# MGMT MFE 406 – Derivative Markets (4 units)

## Part 4: (Introduction to) Monte Carlo Methods

Prof. Eric S. Reiner


UCLA Anderson

Winter 2025

# Outline

# 1. Introduction to Monte Carlo Methods

- Preamble
- Brief History and Readings/References
- Basic Strategy

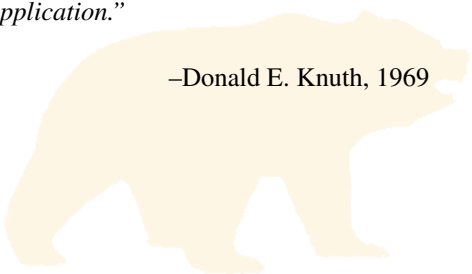
# 1.1. Preamble

*"Anyone who considers arithmetical methods of producing random digits is, of course, in a state of sin."*

– János (John) von Neumann, 1951

*"Every random number generator will fail in at least one application."*

–Donald E. Knuth, 1969



# 1.1. Preamble (2)

- Thus far, we've seen that the option valuation problem can be thought of as a discounted expectation in a risk-neutral (equivalent martingale) measure:

$$C_t(S_t) = e^{-r(T-t)}\mathbb{E}_t^{\mathbf{Q}}[C_T(S_T) \mid S(t)=S_t] = e^{-r(T-t)}\int_0^\infty dS_T\, q(S_T|S_t)C_T(S_T)$$

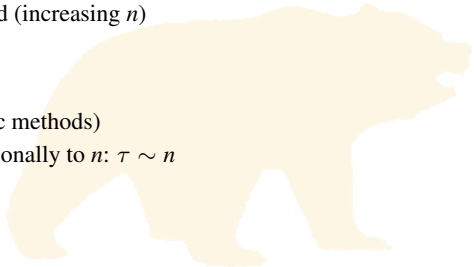
  with, e.g., $dS = (r-y)\,S\,dt + \sigma\,S\,dW_t^{\mathbf{Q}}$

- Much of the remainder of the class will be about ways to evaluate this expectation.
- What are some of the alternatives?
  - Analytical ("closed-form") solution
  - Numerical integration
  - Lattice/grid methods
  - Monte Carlo/simulation methods
  - (Analytical) Approximation methods

# 1.1. Preamble (3)

## Valuation Alternatives

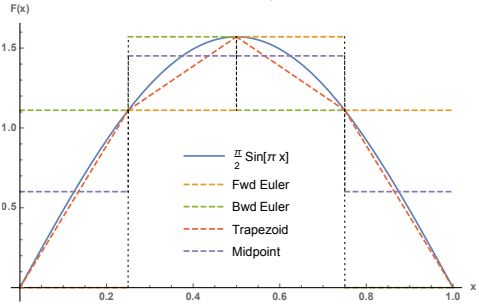
- Analytical ("closed-form") solution
  - Express integral(s) / calculate expectations in terms of (well-) known functions that can be evaluated to arbitrary accuracy in deterministic time
  - E.g., Black-Scholes formula
  - (Infinite) series solutions could probably be characterized as closed-form
- Numerical integration
  - Calculate expectation numerically via a grid of $n$ points in $S_T$
  - Convergence to (arbitrary) desired accuracy by refining the grid (increasing $n$)
  - E.g., binomial model formula
  - Convergence properties:
    - Assume error $\epsilon \sim n^{-\alpha}$ (typically $\alpha = 1$ or 2 for deterministic methods)
    - In one dimension, computation time $\tau$ usually grows proportionally to $n$: $\tau \sim n$
    - Hence $\tau \sim \epsilon^{-1/\alpha}$

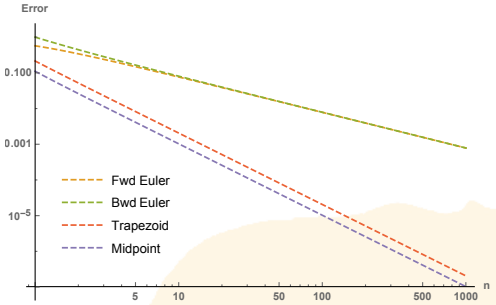


# 1.1. Preamble (4)

## Valuation Alternatives

- Numerical integration (continued)
  - Example: approximation of $\int_0^1 dx \, \frac{\pi}{2} \sin(\pi x) = 1$ with $n$ intervals



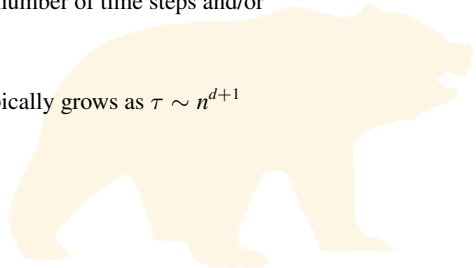




  - Curse of dimensionality:
    - What happens when we have to integrate over more than one dimension $d$?
    - Error still scales as $\epsilon \sim n^{-\alpha}$, but computation time $\tau$ now grows as $n^d$: $\tau \sim n^d$
    - Hence $\tau \sim \epsilon^{-d/\alpha}$

# 1.1. Preamble (5)

**Valuation Alternatives, continued**

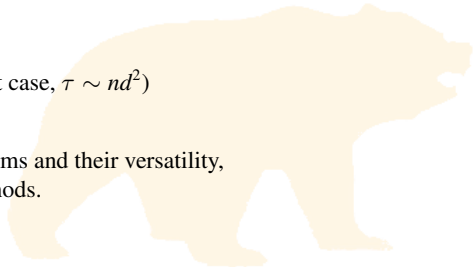
- Lattice/grid methods
  - E.g., binomial/trinomial trees, finite difference methods
  - Allow us to consider:
    - time-dependent parameters,
    - options with payoffs depending on multiple dates (e.g., early exercise),
    - path-dependency (at the cost of introducing additional state variables to track path metrics)
  - Convergence to (arbitrary) desired accuracy by increasing the number of time steps and/or refining the grid in asset price space
  - Curse of dimensionality:
    - error still scales as $\epsilon \sim n^{-\alpha}$, but computation time $\tau$ now typically grows as $\tau \sim n^{d+1}$
    - Hence $\tau \sim \epsilon^{-(d+1)/\alpha}$



# 1.1. Preamble (6)

## Valuation Alternatives, continued

- Monte Carlo methods
  - Sample from the underlying density $q(S_T)$
  - Use independent samples to construct an estimator of $\mathbb{E}_t^{\mathbf{Q}}[C_T(S_T)|S(t)=S_t]$
  - Allow us to consider time-dependent parameters, multiple assets, path-dependency
  - Early exercise also possible, though this is done by forcing the Monte Carlo to behave (at least somewhat) like a grid method
  - Convergence to (arbitrary) desired accuracy by increasing the number of samples $n$
  - Curse of dimensionality (?)
    - error typically scales as $\epsilon \sim n^{-1/2}$ (central limit theorem)
    - computation time $\tau$ now typically grows as $\tau \sim nd$ (or, worst case, $\tau \sim nd^2$)
    - Hence $\tau \sim d\epsilon^{-2}$ (worst case, $\tau \sim d^2\epsilon^{-2}$)
  - Because of their broad applicability to a wide variety of problems and their versatility, we'll spend much of the remainder of the course on these methods.



# 1.1. Preamble (7)

## Valuation Alternatives, continued

- Approximation methods
  - E.g., asymptotic expansions, moment-matching methods
  - Properties similar to closed-form solutions but (usually) without ability to refine error to arbitrary accuracy
  - Time-dependent parameters, multiple assets, some forms of path-dependency possible
  - Challenge is characterizing (and bounding) error
  - No curse of dimensionality
  - If time permits, we'll introduce some of these methods along the way while we're developing Monte Carlo techniques

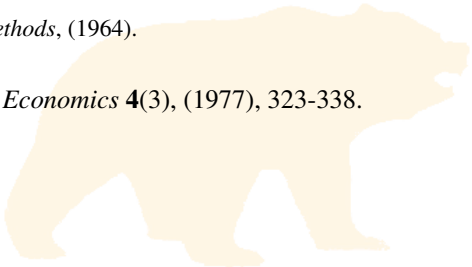
## How to rank alternatives?

- All other things being equal:
  - Closed form preferred to numerical solution
  - Deterministic method preferred to stochastic
  - Shorter execution time preferred to longer
  - Simpler algorithm, code, and maintenance preferred to more complex

# 1.2. Brief History and Readings/References

## History

- Development during WWII (atomic bomb design at Los Alamos):
  von Neumann, Fermi, Ulam, Metropolis, Teller...
  - Original reference (Markov-chain Monte Carlo/Metropolis-Hastings algorithm):
    Metropolis, N.; Rosenbluth, A.W.; Rosenbluth, M.N.; Teller, A.H.; Teller, E., "Equations of State Calculations by Fast Computing Machines," *Journal of Chemical Physics* **21**(6), (1953), 1087-1092.
- Subsequent refinement and applications particularly in the broad physics community.
  By the 1960s, the theory was more-or-less complete.
  - See: Hammersley, J.M. and D.C. Handscomb, *Monte Carlo Methods*, (1964).
- Introduced to option pricing by Phelim Boyle:
  "Options: A Monte Carlo Approach," *Journal of Financial Economics* **4**(3), (1977), 323-338.



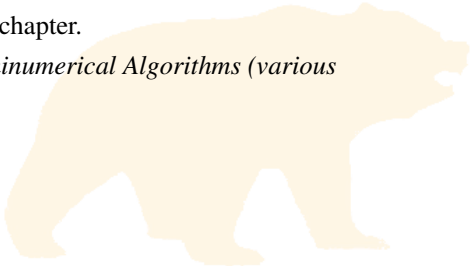
# 1.2. Brief History and Readings/References (2)

## Further Readings

- Tavella, Chapter 5.
- Jäckel, Chapters 7, 9-11, and (lightly) 2, 4.
- Glasserman, Chapters 1, 2, 3.1-3.2, 4.1-4.2, 4.5.
- Seydel, Chapters 2, 3.

## Other References (random number generation)

- *Numerical Recipes (various editions)*, “Random numbers” chapter.
- Knuth, *The Art of Computer Programming, Volume 2 / Seminumerical Algorithms (various editions)*, Chapter 3 – “Random numbers".



# 1.3. Basic Strategy

- We want to sample from the underlying distribution $q(S_T)$ to construct an estimator of $\mathbb{E}_t^{\mathbf{Q}}[C_T(S_T) \,|\, S(t) = S_t]$ and, thereby, $C_t(S_t)$:

$$C_t(S_t) = e^{-r(T-t)}\mathbb{E}_t^{\mathbf{Q}}[C_T(S_T) \,|\, S(t) = S_t] = e^{-r(T-t)}\int_0^\infty dS_T\, q(S_T|S_t)C_T(S_T)$$

  with, e.g., $dS = (r-y)\,S\,dt + \sigma\,S\,dW_t^{\mathbf{Q}}$
- Generate $n$ independent scenarios ($n$ large) for $S_T$ – and thereby $C_T(S_T)$ – from the density $q(S_T|S_t)$.
- Average over scenarios to obtain estimators $\langle C_T \rangle_n$ and $\langle C_t \rangle_n$ of $\mathbb{E}^{\mathbf{Q}}[C_T]$ and $C_t$:

$$\mathbb{E}^{\mathbf{Q}}[C_T] \simeq \langle C_T \rangle_n \doteq \frac{1}{n}\sum_{i=1}^n C_{T,i}(S_{T,i})$$

$$C_t \simeq \langle C_t \rangle_n \doteq \frac{1}{n}\sum_{i=1}^n e^{-r(T-t)}C_{T,i}(S_{T,i}) = \frac{e^{-r(T-t)}}{n}\sum_{i=1}^n C_{T,i}(S_{T,i}) = e^{-r(T-t)}\langle C_T \rangle_n$$

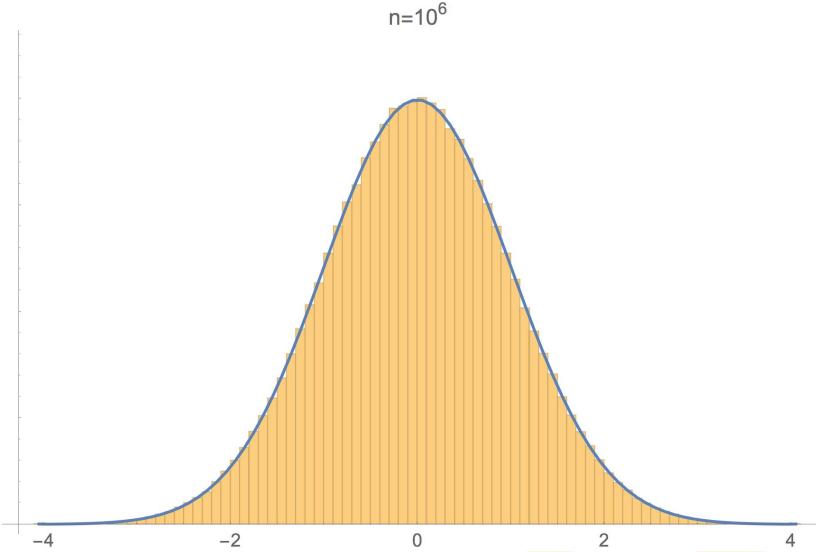
- Central limit theorem tells us about convergence properties of these estimators:

$$\text{var}[\,\langle C_T \rangle_n\,] \simeq \frac{1}{n}\text{var}[C_{T,i}] \simeq \frac{1}{n}\text{var}^{\mathbf{Q}}[C_T] \implies \langle C_T \rangle_n \sim N\left(\mathbb{E}^{\mathbf{Q}}[C_T], \frac{\text{var}^{\mathbf{Q}}[C_T]}{n}\right)$$

  - What does this tell us about convergence of the standard deviation (standard error)?

# 1.3. Basic Strategy (2)

## Monte Carlo – Illustration



# 2. Generating Samples from a Distribution

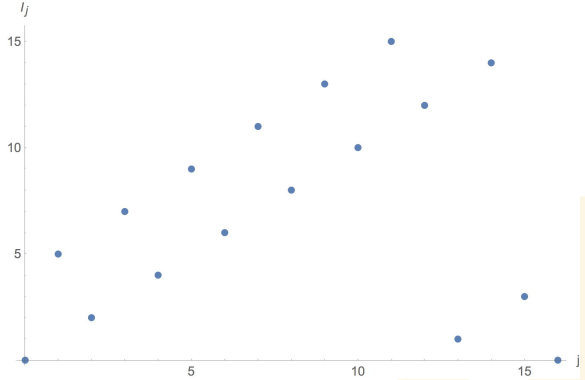
## How do we sample from $q(S_T)$?

- Starting point is most often sampling from the uniform distribution $U(0,1)$:
$$\text{pdf } u(u) = 1 \Rightarrow \text{cdf } \mathcal{U}(u) = u \;\; \forall\, u \in [0,1], (0,1], \text{or } [0,1)$$
  - Why? Hint: the cdf of any distribution can be mapped to $U$.
- Implementation
  - Almost every operating system, programming language, and statistical or algebra package has one or more built-in functions e.g., **rand**() to generate pseudo-random draws $u_j$ from $U(0,1)$.
    Some of them are "good," some are not so good...
  - Almost all of these functions generate the draws by dividing a sequence $I_j$ taken from the set of integers $\{0, 1, ..., N-1\}$ ($N$ large) by $N$
  - The sequence is completely deterministic and recurrent, e.g., linear congruential (LCG) method:
$$I_{j+1} = a\, I_j + c \mod N$$
    with multiplier $a$ and offset or increment $c$ (both integers)
  - If $a$ and $c$ are chosen well (e.g., $a$ is relatively prime to $N$... but there's a good bit more to it than that), the sequence will have (full) period $N$ and successive draws will appear random
  - $u_j$ are both discretely distributed and deterministic!
  - To add insult to injury, the deterministic property is actually highly desirable...
    - Consistency
    - Reproducibility
    - Seeding the random number generator: **srand**() or similar function

# 2. Generating Samples from a Distribution (2)

## Simple Example

- Choose $N = 16, a = 9, c = 5$ and consider an initial $(j = 0)$ value (seed) of 0:
  - $I_1 = (9 \cdot 0 + 5) \bmod 16 = 5$
  - $I_2 = (9 \cdot 5 + 5) \bmod 16 = 50 \bmod 16 = 2$
  - Sequence generated: $\{0, 5, 2, 7, 4, 9, 6, 11, 8, 13, 10, 15, 12, 1, 14, 3, 0, ...\}$




- A simple example of Marsaglia's "Random numbers Fall Mainly in the Planes" effect: *Proc. Natl. Acad. Sci.* **61**(1), (1968), 25-28.

# 2. Generating Samples from a Distribution (3)

## Practical Matters

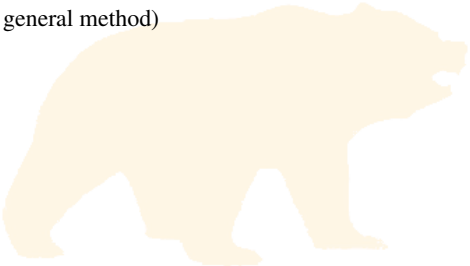
- For prototyping, built-in system functions probably suffice
  - Some programming environments, e.g., MATLAB, R, Mathematica,... have fairly advanced built-in RNGs
- For production code, it may be worthwhile to implement a more sophisticated method such as:
  - the **Ran**() function in *Numerical Recipes:* combines several different single-state methods
  - the "Mersenne Twister" described in Jäckel (pp. 74-5): 623-element (vector) state space
  - More recent: O'Neill's PCG (Permuted Congruential Generator, 2014) www.pcg-random.org
  - Also recent: xorshiro (2016) https://arxiv.org/abs/1805.01407
- Test for randomness
  - Autocorrelation function $\langle u_i, u_{i+k} \rangle$
  - Scatter plots of pairs $\{u_i, u_{i+k}\}$
  - Marsaglia's "Diehard Battery" of statistical tests (see *Numerical Recipes*)
- Watch out...
  - Finite resolution (discreteness) $1/N$ and recurrence properties of $u_i$ may lead to sampling problems, particularly in the tails of the distribution.

    See: Neave, H. R., "On using the Box-Muller transformation with multiplicative congruential pseudo-random number generators," *Applied Statistics* **22**, (1973), 92-97.
  - Transformation methods may require adjustment of "support" of $u_i$ to $[0, 1]$, $(0, 1]$, or $[0, 1)$ to avoid sampling (unreachable) points at 0 or $\infty$ (and/or code blowing up)

# 2. Generating Samples from a Distribution (4)

## Generating (Standard) Normally Distributed Samples

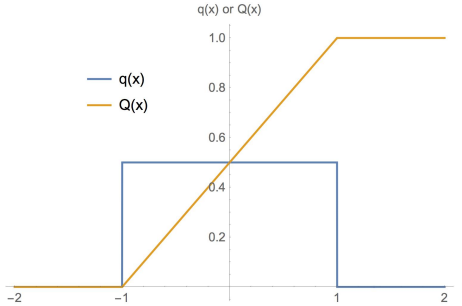
- Several algorithms are available, including:
  - Inverse CDF method (specific application of general method)
  - Box-Muller (polar transformation) method (specific to normal distribution)
  - Appendix: Polar rejection method (specific to normal distribution)
  - Appendix: General rejection method(s) (specific application of general method)

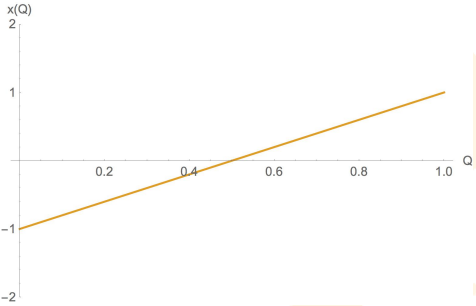


# 2.1. Inverse CDF method

- Leverage the idea that for any probability density $q(x)$ with support (defined on): $\{x : a \le x \le b\}$, the cdf $Q(x) \doteq \int_a^x dx'\, q(x')$ is bounded by $0 \le Q(x) \le 1$.
  - As long as $q(x)$ doesn't vanish anywhere in $[a,b]$, because $q(x)$ is a probability density (everywhere positive), $Q(x)$ is monotonically increasing in $x$ and so $x|_a^b \leftrightarrow Q|_0^1$ is a one-to-one mapping.
  - If we can sample $Q$ from $U(0,1)$ *and* we can evaluate the (inverse) mapping $x(Q) \doteq Q^{-1}$, we can sample from the density $q(x)$.
- Simple example, uniform density $U(a,b)$, e.g., $U(-1,1)$: $q(x) = \frac{1}{b-a}\ \forall x \in [a,b]$:

$$Q(x) = \int_a^x dx' \frac{1}{b-a} = \frac{x-a}{b-a} \implies x = a + (b-a)Q \sim a + (b-a)u \text{ with } u \sim U[0,1]$$



# 2.1. Inverse CDF method (2)

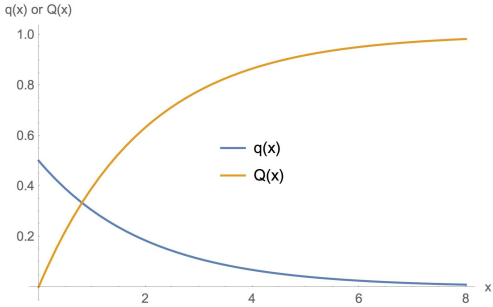
## More interesting example

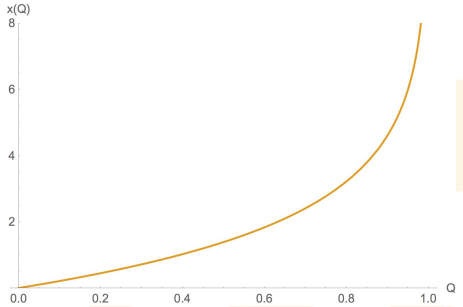
- Exponential density $E(\lambda): q(x) = \lambda e^{-\lambda x} \, \forall x \in [0, \infty)$ with, e.g., $\lambda = 1/2$:

$$Q(x) = \int_0^x dx' \lambda e^{-\lambda x'} = 1 - e^{-\lambda x} \Rightarrow e^{-\lambda x} = 1 - Q$$

$$\implies x = -\ln(1-Q)/\lambda \sim -\ln(1-u)/\lambda \text{ with } u \sim U[0,1)$$

$$\text{or } x \qquad\qquad \sim -\ln(u)/\lambda \text{ with } u \sim U(0,1]$$



# 2.1. Inverse CDF method (3)

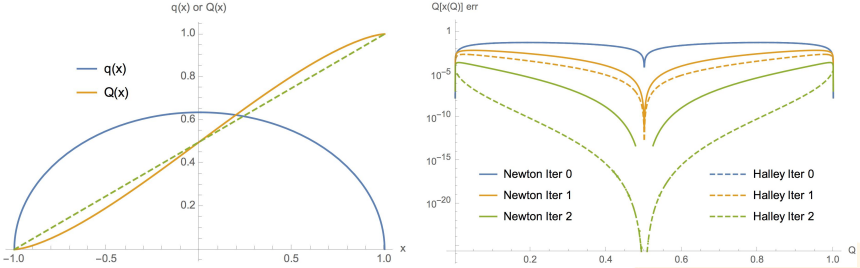
**How can we apply this method to sample from the standard normal distribution?**

- We need $\mathcal{N}^{-1}(Q)$.
  - Unfortunately, closed-form doesn't exist in terms of standard functions available in many environments.
  - However, numerical implementations do exist:
    see references in chapter 2.3 of Jäckel (who is fond of this method) – *caveat emptor*!
    - Acklam, P. J., "An algorithm for computing the inverse normal cumulative distribution function," University of Oslo, Statistics Division, (Jun. 2000). Wayback Machine archive at:
      web.archive.org/web/20070505093933/home.online.no/∼pjacklam/notes/invnorm/
    - Moro, B.,"The Full Monte," *Risk Magazine* **8**(2), (Feb. 1995), 57-58.
    - Worst case: use one of these as a close approximation, then refine (or "polish") the value with one iteration of Newton's or Halley's method.
  - See also:
    - Marsaglia, G.; Zaman, A.; Marsaglia, J., "Rapid evaluation of the inverse of the normal distribution function," *Statistics and Probability Letters* **19**(4), (15 Mar. 1994), 259-266.
    - Wichura, M., "Algorithm AS 241: The Percentage Points of the Normal Distribution," *Applied Statistics*, **37**(3), (1988), 477-484.
    - Shaw, W., "Refinement of the Normal Quantile," King's College London wkg paper (20 Feb 2007)
      web.archive.org/web/20070609171237/mth.kcl.ac.uk/∼shaww/web_page/papers/NormalQuantile1.pdf
    - Giles, M., "Approximating the ERFINV function," Oxford University, Institute of Quantitative Finance, (2011).
      people.maths.ox.ac.uk/gilesm/files/gems_erfinv.pdf.
- Are there more direct alternatives?

# 2.1. Inverse CDF method (4)

## Appendix: What to do if the *quantile function* $Q^{-1}$ doesn't have a simple closed form?

- Solution if $Q$ *is* available and we can find a good initial approximation $Q_0^{-1}$ to $Q^{-1}$
  E.g., $q(x) = \frac{2}{\pi}\sqrt{1-x^2}\ \forall x \in [-1,1]$ for which $Q(x) = \frac{1}{2} + \frac{1}{\pi}[x\sqrt{1-x^2} + \sin^{-1}(x)]$:




  In this example, try $Q(x) \simeq \frac{1+x}{2} \Rightarrow Q_0^{-1} = 2Q - 1$

- Then use Newton's method or Halley's method on the equation $Q\left[Q^{-1}(\hat{Q})\right] = \hat{Q}$ (with $\hat{Q}$ the input value of $Q$) to refine the approximation to the precision needed.
- In practice, start with Taylor or asymptotic series around endpoints and/or midpoint
  - There are standard methods (Padé approximants, Chebyshev series, ...) for improving the approximations so that each is a good fit to $Q^{-1}$ over a finite sub-interval of $Q$ rather than merely near an isolated point.

# 2.2. Box-Muller (polar transformation) method

- Remember polar transformation to prove normalization of Gaussian distribution?
  If we take two independent Gaussian variates $\{z_1, z_2\} = \{x, y\}$, we can change from Cartesian co-ordinates $\{z_1|_{-\infty}^{+\infty}, z_2|_{-\infty}^{+\infty}\}$ to polar co-ordinates $\{r|_0^\infty, \theta|_0^{2\pi}\}$:

$$\int_{-\infty}^{+\infty} dz_1 \frac{e^{-z_1^2/2}}{\sqrt{2\pi}} \int_{-\infty}^{+\infty} dz_2 \frac{e^{-z_2^2/2}}{\sqrt{2\pi}} F(z_1, z_2) = \int_{-\infty}^{+\infty} dz_1 \int_{-\infty}^{+\infty} dz_2 \frac{e^{-(z_1^2+z_2^2)/2}}{2\pi} F(z_1, z_2)$$
$$= \int_0^{2\pi} \frac{d\theta}{2\pi} \int_0^{\infty} dr\, r\, e^{-r^2/2} F(r, \theta)$$

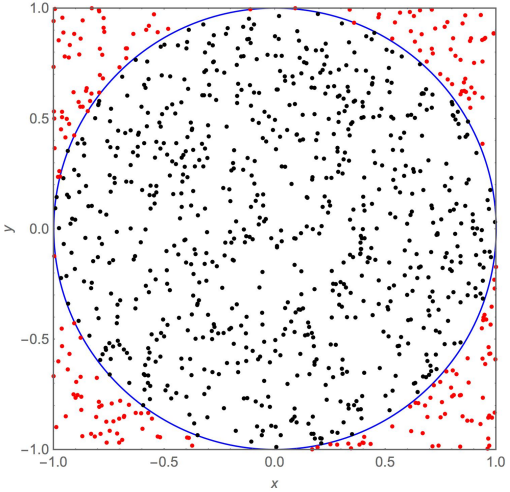
- Since $\int_0^\infty dr\, r\, e^{-r^2/2} = 1$ and $\int_0^{2\pi} \frac{d\theta}{2\pi} = 1$, and the integrands are everywhere positive & bounded, $\int_{r^*}^\infty dr\, r\, e^{-r^2/2} = e^{-r^{*2}/2}$ and $\int_0^{\theta^*} \frac{d\theta}{2\pi} = \frac{\theta^*}{2\pi}$ are independent and uniform $U(0,1)$ distributed.
- Reverse process: take $u_1 = e^{-r^{*2}/2}$ and $u_2 = \frac{\theta^*}{2\pi}$ as independent uniform draws from $(0, 1]$ and let:

$$z_1 = x = r^* \cos(\theta^*) = \sqrt{-2 \ln u_1} \cos(2\pi u_2) \text{ and}$$
$$z_2 = y = r^* \sin(\theta^*) = \sqrt{-2 \ln u_1} \sin(2\pi u_2)$$

  $\Longrightarrow$ $\{z_1, z_2\}$ are independent standard normal random variables

# 2.3. Appendix: Polar rejection method

- Take $u_1^*$ and $u_2^*$ as independent uniform draws from $[-1, 1]$.
- Let $r = \sqrt{(u_1^*)^2 + (u_2^*)^2}$. Discard results if $r = 0$ or $r > 1$.




- Asymptotic acceptance rate = (circle area)/(square area) = $\pi/4 \approx 0.785398\ldots$

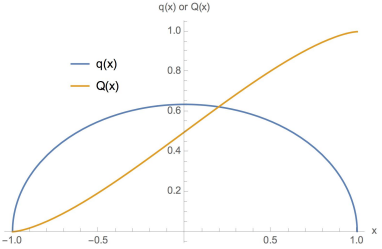
# 2.3. Appendix: Polar rejection method (2)

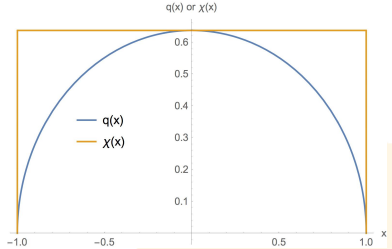
- Conditional on acceptance, we now have points uniformly distributed across the area $(=\pi)$ of a unit circle. Normalize to measure $1 = \frac{1}{\pi}\int_0^{2\pi} d\theta \int_0^1 dr\, r = \int_0^{2\pi} \frac{d\theta}{2\pi} \int_0^1 dr\, 2r$
  - So, conditional on $r \leq 1$, the points are uniformly distributed vs. $\theta$ and the calculation of trigonometric functions can be replaced by ratios with respect to $r$:
    $$u_1^*/r = x/r \leftrightarrow \cos(\theta) \text{ and } u_2^*/r = y/r \leftrightarrow \sin(\theta)\,.$$
  - We've thus used rejection to convert points sampled uniformly from a square to points uniformly distributed over the area of a unit circle: $\{u_1^*|_{-1}^{+1}, u_2^*|_{-1}^{+1}\} \rightarrow \{r|_0^1, \theta|_0^{2\pi}\}$
  - But what we need is points distributed as a 2-dimensional Gaussian on $\{r^*|_0^\infty, \theta|_0^{2\pi}\}$.
    - The (uniform) $\theta$ measure is the same in both cases: $\frac{d\theta}{2\pi}|_0^{2\pi}$.
    - But we need to transform from the cdf for r: $\int_0^r dr'\, 2r' = r^2$ to align with the Box-Muller (complementary) cdf for $r^*$: $\int_{r^*}^\infty dr' r' e^{-r'^2/2} = e^{-r^{*2}/2} \implies r^* = \sqrt{-4\ln r}$.
  - So, setting: $z_1 = \sqrt{-4\ln r}\left(\frac{u_1^*}{r}\right)$ and $z_2 = \sqrt{-4\ln r}\left(\frac{u_2^*}{r}\right)$
    $\implies \{z_1, z_2\}$ are independent standard normal random variables
- Several variations on this theme exist.
- More of historical interest than current practical use:
  dates from the era when $\cos(\cdot)$ and $\sin(\cdot)$ functions were computationally expensive.
- Can wreak havoc on special quasi-random sequences with optimized space-filling properties.

# 2.4. Appendix: General rejection method(s)

**What can we do in the general case where $Q^{-1}$ doesn't have a simple closed form?**

- Take some hints from the previous example:
  If we can enclose the area [$\leftrightarrow$ the pdf $q(x)$] we *want* to sample from inside a larger area [call it $\chi(x)$] that we *can* sample from, then we can sample from that larger area ("the comparison function") and simply throw away the points that don't fall inside the smaller (enclosed) area!
  E.g., $q(x) = \frac{2}{\pi}\sqrt{1-x^2}\ \forall x \in [-1,1]$ for which $Q(x) = \frac{1}{2} + \frac{1}{\pi}[x\sqrt{1-x^2} + \sin^{-1}(x)]$:



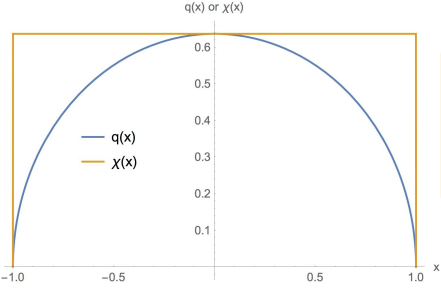




- Simplest idea (inspired by previous example): take $\chi(x) = \frac{2}{\pi}\ \forall x \in [-1,1]$
  - $\chi(x)$ obviously isn't a probability density since $X(x) = \int_{-1}^{x} dx' \chi(x') = \frac{2}{\pi}(x+1)\ \forall x \in [-1,1]$ isn't bounded from above by 1.
  - However, if we normalize by the total area under $\chi(x)$: $\bar{X}(x) \doteq \frac{X(x)}{X(1)} = \frac{\pi}{4}X(x) = \frac{1}{2}(x+1)$, then $\bar{X}(x)$ *is* a cdf! Hint: $\frac{\pi}{4}$ is just the acceptance ratio for the polar rejection method...

# 2.4. Appendix: General rejection method(s) (2)

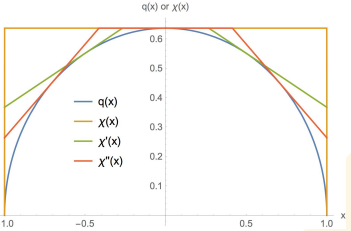
**So, our basic algorithm combines two ideas:**

1. Take the bounding area defined by $\chi(x)$ and integrate/normalize this to obtain the corresponding cdf $\bar{X}(x)$. Apply the inverse cdf method to sample for values of x.
   - I.e. draw a first uniform $U(0,1)$ sample $u_1$ and apply the inverse of $\bar{X}(x)$: $x(\bar{X}) = x(u_1)$ to select a value of $x$.
   - For our example: $\bar{X}(x) = \frac{1}{2}(x+1) \Rightarrow x(\bar{X}) = x(u_1) = (2u_1 - 1)$
2. Now, apply the rejection method to decide whether to retain (and therefore sample) or discard $x(u_1)$.
   - I.e. draw a second uniform $U(0,1)$ sample $u_2$. If $u_2 \leq \frac{q(x)}{\chi(x)}$ retain $x(u_1)$; otherwise discard it.
   - For our example, we'll keep $x(u_1)$ if: $u_2 \leq \frac{q(x)}{\chi(x)} = \frac{(2/\pi)\sqrt{1-x^2}}{(2/\pi)} = \sqrt{1-x^2}$, with average acceptance probability $\frac{\pi}{4} = 0.785398...$



# 2.4. Appendix: General rejection method(s) (3)

- In our example, we chose the simplest $\chi(x)$ possible: a constant.
- But we are free to choose any comparison function $\chi(x)$ that (1) bounds $q(x)$ from above and (2) is (piecewise) integrable, then invertible. E.g.:
  - The piecewise linear envelope obtained by combining the segment $\chi(x)=\frac{2}{\pi}$ with the tangent(s) to $q(x)$ at $x=\pm\frac{1}{2}$ (with average acceptance probability $(3+\sqrt{3})\frac{\pi}{16}=0.929136...$)
  - The optimal 3-segment envelope obtained by combining $\chi(x)=\frac{2}{\pi}$ with the tangent(s) to $q(x)$ at $x=\pm\frac{\sqrt{2}}{2}$ (with average acceptance probability $(1+\sqrt{2})\frac{\pi}{8}=0.948059...$)




- We need not restrict ourselves to piecewise linear $\chi(x)$. E.g., we might choose comparison functions based on distributions whose cdfs we know how to invert.
- *Numerical Recipes* develops many applications of this approach, including a particularly tight method for sampling from the standard normal distribution.

# 3. Monte Carlo Pricing

**To motivate what follows, consider valuation of a vanilla call option**

- Fix the following set of parameters:

$$S = K = 100, T-t = 1, r = 0.04, y = 0.02, \sigma = 0.2, \text{ so that:}$$

$$\mathbb{E}^{\mathbf{Q}}[S_T] = F_T = Se^{(r-y)(T-t)} = 100\,e^{(0.04-0.02)(1)} = 100\,e^{0.02} = 102.020$$

$$\mathbb{E}^{\mathbf{Q}}[C_t] = BSC(S,K,T-t,r,y,\sigma) = BSC(100,100,1,0.04,0.02,0.2) = 8.73949$$

- Examine convergence properties vs $n$, $2^1 \le n \le 2^{20}$, of
  - estimators: $\langle(\cdot)\rangle_n = \frac{1}{n}\sum_{i=1}^{n}(\cdot)_i$, and
  - their asymptotic i.i.d./central limit theorem standard errors: $\text{std}[\langle(\cdot)\rangle_n] = \sqrt{\text{var}[(\cdot)_i]_n/n}$, where $\text{var}[(\cdot)_i]_n$ is the sample variance calculated in the simulation itself.
- Consider both $\langle S_T\rangle_n$ and $\langle C_t\rangle_n$, and examine both:
  - absolute deviations $|\langle S_T\rangle_n - \mathbb{E}^{\mathbf{Q}}[S_T]|$ and $|\langle C_t\rangle_n - \mathbb{E}^{\mathbf{Q}}[C_t]|$ ($\sim \text{std}[\langle(\cdot)\rangle_n]\,|N(0,1)|$) from the respective exact values $\mathbb{E}^{\mathbf{Q}}[S_T]$ and $\mathbb{E}^{\mathbf{Q}}[C_t]$ **(solid dots •)** and
  - (estimator) standard errors $\text{std}[\langle S_T\rangle_n]$ and $\text{std}[\langle C_t\rangle_n]$ **(solid lines and empty dots —○—)** .
- Display results on log scales so that convergence to $n^{-1/2}$ scaling can be observed.

# 3. Monte Carlo Pricing (2)

## Appendix: Sub-sampling/Binning/Batching

- There are occasions in which the "official" (asymptotic CLT) estimator of the standard error doesn't actually describe the distribution of $\langle(\cdot)\rangle_n$.
  - This usually occurs because of some violation of the i.i.d. CLT assumptions.
  - The solution is to implement a corrected estimator using sub-sampling (or "batching" or "binning"), in which the $n$ samples are divided into $m$ buckets of $n' = n/m$ samples each so that $\langle(\cdot)\rangle_n = \left\langle \langle(\cdot)\rangle_{n'} \right\rangle_m$ and the standard error is estimated as $\sqrt{\mathrm{var}[\ \langle(\cdot)\rangle_{n'}\ ]_m/m}$.
  - We implement this approach, denoted by $\mathrm{std}^*[\langle S_T\rangle_n]$ and $\mathrm{std}^*[\langle C_t\rangle_n]$, (shown as **dashed lines and empty squares – ▫ –** in the examples to follow) to test for CLT behavior, using $m = 2^{10} = 1024$.
- N.B. Normally, binning preserves sample means. But, in some instances, we'll need to be careful about how variance reduction techniques interact with this approach.
  - e.g., moment-matching methods, where the whole point is to break the CLT assumptions (and hence the matching must usually be done inside each bin).
- For clarity, what this effectively means is that we will run the model $m$ times (batches), with each batch containing $n' = n/m$ of the $n$ samples.
  - The overall ("$n$") estimators are then calculated as statistics of the $m$ individual batch means.
  - Both $m$ and $n'$ must be sufficiently large for the CLT to apply both within and across batches.
  - Each batch's samples must be independent of all the others.

# 3.1. Bare-bones Monte Carlo (BBMC)

## BBMC Example Code

- Let: $S_{T,i} = S_t \exp\left[(r-y-\sigma^2/2)(T-t) + \sigma\sqrt{T-t}\, z_i\right]$, $C_{T,i} = \max[S_{T,i} - K, 0]$

```
BSCBBMC[ S_, K_, T_, r_, y_, σ_, n_, Seed_ ] :=
  Module[ {i, (* integer iterator *)
    σSqrtT, μ, μtildeT, DiscountFactor, OneOverSqrtN, (* constants *)
    μST, StdST, StdμST, μCT, StdCT, StdμCT, μCt, StdμCt, (* statistics/estimators *)
    Norms, ST, CT}, (* vector results *)

   (* Calculate basic simulation parameters *)
   σSqrtT = σ Sqrt[T]; (* multiplier for std. normal draws *)
   μ = r - y; (* drift *)
   μtildeT = μ T - 0.5 σSqrtT^2; (* log drift *)
   DiscountFactor = Exp[-r T]; (* discount factor *)
   OneOverSqrtN = 1.0 / Sqrt[N[n]]; (* 1/sqrt(n) factor for CLT estimators *)

   (* Seed generator and draw n standard normal variates *)
   SeedRandom[Seed]; (* seed random number generator *)
   Norms = RandomVariate[NormalDistribution[], {n}]; (* fill vector of length n with std. normal draws *)
```

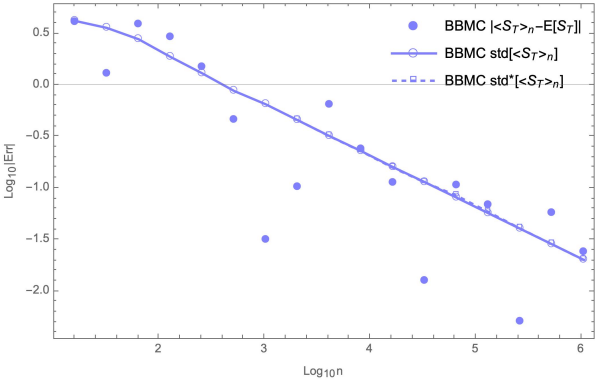
# 3.1. Bare-bones Monte Carlo (BBMC) (2)

- BSCBBMC[$S\_, K\_, T\_, r\_, y\_, \sigma\_, n\_, Seed\_$] code, continued:

```
(* Seed generator and draw n standard normal variates *)
SeedRandom[Seed]; (* seed random number generator *)
Norms = RandomVariate[NormalDistribution[], {n}]; (* fill vector of length n with std. normal draws *)

(* Calculate share prices S_T and statistics *)
ST = S Exp[ μtildeT + σSqrtT Norms ]; (* vector of S_T *)
μST  = Mean[ST]; (* mean of S_T vector *)
StdST = StandardDeviation[ST]; (* sample std. dev. of S_T vector *)
StdμST = OneOverSqrtN StdST; (* CLT estimator of std. dev. of mean of S_T *)

(* Calculate option payoffs C_T and statistics, then discount to get C_t *)
CT = Table[ Max[ ST[[i]] - K, 0], {i, n}]; (* evaluate vector of call payoffs C_T using vector of S_T *)
μCT  = Mean[CT]; (* mean of C_T vector *)
StdCT = StandardDeviation[CT]; (* sample std. dev. of C_T vector *)
StdμCT = OneOverSqrtN StdCT; (* CLT estimator of std. dev. of mean of C_T *)

μCt = DiscountFactor μCT; (* discount mean of C_T vector to get C_t estimator *)
StdμCt = DiscountFactor StdμCT; (* same for estimator of std. dev. of mean of C_t *)

Return[ { {μCt , StdμCt}, {μCT , StdμCT}, {μST , StdμST} } ];
];
```

# 3.1. Bare-bones Monte Carlo (BBMC) (3)

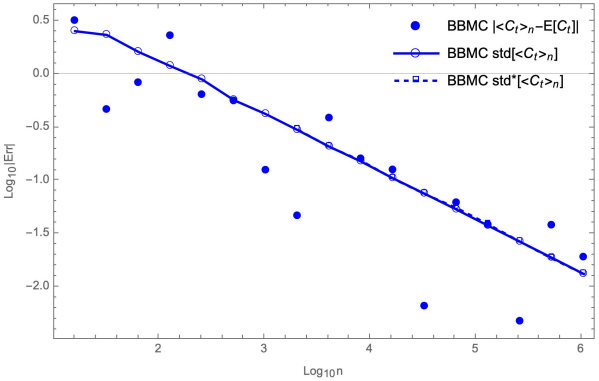
**Start with convergence properties of $\langle S_T \rangle_n \to \mathbb{E}^{\mathbf{Q}}[S_T]$:**




- After an initial transient, CLT behavior seems to kick in by $n \simeq 10^3$
- Binned and CLT estimators of error in $\mathbb{E}^{\mathbf{Q}}[S_T]$ appear to describe the data well

# 3.1. Bare-bones Monte Carlo (BBMC) (4)

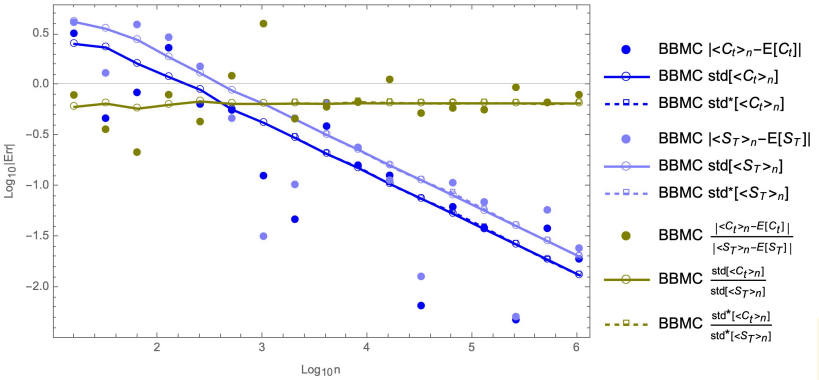
**Next, examine $\langle C_t \rangle_n \to \mathbb{E}^{\mathbf{Q}}[C_t] = e^{-r(T-t)}\mathbb{E}^{\mathbf{Q}}[C_T]$:**




- CLT behavior seems to be in place throughout
- Binned and CLT estimators of error in $\mathbb{E}^{\mathbf{Q}}[C_t]$ also appear to fit the data well

# 3.1. Bare-bones Monte Carlo (BBMC) (5)

**Putting $S_T$ and $C_t$ together:**




- We observe parallel convergence (equivalent error scaling) of $\langle S_T \rangle_n$ and $\langle C_t \rangle_n$
- Looking at ratios in errors of $\langle S_T \rangle_n$ and $\langle C_t \rangle_n$, it appears that the two might be correlated beyond mere scaling with $n$
  - This suggests that if we can reduce sampling error for $S_T$, we might improve estimation of $C_t$

# 3.1. Bare-bones Monte Carlo (BBMC) (6)

## Efficiency Metrics

- We've seen that the variance of a Monte Carlo estimator typically scales as $1/n$ (with $n$ the number of samples/realizations/scenarios), and therefore that the standard deviation of the estimator scales as $1/\sqrt{n}$.
- Since execution time $\tau$ usually scales as $n$, we are led to define an efficiency metric (ratio) $\mathcal{E}$ for comparing one method to another (or even one choice of $n$ to another):

$$\mathcal{E}_{2,1} \doteq \frac{\sigma_1^2 \tau_1}{\sigma_2^2 \tau_2}$$

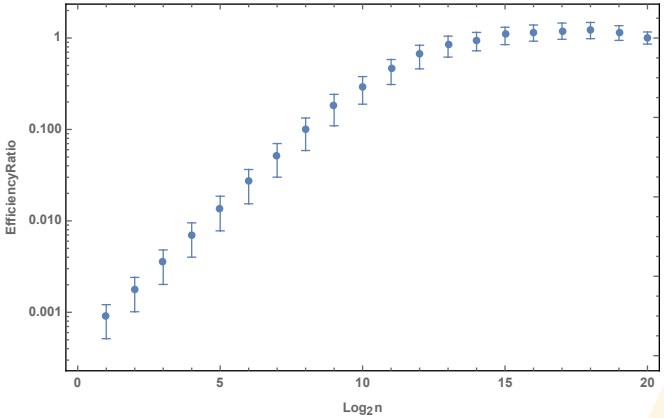
where $\sigma_j^2$ is the variance of the estimator using method $j$ and $\tau_j$ is the corresponding execution time.

  - This should be read as "the efficiency ratio of method 2 relative to method 1."
  - In designing a variance reduction technique, we can make use of assumptions/estimates regarding the additional (proportional) run time required to execute the (more complex) algorithm and the corresponding reduction in variance to estimate $\mathcal{E}$.
  - Empirically, we can use the timing functions built into a programming environment to calculate total execution times and evaluate $\mathcal{E}$.
    - Meaningful results should only be expected within asymptotic scaling regimes for time and variance...

# 3.1. Bare-bones Monte Carlo (BBMC) (7)

## Efficiency Metrics (continued): Example

- Empirical efficiency ratio (with 95% conf. range) for various $n$ relative to $n = 2^{20}$ (run 1000 times each – in randomized order – to reduce idiosyncratic timing effects)




- Even for $n$ of order 1000 or more, timing function and system idiosyncrasies can contaminate or mask asymptotic behavior.

# 3.2. Improvement 1: Antithetic variates (AMC)

- For each of the $n$ independent scenarios, generate an additional perfectly anti-correlated scenario:

$$S_{T,i} = S_t \exp\left[(r-y-\sigma^2/2)(T-t) + \sigma\sqrt{T-t}\, z_i\right], C_{T,i} = \max[S_{T,i} - K, 0]$$
$$S^a_{T,i} = S_t \exp\left[(r-y-\sigma^2/2)(T-t) - \sigma\sqrt{T-t}\, z_i\right], C^a_{T,i} = \max[S^a_{T,i} - K, 0]$$

- Average over both scenarios to obtain an estimator $\langle C_T \rangle_{a,n}$ of $\mathbb{E}^{\mathbf{Q}}[C_T]$:

$$\mathbb{E}^{\mathbf{Q}}[C_T] \simeq \langle C_T \rangle_{a,n} \doteq \frac{1}{n}\sum_i \frac{(C_{T,i+} + C_{T,i-})}{2} = \frac{1}{n}\sum_i \frac{(C_{T,i} + C^a_{T,i})}{2}$$

- Convergence properties of this estimator:

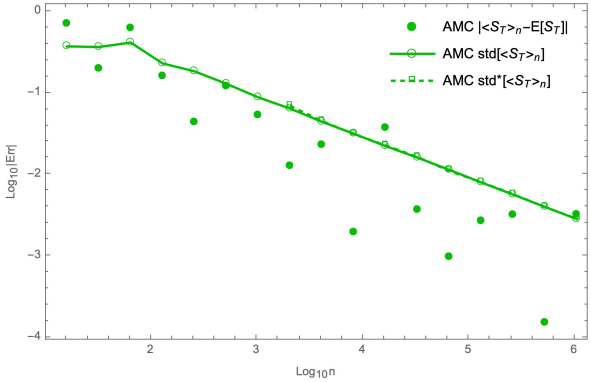
$$\mathrm{var}[\,\langle C_T \rangle_{a,n}\,] \simeq \frac{1}{4n}\mathrm{var}[C_{T,i} + C^a_{T,i}] \simeq \frac{1}{n}\frac{1+\rho(C_{T,i}, C^a_{T,i})}{2}\,\mathrm{var}^{\mathbf{Q}}[C_T]$$

- If the correlation is strongly negative, we can substantially increase efficiency.
  - Worst (?) case scenario: $\rho = 0$ and execution time doubles
    - Is there a worse "worst case" scenario?
  - We can also use the reduction in sampled variance to estimate the correlation $\rho(C_{T,i}, C^a_{T,i})$

# 3.2. Improvement 1: Antithetic variates (AMC) (2)

**Examine convergence properties vs $n$, starting with $\mathbb{E}^{\mathbb{Q}}[S_T]$:**

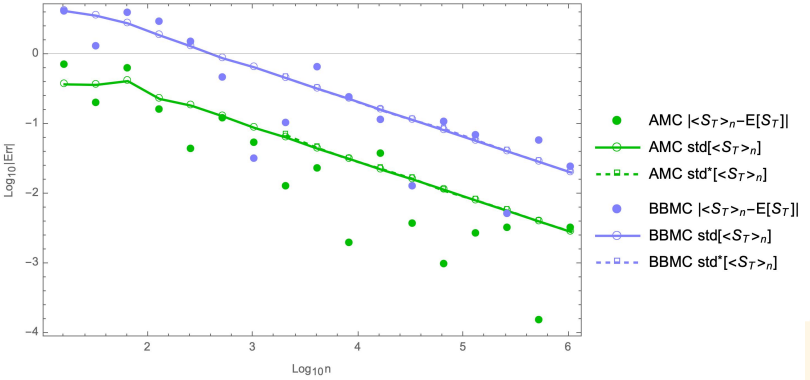
- Use parameters as before




- CLT and binned error estimators suggest $\mathcal{O}\left(n^{-1/2}\right)$ convergence.

# 3.2. Improvement 1: Antithetic variates (AMC) (3)

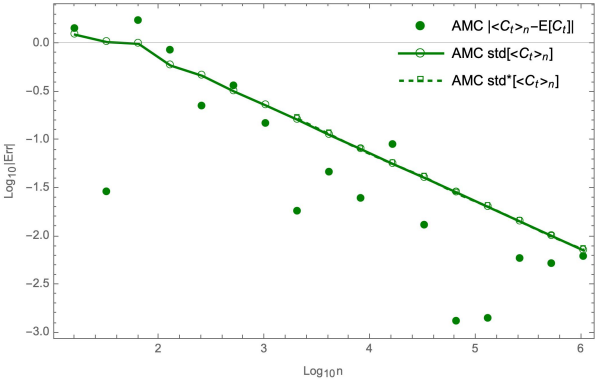
**Compare $\mathbb{E}^{\mathbf{Q}}[S_T]$ results for AMC vs. BBMC:**




- Results appear almost an order of magnitude (std. error ratio ca. 14%) better.
- Efficiency ratio for $\mathbb{E}^{\mathbf{Q}}[S_T]$, assuming $2\times$ execution time: $1/(2 \cdot 0.14^2) \approx 25$!
  - Empirically, $\tau_{AMC} \simeq 1.6 \times \tau_{BBMC}$
- Implied $\rho$: $(1+\rho)/2 = 0.14^2 \Rightarrow \rho = 2 \cdot 0.14^2 - 1 \approx -0.96$

# 3.2. Improvement 1: Antithetic variates (AMC) (4)

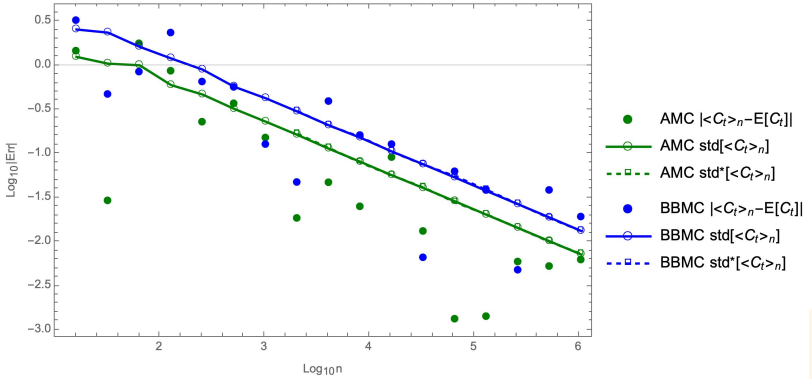
**Next, examine $C_t$:**




- Convergence properties are similar to those for $\mathbb{E}^{\mathbf{Q}}[S_T]$

# 3.2. Improvement 1: Antithetic variates (AMC) (5)

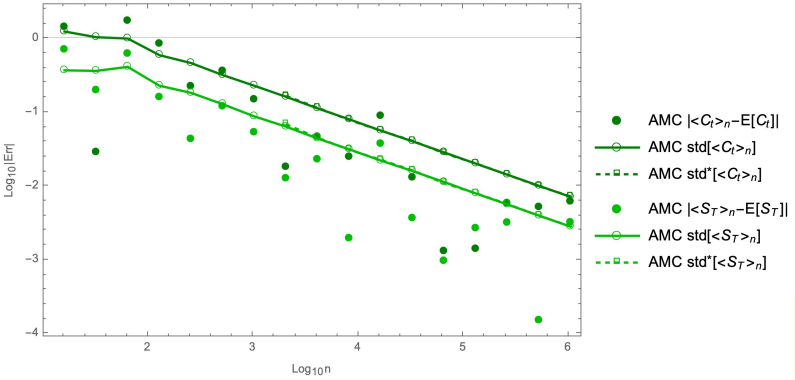
**Compare $C_t$ results for AMC vs. BBMC:**




- Improvement apparent (std. error ratio ca. 54%), but not nearly as good as for $\mathbb{E}^{\mathbf{Q}}[S_T]$
- Efficiency ratio for $\langle C_t \rangle_n$, assuming $2\times$ execution time: $1/(2 \cdot 0.54^2) \approx 1.7$.
- Implied $\rho$: $(1+\rho)/2 = 0.54^2 \Rightarrow \rho = 2 \cdot 0.54^2 - 1 \approx -0.42$

# 3.2. Improvement 1: Antithetic variates (AMC) (6)

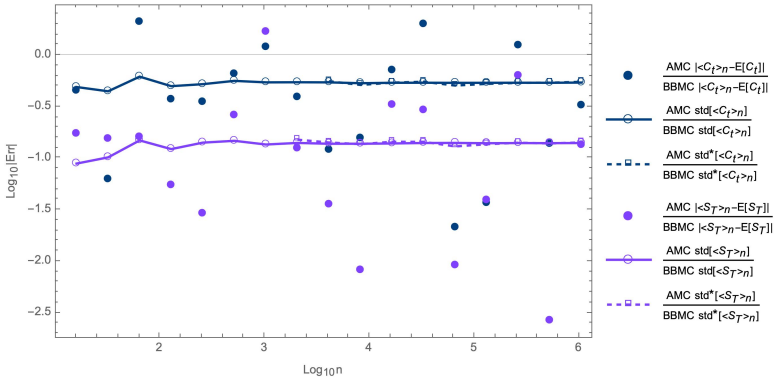
**Putting $S_T$ and $C_t$ together:**




- Just as for BBMC, we observe parallel convergence (equivalent error scaling) of $\langle S_T \rangle_n$ and $\langle C_t \rangle_n$
- In contrast to BBMC, absolute errors in $\langle C_t \rangle_n$ usually seem larger than those in $\langle S_T \rangle_n$

# 3.2. Improvement 1: Antithetic variates (AMC) (7)

**Examining the standard error ratios vs. Bare-Bones MC:**



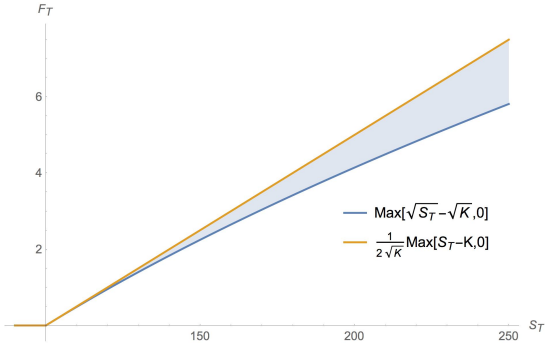

- Why is improvement in convergence of $\langle S_T \rangle_n$ so much better than that in $\langle C_t \rangle_n$?
  - Symmetric vs. asymmetric payoffs has much to do with it, but there's a bit more to the story than that: think about the moments of $\ln(S_T) + \ln(S_T^a)$...

# 3.3. Improvement 2: Control variates (CVMC)

- For each of the $n$ independent paths, calculate the payoff of an option $C_T^*$.
  $C_T^*$ should be both highly correlated with $C_T$ and have a reliably known expected value $\mathbb{E}^{\mathbf{Q}}[C_T^*]$.
- Average the difference $C_T - C_T^*$ over scenarios to obtain an estimator $\langle C_T \rangle_{cv,n}$ of $\mathbb{E}^{\mathbf{Q}}[C_T]$:
$$\mathbb{E}^{\mathbf{Q}}[C_T] \simeq \langle C_T \rangle_{cv,n} \doteq \mathbb{E}^{\mathbf{Q}}[C_T^*] + \frac{1}{n}\sum_i \left(C_{T,i} - C_{T,i}^*\right)$$
- Convergence properties of this estimator:
$$\mathrm{var}[\,\langle C_T \rangle_{cv,n}\,] \simeq \frac{1}{n}\mathrm{var}[C_{T,i} - C_{T,i}^*] \simeq \frac{1}{n}(\mathrm{var}[C_T] + \mathrm{var}[C_T^*] - 2\,\mathrm{cov}[C_T, C_T^*])$$
  - If $\mathrm{var}[C_T^*] \simeq \mathrm{var}[C_T]$: $\quad \mathrm{var}[\,\langle C_T \rangle_{cv,n}\,] \simeq \frac{2}{n}(1 - \rho[C_T, C_T^*])\,\mathrm{var}[C_T]$
- If the covariance is strongly positive, we can substantially increase efficiency.
  - We need $\rho \gtrsim \frac{1}{2}$ to avoid *increasing* variance.
  - Assuming that adding a control variate roughly doubles computation time, we need $\rho \gtrsim \frac{3}{4}$ to increase efficiency.
  - Choosing a control variate to maximize the covariance involves (almost? at least?) as much art as science.

# 3.3. Improvement 2: Control variates (CVMC) (2)

- Example: the Square-Root Option we considered in Part 3 of the notes, with $F_T = [S_T^{1/2} - K^{1/2}]^+$:




- One possibility: take parameters as before: $S = K = 100, T-t = 1, r = 0.04, y = 0.02, \sigma = 0.2$, so that pre-washing:

$$\begin{aligned} S_{eff} = K_{eff} &= \sqrt{100} = 10, \\ T-t &= 1, \\ r_{eff} &= r = 0.04, \\ y_{eff} &= 0.5(r + y + 0.5\sigma^2/2) = 0.5(0.04 + 0.02 + 0.5(0.2)^2/2) = 0.035, \\ \sigma_{eff} &= 0.5\sigma = 0.1. \end{aligned}$$

# 3.3. Improvement 2: Control variates (CVMC) (3)

- But our results won't be directly comparable to our BBMC and AMC for vanilla options, so we'd need to re-generate everything.
- We'd like to use our previous BBMC results and THEN assess the impact of a control variate.
- So, a bit deviously, choose ***effective*** parameters to match the vanilla option inputs we've been using:
  $S_{eff} = K_{eff} = 100, T-t = 1, r_{eff} = 0.04, y_{eff} = 0.02, \sigma_{eff} = 0.2,$
- Then, reverse engineer the real-world inputs that generate the desired effective parameters:
  $S = K = 100^2 = 10^4, T-t = 1, r = r_{eff} = 0.04,$
  $\sigma = \sigma_{eff}/0.5 = 0.4,$
  $y_{eff} = 0.02 = 0.5(r + y + 0.5\sigma^2/2) \Rightarrow y = 2 \cdot 0.02 - 0.04 - (0.4)^2/4 = -0.04.$
- In either case, closed-form valuation is as simple as:

```
SqrtC[S_, K_, T_, r_, y_, σ_] :=
  BSC[Sqrt[S], Sqrt[K], 1, r, 0.5 (r + y + 0.25 σ^2), 0.5 σ];
```

$$\mathbb{E}^{\mathbf{Q}}[\sqrt{S_T}] = \sqrt{S} e^{(r-y_{eff})(T-t)} = 100\, e^{(0.04-0.02)(1)} = 100\, e^{0.02} = 102.020$$

$$F_t = BSC(S_{eff}, K_{eff}, T-t, r, y_{eff}, \sigma_{eff}) = BSC(100, 100, 1, 0.04, 0.02, 0.2) = 8.73949$$

- If we wish, we can also re-use (most of) our bare-bones, antithetic, etc. codes and results:

```
SqrtCBBMCPreWash[S_, K_, T_, r_, y_, σ_] :=
  BSCBBMC[Sqrt[S], Sqrt[K], 1, r, 0.5 (r + y + 0.25 σ^2), 0.5 σ];
```

# 3.3. Improvement 2: Control variates (CVMC) (4)

- It's actually even better than that! Consider path-wise contribution to square-root option value:

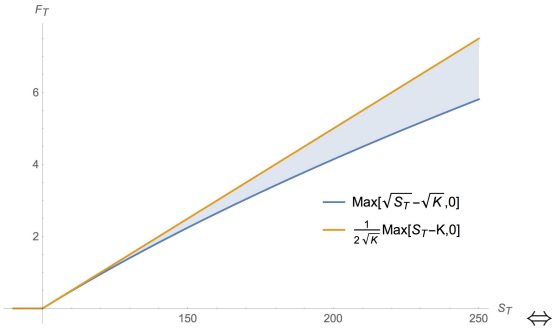
$$F_{t,i} = e^{-r(T-t)}\left[\sqrt{S}e^{\frac{1}{2}[(r-y-\sigma^2/2)(T-t)+\sigma\sqrt{T-t}\,z_i]} - \sqrt{K}\right]^+ = e^{-r(T-t)}\left[S_{eff}e^{(r-y_{eff}-\sigma_{eff}^2/2)(T-t)+\sigma_{eff}\sqrt{T-t}\,z_i} - K_{eff}\right]^+$$

- What would be a good choice of control variate? Match slopes at kink point:

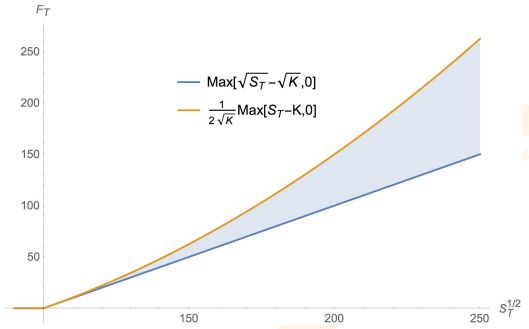
$$\frac{1}{2\sqrt{K}}BSC(S,K,T-t,r,y,\sigma) = \frac{1}{200}BSC(10^4,10^4,1,0.04,-0.04,0.4) = 10.0888$$

  ...about 15% too high, but otherwise a good match overall...

- This will actually serve as a strenuous test compared to the way we would ordinarily implement MC (and CV) for this payoff (why?)

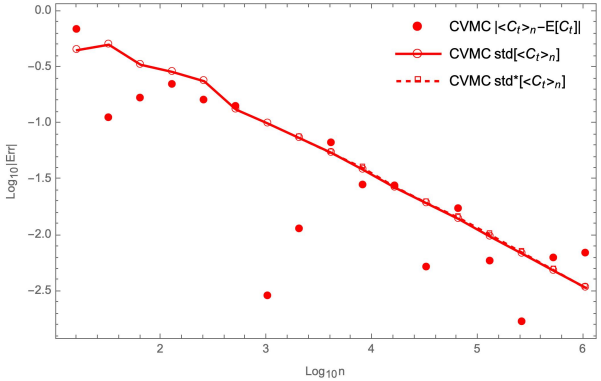



$\Leftrightarrow$



# 3.3. Improvement 2: Control variates (CVMC) (5)

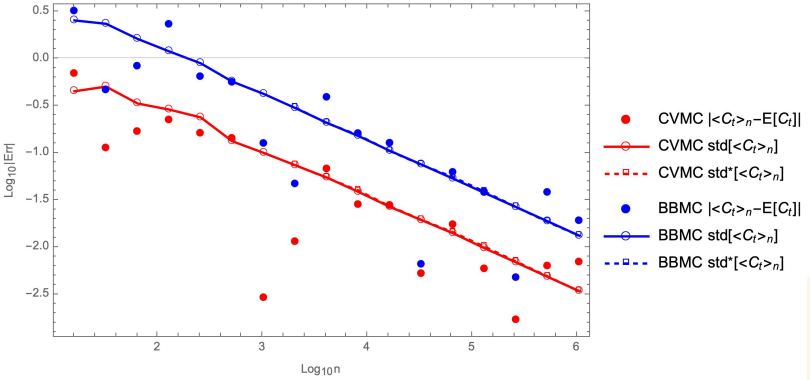
**Examine $C_t$:**




- Convergence in line with CLT

# 3.3. Improvement 2: Control variates (CVMC) (6)

**Compare $C_t$ results for CVMC vs. BBMC:**

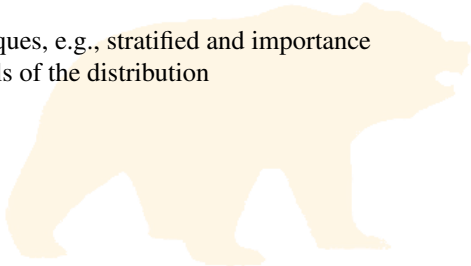



- Errors reduced by about 74% (Efficiency estimate ca. $1/(2 \cdot 0.26^2) \simeq 7.5) \Rightarrow \rho$ ca. 97%.

# 3.3. Improvement 2: Control variates (CVMC) (7)

**A few last comments on variance reduction**

- We can usually combine/mix/match any of the methods we've described.
  - This almost always helps, but variance reduction may not always be fully compounded.
- It's also possible to introduce multiple control variates and optimize their correlation with the desired payoff.
  - ...but remember the caveat that it is sometimes possible to introduce sufficient multi-collinearity that the estimation problem becomes ill conditioned.
- There are also a number of other variance reduction techniques, e.g., stratified and importance sampling, that can be very helpful in sampling from the tails of the distribution



# 3.4. Appendix: Improvement 3: Moment matching (MMMC)

- Adjust $\{z_i\} \to \{z_i'\}$: $z_i' = \alpha + \beta z_i \; \forall i$, to match moments: {either mean or log-mean} and {either variance or log-variance} of $S_T$:

$$S_{T,i} = S_t e^{\tilde{\mu}^{\mathbf{Q}} dt + (\sigma\sqrt{dt}) z_i} \;\to\; S'_{T,i} = S_t e^{\tilde{\mu}^{\mathbf{Q}} dt + (\sigma\sqrt{dt}) z'_i} \text{ with } \tilde{\mu}^{\mathbf{Q}} = r - y - \sigma^2/2, dt = T - t$$

1. First moment:
   1. Log mean: Matching $\left\langle \ln(S'_{T,i}/S_t) \right\rangle_n$ to $\mathbb{E}^{\mathbf{Q}}[\ln(S_T/S_t)]$:

$$\left\langle \tilde{\mu}^{\mathbf{Q}} dt + (\sigma\sqrt{dt}) z'_i \right\rangle_n = \tilde{\mu}^{\mathbf{Q}} dt \;\Rightarrow\; \left\langle (\sigma\sqrt{dt}) z'_i \right\rangle_n = 0 \;\Rightarrow\; \left\langle z'_i \right\rangle_n = 0 \;\Rightarrow\; \alpha + \beta \langle z_i \rangle_n = 0 \;\Rightarrow\; \alpha = -\beta \langle z_i \rangle_n$$

   If we make no variance adjustment, then $\beta = 1$ and $z'_i = z_i - \langle z_i \rangle_n$

   2. "Exponential" mean: Matching $\left\langle S'_{T,i}/S_t \right\rangle_n$ to $\mathbb{E}^{\mathbf{Q}}[S_T/S_t]$:

$$\begin{aligned}
\left\langle e^{\tilde{\mu}^{\mathbf{Q}} dt + (\sigma\sqrt{dt}) z'_i} \right\rangle_n &= e^{(\tilde{\mu}^{\mathbf{Q}} + \sigma^2/2) dt} = e^{\mu^{\mathbf{Q}} dt} \text{ with } \mu^{\mathbf{Q}} = r - y \\
\left\langle e^{\tilde{\mu}^{\mathbf{Q}} dt + (\sigma\sqrt{dt})(\alpha + \beta z_i)} \right\rangle_n &= e^{\tilde{\mu}^{\mathbf{Q}} dt + \alpha\sigma\sqrt{dt}} \left\langle e^{\beta(\sigma\sqrt{dt}) z_i} \right\rangle_n = e^{\mu^{\mathbf{Q}} dt} \\
\Rightarrow\; e^{\alpha\sigma\sqrt{dt}} \left\langle e^{\beta(\sigma\sqrt{dt}) z_i} \right\rangle_n &= e^{(\mu^{\mathbf{Q}} - \tilde{\mu}^{\mathbf{Q}}) dt} = e^{\sigma^2 dt/2} \;\Rightarrow\; \alpha = \frac{\sigma\sqrt{dt}}{2} - \frac{\ln\left\langle e^{\beta(\sigma\sqrt{dt}) z_i} \right\rangle_n}{\sigma\sqrt{dt}}
\end{aligned}$$

   With no variance adjustment, $\beta = 1$ so $\alpha$ can effectively be calculated by multiplying $S_{T,i}$ by the ratio $\mathbb{E}^{\mathbf{Q}}[S_T] / \langle S_{T,i} \rangle_n$

   If we fit the log variance, then $\beta$ is determined independent of $\alpha$, so we can effectively find $\alpha$ by first calculating $S'_{T,i}$ with $\beta$ as found below and $\alpha = 0$, then multiplying $S'_{T,i}$ by the ratio $\mathbb{E}^{\mathbf{Q}}[S_T] / \left\langle S'_{T,i} \right\rangle_n$.

   Fitting the exponential variance: see below

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (2)

- Moment equations (continued):
  2. Second moment:
     1. Log variance: Matching $\text{var}[\ln(S'_{T,i}/S_t)]_n$ to $\text{var}[\ln(S_T/S_t)]$:

$$\begin{aligned}&\text{var}[\tilde{\mu}^{\mathbf{Q}}dt + (\sigma\sqrt{dt})z'_i]_n = \sigma^2 dt\\ &\Rightarrow\ \sigma^2 dt\ \text{var}[z'_i]_n = \sigma^2 dt\ \Rightarrow\ \text{var}[z'_i]_n = 1\ \Rightarrow \text{var}[\alpha+\beta z_i]_n = 1\\ &\Rightarrow\ \beta^2\ \text{var}[z_i]_n = 1\ \Rightarrow\ \beta = 1/\sqrt{\text{var}[z_i]_n}\end{aligned}$$

     2. "Exponential" variance: Matching $\text{var}[(S'_{T,i}/S_t)]_n$ to $\text{var}[(S_T/S_t)]$ or $\left\langle (S'_{T,i}/S_t)^2 \right\rangle_n$ to $\mathbb{E}^{\mathbf{Q}}[(S_T/S_t)^n]$:
        Assume for illustration that we're matching the $2^{nd}$ moment (expectation).
        This is identical to matching variance if we've also matched the "exponential" mean.

$$\begin{aligned}\left\langle e^{2\tilde{\mu}^{\mathbf{Q}}dt+2(\sigma\sqrt{dt})z'_i}\right\rangle_n &= e^{(2\tilde{\mu}^{\mathbf{Q}}+2\sigma^2)dt} = e^{(2\mu^{\mathbf{Q}}+\sigma^2)dt} \text{ with } \mu^{\mathbf{Q}} = r-y\\ \left\langle e^{2\tilde{\mu}^{\mathbf{Q}}dt+(2\sigma\sqrt{dt})(\alpha+\beta z_i)}\right\rangle_n &= e^{2\tilde{\mu}^{\mathbf{Q}}dt+2\alpha\sigma\sqrt{dt}}\left\langle e^{2\beta(\sigma\sqrt{dt})z_i}\right\rangle_n = e^{(2\mu^{\mathbf{Q}}+\sigma^2)dt}\\ \Rightarrow\ e^{2\alpha\sigma\sqrt{dt}}\left\langle e^{2\beta(\sigma\sqrt{dt})z_i}\right\rangle_n &= e^{(2\mu^{\mathbf{Q}}+\sigma^2-2\tilde{\mu}^{\mathbf{Q}})dt} = e^{2\sigma^2 dt}\ \Rightarrow\ \alpha = \sigma\sqrt{dt} - \frac{\ln\left\langle e^{2\beta(\sigma\sqrt{dt})z_i}\right\rangle_n}{2\sigma\sqrt{dt}}\end{aligned}$$

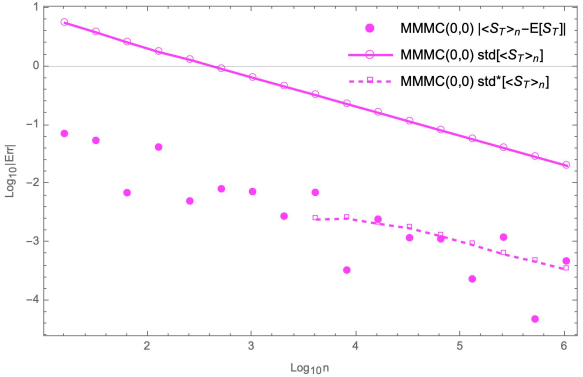
        We must combine this relation between $\alpha$ and $\beta$ with one of the formulas for $\alpha$ in terms of $\beta$ from the first moment conditions on the previous slide, thereby eliminating $\alpha$. E.g.:

$$\sigma\sqrt{dt} - \frac{\ln\left\langle e^{2\beta(\sigma\sqrt{dt})z_i}\right\rangle_n}{2\sigma\sqrt{dt}} = \frac{\sigma\sqrt{dt}}{2} - \frac{\ln\left\langle e^{\beta(\sigma\sqrt{dt})z_i}\right\rangle_n}{\sigma\sqrt{dt}} \Rightarrow \left\langle e^{2\beta(\sigma\sqrt{dt})z_i}\right\rangle_n = e^{\sigma^2 dt}\left\langle e^{\beta(\sigma\sqrt{dt})z_i}\right\rangle_n^2$$

        The resulting non-linear equation in $\beta$ must be solved numerically using, e.g., Newton's method.

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (3)

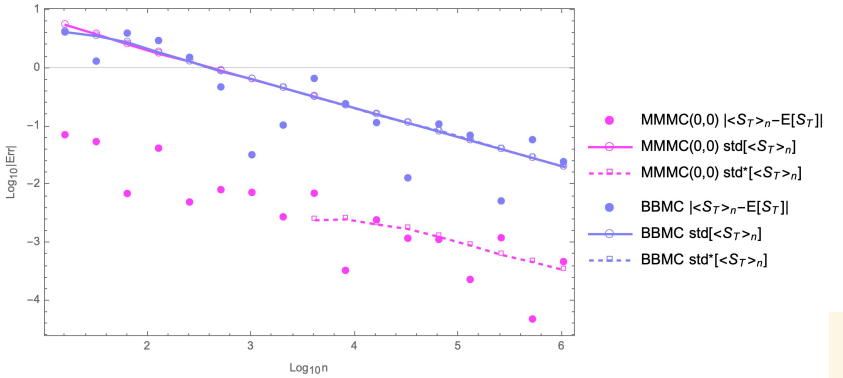
- Match log-mean and log-variance: options to MMMC code shown as $(0, 0)$.
- Use parameters as before; examine convergence properties vs $n$, starting with $\mathbb{E}^{\mathbf{Q}}[S_T]$:




- Data suggest convergence more than an order of magnitude faster than CLT implies (but consistent with binned error estimator, which now diverges from CLT error)
- Note that binning doesn't preserve means exactly for these methods, since each bin must be moment matched. Binned error estimates are still valid asymptotically.

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (4)

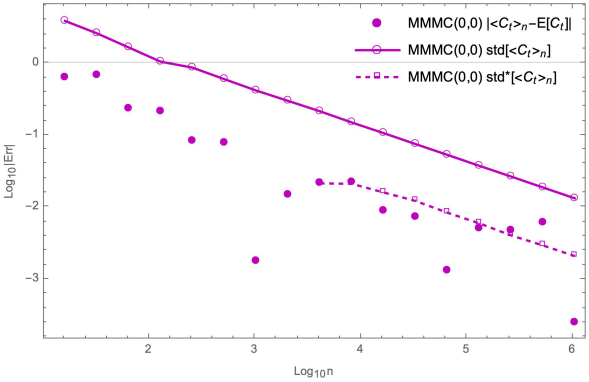
**Compare $\mathbb{E}^{\mathbf{Q}}[S_T]$ results for MMMC vs. BBMC:**




- CLT error estimators are essentially the same for MMMC and BBMC: consistent with BB data but inconsistent with MM.
- MM binned error estimates lie close to the MM data (parallel to, but below the CLT errors), implying a reduction in the coefficient multiplying the $n^{-1/2}$ error scaling.

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (5)

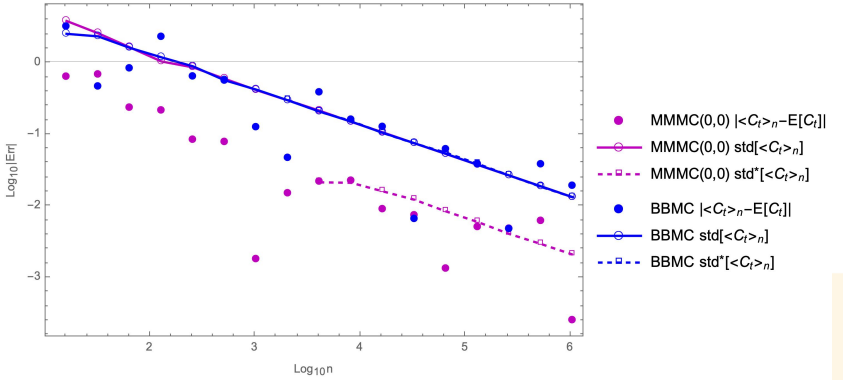
**Next, examine $C_t$:**




- Similar (substantially improved) results to $S_T$, albeit not quite as drastic
- Data are again consistent with binned error estimator, but diverge from CLT error

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (6)

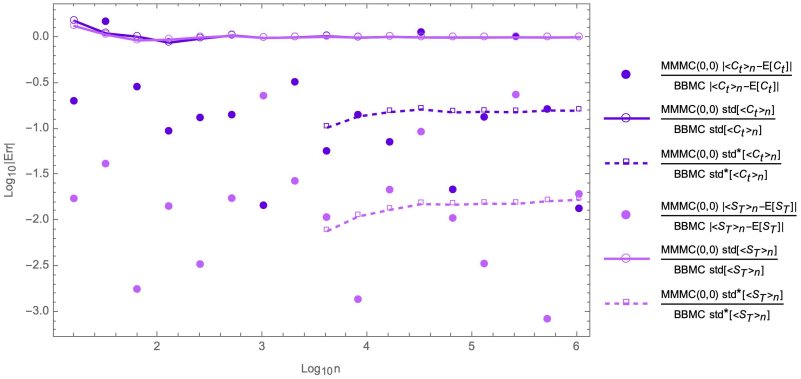
**Compare $C_t$ results for MMMC vs. BBMC:**




- Still about an order of magnitude better!

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (7)

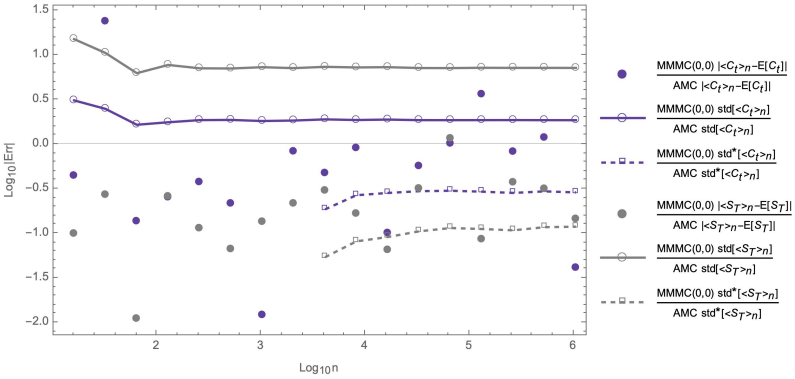
**Examining the standard error ratios vs. Bare-Bones MC:**




- About two orders of magnitude improvement (error ratio ca. 1.6%) in $S_T$;
- About one order of magnitude improvement (error ratio ca. 16%) in $C_t$
- Assuming (conservatively) $2\times$ execution time, efficiency ratios are about 1800 for $\langle S_T \rangle_n$ and 20 for $\langle C_t \rangle_n$, respectively.
  - For moment matching in log space, actual execution time is ca. $1.15-1.20\times$ that of BBMC.

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (8)

**Examining the standard error ratios vs. Antithetic MC:**




- MMMC does globally (across all samples) what AMC does locally (sample-by-sample), but does it "twice as well" (by matching both $1^{st}$ & $2^{nd}$ log-moments).
- Binned error ratios are ca. 12% for $\langle S_T \rangle_n$ and ca. 29% for $\langle C_t \rangle_n$.
  Assuming equal execution times, efficiency ratios are about 72 and 12, respectively.

# 3.4. Appendix: Improvement 3: Moment matching (MMMC) (9)

## A few final thoughts on moment matching

- Some authors (e.g. Jäckel) issue multiple health warnings regarding application of moment matching in path-dependent and multi-asset settings.
- Until fairly recently, I didn't see much point to these warnings...
- My sense on this remains that it is always possible to transform sets of normal samples so that the desired population (log-) statistics for each individual set are matched, and then to combine them so that the desired population correlations are also achieved.
  - The sticky point is the phrase "desired population."
  - Especially for sub-batched/binned estimators there is some question about whether second moments should be corrected using the population or sample forms of variance/covariance calculations.
  - The procedure should be "safe" in a multi-asset setting and probably so for time-dependent applications as well (with some care as to exactly which population statistics we're trying to match).
  - Perhaps the one further generic reminder that should be mentioned in the context of combining variance reduction techniques, particularly the control variate methods we've discussed, is that it is sometimes possible to introduce sufficient multi-collinearity that the estimation problem becomes ill conditioned.
- I do agree that moment matching shouldn't be combined with quasi-random sequences (e.g. Sobol, Halton, ...) where moment properties are in some sense already built-in.

# 4. Monte Carlo Sensitivities

**Three basic methodologies**

- Finite differencing (discrete “bumping")
- Analytical differencing ($\epsilon \searrow 0$ limit of discrete “bumping")
- Likelihood ratio or Radon-Nikodym method

# 4.1. Finite Differencing

- Typically done "outside" the valuation algorithm
  - Easier than programming sensitivities for each model; once you have a value you can bump
  - Ensure consistency across models, inputs
  - Don't have to worry about idiosyncrasies of inputs (in theory, at least)
- Practically, two kinds of bumps:
  - Relative: usually for asset prices [homogeneous of degree 1] $\Rightarrow$ $\Delta, \Gamma$
    - e.g., $S \to S(1 \pm \epsilon)$, with $\epsilon$ typically, say, 1%.
    - In raw form, sensitivities answer questions like: "How much will my delta change for a 1% move in spot?"
  - Absolute: usually for times, rates, yields, vols [homogeneous of degree 0] $\Rightarrow$ $\theta, \rho_r, \rho_y, \upsilon$
    - e.g., $r \to r \pm \epsilon$, with $\epsilon$ typically 1bp for rates and yields.
    - Vol bumps are typically $\sim 1\%$ (in an ideal world, we'd bump variance *rate*).
    - Time bumps:

      $\theta$ (moving today $t$ forward) not necessarily the same as $\theta'$ (moving maturity $T$ backward)

      Typical bump size = 1 day, but note difference between trading, calendar, interest accrual days
      (most firms have their own idiosyncratic treatments of these and code/methods in place to handle them).
- Pet peeve: one-sided bumps

# 4.1. Finite Differencing (2)

## Forward, Backward, and Central Differences

- Calculation of deltas and gammas, using a proportional bump size $\delta S = \epsilon \cdot S$.
  If one defines (holding all inputs other than $S_t$ constant):

$$\begin{aligned} S_0 &\doteq S & , \quad C_0 &\doteq C(S_t = S_0) \\ S_+ &\doteq S + \delta S = (1+\epsilon)S, & C_+ &\doteq C(S_t = S_+) \\ S_- &\doteq S - \delta S = (1-\epsilon)S, & C_- &\doteq C(S_t = S_-) \end{aligned}$$

Then one can define forward ($\Delta_+$), backward ($\Delta_-$), and central ($\Delta_c$) deltas:

$$\begin{aligned} \Delta_+ &\doteq \frac{C_+ - C_0}{S_+ - S_0} = \frac{C_+ - C_0}{\epsilon \cdot S} \\ \Delta_- &\doteq \frac{C_0 - C_-}{S - S_-} = \frac{C_0 - C_-}{\epsilon \cdot S} \\ \Delta_c &\doteq \frac{C_+ - C_-}{S_+ - S_-} = \frac{C_+ - C_-}{2 \cdot \epsilon \cdot S} \end{aligned}$$

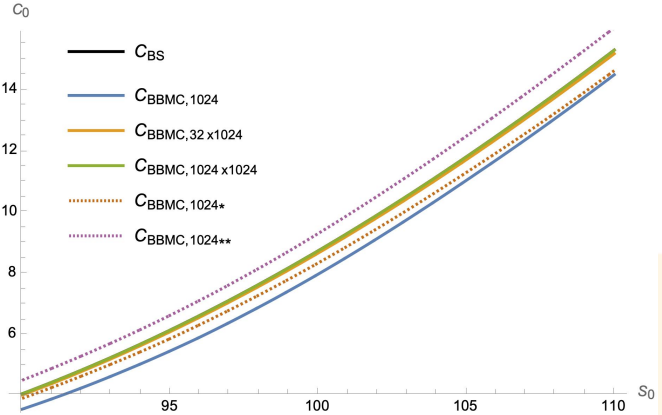
as well as the central gamma ($\Gamma_c$):

$$\Gamma_c \doteq \frac{\Delta_+ - \Delta_-}{\delta S} = \frac{(C_+ - C_0) - (C_0 - C_-)}{(\delta S)^2} = \frac{C_+ + C_- - 2C_0}{(\epsilon \cdot S)^2}$$

# 4.1. Finite Differencing (3)

## Set-up: Closed-form & MC Values vs. $S_0$

- Example parameters: $S_0 = K = 100, T = 1, r = 4\%, y = 2\%, \sigma = 0.2$
- Compare BSM call values vs. $S_0$ to BBMC results for $n = 1024, 32\times1024, 1024\times1024$

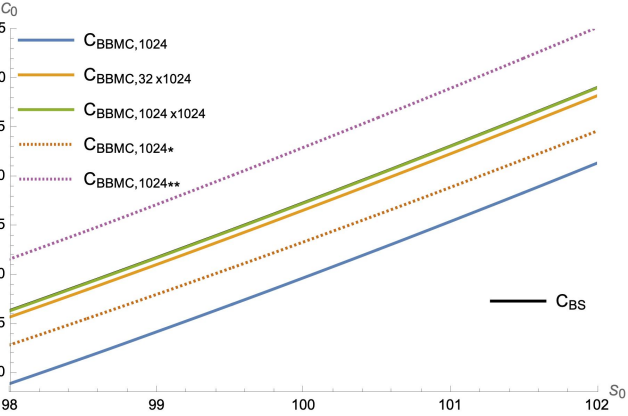



- Simulation error appears as bias locally; error appears to be reducing steadily with $n$.
- Values appear smooth even for small-ish $n$.

# 4.1. Finite Differencing (4)

## Set-up: Closed-form & MC Values vs. $S_0$, continued

- Drill down to $98 \leq S_0 \leq 102$ for $n = 1024, 32{\times}1024, 1024{\times}1024$

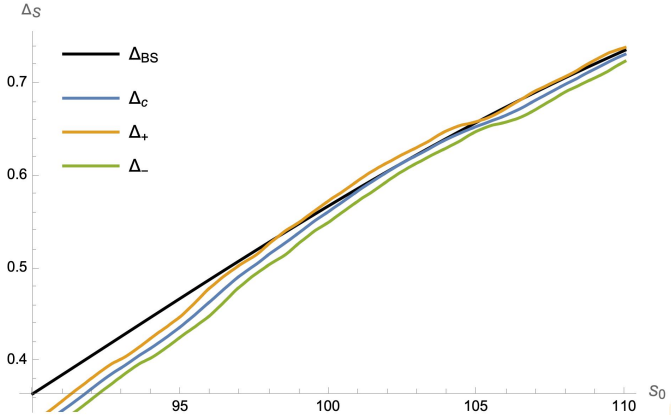



- Values remain smooth even at this scale

# 4.1. Finite Differencing (5)

## Closed-form & MC Delta vs. $S_0$

- Plot $\boldsymbol{\Delta_S}$ vs. $S$ for n = 1024, bump size $\epsilon = 0.01$, $S_0$ points spaced 0.01:

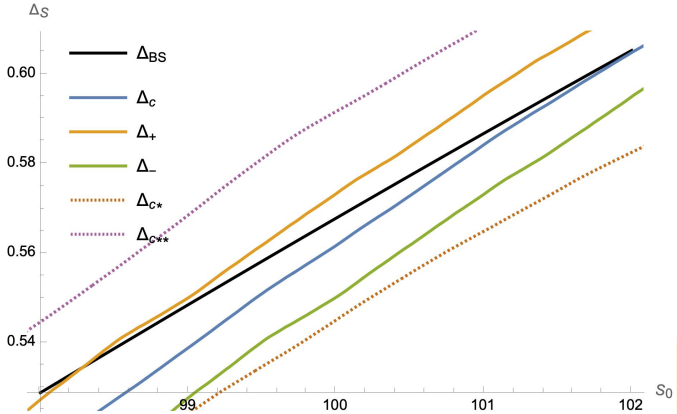



- $\Delta_+, \Delta_-$ bound $\Delta_c$
- Small $n$ effects (both overall error and bumpiness/discretization error) are apparent

# 4.1. Finite Differencing (6)

## Closed-form & MC Delta vs. $S_0$, continued

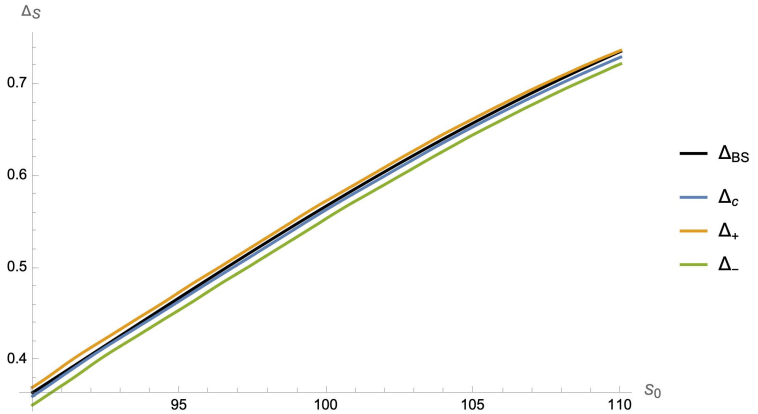
- Drill into $[98, 102]$ domain ($\epsilon = 0.01$, $S_0$ points spaced 0.01):



# 4.1. Finite Differencing (7)

## Closed-form & MC Delta vs. $S_0$, continued

- Increase n to $32 \times 1024$ ($\epsilon = 0.01$, $S_0$ points spaced 0.01):

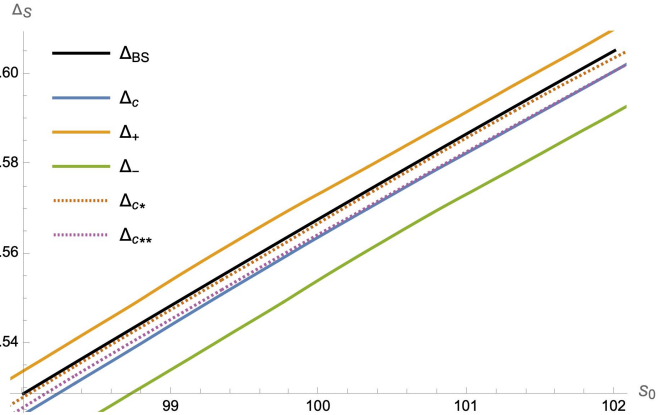



- Convergence toward theoretical value is apparent
- Discretization effects are much smaller

# 4.1. Finite Differencing (8)

## Closed-form & MC Delta vs. $S_0$, continued

- Drill into $[98, 102]$ domain for n $= 32 \times 1024$ ($\epsilon = 0.01$, $S_0$ points spaced 0.01):

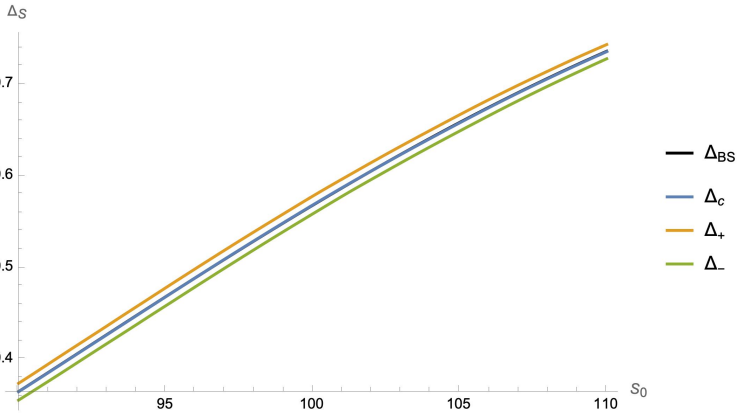



- Bias error in $\Delta_+, \Delta_-$ now dominates sample error
- Discretization effects are not apparent in this view

# 4.1. Finite Differencing (9)

## Closed-form & MC Delta vs. $S_0$, continued

- Increase n further to $1024 \times 1024$ ($\epsilon = 0.01$, $S_0$ points spaced 0.1):

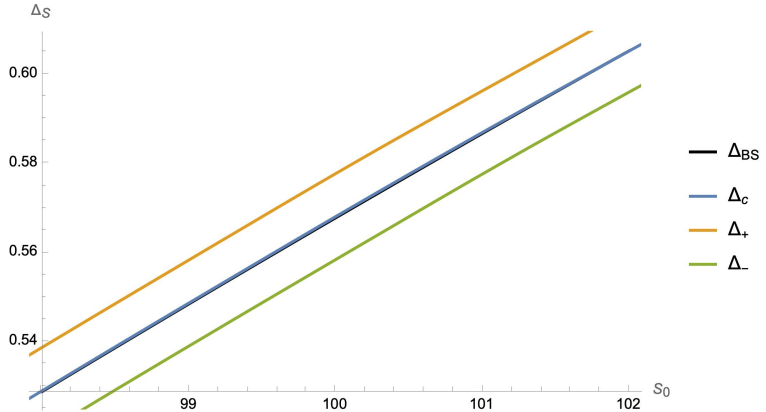



- Centered difference converges to theoretical Delta
- Bias in up, down Deltas apparent

# 4.1. Finite Differencing (10)

## Closed-form & MC Delta vs. $S_0$, continued

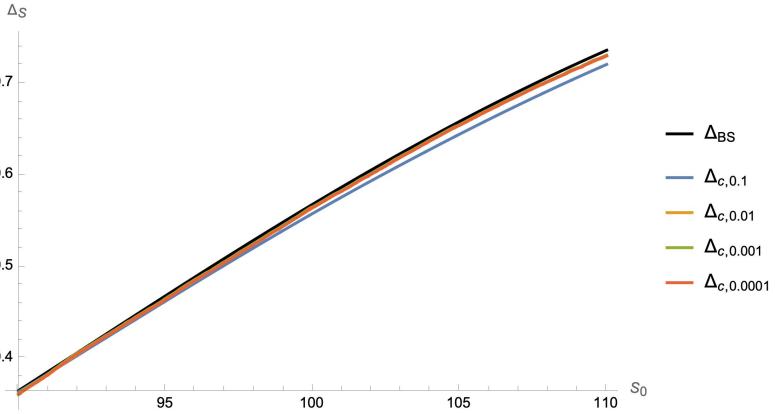
- Drill into $[98, 102]$ domain for n $= 1024 \times 1024$ ($\epsilon = 0.01$, $S_0$ points spaced 0.1):



# 4.1. Finite Differencing (11)

## Closed-form & MC Delta vs. $S_0$ and bump size

- Consider $\Delta_c$ for n fixed at $32 \times 1024$; vary bump sizes $\epsilon = 0.1, 0.01, 0.001, 0.0001$: ($S_0$ points spaced 0.01)

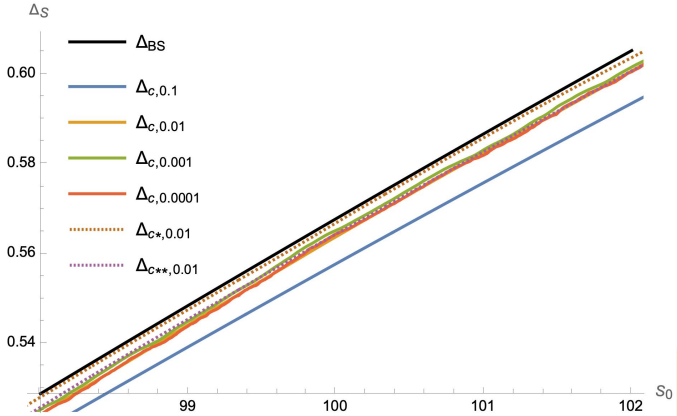



- Bias effects apparent for larger bumps; noise increases as bumps get smaller

# 4.1. Finite Differencing (12)

## Closed-form & MC Delta vs. $S_0$ and bump size, continued

- Drill into $[98, 102]$ domain for $n = 32 \times 1024$, bump sizes $= 0.1, 0.01, 0.001, 0.0001$: ($S_0$ points spaced 0.01)

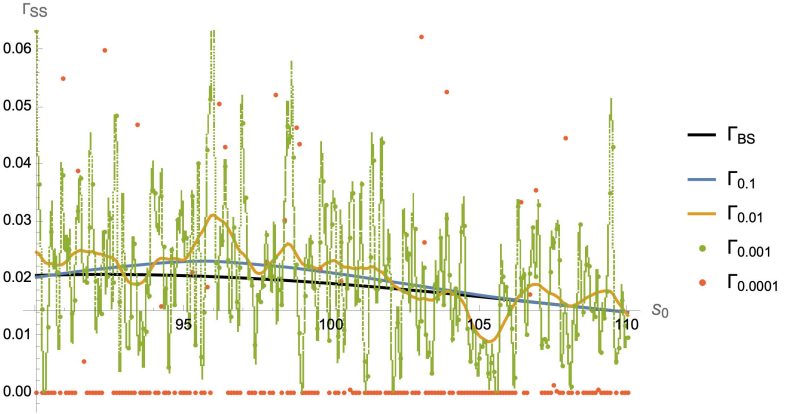



- At some point, convergence to theoretical Delta is limited by overall resolution ($n$) of simulation, as well as worsening truncation error as $\epsilon \searrow 0$

# 4.1. Finite Differencing (13)

## Closed-form & MC Gamma vs. $S_0$ and bump size

- What about Gamma? Start with n $= 1024$, bump sizes $\epsilon = 0.1, 0.01, 0.001, 0.0001$ ($S_0$ points spaced 0.1 except for 0.001 in last view of $\epsilon = 0.001$)

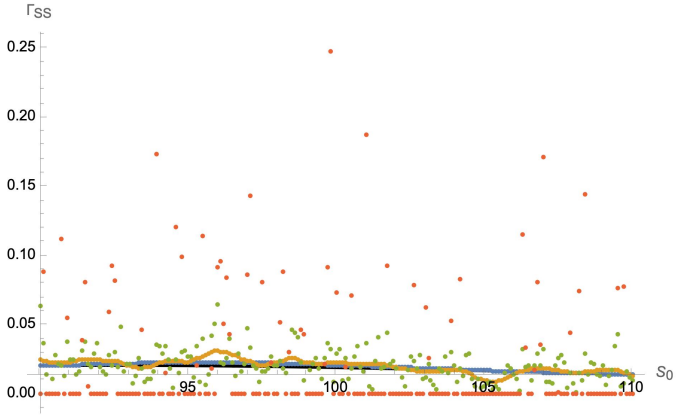



- Trade-off between bias for large shocks and noise for small shocks is striking
- Quantization effects are apparent even for 1% shocks

# 4.1. Finite Differencing (14)

## Closed-form & MC Gamma vs. $S_0$ and bump size, continued

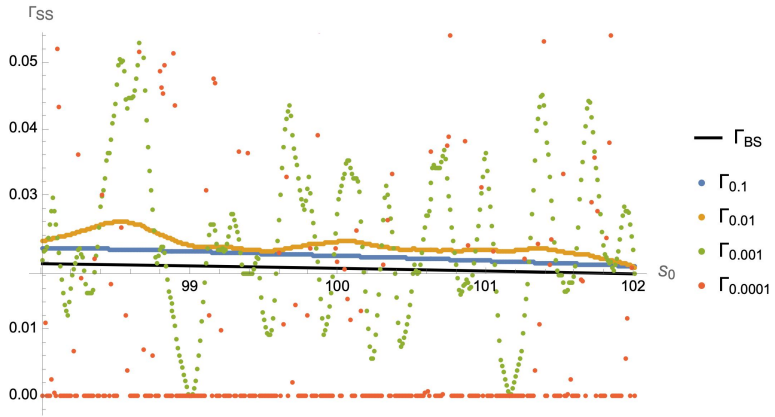
- Things actually look worse if the whole range of values is shown ($S_0$ points spaced 0.1)



# 4.1. Finite Differencing (15)

## Closed-form & MC Gamma vs. $S_0$ and bump size

- Drill down to $[98, 102]$ domain for $n = 1024$ ($S_0$ points spaced 0.01)

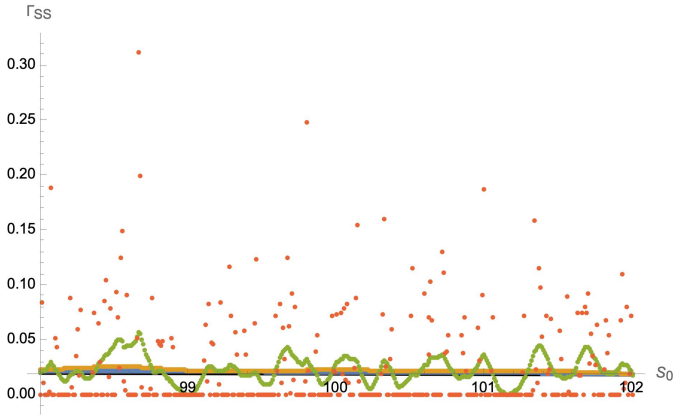



- Simulation Gamma becomes progressively less smooth as shock is reduced
- "Butterfly" structure is revealed for 10bp shocks

# 4.1. Finite Differencing (16)

## Closed-form & MC Gamma vs. $S_0$ and bump size, continued

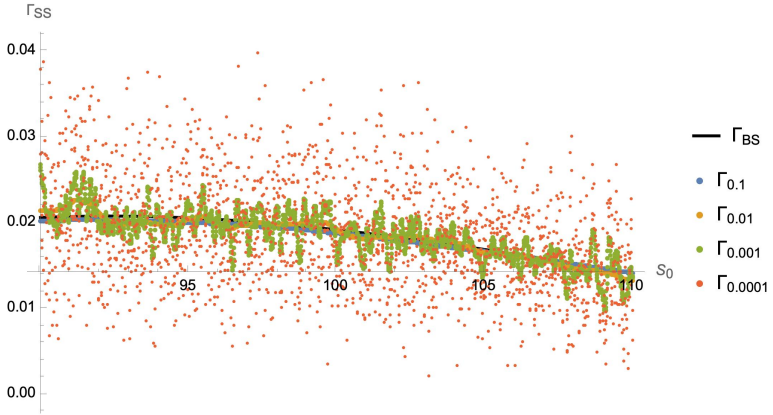
- Worse if the whole range of values is shown ($S_0$ points spaced 0.01):



# 4.1. Finite Differencing (17)

## Closed-form & MC Gamma vs. $S_0$ and bump size, continued

- What happens if we increase n to $32 \times 1024$?
  ($S_0$ points spaced 0.1 in first frame, 0.01 in last frame):

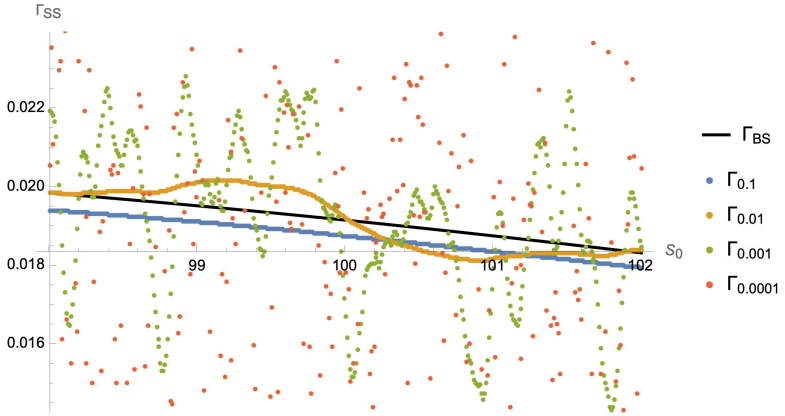



- Some improvement/convergence, but discretization error is still severe for small bumps

# 4.1. Finite Differencing (18)

## Closed-form & MC Gamma vs. $S_0$ and bump size

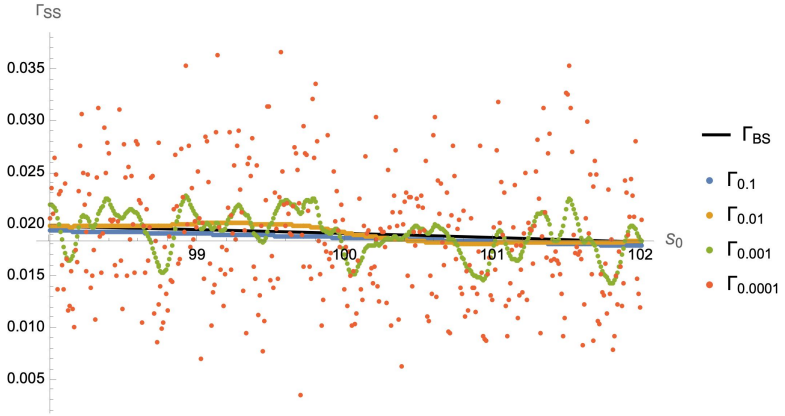
- Drill down to $[98, 102]$ domain for $n = 32 \times 1024$ ($S_0$ points spaced 0.01):



# 4.1. Finite Differencing (19)

## Closed-form & MC Gamma vs. $S_0$ and bump size, continued

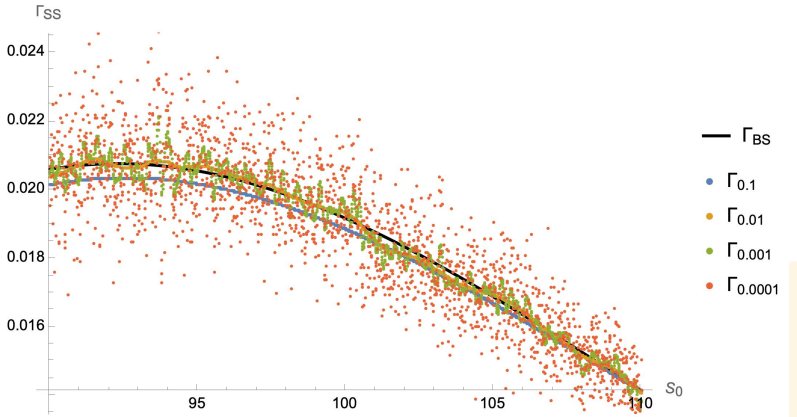
- Worse if the whole range of values is shown ($S_0$ points spaced 0.01):



# 4.1. Finite Differencing (20)

## Closed-form & MC Gamma vs. $S_0$ and bump size, continued

- One last increment of n to $1024 \times 1024$: ($S_0$ points spaced 0.1 in first frame, 0.01 in last frame):

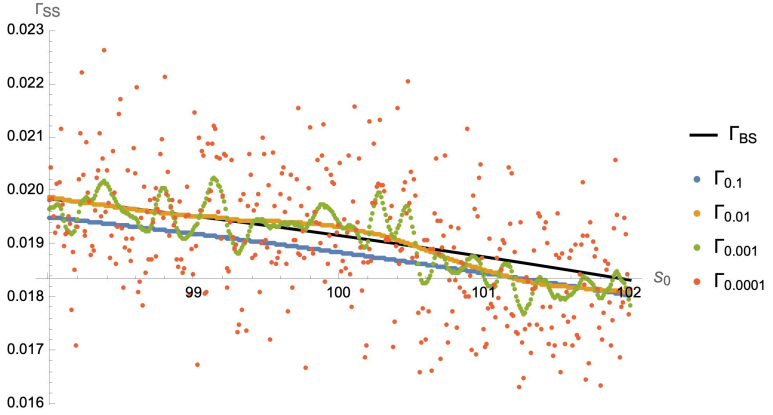



- Convergence continues, but discretization error remains severe for small bumps

# 4.1. Finite Differencing (21)

## Closed-form & MC Gamma vs. $S_0$ and bump size

- Drill down to $[98, 102]$ domain for $n = 1024 \times 1024$ ($S_0$ points spaced 0.01):

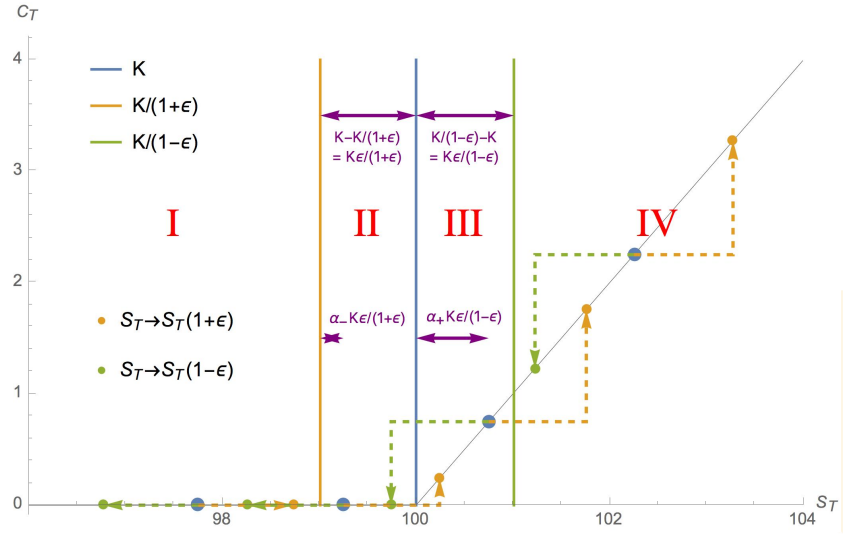



- Local (butterfly) structure for finer bump sizes appears at higher resolution.

# 4.1. Finite Differencing (22)

## What's Going on with Delta and Gamma?

- Schematic of bumping process $(K=100)$, $(\epsilon=0.01)$:



# 4.1. Finite Differencing (23)

## What's Going on with Delta and Gamma, continued?

- Point-wise contributions (w/o discounting) to Delta(s) & Gamma by $S_T$ region, with:
  $\alpha_- \doteq \frac{S_T - K/(1+\epsilon)}{K - K/(1+\epsilon)} = \frac{S_T(1+\epsilon)-K}{\epsilon K}$ ($S_T$ region 2 fraction);
  $\alpha_+ \doteq \frac{S_T - K}{K/(1-\epsilon) - K} = \frac{(S_T-K)(1-\epsilon)}{\epsilon K}$ ($S_T$ region 3 fraction)

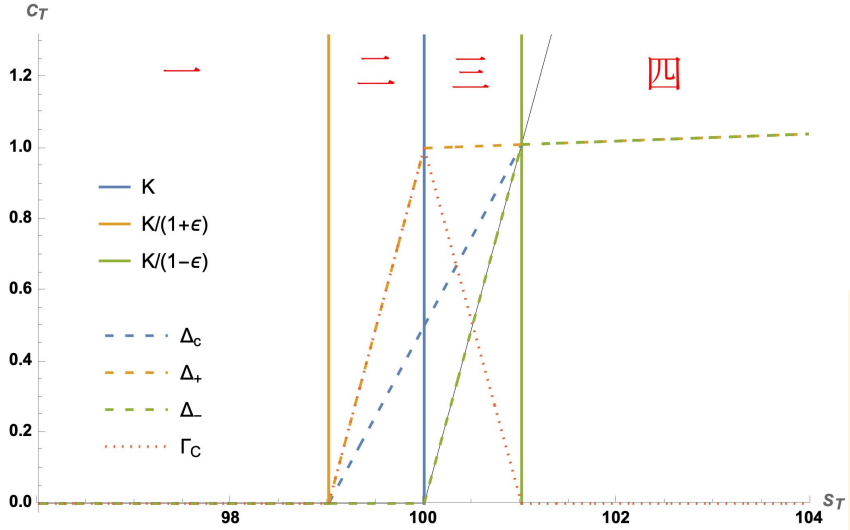
| | I | II | III | IV |
|---|---|---|---|---|
| $\Delta_+$ | 0 | $\frac{S_T(1+\epsilon)-K}{\epsilon S_0} = \frac{K}{S_0}\alpha_-$ | $\frac{S_T(1+\epsilon)-S_T}{\epsilon S_0} = \frac{S_T}{S_0} = \frac{K}{S_0}(1+\frac{\epsilon}{1-\epsilon}\alpha_+)$ | $\frac{S_T(1+\epsilon)-S_T}{\epsilon S_0} = \frac{S_T}{S_0}$ |
| $\Delta_-$ | 0 | 0 | $\frac{S_T-K}{\epsilon S_0} = \frac{K}{S_0}\frac{1}{1-\epsilon}\alpha_+$ | $\frac{S_T-S_T(1-\epsilon)}{\epsilon S_0} = \frac{S_T}{S_0}$ |
| $\Delta_c$ | 0 | $\frac{K}{2S_0}\alpha_-$ | $\frac{S_T(1+\epsilon)-K}{2\epsilon S_0} = \frac{K}{2S_0}(1+\frac{1+\epsilon}{1-\epsilon}\alpha_+)$ | $\frac{S_T}{S_0}$ |
| $\Gamma_c$ | 0 | $\frac{K}{\epsilon S_0^2}\alpha_-$ | $\frac{S_T(\epsilon-1)+K}{(\epsilon S_0)^2} = \frac{K}{\epsilon S_0^2}(1-\alpha_+)$ | 0 |

- $\Delta_+$: "extra" contribution phases in linearly over region II.
  Essentially a central difference formula with $\epsilon \to \epsilon/2$, and $S_0 \to S_0(1+\epsilon/2)$ or $K \overset{\sim}{\to} K(1-\epsilon/2)$.
- $\Delta_-$: converse of forward Delta, with phase-in linearly over region III.
  Essentially a central difference formula with $\epsilon \to \epsilon/2$, and $S_0 \to S_0(1-\epsilon/2)$ or $K \overset{\sim}{\to} K(1+\epsilon/2)$.
- $\Delta_c$: average of $\Delta_+$ & $\Delta_-$. Note decomposition into $\left(\frac{S_T}{S_0}\mathbf{1}_{S_T-K} \pm \frac{\epsilon S_0}{2}\Gamma_c\right)$
- $\Gamma_c$: butterfly pattern is clear.

# 4.1. Finite Differencing (24)

## Delta and Gamma Contributions

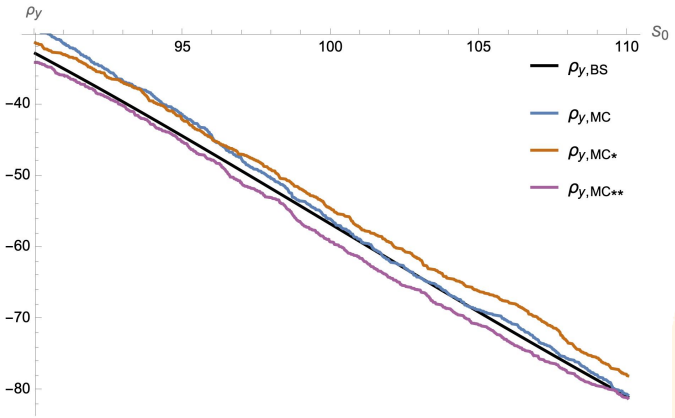
- $K = 100, \epsilon = 0.01$:



# 4.1. Finite Differencing (25)

## $\rho_y$ vs. $S_0$

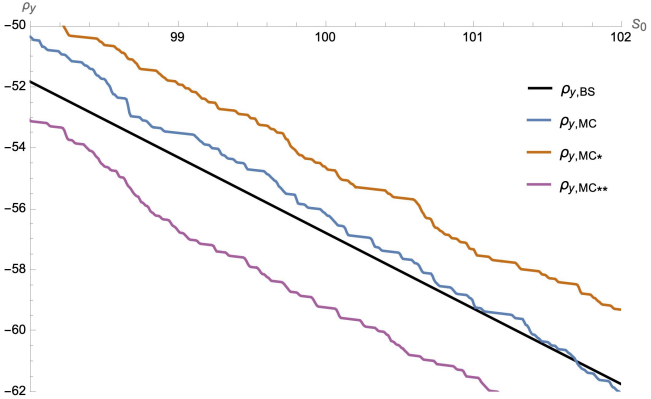
- Example parameters: $S = K = 100, T = 1, r = 4\%, y = 2\%, \sigma = 0.2$
- $n = 1024$, Plot $\rho_y$ vs. $S$ for bump size of 1bp:



# 4.1. Finite Differencing (26)

## $\rho_y$ vs. $S_0$, continued

- Drill into $[98, 102]$ domain:

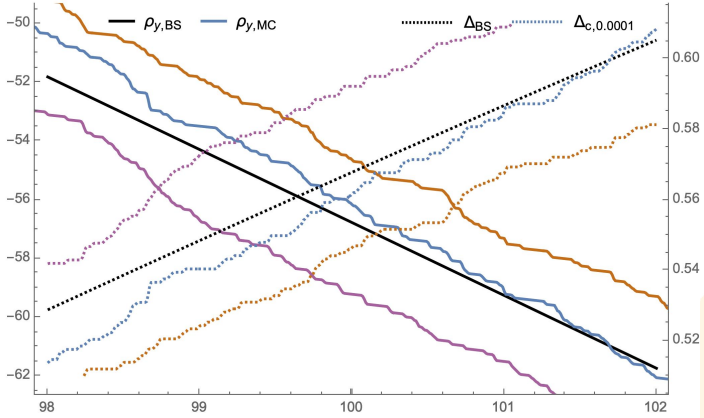



- Step function effects become apparent.

# 4.1. Finite Differencing (27)

## $\rho_y$ vs. $S_0$, continued

- Compare $\Delta_S$ with 1bp shocks in $[98, 102]$ domain:

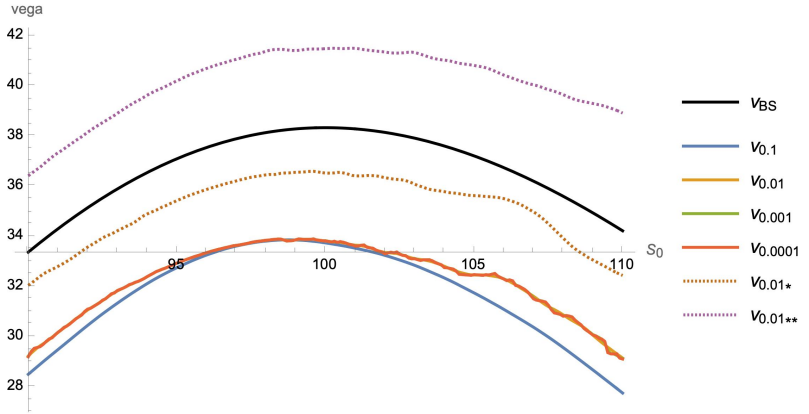



- Hints of path-wise proportionality!

# 4.1. Finite Differencing (28)

## $v$ vs. $S_0$ and bump size

- What about vega ($n = 1024$)?

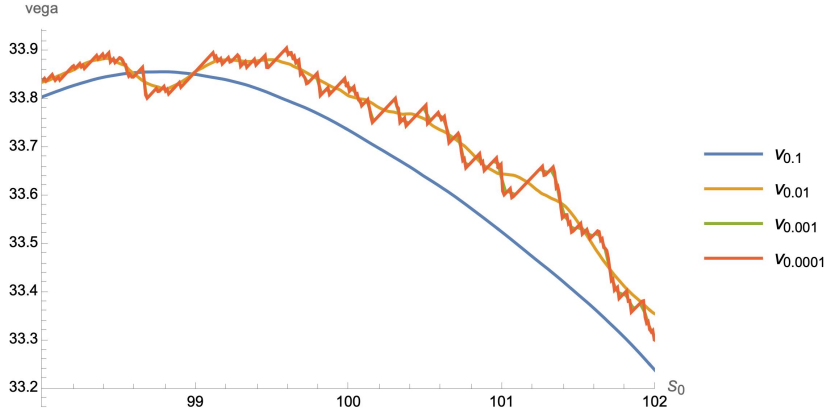



- Convexity bias for 10% shocks, poor convergence for small $n$, somewhat noisy tails.

# 4.1. Finite Differencing (29)

## $v$ vs. $S_0$ and bump size, continued

- Drilling into the $[98, 102]$ domain:

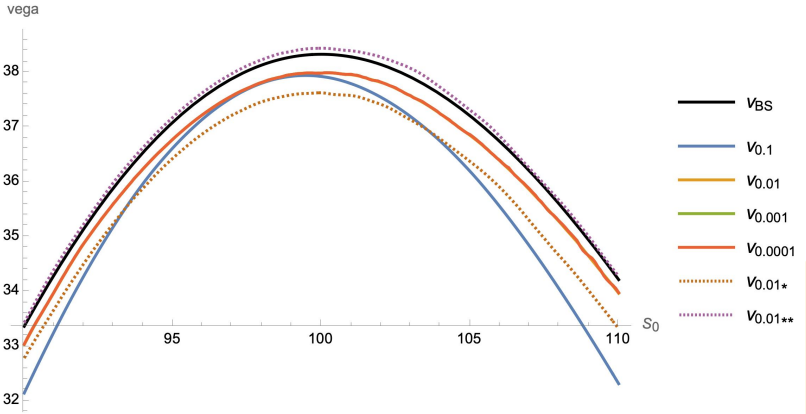



- Increasing noisiness for smaller bump sizes, intermediate between delta and gamma behavior.

# 4.1. Finite Differencing (30)

## $v$ vs. $S_0$ and bump size, continued

- Do matters improve for larger $n = 32 \times 1024$?:

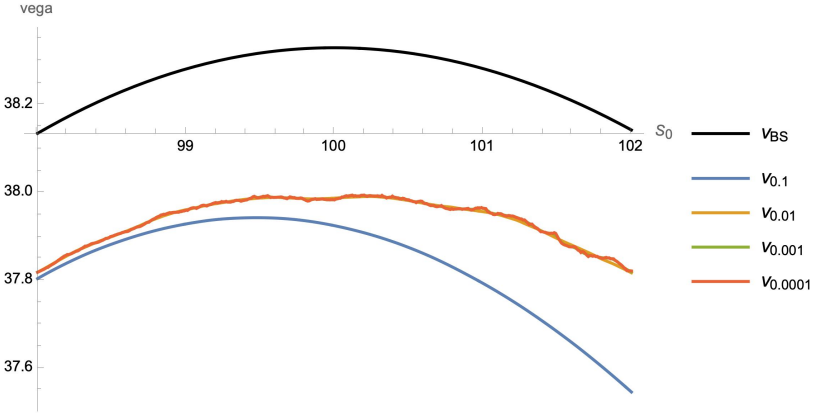



- Improved results, but convergence still somewhat slow.

# 4.1. Finite Differencing (31)

## $v$ vs. $S_0$ and bump size, continued

- Drilling into the $[98, 102]$ domain:

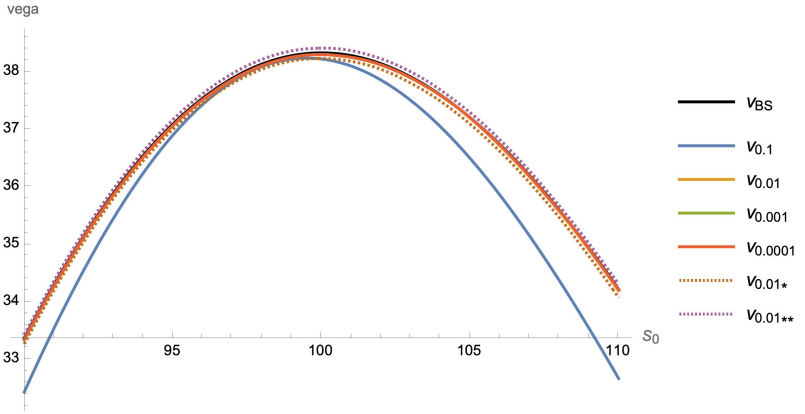



- Improved signal/noise ratio, in context of somewhat slow convergence

# 4.1. Finite Differencing (32)

## $v$ vs. $S_0$ and bump size, continued

- Do matters improve further for larger $n = 1024 \times 1024$?

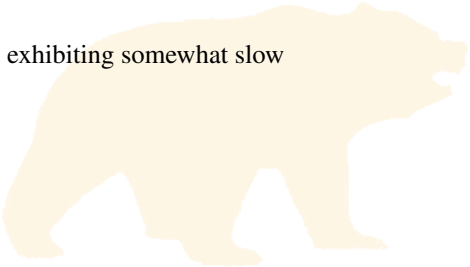



- Unlike $\rho_y$ vs. $\Delta_S$, there is clearly no path-wise proportionality vs. $\Gamma$ here.

# 4.1. Finite Differencing (33)

**What have we learned?**

- Each order of differentiation leads to progressively "rougher" results
- In addition, larger vs. smaller shocks trades off convexity biases vs. truncation errors
  - Delta and $\rho_y$ reveal $1/n$ quantization noise but otherwise appear to be reasonable estimators.
  - AT BEST, obtaining meaningful Gammas requires a careful balancing between $n$ and $\epsilon$.
- $\Delta_S$ and $\rho_y$ seem to have a path-wise proportionality
  - We might surmise a similar relationship between $\Delta_K$ and $\rho_r$
- vega has behavior intermediate between Gamma and Delta, exhibiting somewhat slow convergence with $n$



# 4.2. Analytical Differencing

- For those sensitivities which are reasonably well behaved under finite differencing as $\epsilon \searrow 0$, it makes sense to consider that limit in an analytical framework.
- Begin with $\rho_y$ for simplicity and examine the option value estimator:

$$\begin{aligned}\langle C_0\rangle_n &= \left\langle e^{-rT}C_T\right\rangle_n = e^{-rT}\langle C_T\rangle_n = e^{-rT}\langle C_{T,i}\rangle \\ &= \frac{e^{-rT}}{n}\sum_{i=1}^{n}[S_{T,i}-K]^+ = \frac{e^{-rT}}{n}\sum_{i=1}^{n}[S_0\, e^{\tilde{\mu}^{\mathbf{Q}}T+\sigma\sqrt{T}\,z_i}-K]^+, \text{ with } \tilde{\mu}^{\mathbf{Q}} \doteq r-y-\frac{\sigma^2}{2}\end{aligned}$$

- Consider: $\lim_{\epsilon\searrow 0}\frac{\langle C_0|_{y+\epsilon}\rangle_n-\langle C_0|_y\rangle_n}{\epsilon} = \partial_y\langle C_0\rangle_n = \partial_y\left\langle e^{-rT}C_T\right\rangle_n = e^{-rT}\langle\partial_y C_{T,i}\rangle$
- Define $\rho_{y,AD} \doteq \partial_y\langle C_0\rangle_n$, i.e., the discounted expectation of the path-wise differential of the payoff with respect to $y$:

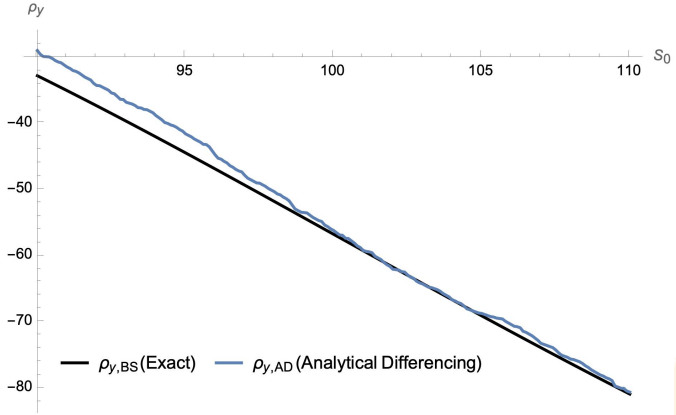
$$\begin{aligned}\rho_{y,AD} = \partial_y\langle C_0\rangle_n &= e^{-rT}\left\langle\partial_y\left[S_0e^{\tilde{\mu}^{\mathbf{Q}}T+\sigma\sqrt{T}\,z_i}-K\right]^+\right\rangle = e^{-rT}\left\langle\left(-T\,S_0e^{\tilde{\mu}^{\mathbf{Q}}T+\sigma\sqrt{T}\,z_i}\right)\mathbf{1}_{(S_{T,i}-K)}\right\rangle \\ &= -Te^{-rT}\left\langle(S_{T,i})\mathbf{1}_{(S_{T,i}-K)}\right\rangle = -T\frac{e^{-rT}}{n}\sum_{i=1}^{n}(S_{T,i})\mathbf{1}_{(S_{T,i}-K)}\end{aligned}$$

- This is just $-T$ times the Monte Carlo value of an asset-or-nothing binary call!
- NB: $\partial_y[F(y)]^+ = \partial_y[F(y)\cdot\mathbf{1}_{F(y)}] = \partial_yF(y)\cdot\mathbf{1}_{F(y)} + F(y)\cdot\partial_y\mathbf{1}_{F(y)} = \partial_yF(y)\cdot[\,\mathbf{1}_{F(y)}+F(y)\cdot\delta_{F(y)}\,]$. $\delta_{F(y)}$ is non-zero only on a set of measure zero (so can't be sampled by discrete Monte Carlo).

# 4.2. Analytical Differencing (2)

## $\rho_y$ vs. $S_0(90, 110)$

- Example parameters $(K = 100, T = 1, \sigma = 0.2, r = 4\%, y = 2\%), n = 1024$:

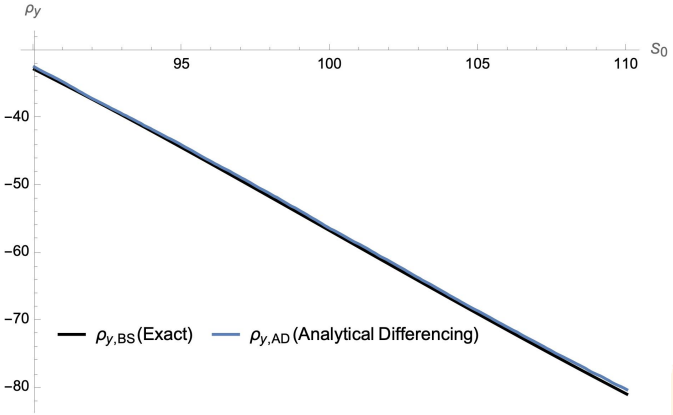



- Lack of smoothness (noise) in results: closely resemble bumping (for good reason!)

# 4.2. Analytical Differencing (3)

## $\rho_y$ vs. $S_0(90, 110)$

- $n = 32 \times 1024$:

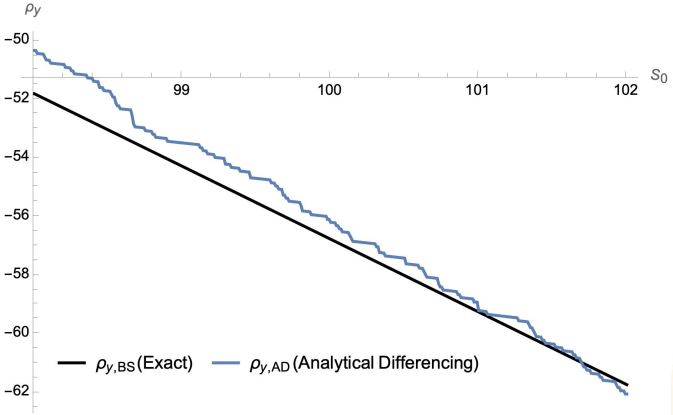



- Improved smoothness and convergence with increased $n$

# 4.2. Analytical Differencing (4)

## $\rho_y$ vs. $S_0(98, 102)$

- Focus in on $98 \leq S_0 \leq 102$ for $n = 1024$:

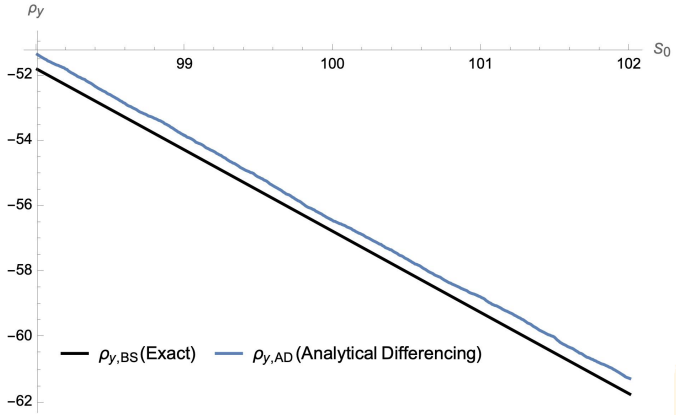



- Estimation error and $\frac{1}{n}$ quantization error are apparent for small $n$

# 4.2. Analytical Differencing (5)

## $\rho_y$ vs. $S_0(98, 102)$

- $n = 32 \times 1024$:




- Improved smoothness and more regular convergence with increased $n$

# 4.2. Analytical Differencing (6)

- AD estimator for Delta:

$$\Delta_{S,AD} \doteq \partial_{S_0} \langle C_0 \rangle_n = \frac{e^{-rT}}{n} \sum_{i=1}^{n} \partial_{S_0} [S_0 \, e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i} - K]^+$$

$$= \frac{e^{-rT}}{n} \sum_{i=1}^{n} (e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i}) \, \mathbf{1}_{(S_{T,i} - K)} = \frac{e^{-rT}}{n \, S_0} \sum_{i=1}^{n} (S_{T,i}) \mathbf{1}_{(S_{T,i} - K)}$$

  - This is just $1/S_0$ times the Monte Carlo value of an asset-or-nothing binary call.
  - It is now clear why we see $1/n$ quantization error in bumping and AD sensitivities:
    - As we move $S_0$ (continuously), $\Delta_{S,AD}$ also changes continuously until another sample point moves into (or out of) the money, at which point Delta jumps by $\frac{e^{-rT} K}{n \, S_0}$.
  - Comparison of the terms within the sum confirms path-wise proportionality of Delta & $\rho_y$.
- What about Gamma? Try differentiating $\Delta_{S,AD}$ by $S_0$:

$$\Gamma_{SS,AD} \doteq \partial_{S_0} \Delta_{S,AD} = \frac{e^{-rT}}{n} \sum_{i=1}^{n} \partial_{S_0} [(e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i}) \, \mathbf{1}_{(S_{T,i} - K)}]$$

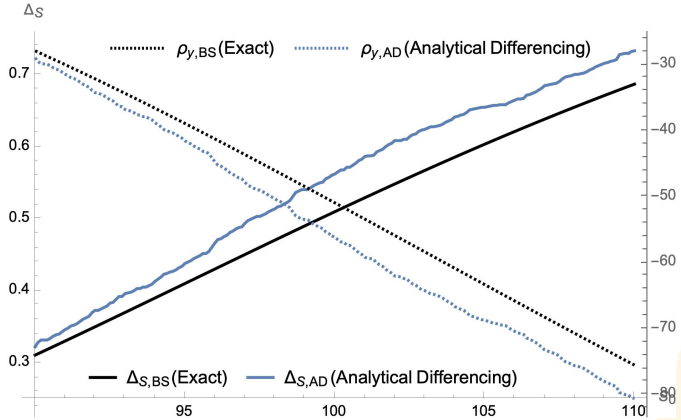
$$= \frac{e^{-rT}}{n} \sum_{i=1}^{n} (e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i})^2 \, \delta_{(S_{T,i} - K)} = \frac{e^{-rT}}{n \, S_0^2} \sum_{i=1}^{n} (S_{T,i})^2 \delta_{(S_{T,i} - K)}$$

  - Discrete sum of (Dirac) delta functions is 0 everywhere except on a point set of measure 0!
  - Explanation of jumpiness in finite differencing Gamma follows reasoning for Delta above.

# 4.2. Analytical Differencing (7)

## Delta vs. $S_0(90, 110)$

- $n = 1024$:




- Path-wise proportionality with $\rho_y$ is clear.

# 4.2. Analytical Differencing (8)

- AD estimator for rho$_r$ requires us to account for both roles of $r$: drift *and* discounting:

$$\begin{aligned}
\rho_{r,AD} = \partial_r \langle C_0 \rangle_n &= \partial_r [e^{-rT} \langle C_{T,i} \rangle] = -Te^{-rT} \langle C_{T,i} \rangle + e^{-rT} \langle \partial_r C_{T,i} \rangle \\
&= -Te^{-rT} \left\langle [S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i} - K]^+ \right\rangle + e^{-rT} \left\langle \partial_r [S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i} - K]^+ \right\rangle \\
&= -Te^{-rT} \left\langle [S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i} - K]^+ \right\rangle + Te^{-rT} \left\langle (S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i}) \mathbf{1}_{(S_{T,i} - K)} \right\rangle \\
&= Te^{-rT} \left\langle -(S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i} - K) \mathbf{1}_{(S_{T,i} - K)} + (S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i}) \mathbf{1}_{(S_{T,i} - K)} \right\rangle \\
&= Te^{-rT} \left\langle K \mathbf{1}_{(S_{T,i} - K)} \right\rangle = T \frac{Ke^{-rT}}{n} \sum_{i=1}^{n} \mathbf{1}_{(S_{T,i} - K)}
\end{aligned}$$

  - This is just $(K) \cdot T$ times the Monte Carlo value of an cash-or-nothing binary call.
- We reach the parallel destination much more easily with the strike Delta:

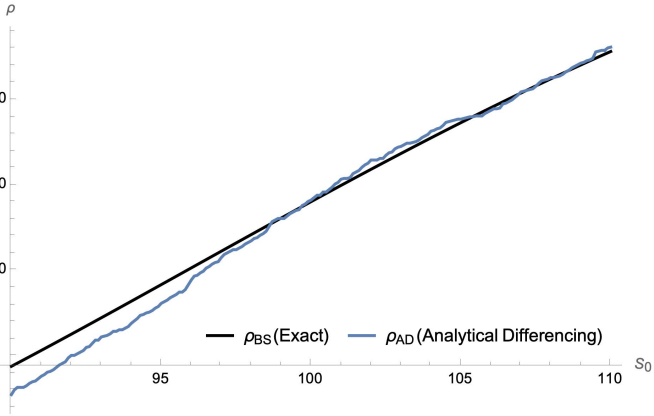
$$\begin{aligned}
\Delta_{K,AD} \doteq \partial_K \langle C_0 \rangle_n &= \frac{e^{-rT}}{n} \sum_{i=1}^{n} \partial_K [S_0 e^{\tilde{\mu}^{\mathbf{Q}} T + \sigma\sqrt{T} z_i} - K]^+ \\
&= \frac{e^{-rT}}{n} \sum_{i=1}^{n} (-1) \mathbf{1}_{(S_{T,i} - K)} = -\frac{e^{-rT}}{n} \sum_{i=1}^{n} \mathbf{1}_{(S_{T,i} - K)}
\end{aligned}$$

  - Quantization effect and proportionality to $\rho_{AD}$ are both apparent.

# 4.2. Analytical Differencing (9)

**rho vs. $S_0(90, 110)$**

- $n = 1024$:




- Results resemble closely those for $\rho_y$.

# 4.2. Analytical Differencing (10)

- Finally, derive the AD estimator for vega:

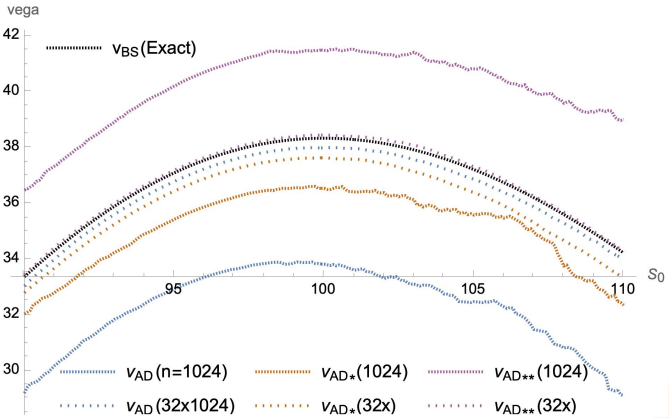
$$\begin{aligned} v_{AD} = \partial_\sigma \left\langle e^{-rT} C_{T,i} \right\rangle &= e^{-rT} \left\langle \partial_\sigma (S_0 e^{(r-y-\sigma^2/2)T+\sigma\sqrt{T}\, z_i} - K)^+ \right\rangle \\ &= e^{-rT} \left\langle (\sqrt{T} z_i - \sigma T) S_{T,i} \mathbf{1}_{(S_{T,i}-K)} \right\rangle \\ &= \sqrt{T} e^{-rT} \left\langle (z_i - \sigma\sqrt{T}) S_{T,i} \mathbf{1}_{(S_{T,i}-K)} \right\rangle \end{aligned}$$

  - Unlike for Gamma, we *do* obtain a meaningful estimator.
    - Clearly, there is no path-wise proportionality between the two.
  - This doesn't map into a simple binary option payoff
    - Translated into $S_T$ terms, it would be a binary paying something like $S_T \ln(S_T)$ conditional on $S_T \geq K$.
  - However, the discontinuousness of the "payoff" is apparent so we should expect noisy results ($1/n$ quantization).
  - In addition, the asset-or-nothing term in the expectation is weighted by $z_i$, so we are in some sense sampling higher moments of the distribution of $S_T$ or $z$.
    - We could expect both greater relative standard errors than the first-order ($\Delta_S, \rho_y, \Delta_K, \rho$) sensitivities and noisier (noise weighted toward the) tails.

# 4.2. Analytical Differencing (11)

## vega vs. $S_0(90, 110)$

- All parameters as before, $n = 1024$ and $n = 32 \times 1024$:




- Expectations regarding noisiness and standard errors appear to be confirmed.

# 4.3. Likelihood Ratio

- Analytical differencing for an option value's sensitivity to an input parameter is based on the $\epsilon \searrow 0$ limit of the finite difference sensitivity to that parameter.
  - In the $\epsilon \searrow 0$ limit, we calculate the (discounted) sample mean of the path-wise sensitivity of the payoff to the parameter under the (simulated) risk-neutral density for $S_T$.
  - We can usually interpret the result as an estimator for the present value of a modified (i.e., binary option) payoff, again under the (unchanged) risk-neutral density for $S_T$.
- The likelihood ratio method takes a different approach:
  It holds the (path-wise) payoff constant and uses the input parameter sensitivity of the risk-neutral density for $S_T$ to obtain an estimator for the sensitivity of the option value.
- Before exploring how this works for calculating option sensitivities, let's consider the likelihood ratio concept more generally.

# 4.3. Likelihood Ratio (2)

- Suppose we had valued an option payoff $C_T(S_T)$ by generating $n$ final asset prices $\{S_{T,i}\}$ from a risk-neutral distribution we know (e.g., the BS log-normal $\mathbf{Q}$ measure).
- How could we value the option under a different, "target" $\mathbf{Q}'$ distribution that is "not too far" from the distribution we know (and have done the hard work to simulate)?
  - Perhaps the target distribution $\mathbf{Q}'$ has a form in which we know the density but don't know how to simulate (sample from), *or*
  - Perhaps the target distribution $\mathbf{Q}'$ is closely related to the original distribution $\mathbf{Q}$ in some way (e.g., is perturbed from the original distribution or is weighted more heavily in one or both tails).
  - The original $\mathbf{Q}$ and target $\mathbf{Q}'$ distributions need to share the same support (i.e., be non-zero on the same set of points, or conversely, share the same zero set).
- The likelihood ratio method is based on a simple idea: sample from the original distribution, but weight the observations by the ratio of the two probability densities (i.e., multiply by the Radon-Nikodym derivative).
- Denote the original density by $q(S_T)$ and the target density by $q'(S_T)$:

$$\begin{aligned}\mathbb{E}^{\mathbf{Q}'}[C_T(S_T)] &= \int_0^\infty dS_T\, q'(S_T)\, C_T(S_T) \\ &= \int_0^\infty dS_T\, q(S_T) \frac{q'(S_T)}{q(S_T)}\, C_T(S_T) = \mathbb{E}^{\mathbf{Q}}\left[\frac{q'(S_T)}{q(S_T)}\, C_T(S_T)\right]\end{aligned}$$

# 4.3. Likelihood Ratio (3)

- Apply this idea to derive a Monte-Carlo estimator of the present value of the option under $\mathbf{Q}'$ using the samples generated from $\mathbf{Q}$:

$$\langle C_0 \rangle_{\mathbf{Q},n} = e^{-rT} \langle C_{T,i} \rangle_{\mathbf{Q}} = \frac{e^{-rT}}{n} \sum_{i=1}^{n} C_T(S_{T,i})$$

$$\langle C_0 \rangle_{\mathbf{Q}',n} = e^{-rT} \left\langle \frac{q'(S_T)}{q(S_T)} C_{T,i} \right\rangle_{\mathbf{Q}} = \frac{e^{-rT}}{n} \sum_{i=1}^{n} \frac{q'(S_{T,i})}{q(S_{T,i})} C_T(S_{T,i})$$

- One interpretation is in terms of sample weights $w_i$ under the two distributions:
  - Under $\mathbf{Q}$, each sample is weighted equally, i.e., $w_i = \frac{1}{n} \; \forall i$
  - Under $\mathbf{Q}'$, the samples are re-weighted in proportion to the likelihood ratio: $w_i' = \frac{1}{n} \frac{q'(S_{T,i})}{q(S_{T,i})}$
  - One should be cautious of finite-size effects for small $n$, i.e., $\sum_i w_i'$ generally $\neq 1$. Re-normalizing the weights (analogous to moment matching) may be helpful.
- The likelihood ratio approach forms the basis of the importance sampling method, with wide applications beyond calculation of sensitivities.
  - E.g.: valuation of deep out-of-the-money options by simulating $S_T$ under a distribution centered within the option's positive payoff region and then transforming back to the BS measure (using the Radon-Nikodym derivative).

# 4.3. Likelihood Ratio (4)

- Now, use this method to derive likelihood-ratio estimators of option sensitivities.
- Illustrate for $\rho_y$ (algebra is simplest), but approach is generic to all sensitivities (caveat for $\rho_r$: as for $\rho_{r,AD}$, we must also account for the discount factor sensitivity).
- Start by assuming a finite perturbation $y \to y + \epsilon$. Then:

$$\langle C_0 \rangle_{y+\epsilon} = e^{-rT} \left\langle \frac{q'(S_{T,i}|y+\epsilon)}{q(S_{T,i}|y)} C_{T,i} \right\rangle_y$$

  - Expand the likelihood ratio in $\epsilon$:

$$\frac{q'(S_{T,i}|y+\epsilon)}{q(S_{T,i}|y)} = \frac{q(S_{T,i}|y) + \epsilon\, \partial_y q(S_{T,i}|y)}{q(S_{T,i}|y)} + \mathcal{O}(\epsilon^2)$$

$$\text{Then: } \rho_{y,LR} \doteq \lim_{\epsilon \searrow 0} \frac{\langle C_0 \rangle_{y+\epsilon} - \langle C_0 \rangle_y}{\epsilon} = e^{-rT} \left\langle \frac{\partial_y q(S_{T,i}|y)}{q(S_{T,i}|y)} C_{T,i} \right\rangle_y = e^{-rT} \langle \partial_y \ln[q(S_{T,i}|y)]\, C_{T,i} \rangle$$

$$\text{Since } q(S_{T,i}|y) = \exp\left( -\frac{1}{2} \left[ \frac{\ln(S_{T,i}/S_0) - \tilde{\mu}^{\mathbf{Q}} T}{\sigma\sqrt{T}} \right]^2 \right) / (\sqrt{2\pi T}\sigma\, S_{T,i}),$$

$$\ln[q(S_{T,i}|y)] = -\frac{1}{2} \left[ \frac{\ln(S_{T,i}/S_0) - \tilde{\mu}^{\mathbf{Q}} T}{\sigma\sqrt{T}} \right]^2 - \ln(\sqrt{2\pi T}\sigma\, S_{T,i}), \text{ and}$$

$$\partial_y \ln[q(S_{T,i}|y)] = -\frac{\sqrt{T}}{\sigma} \left[ \frac{\ln(S_{T,i}/S_0) - \tilde{\mu}^{\mathbf{Q}} T}{\sigma\sqrt{T}} \right] = -\frac{\sqrt{T}}{\sigma}\, z_i$$

# 4.3. Likelihood Ratio (5)

- Hence:

$$\rho_{y,LR} = e^{-rT}\left\langle -\frac{\sqrt{T}}{\sigma}\left[\frac{\ln(S_{T,i}/S_0) - \tilde{\mu}^{\mathbf{Q}}T}{\sigma\sqrt{T}}\right] C_{T,i}\right\rangle$$
$$= -\frac{\sqrt{T}}{\sigma}e^{-rT}\langle z_i\, C_{T,i}\rangle = -\frac{\sqrt{T}}{\sigma}\frac{e^{-rT}}{n}\sum_{i=1}^{n} z_i(S_{T,i}-K)^+$$

- Comparing to the analytical differencing formulæ for $\rho_y$ and vega:

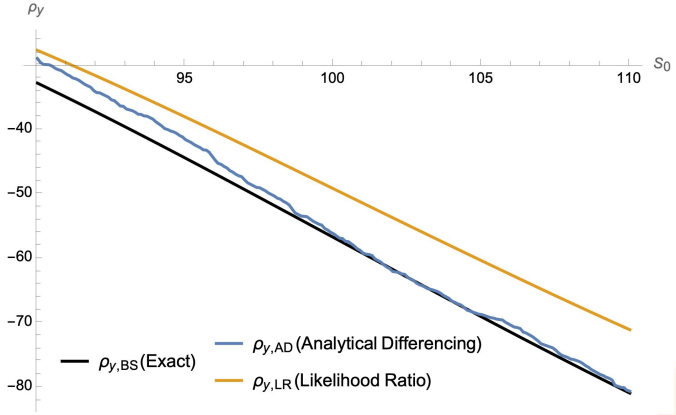
$$(\rho_y):\ \rho_{y,AD} = -Te^{-rT}\left\langle (S_{T,i})\mathbf{1}_{(S_{T,i}-K)}\right\rangle$$
$$(\text{vega}):\ v_{AD} = \sqrt{T}e^{-rT}\left\langle (z_i - \sigma\sqrt{T})(S_{T,i})\mathbf{1}_{(S_{T,i}-K)}\right\rangle$$

- We can make a few observations:
  - We can again interpret the estimator as sampling from a modified payoff function
  - In comparison to $\rho_{y,AD}$, the payoff is smooth (proportional to the call option payoff, not a binary option payoff).
    - $\rho_{y,LR}$ is likely to be smoother than $\rho_{y,AD}$ and not suffer from $1/n$ quantization.
  - However, more like (vega) $v_{AD}$, the payoff function has an extra proportionality to $z_i$ and so is sampling higher moments of $S_T$ and $x$.
    - $\rho_{y,LR}$ is likely to have higher (standard) errors and slower convergence than $\rho_{y,AD}$.

# 4.3. Likelihood Ratio (6)

## $\rho_y$ vs. $S_0(90, 110)$

- Example parameters as before, $n = 1024$:

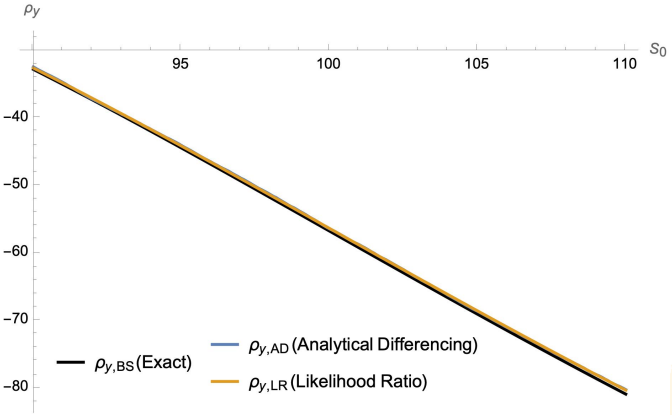



- Superior smoothness, but slower convergence

# 4.3. Likelihood Ratio (7)

## $\rho_y$ vs. $S_0(90, 110)$

- $n = 32 \times 1024$:

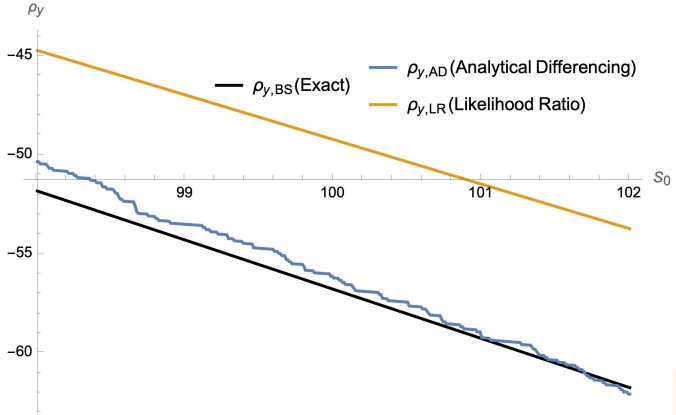



- Improved convergence with increased $n$

# 4.3. Likelihood Ratio (8)

## $\rho_y$ vs. $S_0(98, 102)$

- Focus in on $98 \leq S_0 \leq 102$ for $n = 1024$:

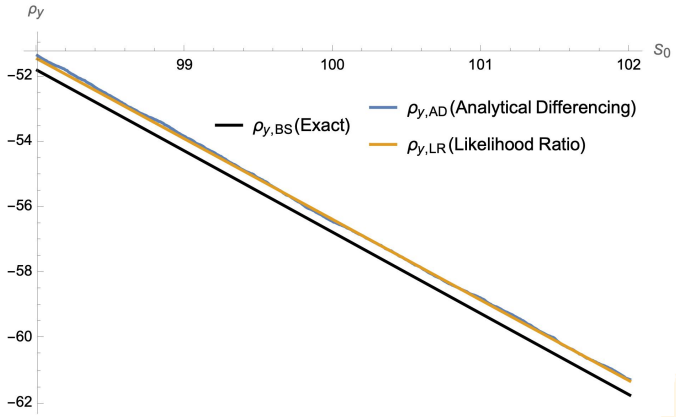



- $\frac{1}{n}$ quantization error is absent from $\rho_{y,LR}$ but convergence is much slower

# 4.3. Likelihood Ratio (9)

## $\rho_y$ vs. $S_0(98, 102)$

- $n = 32 \times 1024$:

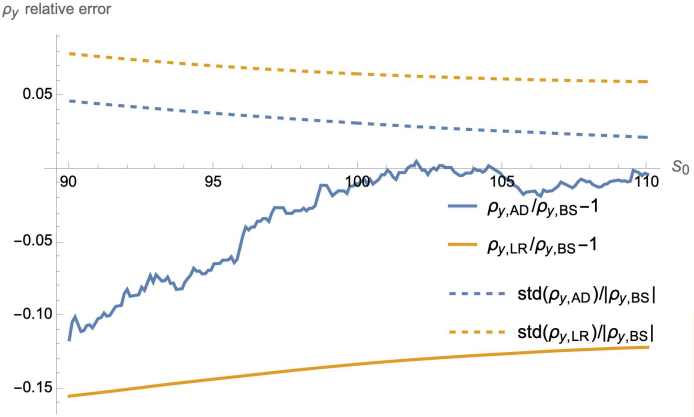



- Convergence appears much improved with increased $n$ (but we are probably benefitting from a "lucky draw").

# 4.3. Likelihood Ratio (10)

## $\rho_y$ vs. $S_0(90, 110)$

- Estimator errors and standard errors (relative to exact BS $\rho_y$) for $n = 1024$:

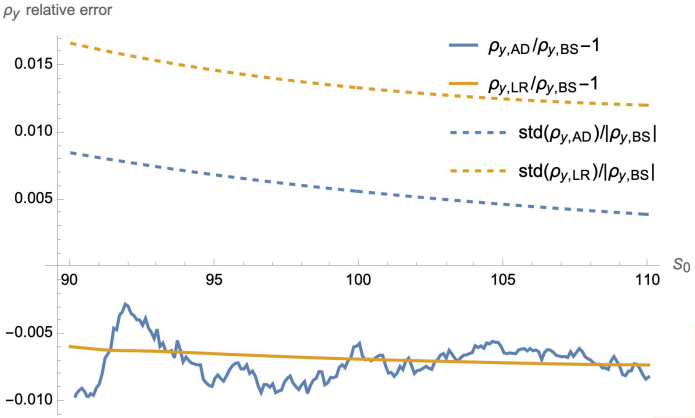



- Relative smoothness and convergence properties of $\rho_{y,AD}$ and $\rho_{y,LR}$ are clear in this view.

# 4.3. Likelihood Ratio (11)

## $\rho_y$ vs. $S_0(90, 110)$

- Estimator errors and standard errors (relative to exact BS $\rho_y$) for $n = 32 \times 1024$:

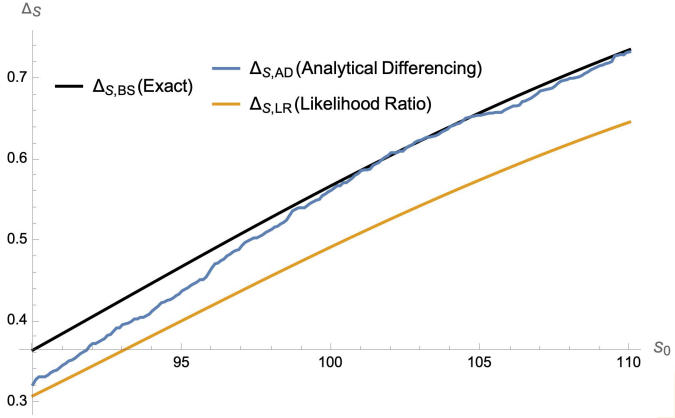



- Both $\rho_{y,AD}$ and $\rho_{y,LR}$ benefit from increased $n$, but standard errors of the latter are still more than twice as large.

# 4.3. Likelihood Ratio (12)

## $\Delta_S$ vs. $S_0(90, 110)$

- Example parameters as before, $n = 1024$:

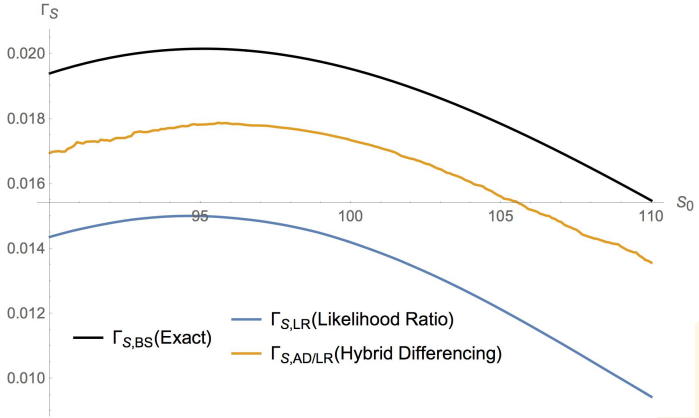



- Similar conclusions as for $\rho_y$ (unsurprisingly)

# 4.3. Likelihood Ratio (13)

## $\Gamma_{SS}$ vs. $S_0(90, 110)$

- Example parameters as before, $n = 1024$:

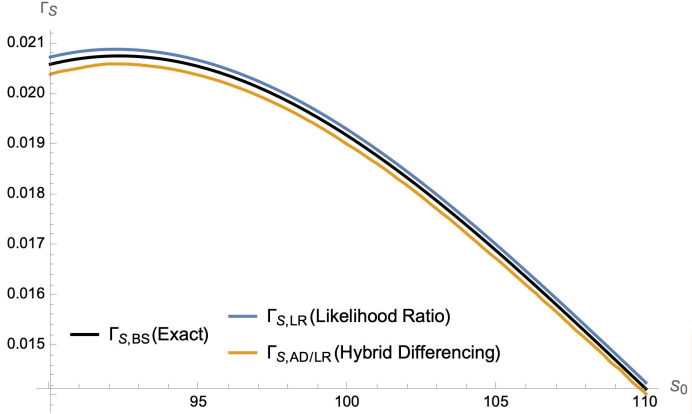



- For the first time, we have a meaningful simulation estimator of Gamma!

# 4.3. Likelihood Ratio (14)

## $\Gamma_{SS}$ vs. $S_0(90, 110)$ (continued)

- Increase $n$ to $32 \times 1024$:

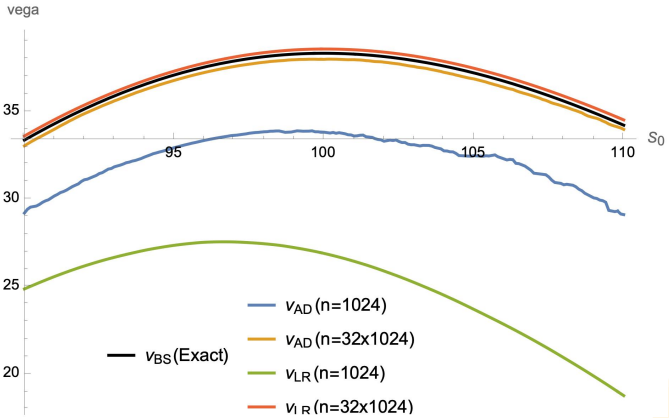



- And we do converge toward the exact value as $n$ increases!

# 4.3. Likelihood Ratio (15)

## vega vs. $S_0(90, 110)$

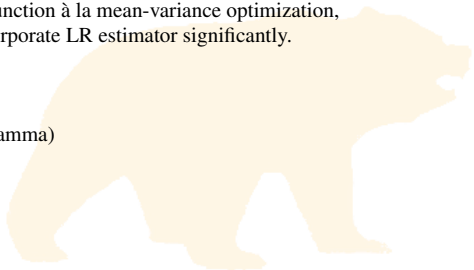
- Example parameters as before, $n = 1024, 32 \times 1024$:




- Convergence as expected

# 4.3. Likelihood Ratio (16)

- Trade-offs between Analytical Differencing and Likelihood Ratio estimators:
  - Noisiness ($1/n$ quantization) for AD vs. global smoothness for LR;
  - Faster (for AD) vs. slower (for LR) convergence.
- There doesn't have to be an either/or choice between the two approaches:
  - Possibility of forming a linear combination of the two that optimizes some objective function
    - e.g., minimize estimator variance à la multiple control variates
      (though beware: high correlation between AD and LR "payoffs" together with larger LR variance may lead to negative weight on LR estimator!).
    - Weight both $1/n$ quantization and estimator variance in objective function à la mean-variance optimization, but note that noise weighting will have to be high for results to incorporate LR estimator significantly.
  - Hybrid estimators for second-order sensitivities, e.g., Gamma
    - LR estimator for $S_0$ sensitivity of AD Delta, or vice-versa.
    - Possibility of combining these (perhaps together with "pure" LR Gamma) to form minimum variance estimator.



# 4.4. Appendix: A Glance at Importance Sampling

- Return to the idea of pricing a payoff $C_T(S_T)$ using a reference (original) probability measure $\mathbf{Q}$ with terminal asset price density $q(S_T)$ to simulate a target (desired) measure $\mathbf{Q}'$ with density $q'(S_T)$:

$$\begin{aligned}\mathbb{E}^{\mathbf{Q}'}[C_T(S_T)] &= \int_0^\infty dS_T\, q'(S_T)\, C_T(S_T) \\ &= \int_0^\infty dS_T\, q(S_T) \frac{q'(S_T)}{q(S_T)}\, C_T(S_T) = \mathbb{E}^{\mathbf{Q}}\left[\frac{q'(S_T)}{q(S_T)}\, C_T(S_T)\right]\end{aligned}$$

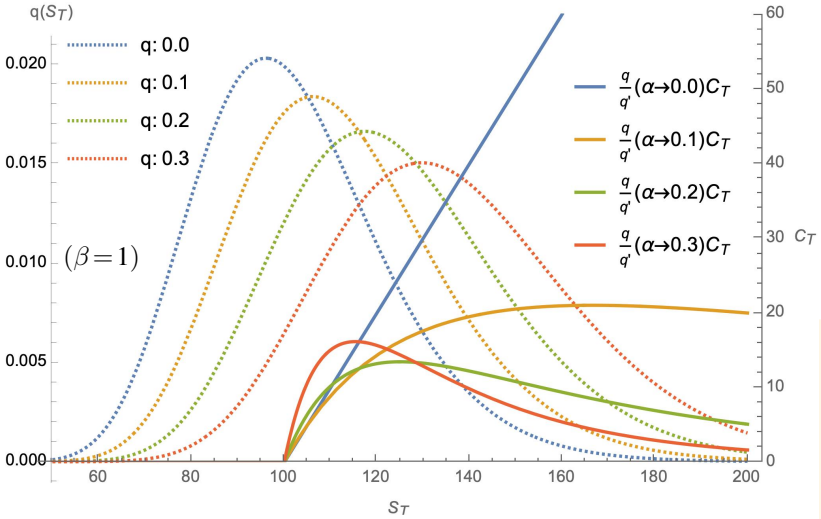
  i.e., we simulate $C_T$ *times* the likelihood ratio $\frac{q'(S_T)}{q(S_T)}$ using samples drawn from $\mathbf{Q}$:

$$\langle C_0 \rangle_{\mathbf{Q}',n} = e^{-rT}\left\langle \frac{q'(S_T)}{q(S_T)} C_{T,i} \right\rangle_{\mathbf{Q}} = \frac{e^{-rT}}{n} \sum_{i=1}^{n} \frac{q'(S_{T,i})}{q(S_{T,i})} C_T(S_{T,i})$$

- Let's reverse the sense in which we've been thinking about $\mathbf{Q}$ and $\mathbf{Q}'$ so far.
  - Let our target $q'$ be the BS density for $S_T$ or for $x_T \doteq \ln(S_T/S_0)$: $x_T^{\mathbf{Q}'} \sim N[\tilde{\mu}^{\mathbf{Q}} T, \sigma^2 T]$
  - Let the reference $q$ be some other density for $S_T$ or $x_t$ that we know how to simulate, e.g., (though not necessarily limited to) $x_T^{\mathbf{Q}} \sim N[(\tilde{\mu}^{\mathbf{Q}} + \alpha)T, \beta^2\sigma^2 T)$ with parameters $\alpha$, $\beta$
  - Why do we think this might be a useful path to go down?

# 4.4. Appendix: A Glance at Importance Sampling (2)

- Probability density isn't well-matched to the payoff:




- By shifting (i.e., $\alpha$), & possibly even scaling (i.e., $\beta$) $q$, can we obtain a better fit?

# 4.4. Appendix: A Glance at Importance Sampling (3)

- How should we (optimally) select $q$ (or its parameters)?
  - We know that (population) expectations are unaffected by the choice of $q$, while sample expectations will only differ by $n^{-1/2}$ noise.
  - In MC simulation, we seek to minimize (or at least reduce) the estimator variance.
  - Equivalently, since the mean is fixed, we can minimize the second (total) moment, either in population or in sample:

$$\mathbb{E}^{\mathbf{Q}}\left[\left(\frac{q'(S_T)}{q(S_T)}C_T(S_T)\right)^2\right] = \int_0^\infty dS_T\, q(S_T)\left(\frac{q'(S_T)}{q(S_T)}C_T(S_T)\right)^2 = \int_0^\infty dS_T\, q'(S_T)\frac{q'(S_T)}{q(S_T)}C_T^2(S_T)$$

$$\simeq \left\langle\left(\frac{q'(S_{T,i})}{q(S_{T,i})}C_{T,i}\right)^2\right\rangle_{\mathbf{Q}} = \frac{1}{n}\sum_{i=1}^{n}\left(\frac{q'(S_{T,i})}{q(S_{T,i})}C_{T,i}\right)^2$$

- Example: vanilla call option with:

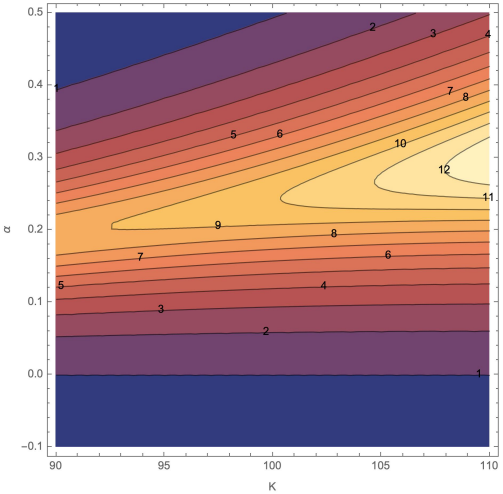
$$x_T^{\mathbf{Q}'} \sim N[\tilde{\mu}^{\mathbf{Q}}T, \sigma^2 T]$$

$$x_T^{\mathbf{Q}} \sim N[(\tilde{\mu}^{\mathbf{Q}} + \alpha)T, \sigma^2 T]\ (\beta = 1)$$

$$\frac{q'(S_{T,i})}{q(S_{T,i})} = e^{\frac{\alpha[(\alpha+2\mu^{\mathbf{Q}})T - 2z_T]}{2\sigma^2}}$$

# 4.4. Appendix: A Glance at Importance Sampling (4)

- (Ratio of variance with $\alpha = 0$ to variance with $\alpha$) vs. $K$ and $\alpha$:



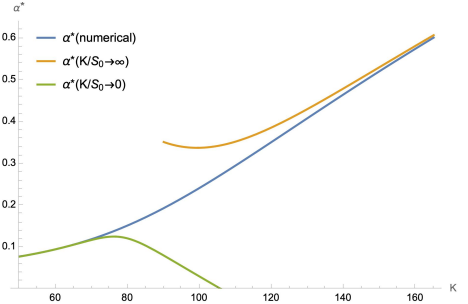

- Significant improvement ($10\times$ variance reduction) is possible even near the money!

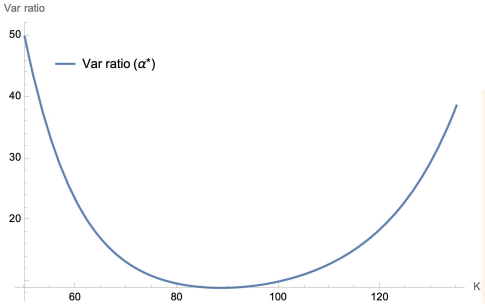
# 4.4. Appendix: A Glance at Importance Sampling (5)

- Population $2^{nd}$ moment (and variance) for vanilla options can be written explicitly in terms of $\mathcal{N}(\cdot)$ functions, but...
- No closed-form solution for the optimal $\alpha^*$ can be obtained in general.
- Asymptotic expansions can be derived in the limits of small & large $(K/S_0)$:

$$\alpha^*(K/S_0 \searrow 0\ ) \sim \sigma^2\left[1 + e^{-(\tilde{\mu}^{\mathbf{Q}}+\sigma^2/2)T}\left(\frac{K}{S_0}\right) + (2+\sigma^2 T - e^{\sigma^2 T})e^{-(2\tilde{\mu}^{\mathbf{Q}}+\sigma^2)T}\left(\frac{K}{S_0}\right)^2 + ...\right]$$

$$\alpha^*(K/S_0 \nearrow \infty) \sim \frac{\ln(K/S_0)}{T} - \tilde{\mu}^{\mathbf{Q}} + \sigma^2\left[\frac{3}{2\ln(K/S_0)} + \frac{3(\tilde{\mu}^{\mathbf{Q}}+\sigma^2/2)T}{2\ln^2(K/S_0)} + ...\right]$$

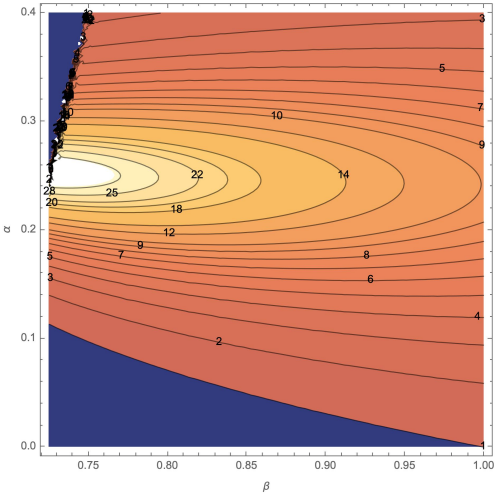



- In simulation, optimization over small sub-sample may be best strategy

# 4.4. Appendix: A Glance at Importance Sampling (6)

- $\beta > 1/\sqrt{2}$ required for $2^{nd}$ moment convergence, $\beta > \sqrt{1-1/n}$ for $n^{th}$ moment
- (Variance ratio for $\{\alpha = 0, \beta = 1\}$ to $\{\alpha, \beta\}$) vs. $\{\alpha, \beta\}$ for $K = 100$:




- Potential for substantial improvement by optimizing $\beta$... but not without hazard!