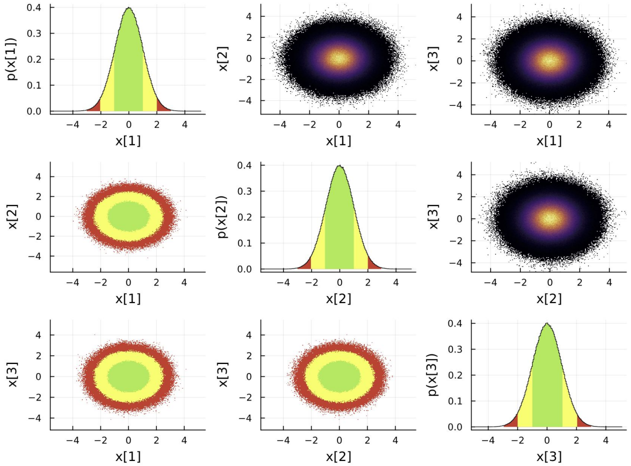
```
testcases{DiagNormal, NamedTupleDist{(:x,), Tuple{Product{Continuous, Uniform{Float64}, Vector{Uniform{Fl
dim: 3
μ: [0.0, 0.0, 0.0]
Σ: [1.0 0.0 0.0; 0.0 1.0 0.0; 0.0 0.0 1.0]
```

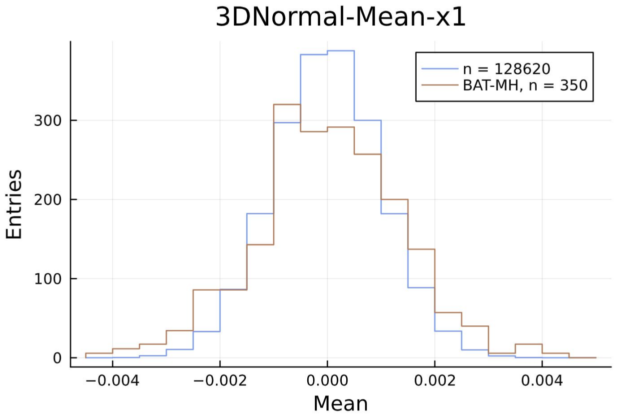
- Building test statistic for metrics with IID and BAT MH sampler

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

- Example: Mean of marginal distribution
    - Normalize in terms of IID test statistic

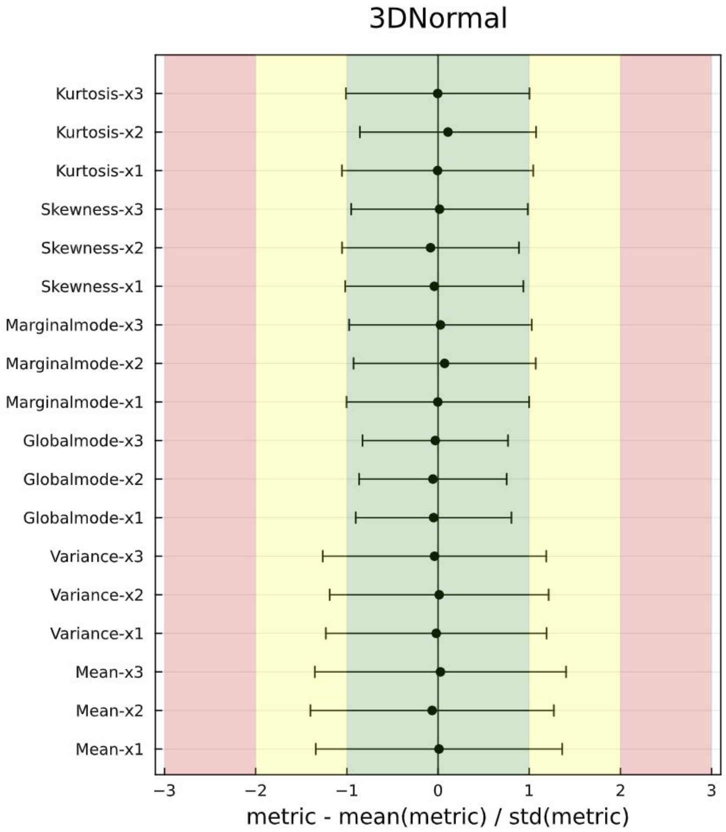






Simple Example: 3D Unit Normal

- Overview of all metrics for one sampler at a time:



Next Steps:

- Test case/samplers:
  - Collect more (complex) test functions
  - Run tests with different samplers
- Test Metrics:
  - So far: 1D marginal metrics
  - Use two sample tests like chi squared
  - Multidimensional point cloud comparisons?

⇨ With Wasserstein distance?

technische universität dortmund

Introduction to sliced-Wasserstein distance

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

February 13, 2024

Introduction to Wasserstein Distance

The Wasserstein distance, also known as Earth Mover's Distance (EMD), is a measure of the distance between two probability distributions over a metric space.

$$W_p(\mu, \nu) = \left(\inf_{\gamma \in \Gamma(\mu,\nu)} \int_{\mathbb{R}^d \times \mathbb{R}^d} \|x - y\|^p d\gamma(x, y) \right)^{\frac{1}{p}}$$

where:

- μ and ν are the probability measures.
- $\Gamma(\mu, \nu)$ is the set of all joint distributions γ with marginals μ and ν.
- $\|x - y\|$ is the distance between points in support of μ and ν.

What is Sliced Wasserstein Distance?

- Sliced Wasserstein Distance (SWD) provides a way to calculate the distance between high-dimensional probability distributions.
- It simplifies the calculation by projecting high-dimensional distributions onto multiple one-dimensional spaces.
- Particularly useful in applications like computer vision and machine learning.
- SWD is defined as the average Wasserstein distance between the one-dimensional projections of the distributions.

Algorithm 1: Computational algorithm of the SW distance

Input: Probability measures μ and ν, $p \geq 1$, and the number of projections L.

for $l = 1$ *to* L **do**

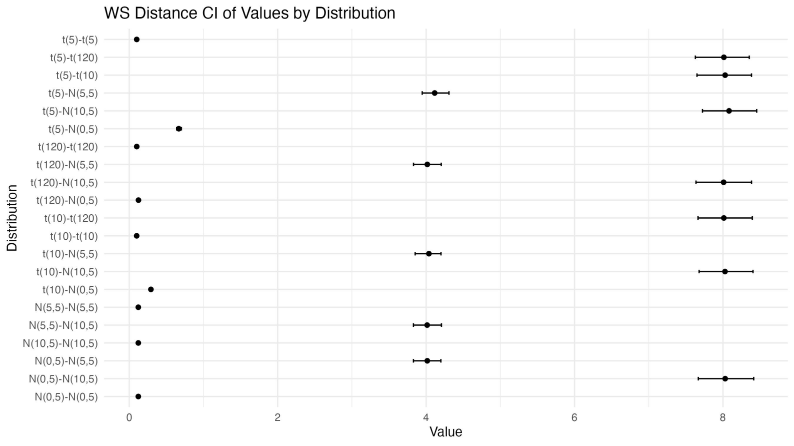
  Sample $\theta_l \sim U(S^{d-1})$

  Compute $v_l = W_p(\theta_l \# \mu, \theta_l \# \nu)$

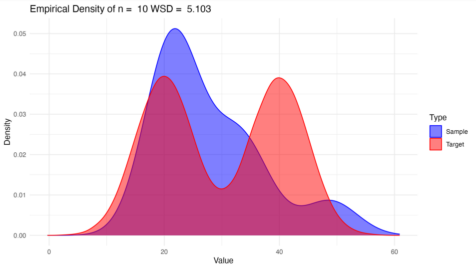
Compute $SW_p(\mu, \nu; L) = \left(\frac{1}{L}\sum_{l=1}^{L} v_l^p\right)^{\frac{1}{p}}$

Output: $SW_p(\mu, \nu; L)$

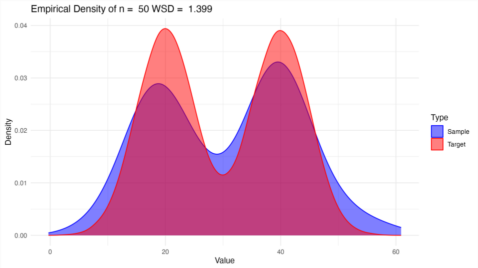
Wasserstein distance of different distributions



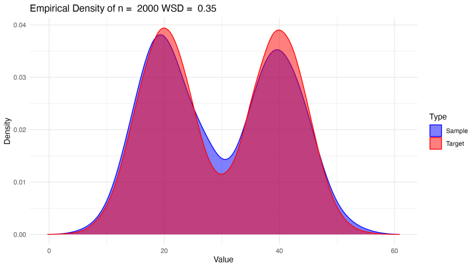
Sampling density and WS distance

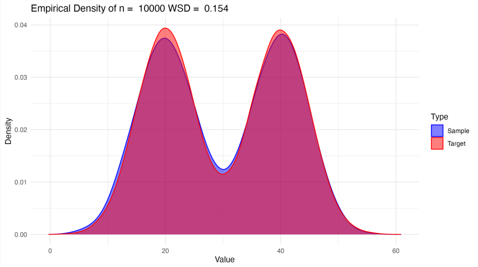


(a)



(b)





Sampling density and WS distance