Towards a turnkey approach to unbiased Monte Carlo estimation of smooth functions of expectations

Francesca R. Crucinio

King's College London

Introduction

Joint work with Nicolas Chopin & Sumeetpal S. Singh

The math problem

Given:

- A function f

The math problem

Given:

- A function f
- a stream of IID random variables $X_1, X_2, ...$ with expectation m and variance σ^2

The math problem

Given:

- A function f
- a stream of IID random variables $X_1, X_2, \dots$ with expectation m and variance σ^2

The math problem

Given:

- A function f
- a stream of IID random variables $X_1, X_2, \ldots$ with expectation m and variance σ^2

generate unbiased estimates of $f(m)$.

Motivation I: Log and MLE

In many situations (e.g. latent variable models), we have access to unbiased estimates of the likelihood $p(y|\theta)$, but we would like unbiased estimates of the log-likelihood (and its gradient).

Motivation I: Log and MLE

In many situations (e.g. latent variable models), we have access to unbiased estimates of the likelihood $p(y|\theta)$, but we would like unbiased estimates of the log-likelihood (and its gradient).

Often, unbiased estimates of $p(y|\theta)$ are available, but we would like to estimate unbiasedly $\log p(y|\theta)$, e.g. to use within **gradient descent**.

Motivation II: Reciprocal and un-normalised models

A model whose likelihood is of the form:

$$p(y|\theta) = g(y, \theta)/Z(\theta)$$

where $Z(\theta)$ is intractable.

Motivation II: Reciprocal and un-normalised models

A model whose likelihood is of the form:

$$p(y|\theta) = g(y, \theta)/Z(\theta)$$

where $Z(\theta)$ is intractable.

Often, unbiased estimates of $Z(\theta)$ are available, but we would like to estimate unbiasedly $1/Z(\theta)$, e.g. to implement a **pseudo-marginal MCMC** sampler.

Taylor expansion

Taylor expansion of f around some x_0:

$$f(m) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(m - x_0)^k$$
$$= \sum_{k=0}^{\infty} \gamma_k \left(\frac{m}{x_0} - 1\right)^k$$

where $\gamma_k := f^{(k)}(x_0)x_0^k/k!$.

Taylor expansion

Taylor expansion of f around some x_0:

$$f(m) = \sum_{k=0}^{\infty} \frac{f^{(k)}(x_0)}{k!}(m - x_0)^k$$
$$= \sum_{k=0}^{\infty} \gamma_k \left(\frac{m}{x_0} - 1\right)^k$$

where $\gamma_k := f^{(k)}(x_0)x_0^k/k!$.

For $f(x) = \log(x)$, $\gamma_k = (-1)^{k-1}/k$ (sub-geometric). For $f(x) = 1/x$, $\gamma_k = (-1)^{k-1}$ (sub-geometric).

A Taylor-based sum estimator

This suggests using a **sum estimator** (McLeish, 2011; Glynn and Rhee, 2014; Rhee and Glynn, 2015):

$$\hat{f} = \sum_{k=0}^{R} \frac{\gamma_k U_{r,k}}{\mathbb{P}(R \geq k)} = \sum_{k=0}^{\infty} \gamma_k U_{r,k} \frac{\mathbf{1}\{R \geq k\}}{\mathbb{P}(R \geq k)}$$

where R takes values in $\mathbb{N} = \{0, 1, ...\}$, and

$$U_{r,k} = \prod_{i=1}^{k} (X_i / x_0 - 1).$$

Main points to address

Robust/automatic guidelines on how to choose:

- x_0

Main points to address

Robust/automatic guidelines on how to choose:

- x_0
- the distribution of R

Main points to address

Robust/automatic guidelines on how to choose:

- x_0
- the distribution of R
- the form of the $U_{r,k}$'s (see later).

Objectives

- variance of estimate should be finite;

Objectives

- variance of estimate should be finite;
- CPU time is random; make sure its variance is also finite (and has light tails).

Objectives

- variance of estimate should be finite;
- CPU time is random; make sure its variance is also finite (and has light tails).

Objectives

- variance of estimate should be finite;
- CPU time is random; make sure its variance is also finite (and has light tails).

The second point is related to the behaviour of

- **sum** of random variables (when estimates are computed sequentially)

Objectives

- variance of estimate should be finite;
- CPU time is random; make sure its variance is also finite (and has light tails).

The second point is related to the behaviour of

- **sum** of random variables (when estimates are computed sequentially)
- **max** of random variables (when estimates are computed in parallel)

Theoretical properties

Variance decomposition

$$\mathrm{var}[\hat{f}] = \mathrm{var}\left[\mathbb{E}[\hat{f}|R]\right] + \mathbb{E}\left[\mathrm{var}[\hat{f}|R]\right]$$

where:

- the first term measures the variability induced by the random truncation.

Variance decomposition

$$\mathrm{var}[\hat{f}] = \mathrm{var}\left[\mathbb{E}[\hat{f}|R]\right] + \mathbb{E}\left[\mathrm{var}[\hat{f}|R]\right]$$

where:

- the first term measures the variability induced by the random truncation.
- the second term measures the variability due to the $U_{r,k}$'s (the unbiased estimates of $(m/x_0 - 1)^k$).

The first term

$$
\begin{aligned}
\operatorname{var}\left[\mathbb{E}[\hat{f}|R]\right] = \sum_{k=0}^{\infty} \gamma_k^2 \left(\frac{m}{x_0} - 1\right)^{2k} \left(\frac{1}{\mathbb{P}(R \geq k)} - 1\right) \\
+ 2 \sum_{k=0}^{\infty} \sum_{l=k+1}^{\infty} \gamma_k \gamma_l \left(\frac{m}{x_0} - 1\right)^{k+l} \left(\frac{1}{\mathbb{P}(R \geq k)} - 1\right).
\end{aligned}
$$

This expression suggests to take:

- x_0 such that $\beta_0^2 := |m/x_0 - 1| < 1$;

The first term

$$\operatorname{var}\left[\mathbb{E}[\hat{f}|R]\right] = \sum_{k=0}^{\infty} \gamma_k^2 \left(\frac{m}{x_0} - 1\right)^{2k} \left(\frac{1}{\mathbb{P}(R \geq k)} - 1\right)$$
$$+ 2 \sum_{k=0}^{\infty} \sum_{l=k+1}^{\infty} \gamma_k \gamma_l \left(\frac{m}{x_0} - 1\right)^{k+l} \left(\frac{1}{\mathbb{P}(R \geq k)} - 1\right).$$

This expression suggests to take:

- x_0 such that $\beta_0^2 := |m/x_0 - 1| < 1$;
- $R \sim \mathrm{Geometric}(p)$, with $p < 1 - \beta_0^2$, so that $\mathbb{P}(R \geq k) > \beta_0^{2k}$.

The first term: Limiting behaviour

If x_0 such that $\beta_0^2 := |m/x_0 - 1| < 1$ and $R \sim \mathrm{Geometric}(p)$, with $p < 1 - \beta_0^2$, then

$$\mathrm{var}\left[\mathbb{E}[\widehat{f}|R]\right] \to 0 \qquad \text{as } p \to 0.$$

(i.e. as computational cost increases)

The second term

$$\mathbb{E}\left[\mathrm{var}[\hat{f}|R]\right] = \sum_{k=0}^{\infty} \frac{\gamma_k^2}{\mathbb{P}(R \geq k)^2} \left\{ \sum_{r=k}^{\infty} \mathbb{P}(R = r)\, \mathrm{var}(U_{r,k}) \right\}$$

$$+2 \sum_{k=0}^{\infty} \sum_{l=k+1}^{\infty} \frac{\gamma_k \gamma_l}{\mathbb{P}(R \geq k)\mathbb{P}(R \geq l)} \left\{ \sum_{r=l}^{\infty} \mathbb{P}(R = r)\, \mathrm{cov}\left(U_{r,k}, U_{r,l}\right) \right\}.$$

This expression suggests to take:

- x_0 such that $\beta^2 := \frac{\sigma^2}{x_0^2} + \beta_0^2 < 1$;

The second term

$$\mathbb{E}\left[\mathrm{var}[\hat{f}|R]\right] = \sum_{k=0}^{\infty} \frac{\gamma_k^2}{\mathbb{P}(R \geq k)^2} \left\{ \sum_{r=k}^{\infty} \mathbb{P}(R = r) \, \mathrm{var}(U_{r,k}) \right\}$$

$$+2 \sum_{k=0}^{\infty} \sum_{l=k+1}^{\infty} \frac{\gamma_k \gamma_l}{\mathbb{P}(R \geq k)\mathbb{P}(R \geq l)} \left\{ \sum_{r=l}^{\infty} \mathbb{P}(R = r) \, \mathrm{cov}\left(U_{r,k}, U_{r,l}\right) \right\}.$$

This expression suggests to take:

- x_0 such that $\beta^2 := \frac{\sigma^2}{x_0^2} + \beta_0^2 < 1$;
- $R \sim \mathrm{Geometric}(p)$, with $p < 1 - \beta^2$, so that $\mathbb{P}(R \geq k) > \beta^{2k}$.

Improving the $U_{r,k}$

Currently, we use the following unbiased estimate of $(m/x_0 - 1)^k$:

$$U_{r,k} = U_k = \prod_{i=1}^{k} (X_i/x_0 - 1).$$

Improving the $U_{r,k}$

Currently, we use the following unbiased estimate of $(m/x_0 - 1)^k$:

$$U_{r,k} = U_k = \prod_{i=1}^{k} (X_i/x_0 - 1).$$

Computing $\hat{f}$ requires generating $X_1, \ldots, X_R$ in order to compute the last term $U_{R,R}$, but we use only the k first inputs to estimate $(m/x_0 - 1)^k$. Seems inefficient.

Cycling estimator

In order to use the **whole sample**, consider the following cycling estimator:

$$U_{r,k}^{\mathrm{C}} := \frac{1}{r}\Big[\prod_{i=1}^{k}\left(\frac{X_i}{x_0}-1\right)+\prod_{i=2}^{k+1}\left(\frac{X_i}{x_0}-1\right) \\ +\ldots+\left(\frac{X_r}{x_0}-1\right)\prod_{i=1}^{k-1}\left(\frac{X_i}{x_0}-1\right)\Big].$$

Second term of the decomposition: cycling

For the cycling estimator, one has (under weak assumptions)

$$\mathbb{E}\left[\mathrm{var}[\hat{f}^{\mathrm{C}}|R]\right] = \mathcal{O}(p\log(1/p))$$

as $p \to 0$.

- Average number of required inputs is $\mathbb{E}[R] \approx 1/p$.

Second term of the decomposition: cycling

For the cycling estimator, one has (under weak assumptions)

$$\mathbb{E}\left[\mathrm{var}[\hat{f}^{\mathrm{C}}|R]\right] = \mathcal{O}(p\log(1/p))$$

as $p \to 0$.

- Average number of required inputs is $\mathbb{E}[R] \approx 1/p$.
- We can have $\mathrm{var}[\hat{f}^{\mathrm{C}}] \to 0$ by taking $\mathbb{E}[R] \to +\infty$. (This is not the case for the simple estimator.)

Second term of the decomposition: cycling

For the cycling estimator, one has (under weak assumptions)

$$\mathbb{E}\left[\mathrm{var}[\hat{f}^{\mathrm{C}}|R]\right] = \mathcal{O}(p\log(1/p))$$

as $p \to 0$.

- Average number of required inputs is $\mathbb{E}[R] \approx 1/p$.
- We can have $\mathrm{var}[\hat{f}^{\mathrm{C}}] \to 0$ by taking $\mathbb{E}[R] \to +\infty$. (This is not the case for the simple estimator.)
- Up to log factor, standard Monte Carlo rate,
  i.e. $\mathrm{var}[\hat{f}^{\mathrm{C}}] = \mathcal{O}(\log \mathbb{E}[R]/\mathbb{E}[R])$.

Second term of the decomposition: without cycling

On the other hand,

$$\mathbb{E}\left[\operatorname{var}[\widehat{f}^{\mathrm{S}}|R]\right] \to v > 0$$

as $p \to 0$.

The simple estimator does not converge as $\mathbb{E}[R] \to +\infty$.

Calibration of tuning parameters

Tuning x_0

Must ensure that $\sigma^2/x_0^2 + |m/x_0 - 1|^2 < 1$, but m, σ^2 unknown.

Tuning x_0

Must ensure that $\sigma^2/x_0^2 + |m/x_0 - 1|^2 < 1$, but m, σ^2 unknown.

$\Rightarrow$ pilot run, bootstrap to ensure this condition with high probability.

Numerical experiments: log

log-likelihood (and gradient) of a latent variable model

Model with data y, latent z, parameter θ;

$$p(y|\theta) = \int p(y|z,\theta)p(z|\theta)dz$$

log-likelihood (and gradient) of a latent variable model

Model with data y, latent z, parameter θ;

$$p(y|\theta) = \int p(y|z,\theta)p(z|\theta)dz$$

For a fixed θ, **importance sampling** gives unbiased estimates of the likelihood:

$$X_i = w_i = \frac{p(y|Z_i,\theta)p(Z_i|\theta)}{q(Z_i)}, \quad Z_i \sim q$$

log-likelihood (and gradient) of a latent variable model

Model with data y, latent z, parameter θ;

$$p(y|\theta) = \int p(y|z,\theta)p(z|\theta)dz$$

For a fixed θ, **importance sampling** gives unbiased estimates of the likelihood:

$$X_i = w_i = \frac{p(y|Z_i,\theta)p(Z_i|\theta)}{q(Z_i)}, \quad Z_i \sim q$$

To compute the MLE, use our approach to derive unbiased estimate of the **log-likelihood** and its **gradient** (stochastic gradient descent).

Alternative approach: SUMO (Luo et al, 2021)

Consider biased (but consistent) IWAE estimate:

$$\ell_k(\theta) = \log\left(\frac{1}{k}\sum_{i=1}^{k} w_i\right)$$

The SUMO estimator is a sum estimator based on the series:

$$\log p(y|\theta) = \mathbb{E}[\ell_1(\theta)] + \sum_{k=1}^{\infty} \mathbb{E}[\Delta_k], \quad \Delta_k = \ell_{k+1}(\theta) - \ell_k(\theta)$$

Alternative approach: SUMO (Luo et al, 2021)

Consider biased (but consistent) IWAE estimate:

$$\ell_k(\theta) = \log\left(\frac{1}{k}\sum_{i=1}^{k} w_i\right)$$

The SUMO estimator is a sum estimator based on the series:

$$\log p(y|\theta) = \mathbb{E}[\ell_1(\theta)] + \sum_{k=1}^{\infty} \mathbb{E}[\Delta_k], \quad \Delta_k = \ell_{k+1}(\theta) - \ell_k(\theta)$$

Main issue: infinite variance.

Alternative approach: MLMC (Shi & Cornish, 2021)

Adaptation of SUMO, truncation at $R = 2^K$.

Alternative approach: MLMC (Shi & Cornish, 2021)

Adaptation of SUMO, truncation at $R = 2^K$.

Variance is finite, but the random CPU time may have infinite variance (and has always heavy tails).

Comparison on a toy model from Shi & Cornish (2021)

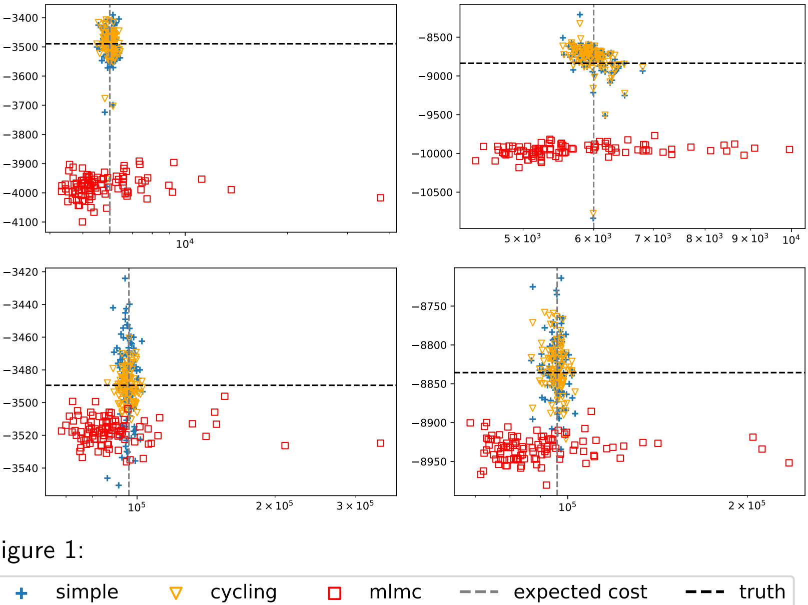

Figure 1:

Independent component analysis (Allassonnière and Younes, 2012)

$$y = \sum_{j=1}^{k} z_j a_j + \sigma\epsilon \quad \text{where}$$

- y is a high-dim object (e.g. image)

Independent component analysis (Allassonnière and Younes, 2012)

$$y = \sum_{j=1}^{k} z_j a_j + \sigma\epsilon \quad \text{where}$$

- y is a high-dim object (e.g. image)
- the z_j are independent latent variables

Independent component analysis (Allassonnière and Younes, 2012)

$$y = \sum_{j=1}^{k} z_j a_j + \sigma\epsilon \quad \text{where}$$

- y is a high-dim object (e.g. image)
- the z_j are independent latent variables
- $\theta = (A, \sigma)$, with $A = (a_j)_{j=1,\ldots,k}$.

Independent component analysis (Allassonnière and Younes, 2012)

$$y = \sum_{j=1}^{k} z_j a_j + \sigma\epsilon \quad \text{where}$$

- y is a high-dim object (e.g. image)
- the z_j are independent latent variables
- $\theta = (A, \sigma)$, with $A = (a_j)_{j=1,\ldots,k}$.

Independent component analysis (Allassonnière and Younes, 2012)

$$y = \sum_{j=1}^{k} z_j a_j + \sigma\epsilon \quad \text{where}$$

- y is a high-dim object (e.g. image)
- the z_j are independent latent variables
- $\theta = (A, \sigma)$, with $A = (a_j)_{j=1,\ldots,k}$.

Aim is to estimate θ using SGD.

Stochastic gradient descent

Given data $(y_1, \ldots, y_n)$, do gradient descent, where at each step, the gradient is replaced by an unbiased estimate of the gradient of a **single** term (chosen uniformly).

Stochastic gradient descent

Given data $(y_1, \dots, y_n)$, do gradient descent, where at each step, the gradient is replaced by an unbiased estimate of the gradient of a **single** term (chosen uniformly).

A good illustration on the need for automation, i.e. at each iteration, the actual value of m, σ^2, and thus x_0 and p must change.

Estimated images

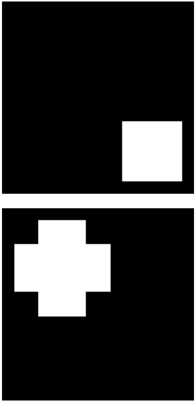

Figure 2: true

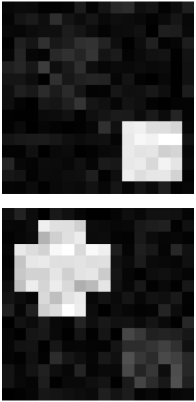

Figure 3: simple

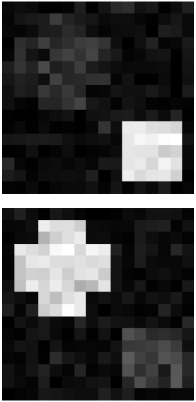

Figure 4: cycling

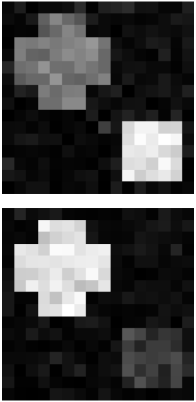

Figure 5: SAEM

Numerical experiments: reciprocal

Exponential random graph model

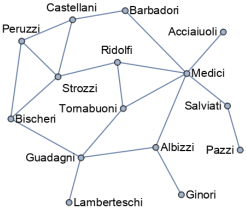

Figure 6: Florentine family business network

$$p(y|\theta) = \exp\{\theta^T s(y)\}/Z(\theta) \quad \text{where}$$

- $y = (y_{ij})$, with $y_{ij} = 1$ (resp. 0) if nodes i and j are connected

Exponential random graph model

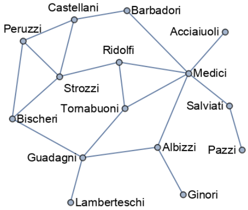

Figure 6: Florentine family business network

$$p(y|\theta) = \exp\{\theta^T s(y)\}/Z(\theta) \quad \text{where}$$

- $y = (y_{ij})$, with $y_{ij} = 1$ (resp. 0) if nodes i and j are connected
- $s(y)$ is a collection of network statistics (number of edges, number of $k-$stars, etc.)

Exponential random graph model

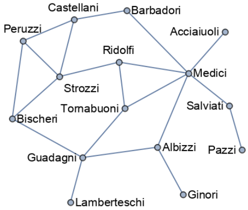

Figure 6: Florentine family business network

$$p(y|\theta) = \exp\{\theta^T s(y)\}/Z(\theta) \quad \text{where}$$

- $y = (y_{ij})$, with $y_{ij} = 1$ (resp. 0) if nodes i and j are connected
- $s(y)$ is a collection of network statistics (number of edges, number of $k-$stars, etc.)
- $Z(\theta)$ is a sum over $2^{\binom{k}{2}}$ terms (intractable)

Bayesian inference (and model choice)

Typically the dimension of θ is 2-3, so even importance sampling could work reasonably well to approximate the posterior:

- Sample $\theta_j \sim q$

Bayesian inference (and model choice)

Typically the dimension of θ is 2-3, so even importance sampling could work reasonably well to approximate the posterior:

- Sample $\theta_j \sim q$
- compute $w_j = \frac{p(\theta_j)\exp\{\theta_j^T s(y)\}}{q(\theta_j)} \times \frac{1}{Z(\theta_j)}$

Bayesian inference (and model choice)

Typically the dimension of θ is 2-3, so even importance sampling could work reasonably well to approximate the posterior:

- Sample $\theta_j \sim q$
- compute $w_j = \frac{p(\theta_j)\exp\{\theta_j^T s(y)\}}{q(\theta_j)} \times \frac{1}{Z(\theta_j)}$

Bayesian inference (and model choice)

Typically the dimension of θ is 2-3, so even importance sampling could work reasonably well to approximate the posterior:

- Sample $\theta_j \sim q$
- compute $w_j = \frac{p(\theta_j)\exp\{\theta_j^T s(y)\}}{q(\theta_j)} \times \frac{1}{Z(\theta_j)}$

Pseudo-marginal approach: replace $1/Z(\theta)$ by an unbiased estimate.

Details

For a fixed θ, run a tempering SMC algorithm to obtain an unbiased estimate of $Z(\theta)$.

Cycling vs simple estimator

Table 1: Efficiency and proportion of negative estimates out of 10^3 replicates in a moderate variance setting.

	Simple $\mathbb{E}[R] = 1$	Cycling $\mathbb{E}[R] = 1$	Simple $\mathbb{E}[R] = 10$	Cycling $\mathbb{E}[R] = 10$
WNV	111	102	61	24
$\mathbb{P}(-)$	0.026	0.028	0.027	0.01

The CPU cost of cycling is less than 0.01 seconds higher than that of the simple estimator.

Posterior approximation

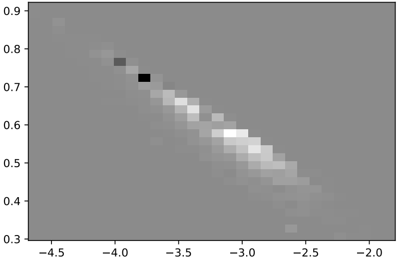

Figure 7: Simple

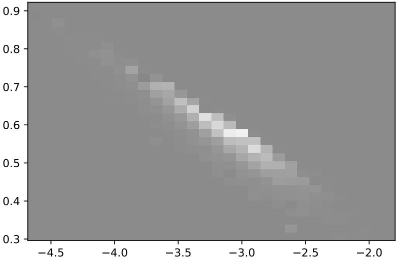

Figure 8: Cycling

Figure 9: Bivariate weighted histograms approximating the posterior distributions obtained with the simple and the cycling estimator using $n = 1024$ samples from proposal q.

Cycling gives better performance II

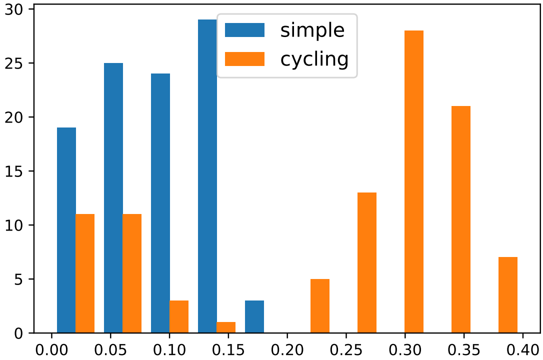

Figure 10: ESS for 100 repetitions.

Conclusion

Concluding remarks

- Debiasing estimators is hard

Concluding remarks

- Debiasing estimators is hard
- Many estimators do not guarantee finite variance

Concluding remarks

- Debiasing estimators is hard
- Many estimators do not guarantee finite variance
- ... or finite CPU cost

Concluding remarks

- Debiasing estimators is hard
- Many estimators do not guarantee finite variance
- ... or finite CPU cost
- ... or CPU cost with finite variance!

Concluding remarks

- A step towards making unbiased estimates of smooth function more **reliable** and **user-friendly**.

Concluding remarks

- A step towards making unbiased estimates of smooth function more **reliable** and **user-friendly**.
- No free lunch. Cannot work without pilot runs.

Concluding remarks

- A step towards making unbiased estimates of smooth function more **reliable** and **user-friendly**.
- No free lunch. Cannot work without pilot runs.
- Garbage in, garbage out: if the variance of the inputs is very large, the variance of our estimator will be large as well.

Paper

Chopin N., Crucinio F.R. and S. S. Singh (2024). Towards a turnkey approach to unbiased Monte Carlo estimation of smooth functions of expectations, arxiv 2403.20313.