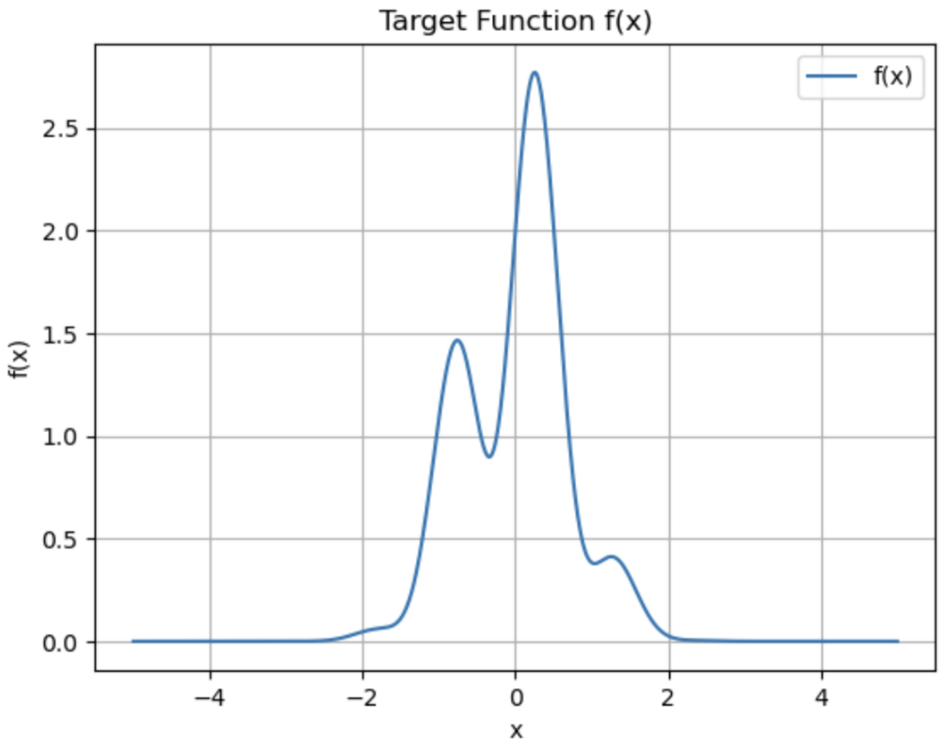
Adaptive Importance Sampling for Monte Carlo Integration

Numerical Methods, Spring 2025

Gandhar M. + Julia Z.

Motivating the Algorithm

Say you want to compute the integral of some wacky function:

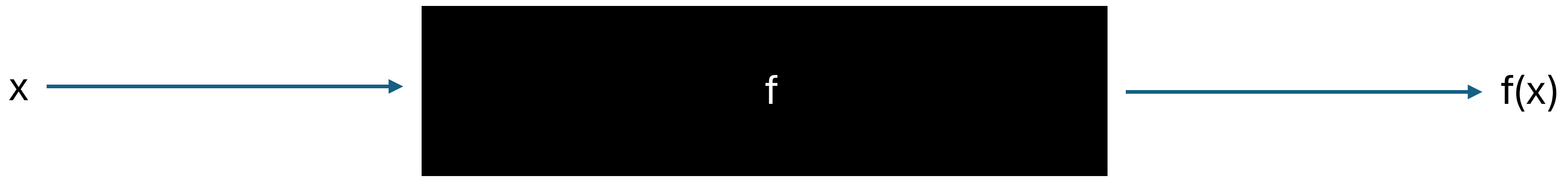


$$f(x) = (e^{-x^2}) \cdot (2 + sin(5x))$$

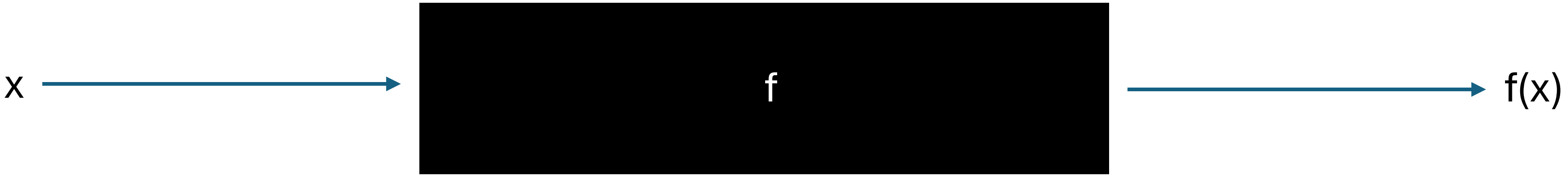
f

We want to estimate:

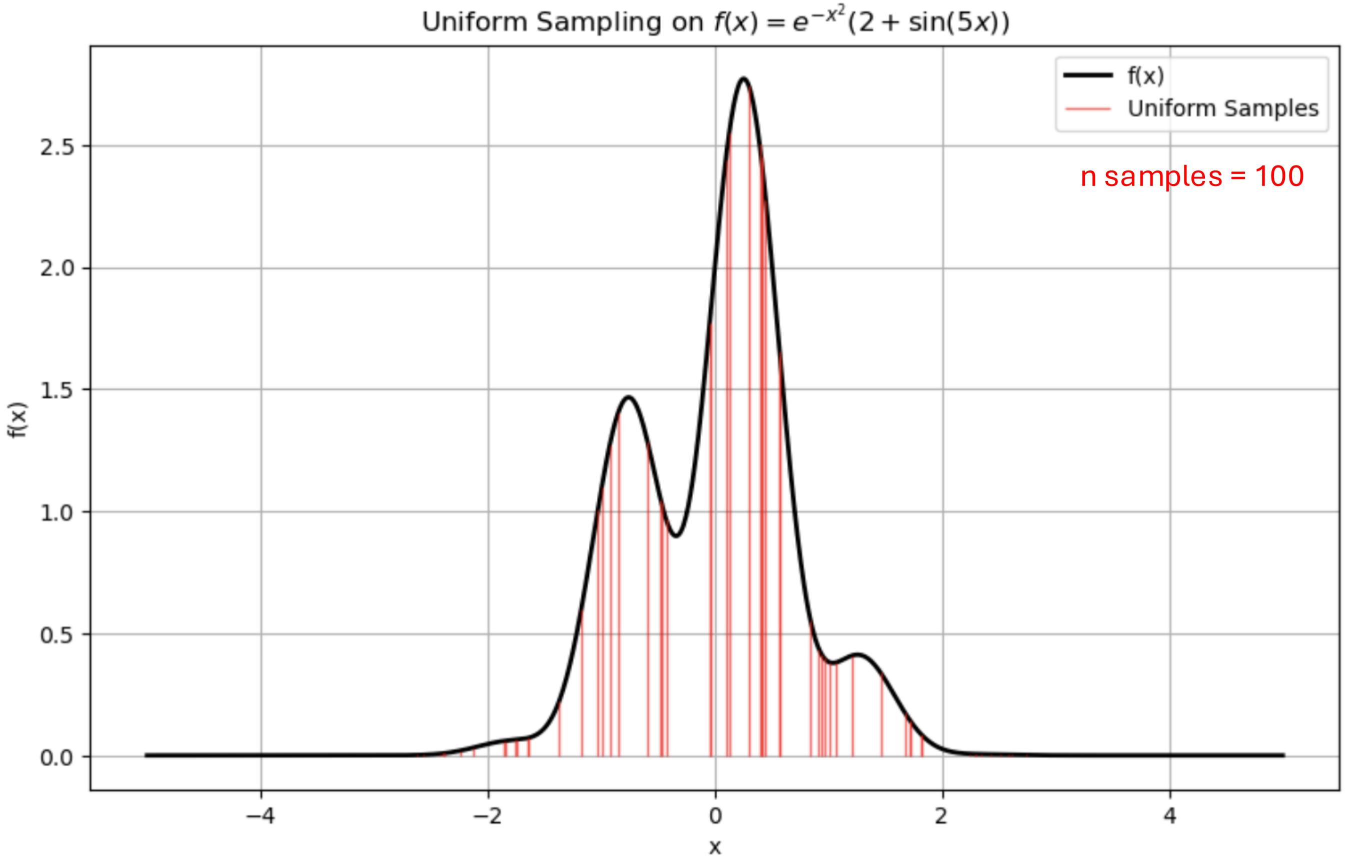
$$I = \int_{-\infty}^{\infty} f(x)$$

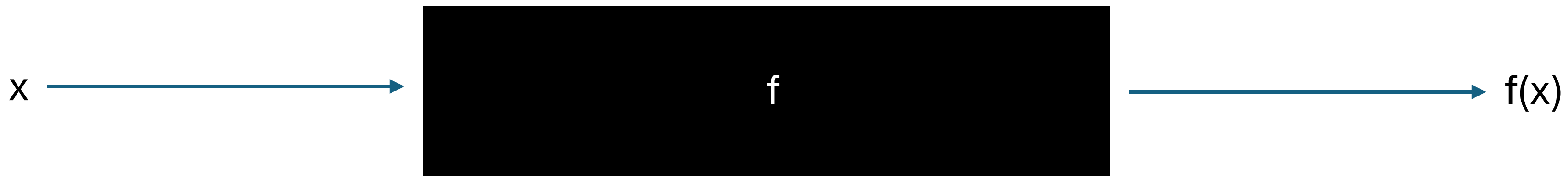


We want to estimate: $$I = \int_{-\infty}^{\infty} f(x)$$

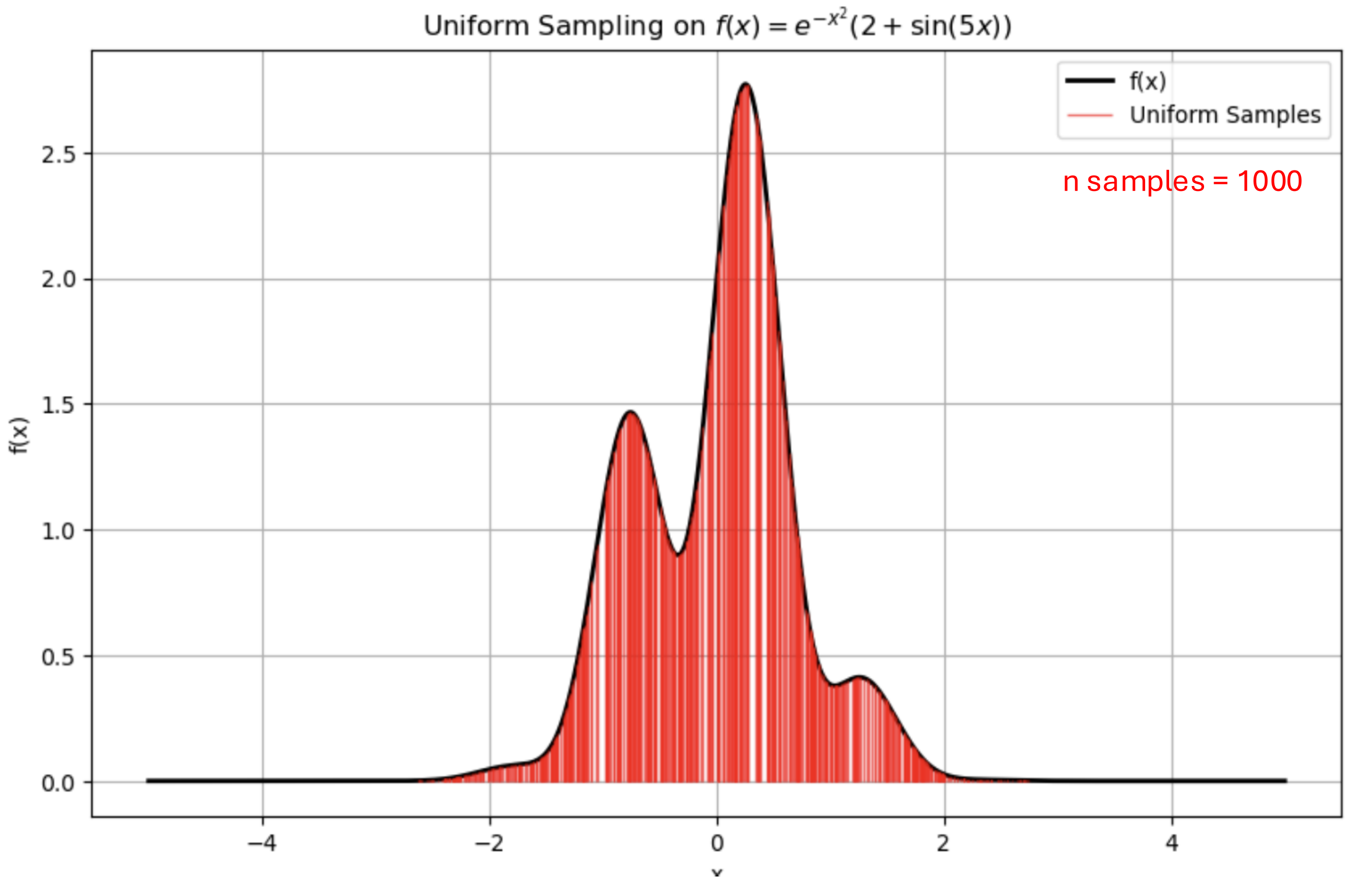


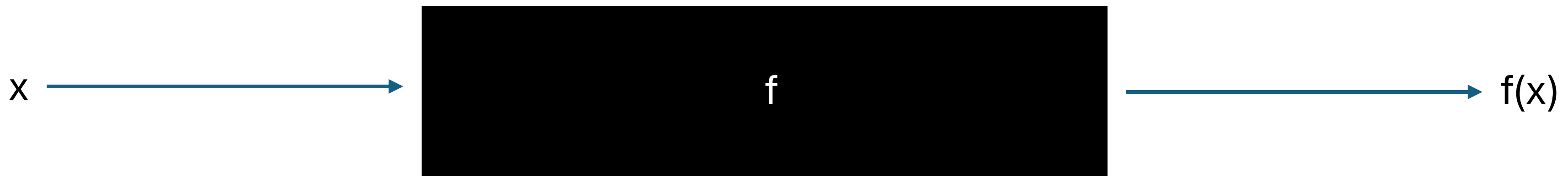
We want to estimate: $$I = \int_{-\infty}^{\infty} f(x)$$





We want to estimate: $$I = \int_{-\infty}^{\infty} f(x)$$





We want to estimate: $$I = \int_{-\infty}^{\infty} f(x)$$

More formally, this is an unbiased Monte Carlo estimator:

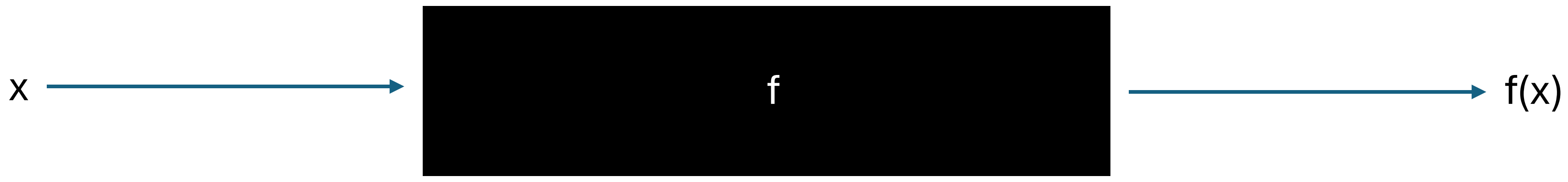
To evaluate $$I = \mathop{\mathbb{E}}_{X \sim \pi}[f(X)] = \int f(x)\pi(x)dx$$

where $\pi(x)$ is a target PDF
and $f(x)$ is the function of interest

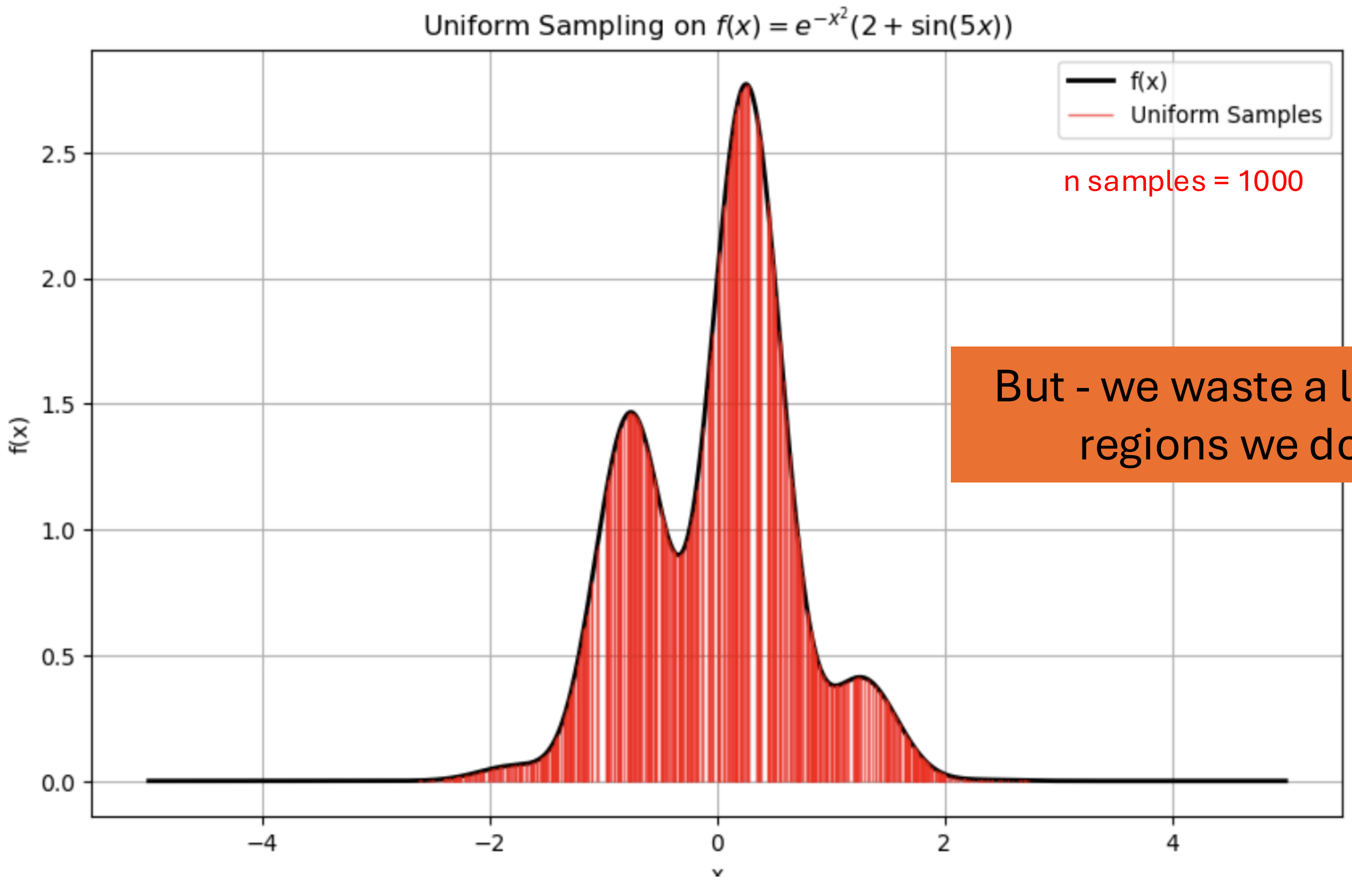
Draw N independent samples X_i from $\pi(x)$ and use $$I_N = \frac{1}{N}\sum_{i=1}^{N} f(X_i)$$

The estimator is unbiased and by the law of large numbers $I_N \to I$ as $N \to \infty$

Precision $\mathrm{Var}(I_N) = \mathrm{Var}_\pi[f(X)]/N$ which increases inversely with N



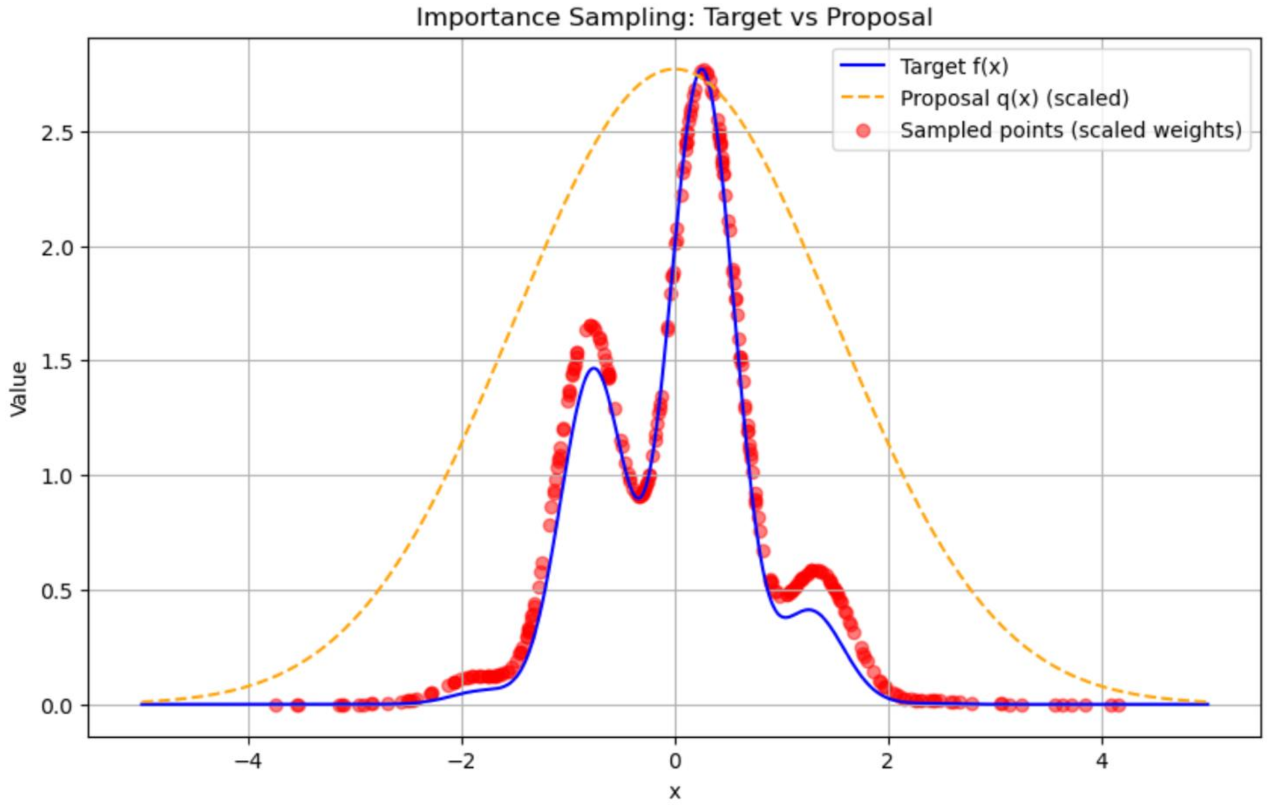
We want to estimate: $$I = \int_{-\infty}^{\infty} f(x)$$



But - we waste a lot of time sampling regions we don't care about!

Importance Sampling

Bias the sampler towards some “proposal distribution” q(x), then correct the bias:



Importance Sampling

More formally, $X_i \sim q(x)$

we compute $w_i = \frac{\pi(X_i)}{q(X_i)}$

$$I_N = \frac{1}{N} \sum_{i=1}^{N} w_i f(X_i)$$

Intuitively, samples drawn from regions where $q(x)$ over-samples relative to are down-weighted, and samples from regions that $q(x)$ under-sampled are up-weighted.

Why do we need Importance Sampling (IS)?

Difficult integrals.

If target distribution concentrates in a small region of the sample space, naive Monte Carlo that samples widely (or an ill-chosen MCMC that explores slowly) will waste many samples in negligible regions.

In Bayesian inference, importance sampling allows one to estimate properties of a posterior distribution by sampling from a simpler distribution (e.g. a multivariate normal approximation) when direct sampling from the posterior is hard

Challenge: Importance sampling's effectiveness depends entirely on the choice of the proposal distribution $q(x)$

Telltale signs of IS failing:

- weight degeneracy: variance blows up because effective sample size << N
- biased sampler if support of q doesn't adequately cover the support of the target

Unbiasedness and variance

with q such that $\mathbb{E}_q[|w(X)f(X)|] < \infty$ and $\pi(x)/q(x)$ well defined

$$\mathbb{E}_q[\hat{I}_N] = \mathbb{E}_q[w(X)f(X)] = \int q(x)\frac{\pi(x)}{q(x)}f(x)dx = \int \pi(x)f(x)dx = I$$

$$\mathrm{Var}_q(\hat{I}_N) = \frac{1}{N}\mathrm{Var}_q(w(X)f(X)) = \frac{1}{N}(\mathbb{E}_q[w(X)^2 f(X)^2] - I^2)$$

CLT holds under mild conditions $\sqrt{N}(\hat{I}_N - I) \to \mathcal{N}(0, \sigma^2)$

convergence rate: $\mathcal{O}\left(\frac{1}{\sqrt{N}}\right)$

Efficiency vs MCMC

Can AIS beat MCMC in terms of convergence rate?

AIS asymptotic variance: $\frac{1}{N}\mathrm{Var}_q(wf)$

MCMC asymptotic variance: $\frac{\sigma^2}{N/\tau}$ where τ is the integrated autocorrelation time and σ^2 is the variance under the target distribution

If AIS can find a q that makes wf much less variable than f itself, it will have lower variance than MCMC

If AIS does not find a good q, its weights might have huge variance, making things worse.

A Brief History of AIS

- Adaptive Importance Sampling on Discrete Markov Chains (Baggerly, Cox, Piccard, 1997): shows that an adaptive algorithm can achieve very fast variance reduction for rare event probability distribution

- Adaptive Mixture Kernels: Douc, et al. 2007: D-kernel algorithm fits a mixture of kernels to the target by either minimizing variance or KL divergence at each step, with theoretical guarantees of convergence to the best mixture in a certain class

- Adaptive Multiple Importance Sampling (AMIS) – Cornuet *et al.*, 2009/2012: Aims to recycle past simulations optimally. AMIS instead keeps all samples from all iterations and recomputes their weights in a way that each sample is weighted with respect to the latest mixture of proposals.

Recent Developments

- Regularization of Weights: A recent idea is to intentionally introduce a small bias to gain a large variance reduction, by "tempering" the importance weights.
- Integration with Reinforcement Learning and SGD: Adaptive sampling has found uses in speeding up stochastic gradient descent by preferentially sampling data points with high loss (this is essentially importance sampling in the space of data indices)

Adaptive Importance Sampling (AIS)

Importance sampling, but in a loop.
At each iteration, improve your sampler so that it gets closer to your target.

```
Initialize q_1(x) (initial proposal distribution with support covering π)
for t = 1 to T:
    Draw X_{t,1}, ..., X_{t,N_t} ~ q_t(x)  independently
    Compute weights w_{t,i} = π(X_{t,i}) / q_t(X_{t,i})  for i=1,...,N_t
    Optionally normalize weights if working with unnormalized π (self-normalization)
    (Store samples and weights; check for convergence)
    Update proposal: determine q_{t+1}(x) based on {X_{s,i}, w_{s,i}} from s=1...t (past iterations)
end for
Output estimate: e.g.,
    \hat{I} = \frac{\sum_{t=1}^T \sum_{i=1}^{N_t} w_{t,i} f(X_{t,i})}{\sum_{t=1}^T \sum_{i=1}^{N_t} w_{t,i}}
    (self-normalized form using all samples)
```

How do we update the proposal?

Proposal update strategies

(1) **Parametrized proposals: moment matching**

choose some family of distributions $q(x|\theta)$

set parameters of q_{t+1} to match the momenta of the weighted sample distribution
(which should be an approximation of the target)

$$\mu_{t+1} = \frac{\sum_{s=1}^{t} \sum_{i=1}^{N_s} w_{s,i} X_{s,i}}{\sum_{s=1}^{t} \sum_{i=1}^{N_s} w_{s,i}}$$

Update Σ_{t+1} similarly

Works well if your target is a Gaussian. Can only predict a single mode.

Proposal update strategies

(2) **Cross-entropy approach**

Choose q_{t+1} to minimize KL divergence / maximize expected log likelihood of q over the target distribution

$$\max_\theta \sum_{s=1}^{t} \sum_{i=1}^{N_s} w_{s,i} \log q(x_s, i|\theta)$$

(treat weights like they were importance weights, then find the parameter that makes the current sample batch most likely)

This is a general form solution- for many distribution families, there are closed form solutions
(e.g. if q is a Gaussian, the above criteria yields the same moment formulae as (1))

Proposal update strategies

(3) **Maintaining multiple proposals (mixture adaptation)**

q_t can be a mixture of M Gaussian components

$$q_t(x) = \sum_{m=1}^{M} \alpha_{t,m} q(x|\theta_{t,m})$$

update mixture weights $\alpha_{t,m}$

update each component's parameters $\theta_{t,m}$

Algorithmically,

update component parameters θ_m
by looking at the samples that were drawn from that component and computing their weights relative to that component's contribution

update mixture weights α_m

by how much total weight each component's samples received

Can dynamically allocate more sampling effort to components that perform well (high total weight) and adjust components to cover different modes

Proposal update strategies

(4) **Resampling-based adaptation**

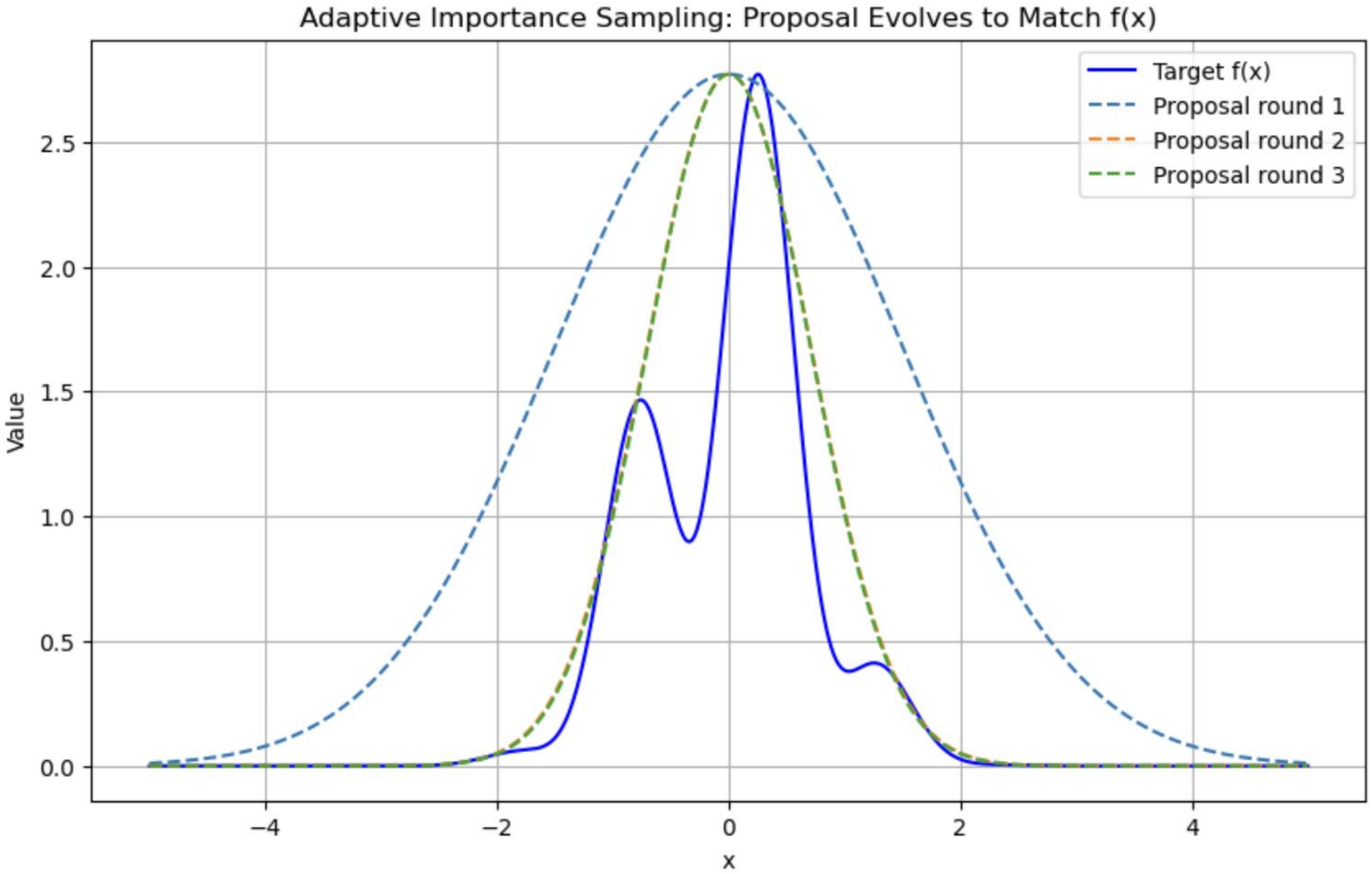
Among the samples you have, you probabilistically select some to serve as “progenitors” of the next proposal distribution.

(5) **Hybrid MCMC steps**

Run short MCMC chains to help propose new sample locations. Take each sample from iteration t, treat it as a seed, run a short Metropolis-Hastings chain targeting your target, and use the endpoint as part of the next proposal set.

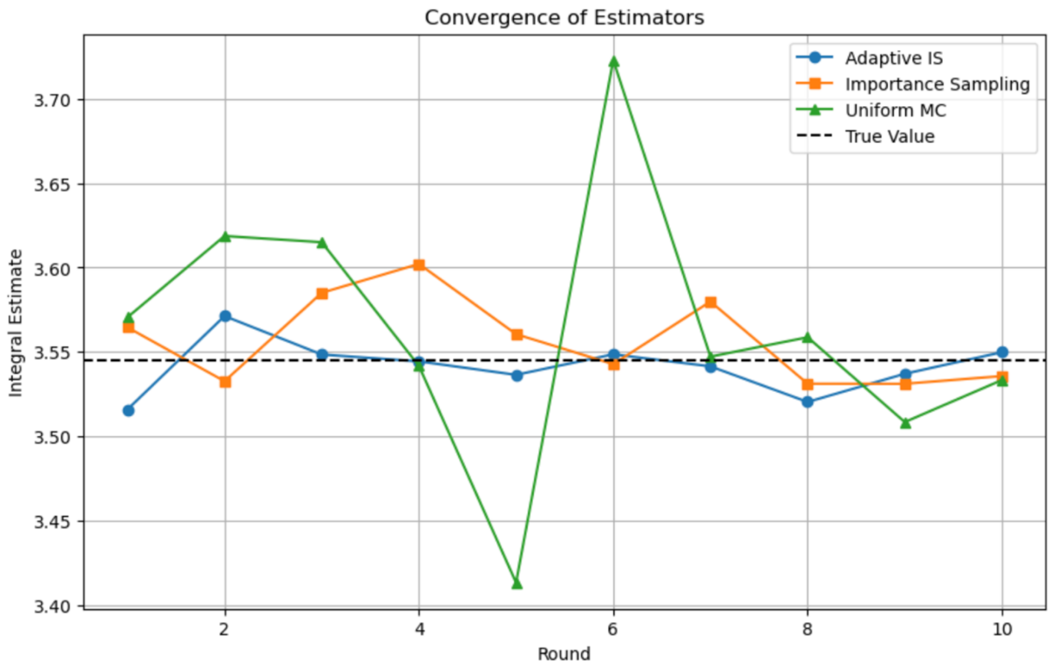
$$f(x) = (e^{-x^2}) \cdot (2 + sin(5x))$$

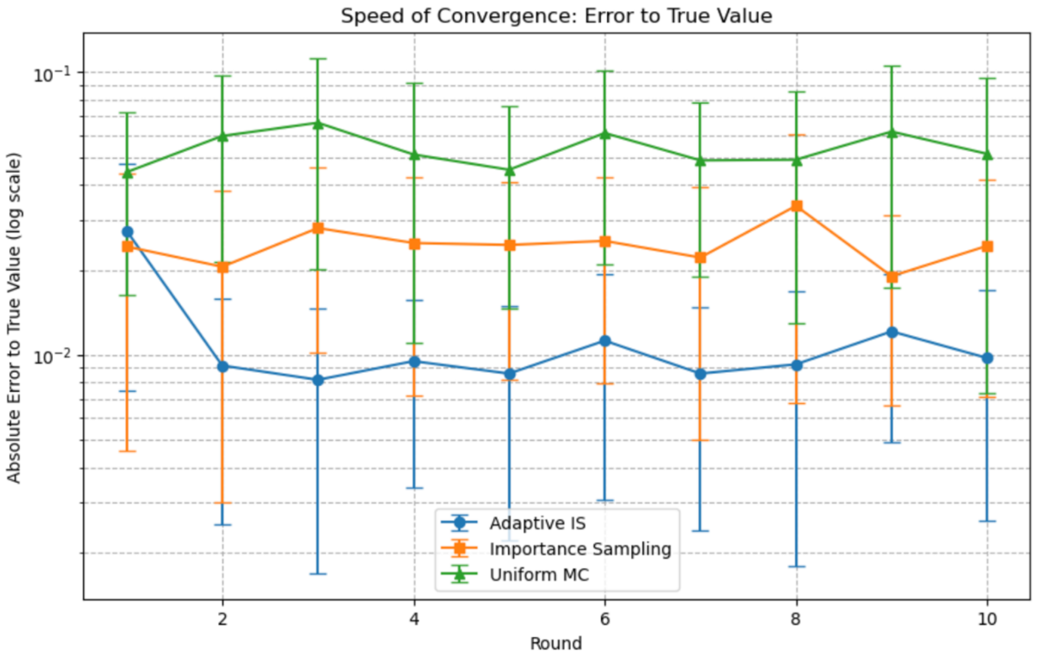
$$I = \int_{-\infty}^{\infty} f(x)$$



$$f(x) = (e^{-x^2}) \cdot (2 + sin(5x))$$

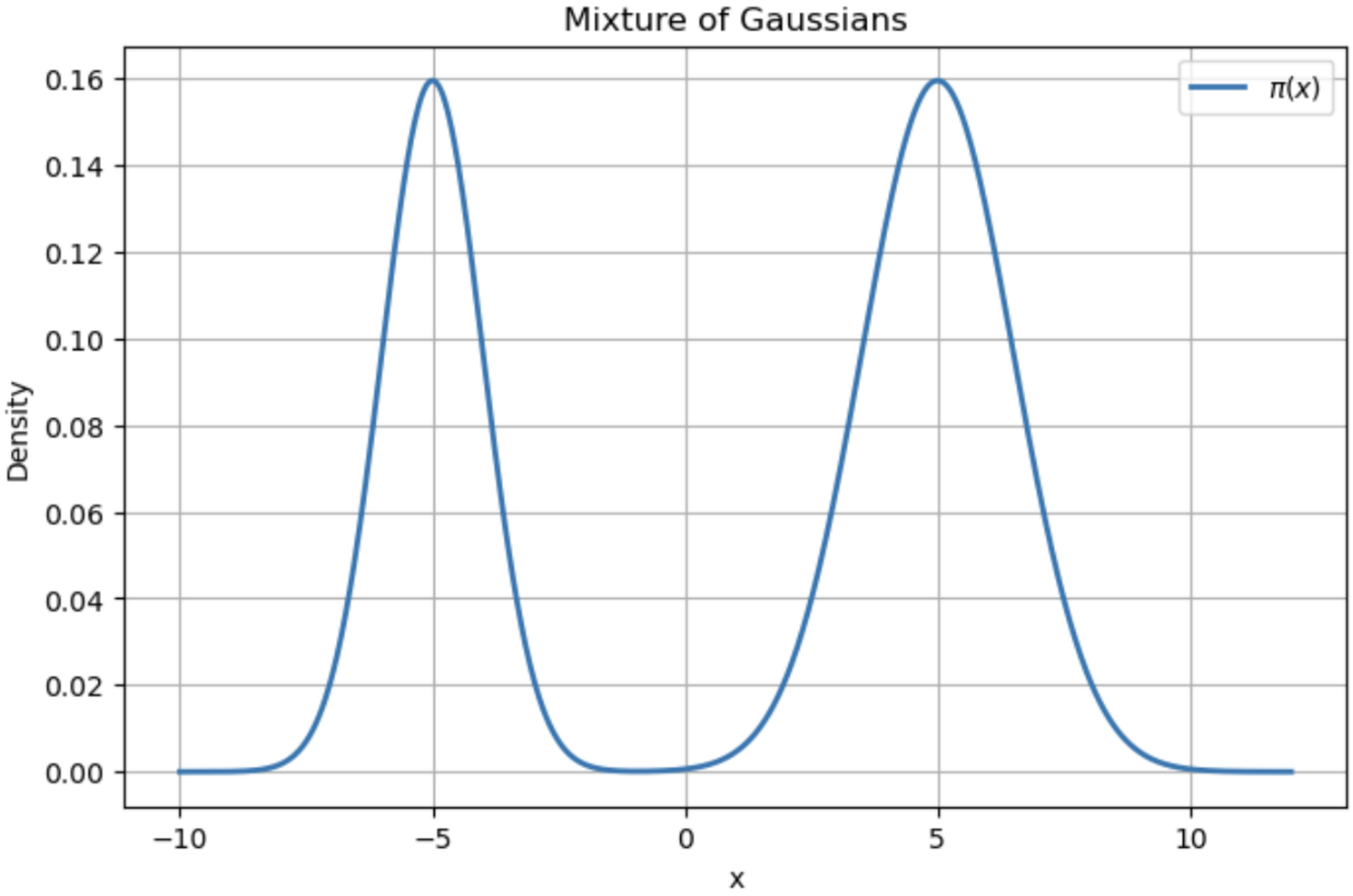
$$I = \int_{-\infty}^{\infty} f(x)$$

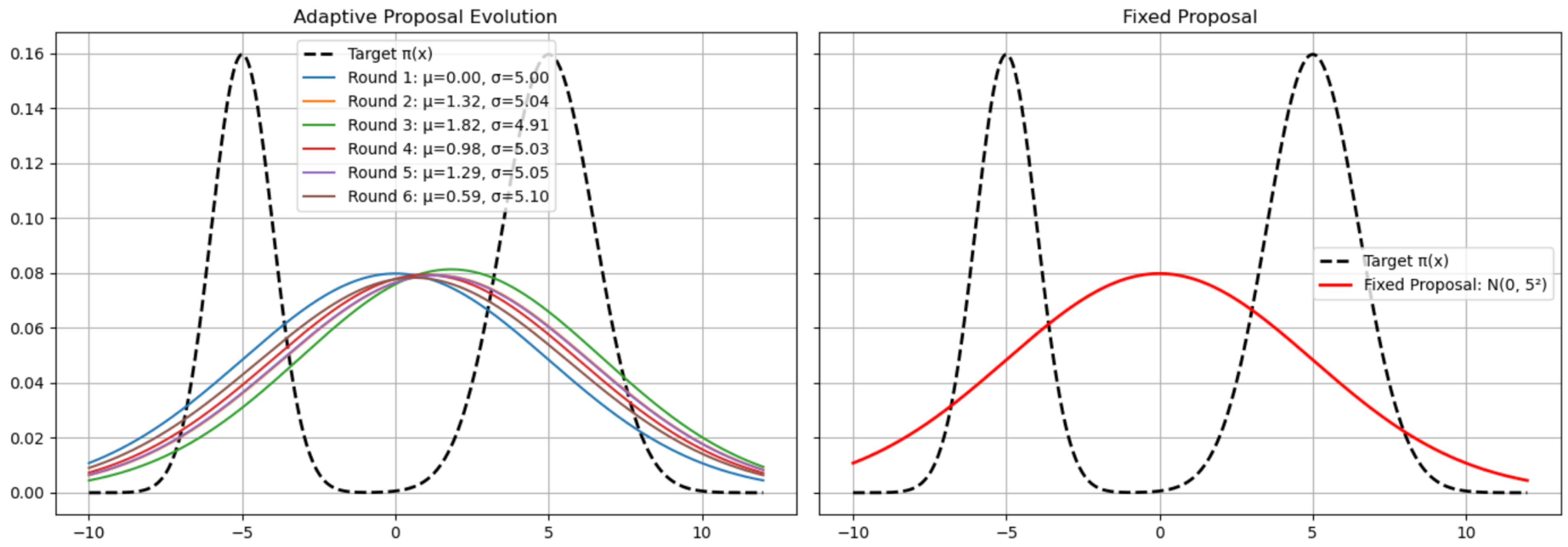




Tackling multi-modal distributions

$$\pi(x) = 0.4\,\mathcal{N}(x| -5,\ 1^2)\ +\ 0.6\,\mathcal{N}(x| +5,\ 1.5^2)$$




Adaptive Proposal Evolution
Target π(x)
Round 1: μ=0.00, σ=5.00
Round 2: μ=1.32, σ=5.04
Round 3: μ=1.82, σ=4.91
Round 4: μ=0.98, σ=5.03
Round 5: μ=1.29, σ=5.05
Round 6: μ=0.59, σ=5.10
0.00
0.02
0.04
0.06
0.08
0.10
0.12
0.14
0.16
−10
−5
0
5
10
Fixed Proposal
Target π(x)
Fixed Proposal: N(0, 5²)

Our approach: “Hard-mixture” AIS

$$q_0(x) \;=\; \sum_{m=1}^{n} \alpha_{0,m}\,\mathcal{N}\!\left(x \mid \mu_{0,m}, \sigma_{0,m}^2\right)$$

User defines n means/variances – by default, n evenly distributed, nonoverlapping Gaussians across domain

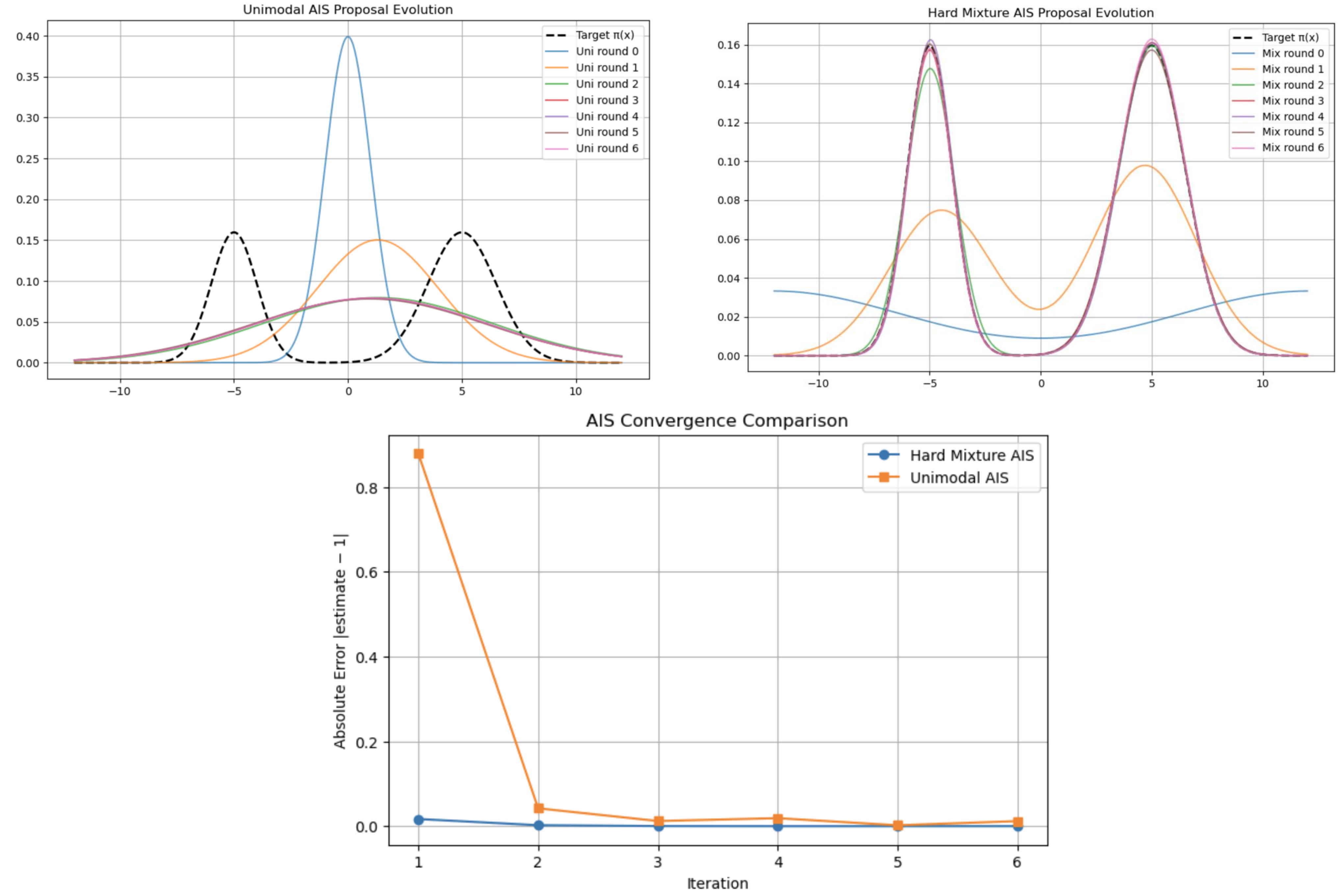
At iteration t,

For $i = 1 \ldots N$

- Draw component index $m_i \sim \text{Categorical}(\{\alpha_{t,m}\})$.
- Draw $x_i \sim \mathcal{N}(\mu_{t,m_i}, \sigma_{t,m_i}^2)$

$$w_i \;=\; \frac{\pi(x_i)}{\alpha_{t,m_i}\, q(x_i \mid \mu_{t,m_i}, \sigma_{t,m_i})}$$

Update $\alpha_{t+1,m} = \dfrac{\sum_{i:m_i=m} w_i}{\sum_i w_i}$ $\quad \mu_{t+1,m} = \dfrac{\sum_{i:m_i=m} w_i\, x_i}{\sum_{i:m_i=m} w_i}$ $\quad \sigma_{t+1,m}^2 = \dfrac{\sum_{i:m_i=m} w_i\,(x_i - \mu_{t+1,m})^2}{\sum_{i:m_i=m} w_i}$



Unimodal AIS Proposal Evolution
Target π(x)
Uni round 0
Uni round 1
Uni round 2
Uni round 3
Uni round 4
Uni round 5
Uni round 6
0.40
0.35
0.30
0.25
0.20
0.15
0.10
0.05
0.00
−10
−5
0
5
10
Hard Mixture AIS Proposal Evolution
Target π(x)
Mix round 0
Mix round 1
Mix round 2
Mix round 3
Mix round 4
Mix round 5
Mix round 6
0.16
0.14
0.12
0.10
0.08
0.06
0.04
0.02
0.00
−10
−5
0
5
10
AIS Convergence Comparison
Hard Mixture AIS
Unimodal AIS
Absolute Error |estimate − 1|
0.8
0.6
0.4
0.2
0.0
1
2
3
4
5
6
Iteration

Application example: Options trading

(1) **Options**: the right but not obligation to buy or sell an underlying asset (e.g. a stock) at a specified price (K) that expires in time T.

Goal: determine a **fair value** for an option based on the **probability distribution** of the underlying asset's price at expiration: discounted expected value of its **payoff**

$$\text{Call Price} = e^{-rT}\,\mathbb{E}_{\mathbb{Q}}\left[\max(\underbrace{S_T - K}_{>0},0)\right]$$

>0: "in-the-money", asset has intrinsic value

Typically rare events!

$$\text{Put Price} = e^{-rT}\,\mathbb{E}_{\mathbb{Q}}\left[\max(K - S_T,0)\right]$$

- S_T: Price of the underlying asset at option expiry
- K: Strike price
- r: Risk-free interest rate
- T: Time to maturity in years
- $\mathbb{E}_{\mathbb{Q}}$: Expectation under the risk-neutral measure

Application example: Options trading

(2) Parameterization: Terminal Asset Price Simulation

Under the Black-Scholes risk-neutral framework, the terminal price of the underlying asset at expiry is given by:

$$S_T = S_0 \cdot \exp\left(\left(r - \frac{1}{2}\sigma^2\right)T + \sigma\sqrt{T} \cdot Z\right)$$

Where:

- S_0 = current spot price
- r = risk-free rate
- σ = volatility (implied or estimated)
- T = time to maturity (in years)
- $Z \sim \mathcal{N}(0, 1)$

This produces a **log-normal distribution** for S_T.

Application example: Options trading

(2) Parameterization: **Payoff**

Call Option Payoff:

$$\text{Payoff}_{\text{call}}(S_T) = \max(S_T - K,\, 0)$$

Put Option Payoff:

$$\text{Payoff}_{\text{put}}(S_T) = \max(K - S_T,\, 0)$$

Where:

- K = strike price
- S_T = simulated terminal price
- The `max` function ensures that the payoff is non-negative

In the risk-neutral framework, the fair value of the option at time 0 is:

$$V_0 = e^{-rT} \cdot \mathbb{E}_{\mathbb{Q}}\left[\text{Payoff}(S_T)\right]$$

Where:

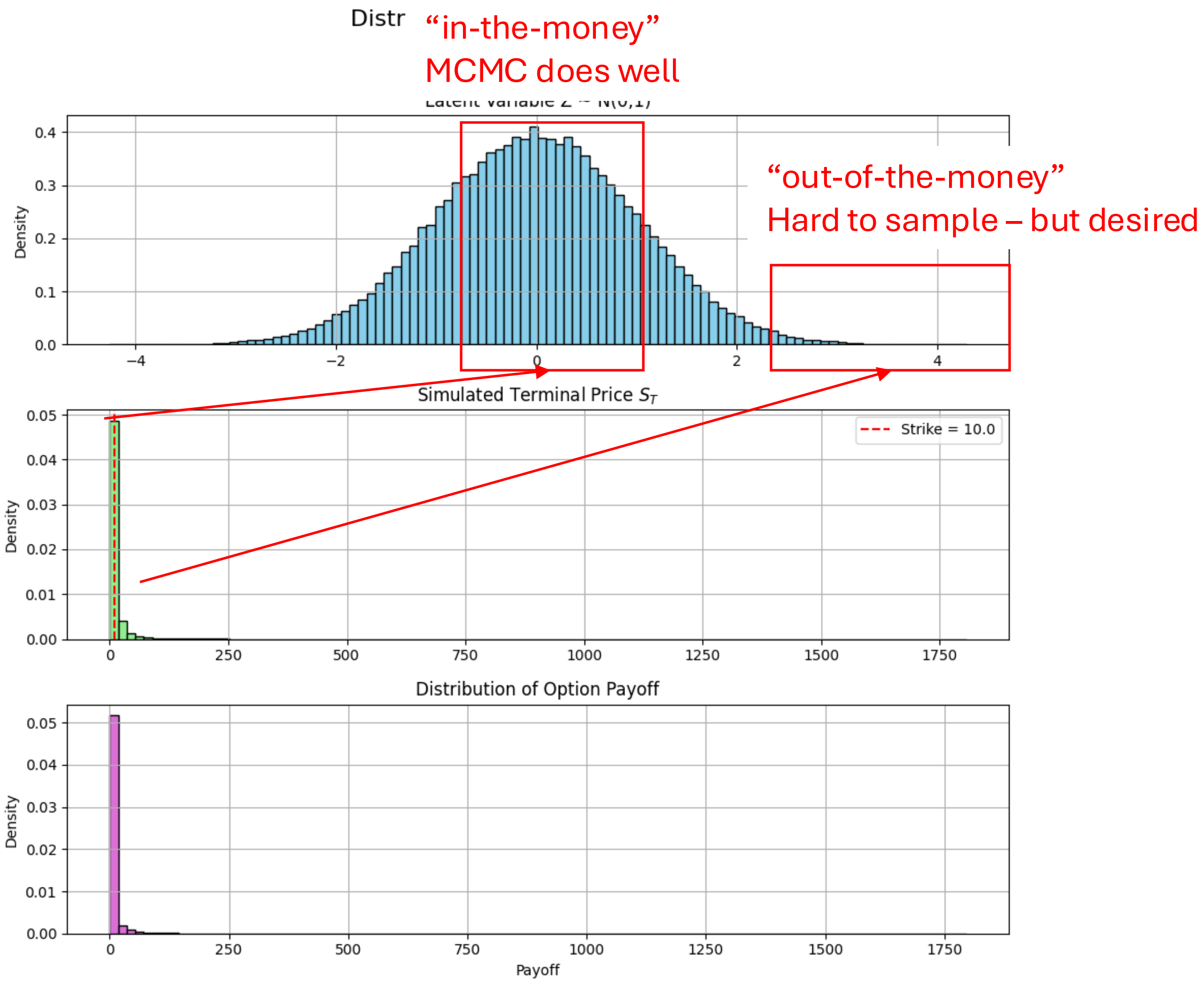
- $\mathbb{E}_{\mathbb{Q}}$ is the expectation under the risk-neutral measure
- No closed-form in simulation: this is estimated via Monte Carlo.

For Monte Carlo simulation:

$$V_0 \approx \frac{e^{-rT}}{N} \sum_{i=1}^{N} \text{Payoff}(S_T^{(i)})$$

Application example: Options trading

(2) Parameterization: **Payoff**



Why AIS for option pricing?

(1) Problems with Standard Monte Carlo:
- Insufficient for tail events - low probability of payoff.
- High variance in estimates and slow convergence.

(2) AIS addresses this by:
- **Biasing the sampling distribution** toward regions that contribute most to the payoff.
- **Re-weighting samples** to maintain unbiasedness.
- **Adapting** the proposal distribution iteratively to improve convergence.

AIS formulation

We rewrite the original expectation:

$$\mathbb{E}_{p(z)}[f(z)] = \int f(z)p(z)\,dz = \int \frac{f(z)p(z)}{q(z)}q(z)\,dz = \mathbb{E}_{q(z)}\left[\frac{f(z)p(z)}{q(z)}\right]$$

In AIS:

- $f(z)$: Payoff function, e.g., $\max(S_T - K, 0)$ written as a function of $Z \sim \mathcal{N}(0,1)$ $S_T = S_0 \cdot \exp\left(\left(r - \frac{1}{2}\sigma^2\right)T + \sigma\sqrt{T} \cdot Z\right)$
- $p(z)$: Target distribution (e.g., standard normal $\mathcal{N}(0,1)$)
- $q(z)$: Adaptive **proposal distribution** (biased distribution used to sample z)
- $\frac{p(z)}{q(z)}$: **Importance weight**, corrects for bias in the proposal

As the AIS procedure evolves, it **updates the parameters** (mean and variance) of the proposal distribution $q(z; \mu, \sigma^2)$ at each iteration to focus sampling more efficiently.

AIS formulation

- $f(z)$: Payoff function, e.g., $\max(S_T - K, 0)$ written as a function of $Z \sim \mathcal{N}(0,1)$
- $p(z)$: Target distribution (e.g., standard normal $\mathcal{N}(0,1)$)
- $q(z)$: Adaptive **proposal distribution** (biased distribution used to sample z)

Becomes a simple problem of sampling a Gaussian with a Gaussian

The data

	Time	Sym	C/P	Exp	Strike	Spot	BidAsk	Orders	Vol	Prems	OI	Diff(%)
0	6/17/2022 15:07	ISEE	Call	10/21/2022	10.0	9.54	5.05	7	360	183.60K	4.07K	4.71
1	6/17/2022 15:05	CVNA	Call	1/19/2024	60.0	23.52	4.60	7	634	310.66K	130	155.05
2	6/17/2022 14:51	PTLO	Put	2/17/2023	15.0	15.19	3.50	7	800	281.00K	0	1.39

(https://www.kaggle.com/datasets/muhammadanas0716/tradyflow-options-trading)

Symbol	Name	Description
S_0	Spot Price	Current price of the underlying asset
K	Strike Price	Strike at which the option can be exercised
T	Time to Maturity	Time until expiration (in years)
σ	Volatility	Typically implied from market data
r	Risk-Free Rate	Constant rate used for discounting
Option Type	Call/Put	Determines payoff structure: $\max(S_T - K, 0)$ or $\max(K - S_T, 0)$

AIS Implementation

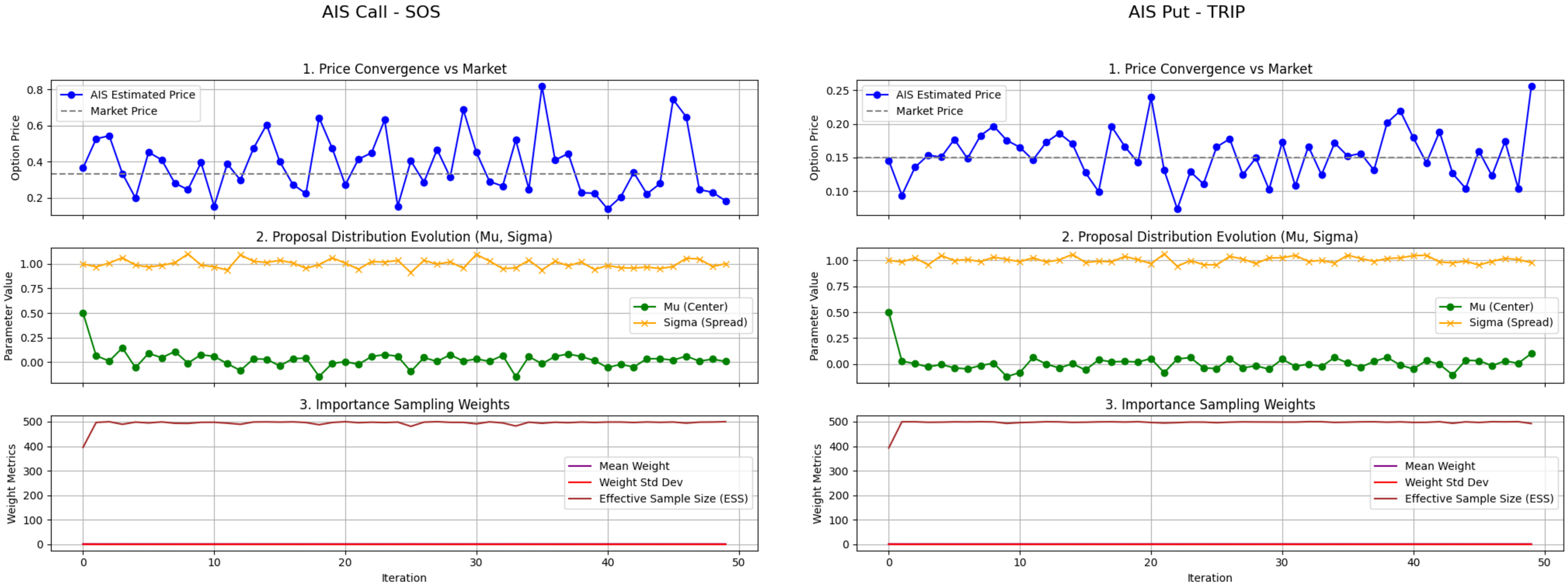
Algorithm 1 Adaptive Importance Sampling for Option Pricing

Require: Option data $(S_0, K, T, \sigma, \text{option type, market price})$
Require: Parameters: risk-free rate r, sample size N, tolerance tol, maximum iterations max_iter

```
Initialize proposal parameters: μ ← 0.5, σ_prop ← 1.0
Set previous price estimate: prev_price ← 0
for it = 1 to max_iter do
    Sample Z_1, ..., Z_N ~ N(μ, σ_prop)
    Compute terminal prices: S_T^(i) = S_0 · exp((r − 0.5 · σ^2)T + σ√T Z_i)
    if call option then
        payoff^(i) ← max(S_T^(i) − K, 0)
    else
        payoff^(i) ← max(K − S_T^(i), 0)
    end if
    Compute weights: w_i ← p(Z_i) / q(Z_i)
    Estimate price: price ← (1/N) Σ_{i=1}^N e^(−rT) · payoff^(i) · w_i
    if |(price − prev_price) / (prev_price + 10^−6)| < tol and it > 1 then
        break
    end if
    Set prev_price ← price
    Normalize weights: w_i ← w_i / Σ_j w_j
    Update proposal mean: μ ← Σ_{i=1}^N w_i Z_i
    Update proposal variance: σ_prop^2 ← Σ_{i=1}^N w_i (Z_i − μ)^2
    Set σ_prop ← √(σ_prop^2)
end for
return Estimated price, market price, price error, final μ, σ_prop, iteration count
```

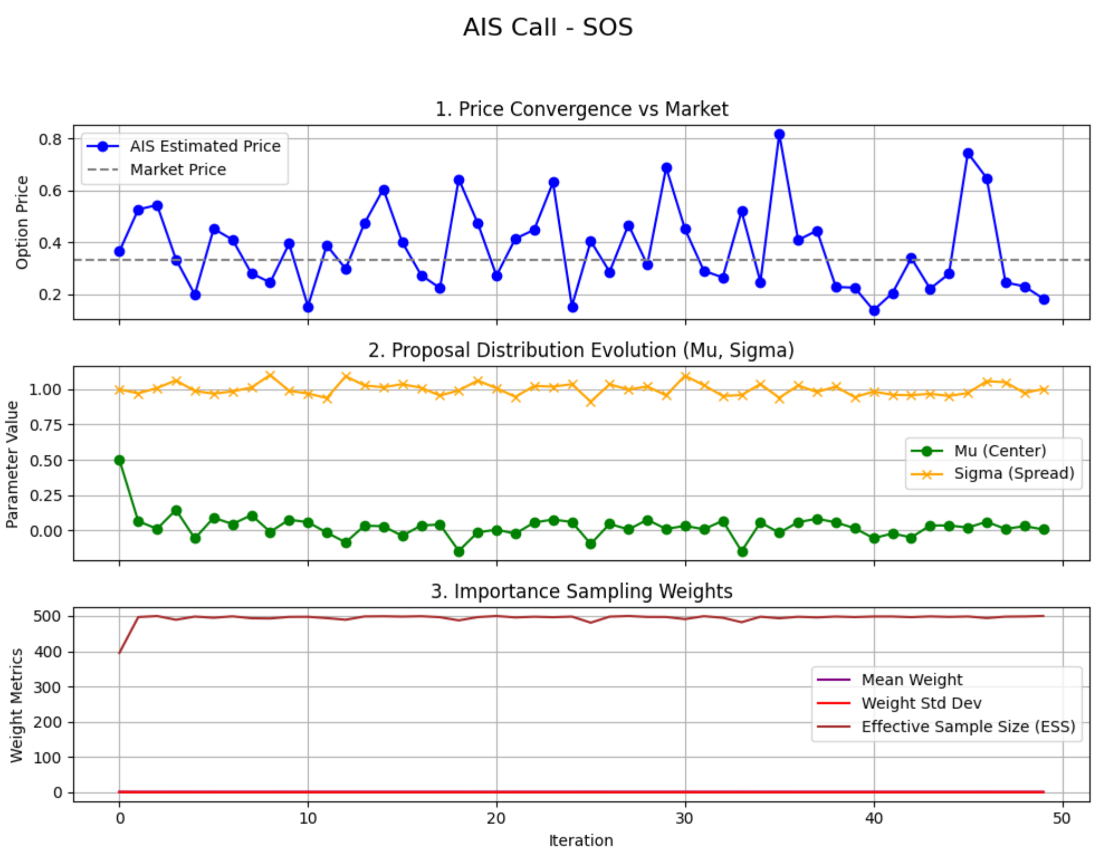
Variable	Description
N	Number of Monte Carlo samples per iteration
μ	Mean of proposal distribution $q(z; \mu, \sigma^2)$
σ_{prop}	Standard deviation of the proposal distribution
`max_iter`	Maximum number of AIS iterations
`tol`	Stopping threshold for convergence (based on price change)
`weights`	The importance weights $\frac{p(z)}{q(z)}$ used to reweight payoffs

AIS Implementation (N=500)

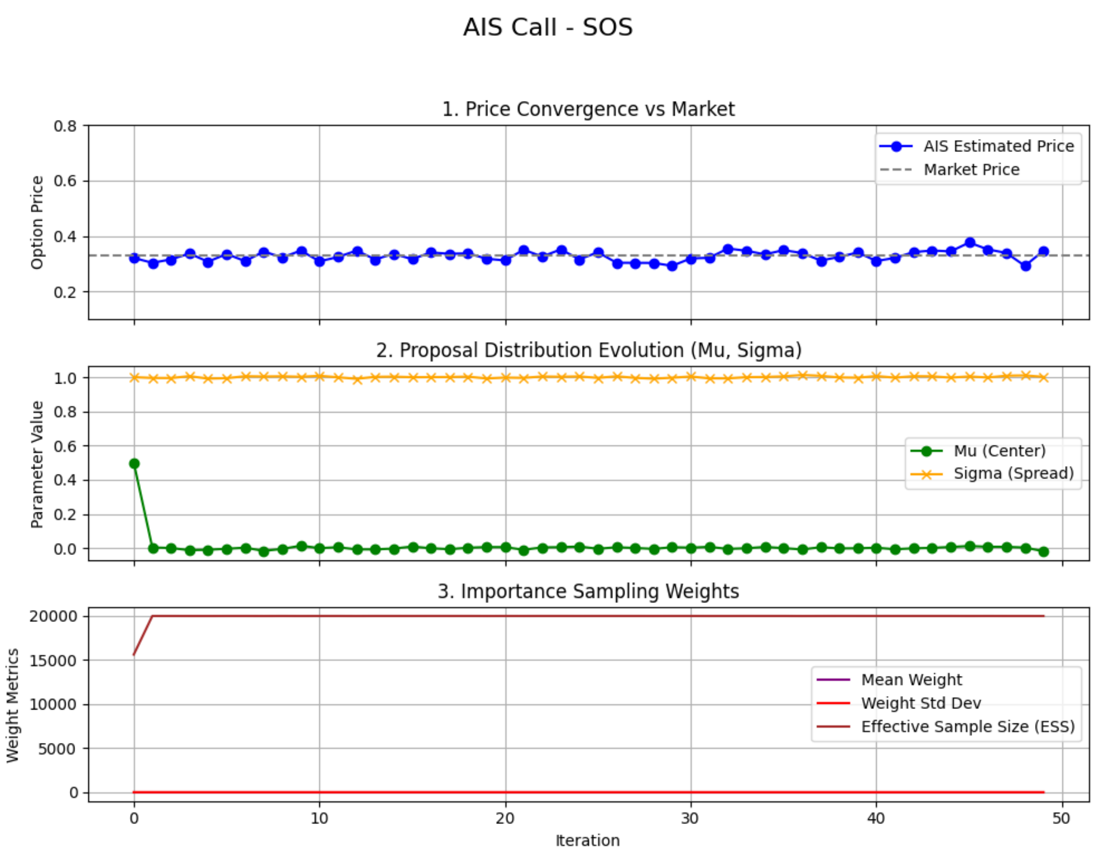


Effect of more samples (N=20000)

N=500



N=20000

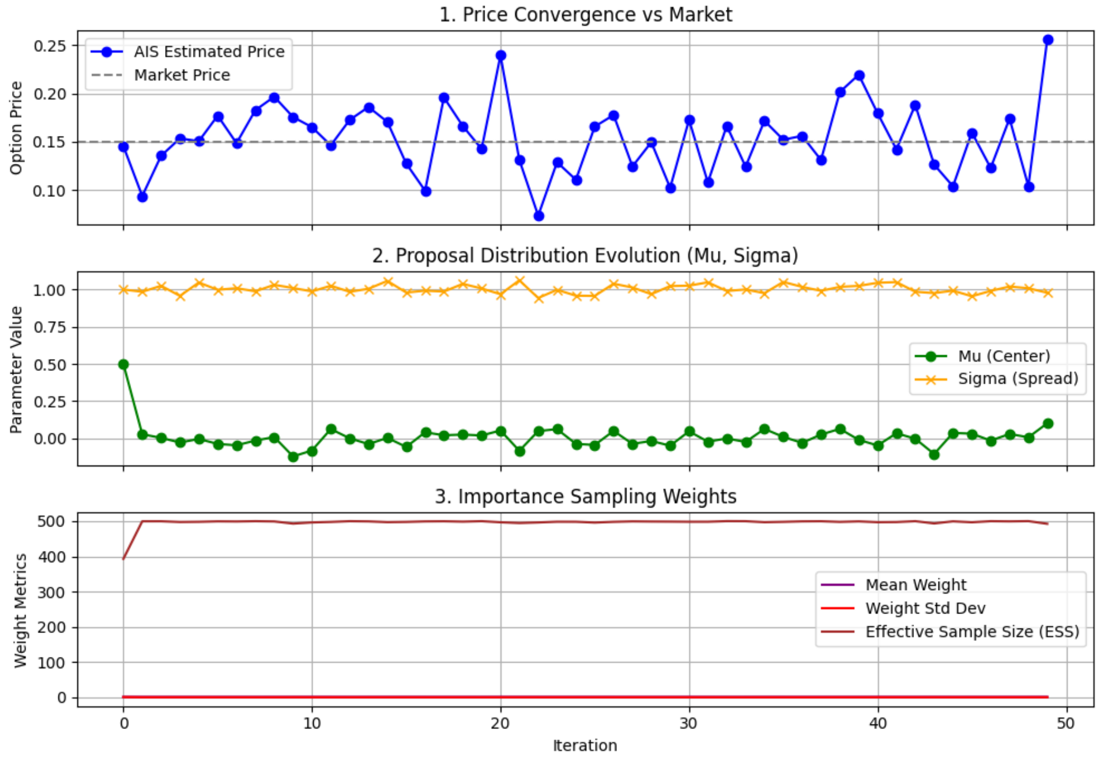


$$\hat{I}_N \xrightarrow{\text{a.s.}} \mathbb{E}_p[\phi(Z)] \quad \text{as } N \to \infty$$

Effect of more samples (N=20000)

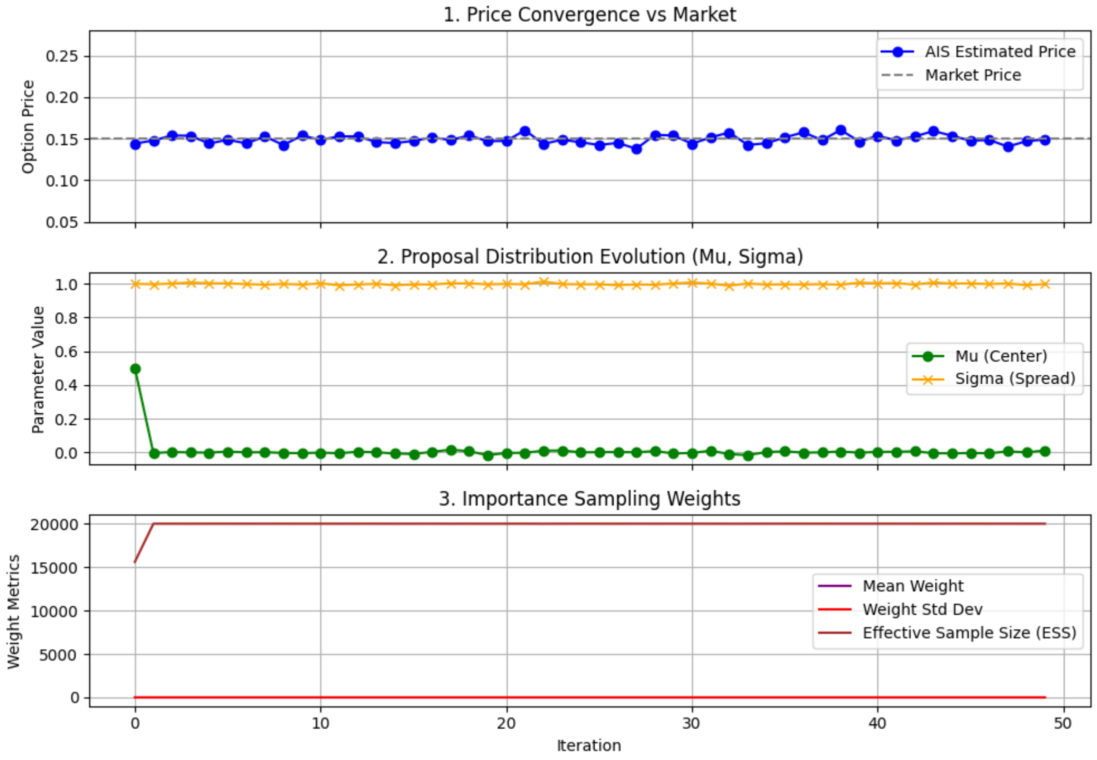
N=500

AIS Put - TRIP



N=20000

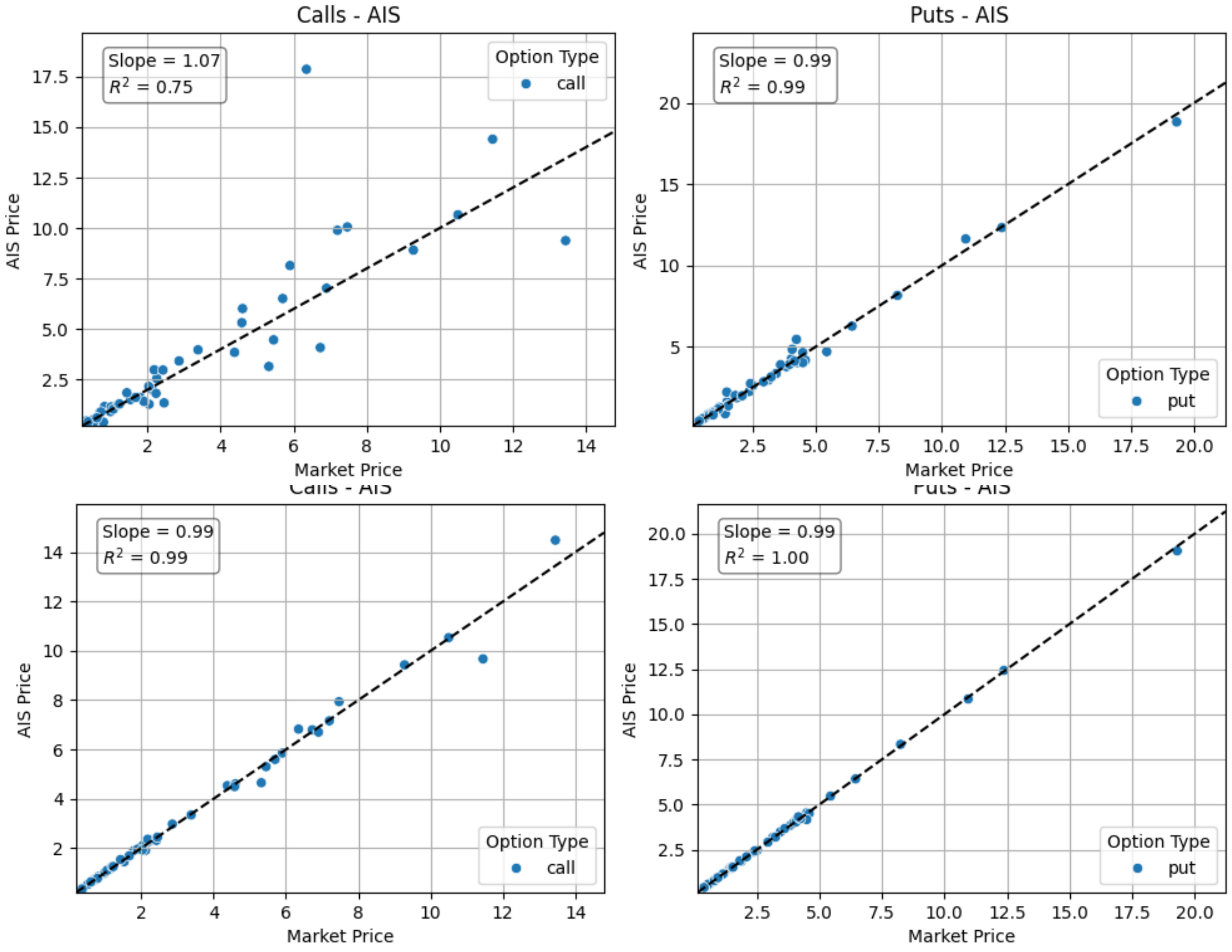
AIS Put - TRIP



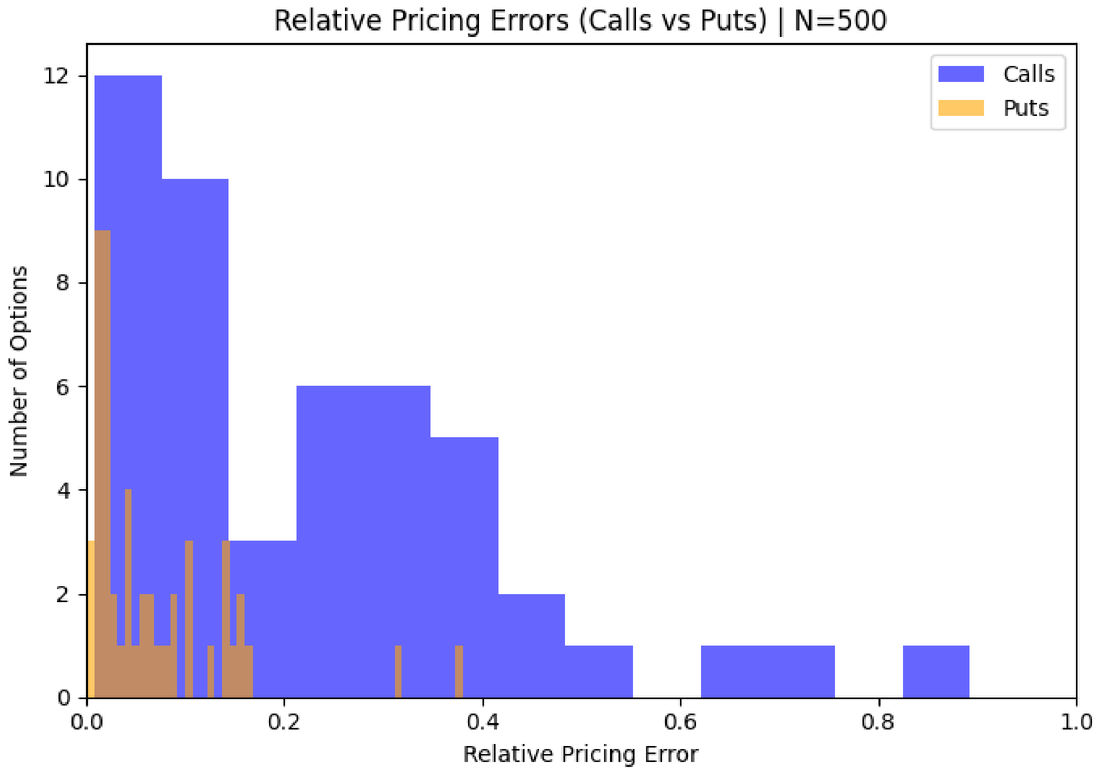
Effect of more samples (N=20000)

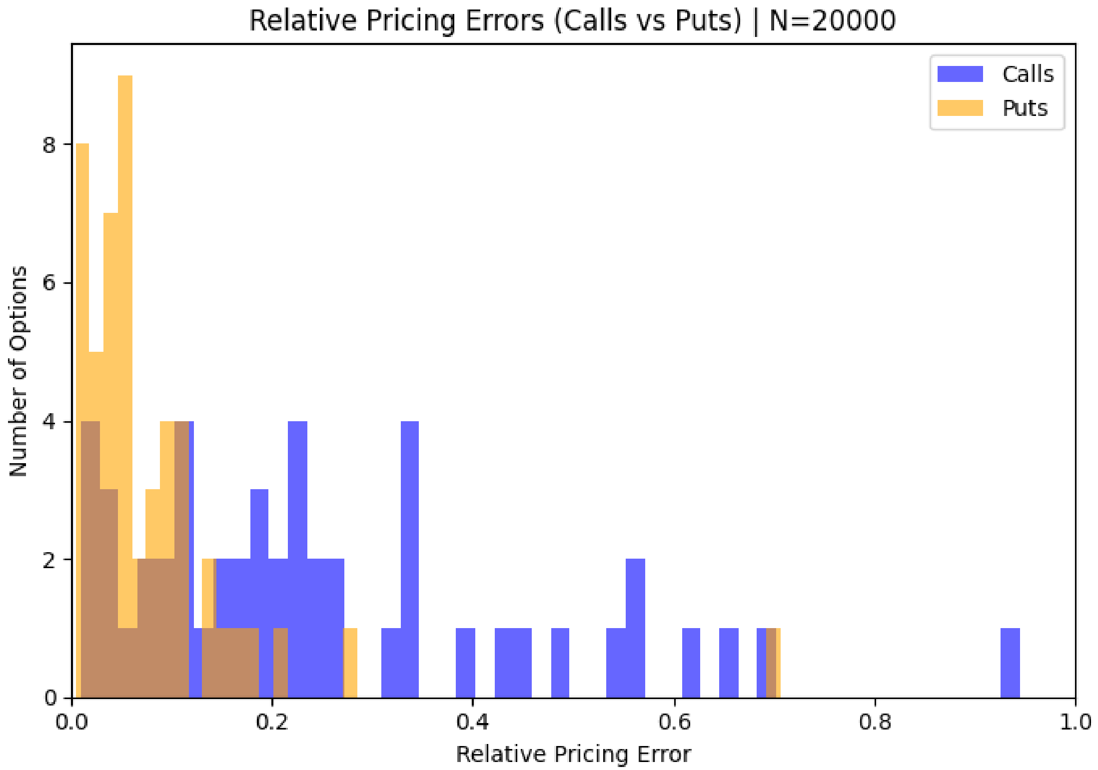
N=500

N=20000

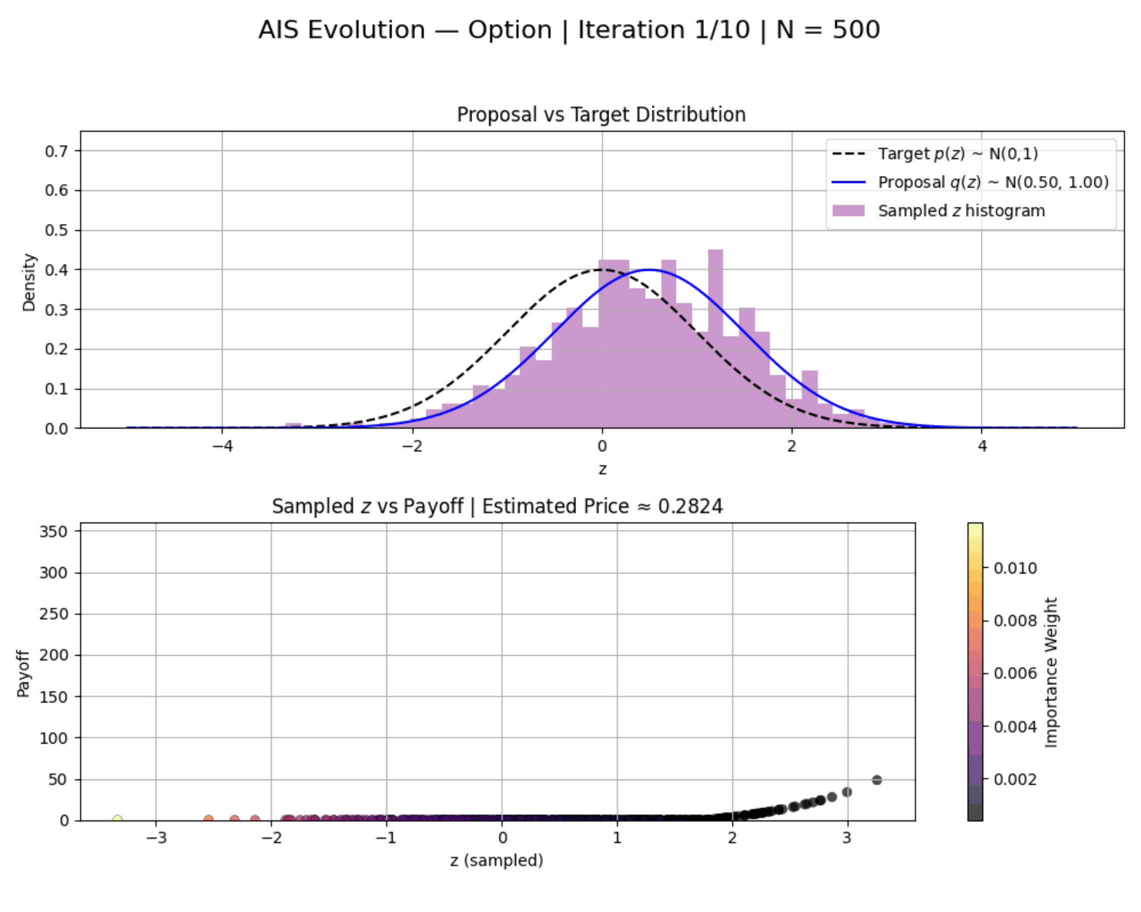


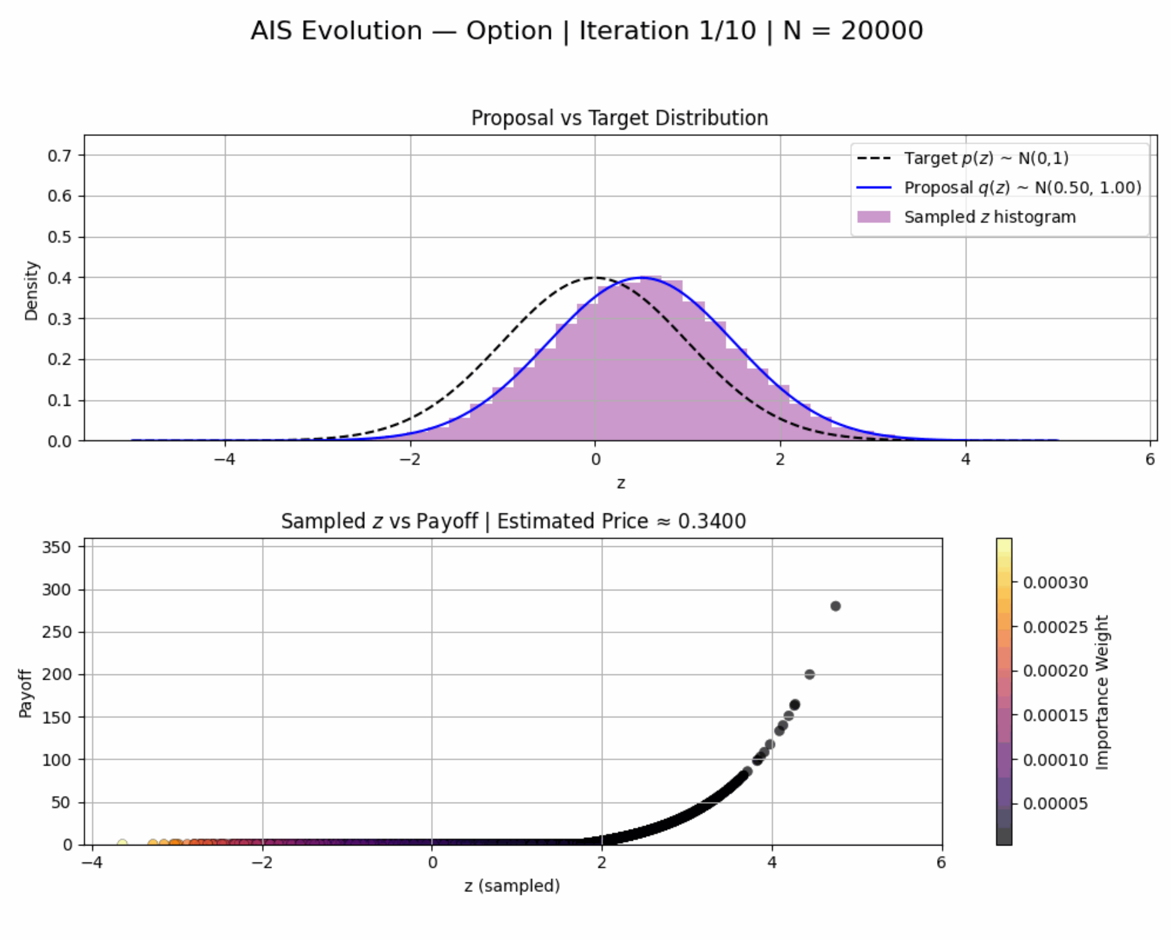
Effect of more samples (N=20000)





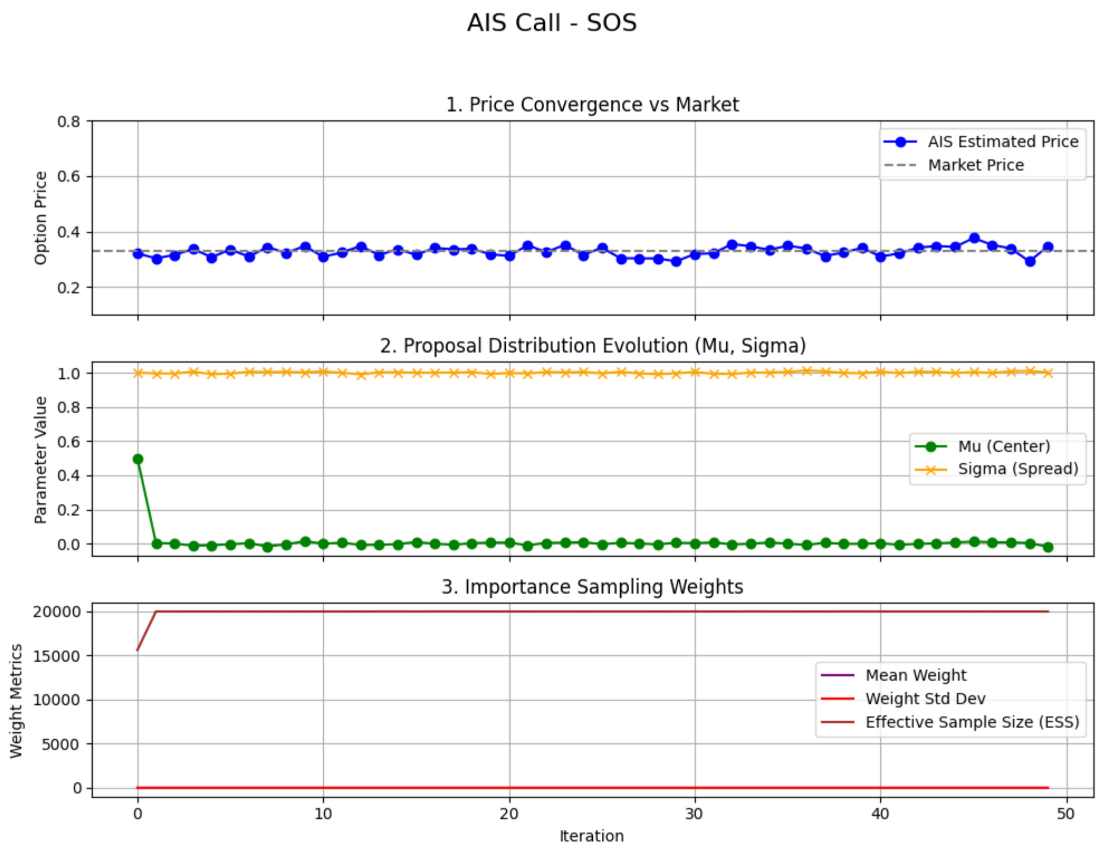
Effect of more samples (N=2000)



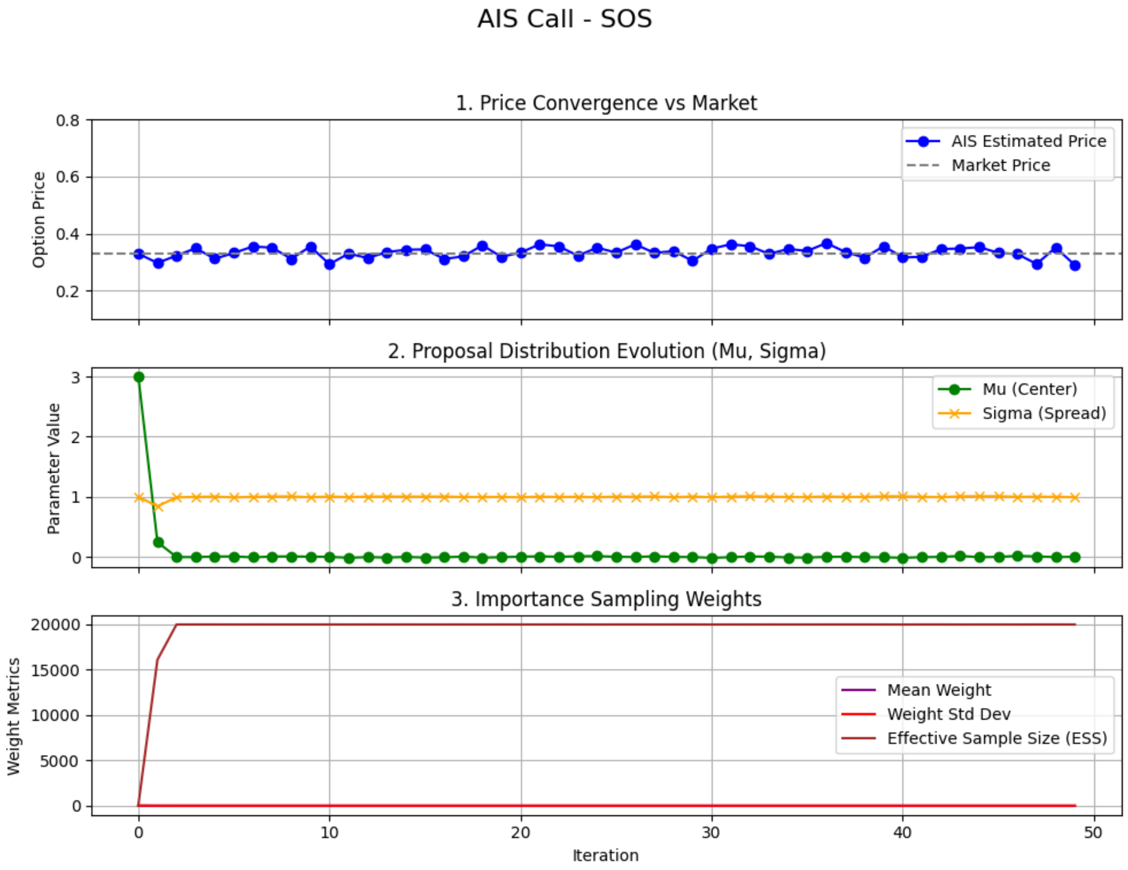


Effect of bad proposal distribution is minimal on AIS

μ=0.5

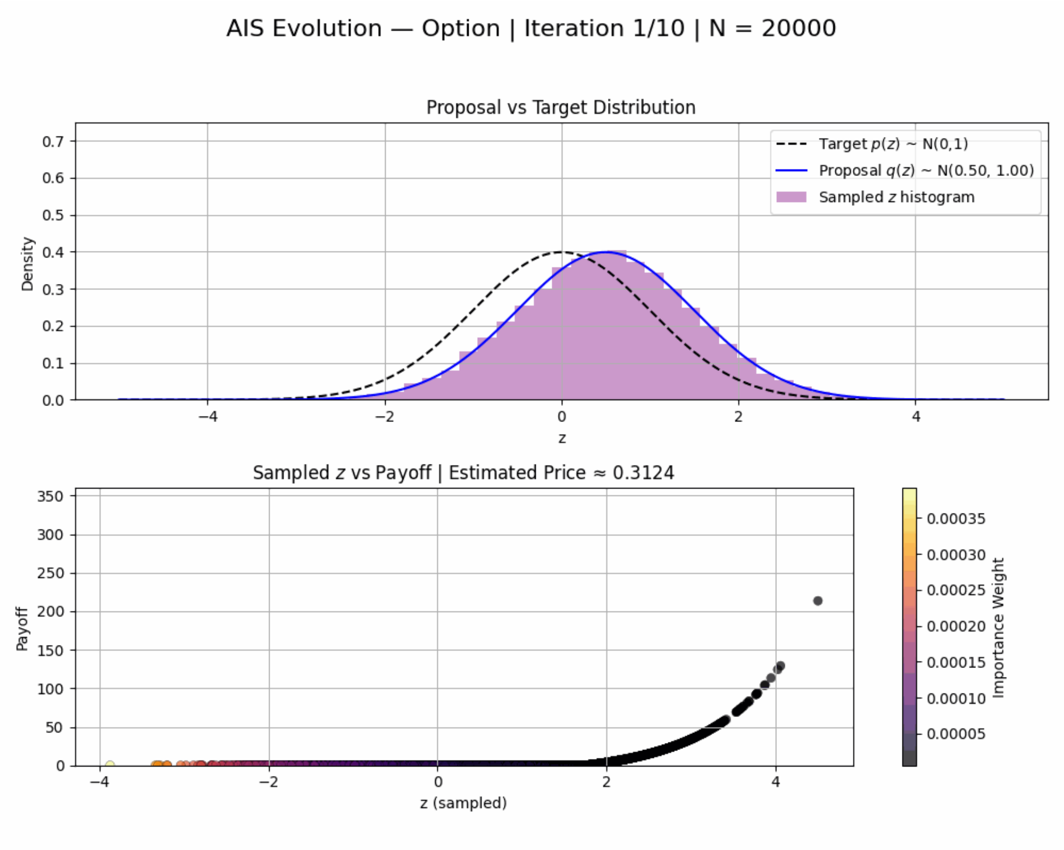


μ=3

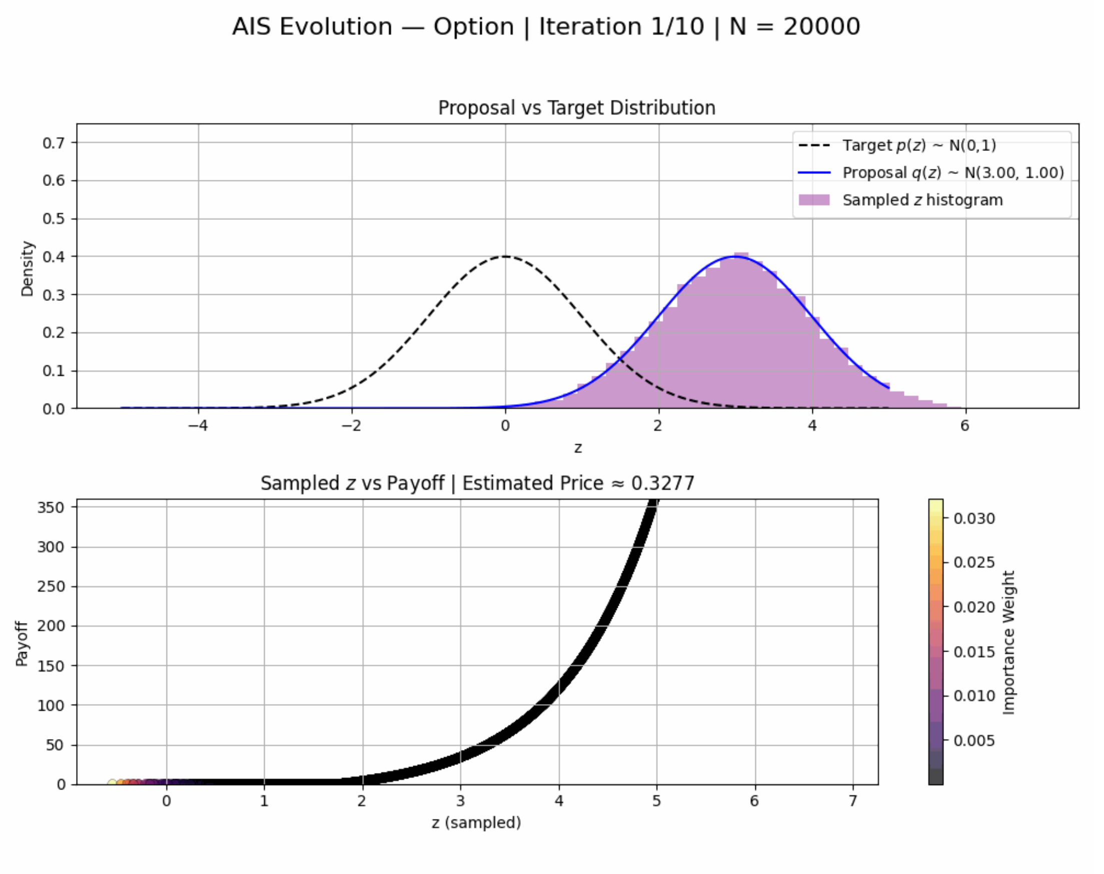


Effect of bad proposal distribution is minimal

μ=0.5



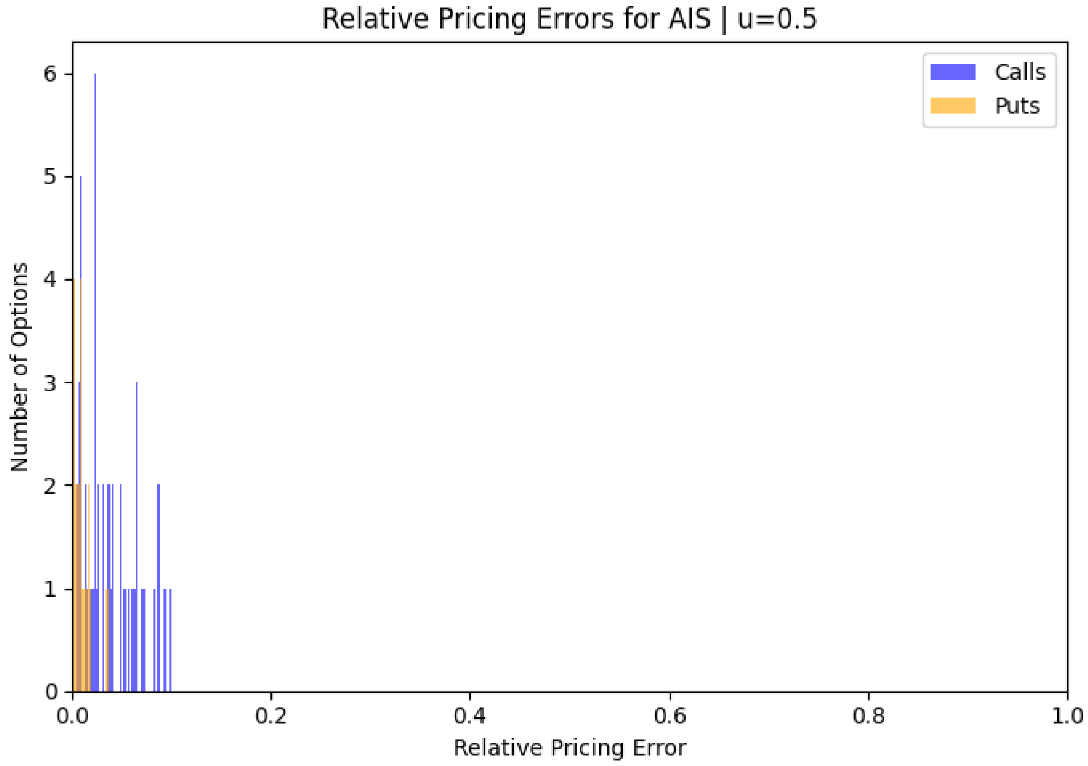
μ=0.3

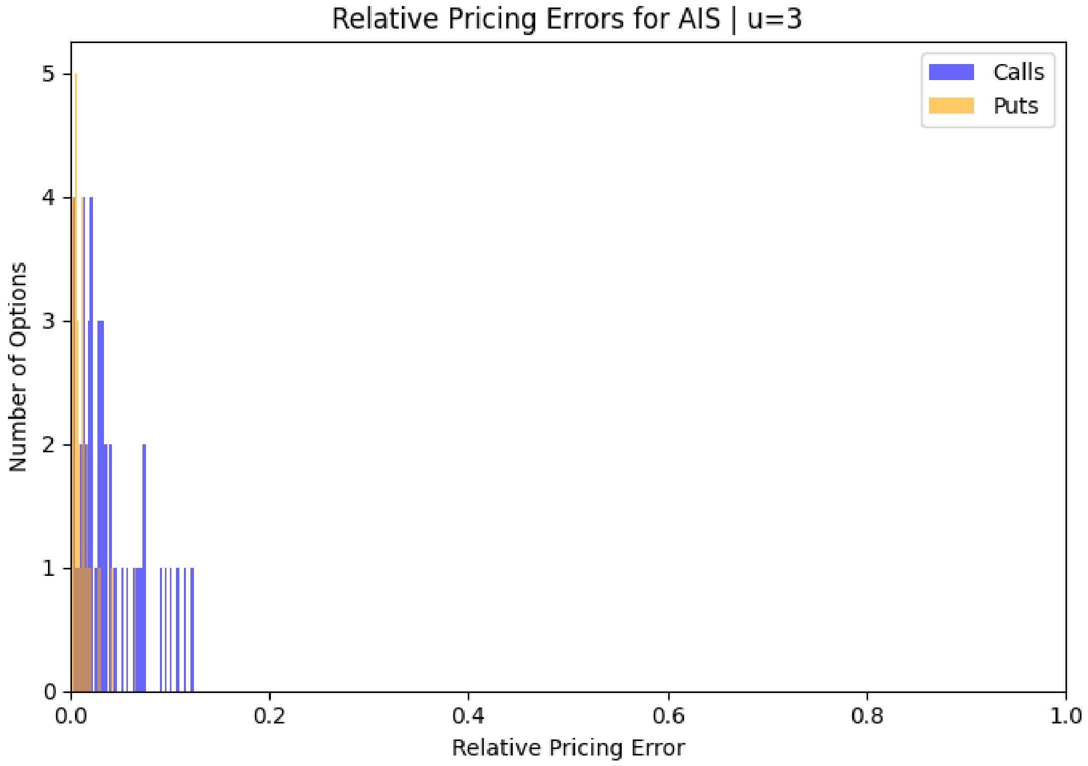


Effect of bad proposal distribution is minimal on AIS

μ=0.5

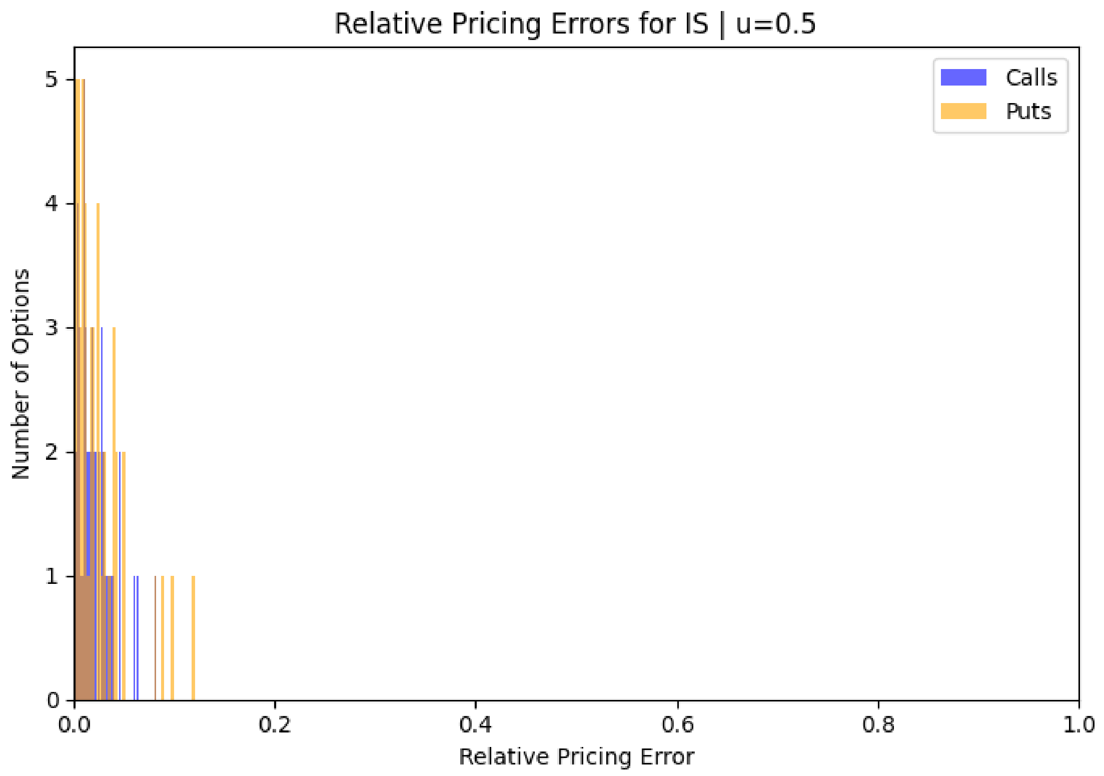
μ=3



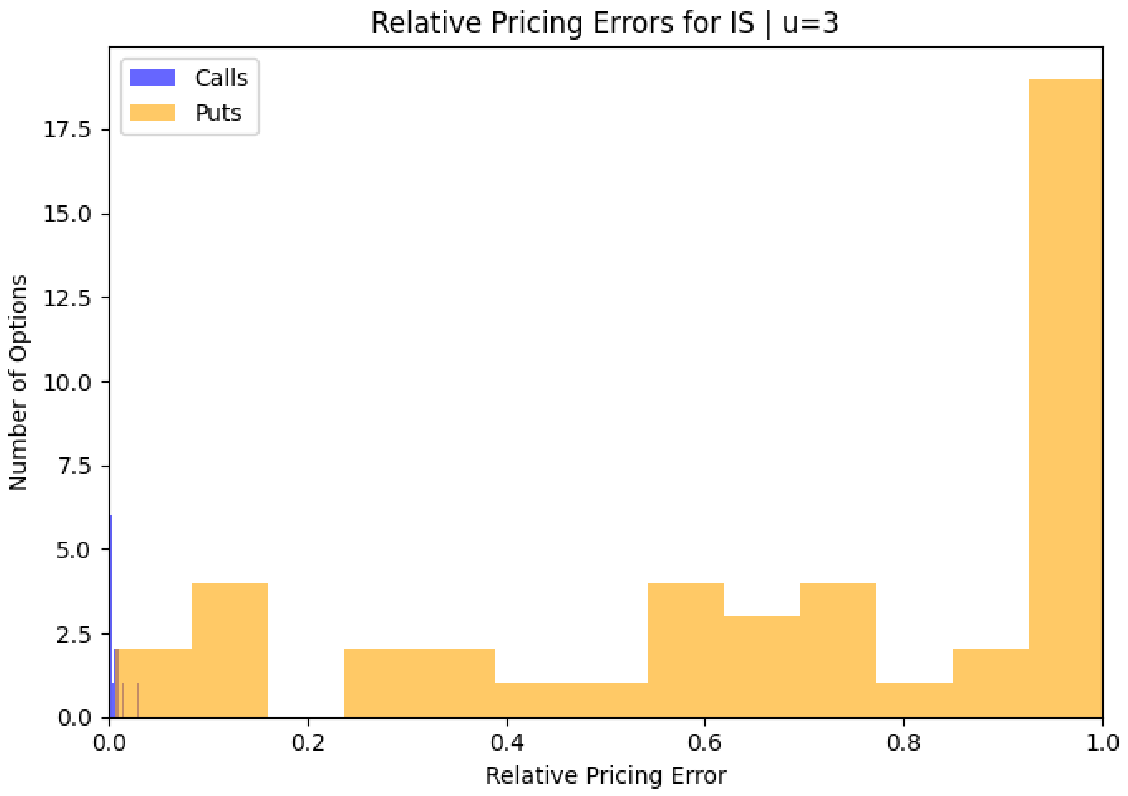


But obvious on IS

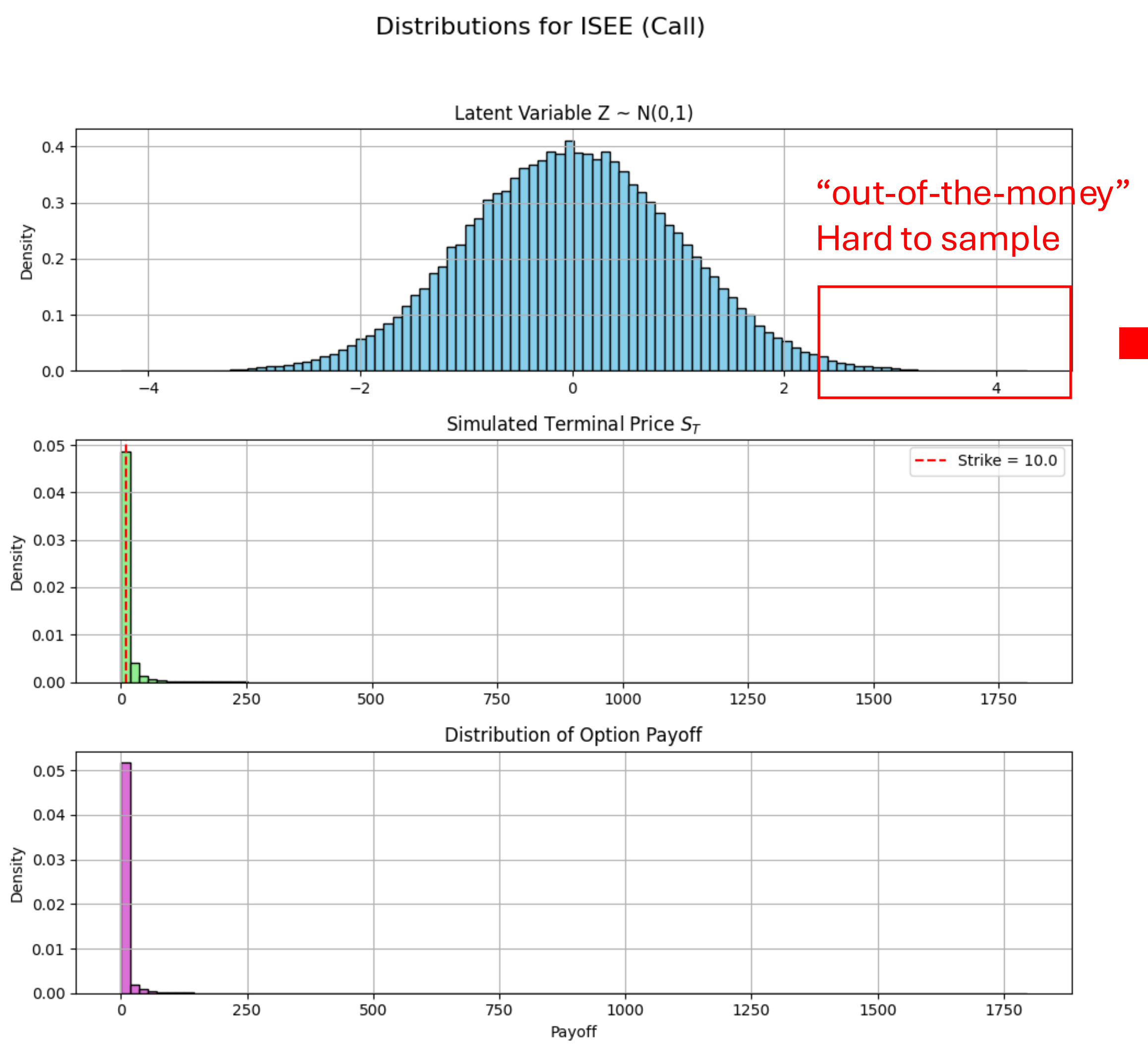
μ=0.5



μ=3



AIS better models OTM options

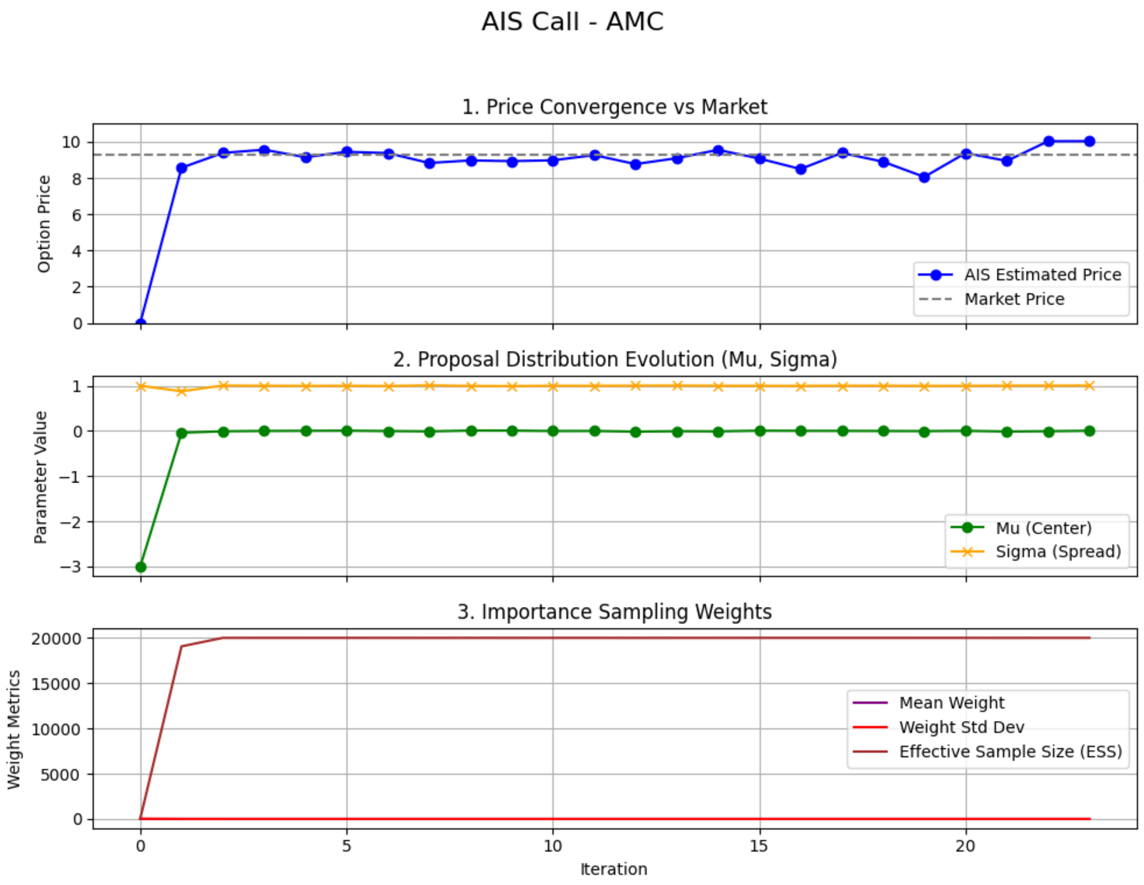


AIS can upweight sampling on long tails!

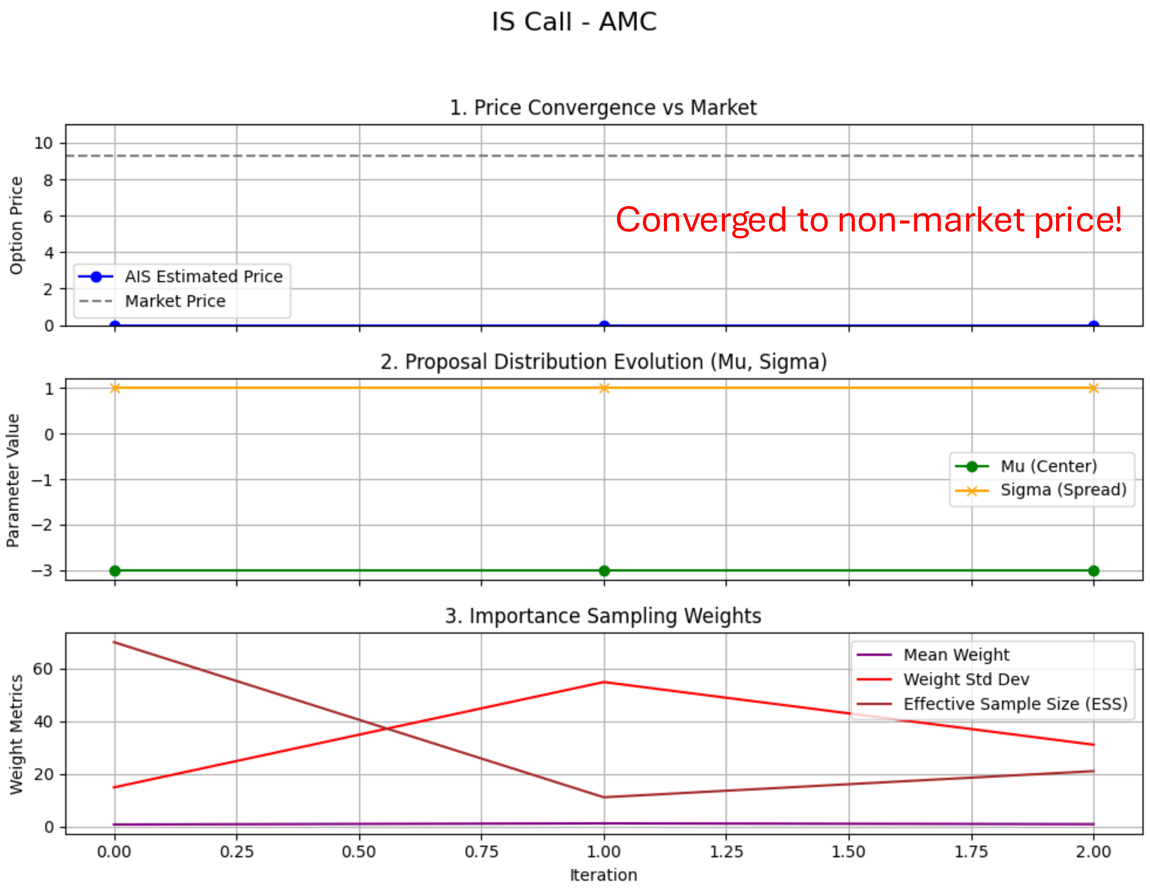
AIS better models OTM options

Out-of-the-market

AIS



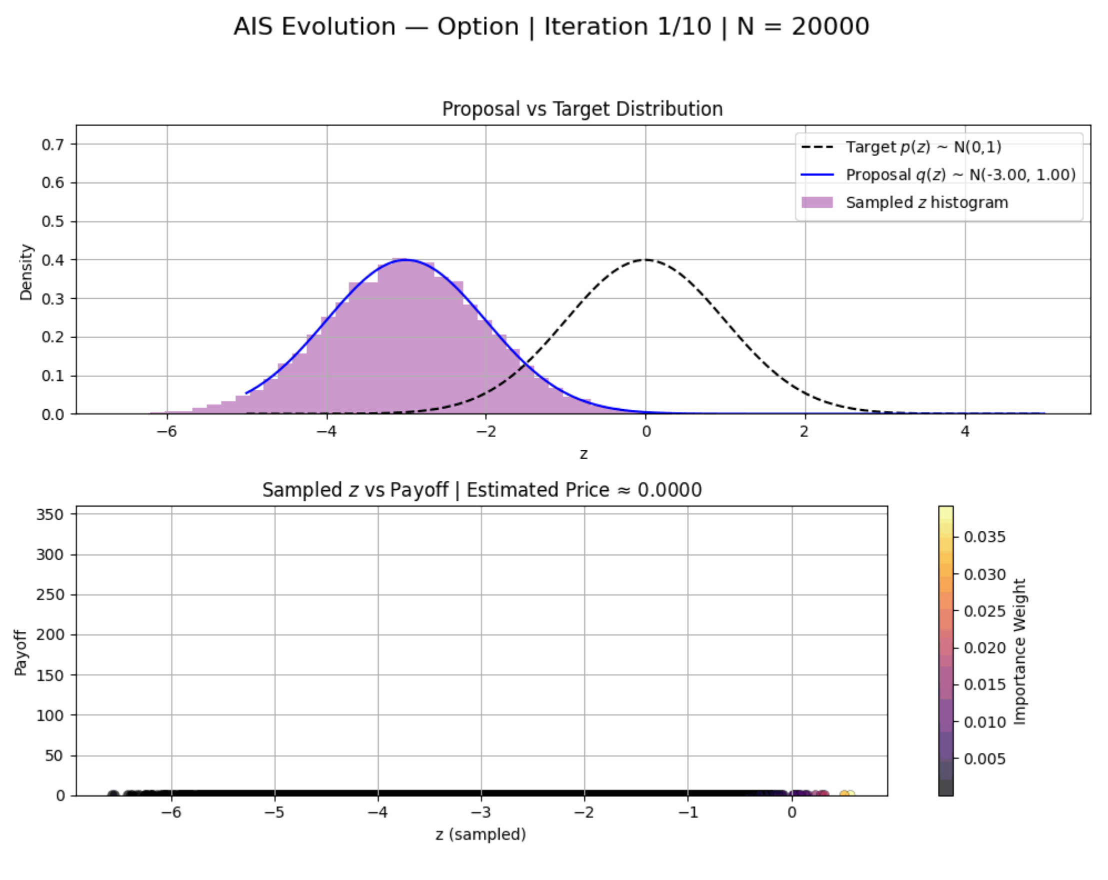
IS



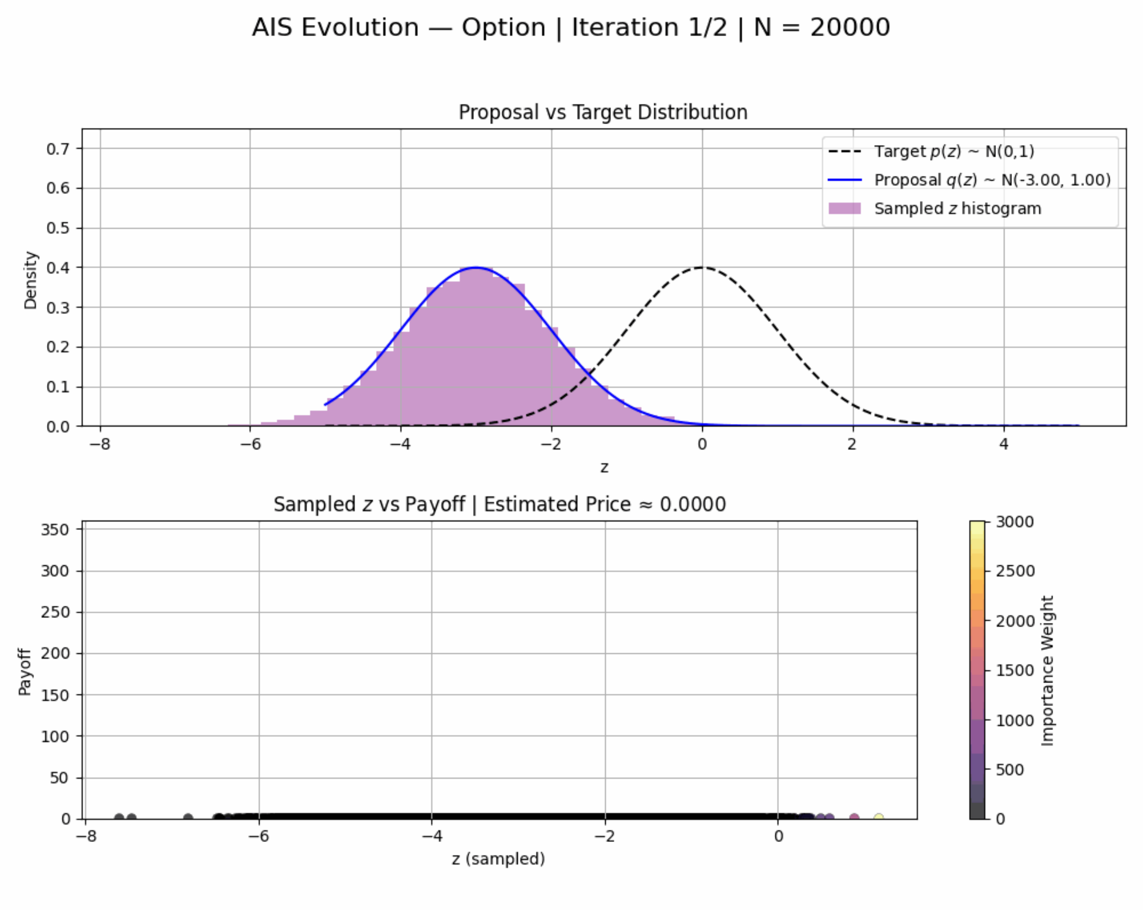
AIS better models OTM options

Out-of-the-market

AIS



IS



Conclusion

(1) **AIS adapts to market variability**, outperforming IS in volatile and changing conditions.
(2) **Better convergence** — AIS reduces variance faster as sample size N increases, especially in low-probability (OTM) regions.
(3) **Handles poor proposals well** — AIS is more robust than IS when the target distribution is unknown or poorly matched.
(4) **Stronger performance on rare events** — AIS prices OTM options more accurately due to its adaptability in the distribution tails.
(5) **No manual tuning needed** — AIS learns optimal sampling paths automatically, reducing modeling overhead.

References

Kahn, H. (1950). “Random sampling (Monte Carlo) techniques in neutron attenuation problems.” Nucleonics 6(5): 27–33.
Kloek, T. & van Dijk, H. K. (1978). “Bayesian Estimates of Equation System Parameters: An Application of Integration by Monte Carlo.” Econometrica 46(1): 1–19.
Kollman, C., Baggerly, K., Cox, D., & Picard, R. (1999). “Adaptive Importance Sampling on Discrete Markov Chains.” Annals of Applied Probability 9(2): 391–412.
Rubinstein, R. Y. & Kroese, D. P. (2004). “The Cross-Entropy Method: A Unified Approach to Combinatorial Optimization, Monte-Carlo Simulation, and Machine Learning.” Springer.
Cappe, O., Guillin, A., Marin, J.-M., & Robert, C. (2004). “Population Monte Carlo.” Journal of Computational and Graphical Statistics 13(4): 907–929
Douc, R., Cappe, O., & Moulines, E. (2007). “Comparison of resampling schemes for particle filtering.”
Cornuet, J.-M., Marin, J.-M., Mira, A., & Robert, C. (2012). “Adaptive Multiple Importance Sampling.” Scandinavian Journal of Statistics 39(4): 798–812.
Bugallo, M. F., Elvira, V., Martino, L., Luengo, D., Míguez, J., & Djuric, P. M. (2017). “Adaptive Importance Sampling: The past, the present, and the future.” IEEE Signal Processing Magazine 34(4): 60–79. 30
Martino, L., Elvira, V., Luengo, D., & Corander, J. (2017). “Layered Adaptive Importance Sampling.” Statistics and Computing 27: 599–623. Guetz, A. N. & Holmes, S. P. (2011). “Adaptive Importance Sampling for Network Growth Models.” Annals of Operations Research 189: 187–203.
Paananen, T., Piironen, J., & Vehtari, A. (2021). “Implicitly adaptive importance sampling.” Statistics and Computing 31(16).
Vehtari, A., Gelman, A., & Gabry, J. (2015). “Pareto smoothed importance sampling.”
Korba, A. & Portier, F. (2022). “Adaptive Importance Sampling meets Mirror Descent: a Bias-Variance Tradeoff.” Proceedings of AISTATS 2022.
Botev, Z. I., & Kroese, D. P. (2019). “Variance reduction.” Wiley StatsRef: Statistics Reference Online.

Thank you all for a great semester!