Advanced Monte Carlo

Fabio Cannizzo

Multiple Correlated State Variables

- Consider the state variables X_t and Y_t following respectively a OH and an ABM processes, and driven by correlated Brownian motions.

$$dZ_t = -\alpha Z_t dt + dW_t^Z \qquad dW_t^Z dW_t^Y = \rho dt$$
$$dY_t = dW_t^Y$$

- Both variables are Gaussian
- In order to diffuse both variables simultaneously, we need to generate Gaussian variables with the appropriate covariance structure

Multiple Correlated State Variables

- If we use some discretization scheme, it is straightforward. For instance, for Euler:

$$\begin{aligned} Z_{t+\Delta t} &= (1-\alpha\,\Delta t)\,Z_t + \Delta W_t^Z \\ \Delta Y_t &= \Delta W_t^Y \end{aligned} \qquad \begin{bmatrix} \Delta W_t^Z \\ \Delta W_t^Y \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} \Delta t & \rho\,\Delta t \\ \rho\,\Delta t & \Delta t \end{bmatrix} \right)$$

- or, working with random variables with unitary variance (which is equivalent):

$$\begin{aligned} Z_{t+\Delta t} &= (1-\alpha\,\Delta t)\,Z_t + \sqrt{\Delta t}\,\varepsilon_t^Z \\ \Delta Y_t &= \sqrt{\Delta t}\,\varepsilon_t^Y \end{aligned} \qquad \begin{bmatrix} \varepsilon_t^Z \\ \varepsilon_t^Y \end{bmatrix} \sim N\left(\begin{bmatrix} 0 \\ 0 \end{bmatrix}, \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix} \right)$$

Multiple Correlated State Variables

- Consider now the exact solution:

$$Z_{t+\Delta t} = Z_t e^{-\alpha \Delta t} + \underbrace{e^{-\alpha \Delta t} \int_t^{t+\Delta t} e^{-\alpha (t-u)} dW_u^Z}_{H_t^Z}$$

$$Y_{t+\Delta t} = Y_t + \underbrace{\int_t^{t+\Delta t} dW_u^Y}_{H_t^Y}$$

$$H_t \sim N(0, \Sigma),$$

$$\ldots \quad Cov\left[H_t^Z, H_t^Y\right] = e^{-\alpha \Delta t} Cov\left[\int_t^{t+\Delta t} e^{-\alpha (t-u)} dW_u^Z, \int_t^{t+\Delta t} dW_u^Y\right] = \rho e^{-\alpha \Delta t} \int_t^{t+\Delta t} e^{-\alpha (t-u)} du$$

$$\Sigma = \begin{bmatrix} Var\left[H_t^Z\right] & Cov\left[H_t^Z, H_t^Y\right] \\ Cov\left[H_t^Z, H_t^Y\right] & Var\left[H_t^Y\right] \end{bmatrix} = \begin{bmatrix} \frac{1^2}{2\alpha}\left(1 - e^{-2\alpha \Delta t}\right) & \rho \frac{1}{\alpha}\left(1 - e^{-\alpha \Delta t}\right) \\ \rho \frac{1}{\alpha}\left(1 - e^{-\alpha \Delta t}\right) & \Delta t \end{bmatrix}$$

Multiple Correlated State Variables

- or, working with random variables with unitary variance (which is equivalent):

$$Z_{t+\Delta t} = Z_t e^{-\alpha \Delta t} + \sqrt{\frac{1^2}{2\alpha}\left(1 - e^{-2\alpha \Delta t}\right)}\,\varepsilon_t^Z \qquad \varepsilon_t \sim N\left(0, \Sigma\right)$$

$$Y_{t+\Delta t} = Y_t + \sqrt{\Delta t}\,\varepsilon_t^Y$$

$$\Sigma = \begin{bmatrix} 1 & Corr\left[H_t^Z, H_t^Y\right] \\ Corr\left[H_t^Z, H_t^Y\right] & 1 \end{bmatrix} = \begin{bmatrix} 1 & \rho\sqrt{\frac{2\left(1 - e^{-\alpha \Delta t}\right)}{\kappa \Delta t\left(1 + e^{-\alpha \Delta t}\right)}} \\ \rho\sqrt{\frac{2\left(1 - e^{-\alpha \Delta t}\right)}{\kappa \Delta t\left(1 + e^{-\alpha \Delta t}\right)}} & 1 \end{bmatrix}$$

Achieving Market Consistency

- Suppose we want to make a spot model consistent with some forward curve and/or some implied volatility term structure. We need to manipulate parameters at each step
- Example: consider a GBM with deterministic volatility. The SDE is written:

$$\frac{dS_t}{S_t} = \mu(t)\,dt + \sigma\,dW_t, \qquad dW_t \sim N\left(0, \sqrt{dt}\right)$$

- Note here the explicit dependence of μ on time
- We want to compute the drift $\mu(t)$ which makes the process consistent with the bizarre forward curve

$$E_0\left[S_t\right] = S_0\left[1 + \frac{3}{5}\sin\left(\frac{t}{2}\pi\right)\right]$$

Achieving Market Consistency

- The SDE has the solution in [0,t]

$$S_t = S_0 \exp\left[\int_0^t \mu(s)\,ds - \frac{\sigma^2}{2}t + \sigma W_t\right]$$

- and the expectation is:

$$E_0\left[S_t\right] = S_0 \exp\left[\int_0^t \mu(s)\,ds\right]$$

- we conclude that

$$S_0\left[1+\frac{3}{5}\sin\left(\frac{t}{2}\pi\right)\right] = S_0 \exp\left[\int_0^t \mu(s)\,ds\right] \quad \Rightarrow \quad \mu(t) = \frac{d}{dt}\ln\left[1+\frac{3}{5}\sin\left(\frac{t}{2}\pi\right)\right]$$

Achieving Market Consistency

- From a discretization point of view, we are not interested in the exact shape of $\mu(t)$
- We only care that its integrated value yields the correct probabilistic properties at each discretization step
- This means that we can focus on computing the value of the integrals over each time step, rather than in determining the exact function $\mu(t)$

Achieving Market Consistency

- The SDE has the solution in [t, t+Δt]

$$S_{t+\Delta t} = S_t \exp\left[\int_t^{t+\Delta t} \mu(s)\,ds - \frac{\sigma^2}{2}\Delta t + \sigma\sqrt{\Delta t}\,\varepsilon_t\right]$$

- Thanks to the mean value theorem, the integral term can be expressed as:

$$\int_t^{t+\Delta t} \mu(s)ds = \mu(\xi)\,\Delta t = \mu_t\,\Delta t$$

- where ξ is some point in the interval [t, t+Δt]

Achieving Market Consistency

- Assuming for simplicity that μ is constant over a time step Δt

$$E_0\left[S_t\right]=S_0\exp\left[\int_0^t \mu(s)\,ds\right]=S_0\exp\left[\sum_{i=0}^{T/\Delta t-1}\int_{i\Delta t}^{(i+1)\Delta t}\mu(s)\,ds\right]=S_0\exp\left[\sum_{i=0}^{T/\Delta t-1}\mu_i\,\Delta t\right]$$

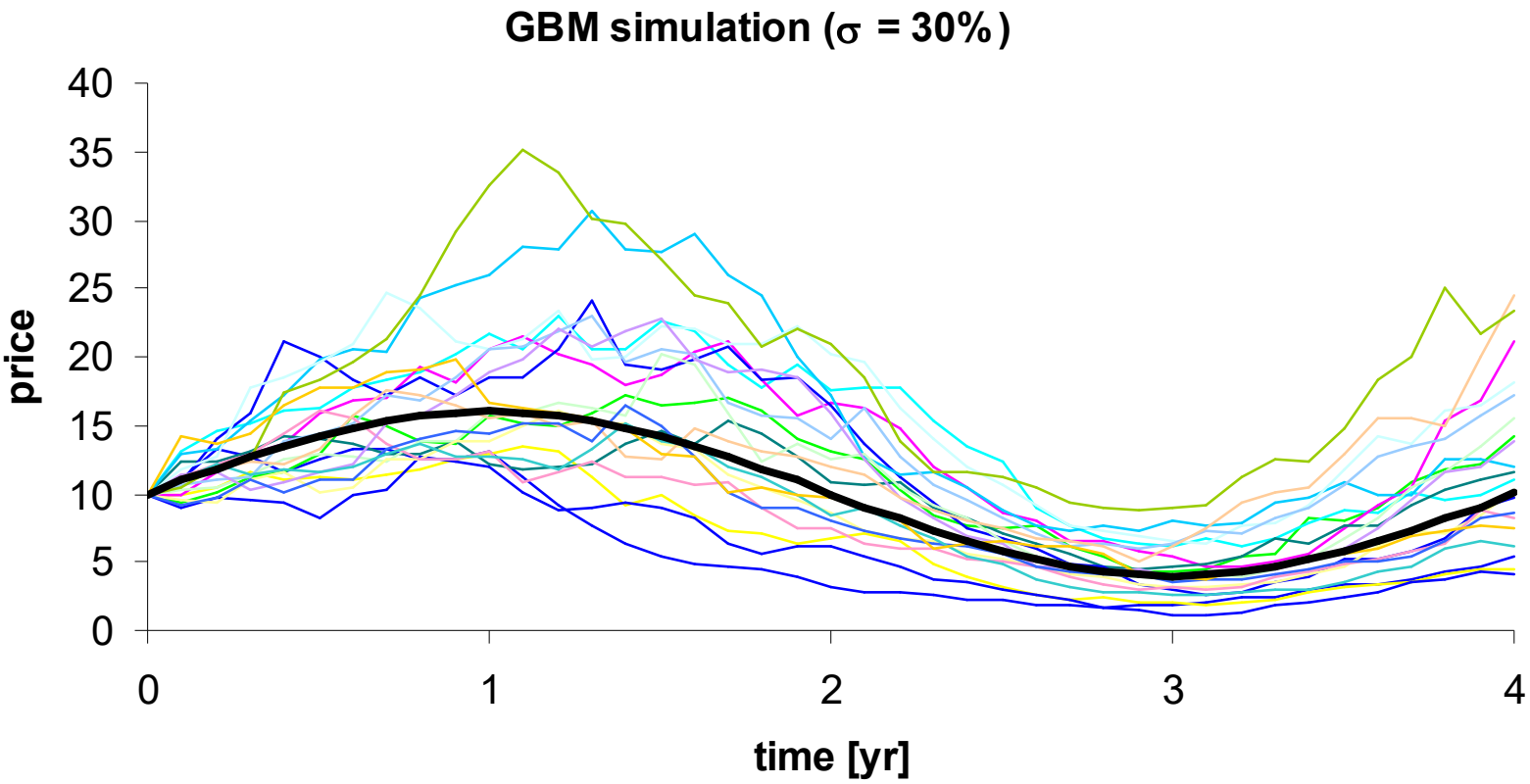
- Therefore we can solve for the unknowns μ_t:

$$\begin{cases} E_0\left[S_{\Delta t}\right]=S_0e^{\mu_0\Delta t} \\ E_0\left[S_{2\Delta t}\right]=S_0e^{(\mu_0+\mu_1)\Delta t} \\ \cdots \end{cases} \Rightarrow \begin{cases} e^{\mu_0\Delta t}=E_0\left[S_{\Delta t}\right]/S_0 \\ e^{\mu_1\Delta t}=E_0\left[S_{2\Delta t}\right]/E_0\left[S_{\Delta t}\right] \\ \cdots \end{cases}$$

- and the MC diffusion equation becomes:

$$S_{t+\Delta t}=S_t\left(\frac{E_0\left[S_{t+\Delta t}\right]}{E_0\left[S_t\right]}\right)\exp\left[-\frac{\sigma^2}{2}\Delta t+\sigma\sqrt{\Delta t}\,\varepsilon_t\right]$$

Achieving Market Consistency



Achieving Market Consistency

- This technique is called **boot-strapping**
- It answers the question: what could a constant value for μ be in the interval $[t_i, t_{i+1}]$, if I know the values of μ at any time before t_i and I know that the process must match some properties at time t_{i+1}?
- Analogous tricks can be used to match volatility term structures or discount factors
- The same ideas can also be used with other models, e.g mean reversion, and other numerical frameworks, e.g. trees

Achieving Market Consistency
Bootstrapping example with volatilities

- We observe that the implied volatility for 1M, 2M and 3M options is σ_{1M}, σ_{2M} and σ_{3M} (for simplicity, let's assume there is no smile)
- What does *implied volatility* mean? The implied volatility is the number that fed into the Black Scholes formula yields a given price observed in the market, i.e.

  C_{1M}=BS(F=F_{1M}, s=σ_{1M}, K, T=1M)
- But what is the meaning of s which goes into the BS formula? Note that the BS formula uses s and T always together as s*sqtr(T), so we can say that the inputs to the B&S formula is s = σ sqtr(T):

  C_{1M}=BS(F=F_{1M}, s = σ_{1M} sqrt(1M), K)

Achieving Market Consistency
Bootstrapping example with volatilities

- What is σ sqtr(T)?
- Remember the derivation of the B&S formula in the previous lecture pack:

$$V = e^{-rT} E\left[\max\left(S_T - K, 0\right)\right]$$

$$\text{let } X = \ln S_T$$

$$V = e^{-rT} E\left[\max\left(e^X - K, 0\right)\right] = e^{-rT} \int_{-\infty}^{+\infty} \max\left(e^X - K, 0\right) p(X)\, dX$$

$$= \ldots$$

$$= e^{-rT}\left[S_0 e^{rT} N(d_1) - K\, N\left(d_2\right)\right], \quad d_{1,2} = \frac{\ln\left(\frac{S_0 e^{rT}}{K}\right)}{\sigma\sqrt{T}} \pm \frac{1}{2}\sigma\sqrt{T}$$

Achieving Market Consistency
Bootstrapping example with volatilities

- More generically, with reference to the properties of the lognormal probability distribution of S_T, the B&S formula can be rewritten as:

$$V = e^{-rT}\left[E\left[S_T\right]N\left(d_1\right) - K\,N\left(d_2\right)\right], \quad d_{1,2} = \frac{\ln\left(\frac{E\left[S_T\right]}{K}\right)}{\sqrt{Var\left[\ln S_T\right]}} \pm \frac{1}{2}\sqrt{Var\left[\ln S_T\right]}$$

Achieving Market Consistency
Bootstrapping example with volatilities

- From the integral we can see clearly that σ sqtr(T) comes from the factor p(X). It is the standard deviation of $\ln(S_T)$ under the B&S assumption, i.e. assuming that *S follows a GBM with* constant volatility σ
- This means that

$$\begin{cases} Var\left[\ln S_{1M}\right] = \sigma_{1M}^2 \left(1/12\right) \\ Var\left[\ln S_{2M}\right] = \sigma_{2M}^2 \left(2/12\right) \\ Var\left[\ln S_{3M}\right] = \sigma_{3M}^2 \left(3/12\right) \end{cases}$$

- But this is contradicting BS assumptions, because the volatility is not constant!
- B&S is just a common convention to exchange information amongst traders: every equation represent a different B&S model, with different parameters

Achieving Market Consistency
Bootstrapping example with volatilities

- This does not work for pricing, where we need a common model we can diffuse
- A simplistic approach is to assume that our GBM has time dependent volatility

$$\frac{dS_t}{S_t} = rdt + \sigma(t)dW_t$$

- The derivation of the B&S formula still holds, but the computation of $Var[\ln S_T]$ change

Achieving Market Consistency
Bootstrapping example with volatilities

- When we introduce in the mode time-dependent volatility the formula for the variance becomes:

$$\frac{dS_t}{S_t} = rdt + \sigma(t)\,dW_t$$

$$(\text{by Ito's lemma}): \quad d\ln S_t = \left(r - \frac{1}{2}\sigma(t)^2\right)dt + \sigma(t)\,dW_t$$

$$(\text{integrating}): \quad \int_0^t d\ln S_u = \int_0^t \left(r - \frac{1}{2}\sigma(u)^2\right)du + \int_0^t \sigma(u)\,dW_u$$

$$Var[\ln S_t] = Var\left[\int_0^t \sigma(u)\,dW_u\right] = \int_0^t \sigma(u)^2\,du$$

(obviously this degenerates to $\sigma^2 T$ if σ is constant)

Achieving Market Consistency
Bootstrapping example with volatilities

- Using the formula obtained for the variance in the previous equations gives us some conditions for the unknown function $\sigma(t)$:

$$\begin{cases} \int_0^{1/12} \sigma(u)^2\,du = \sigma_{1M}^2(1/12) \\ \int_0^{2/12} \sigma(u)^2\,du = \sigma_{2M}^2(2/12) \\ \int_0^{3/12} \sigma(u)^2\,du = \sigma_{3M}^2(3/12) \end{cases} \Rightarrow \begin{cases} \int_0^{1/12} \sigma(u)^2\,du = \sigma_{1M}^2(1/12) \\ \int_{1/12}^{2/12} \sigma(u)^2\,du = \sigma_{2M}^2(2/12) - \sigma_{1M}^2(1/12) \\ \int_{2/12}^{3/12} \sigma(u)^2\,du = \sigma_{3M}^2(3/12) - \sigma_{2M}^2(2/12) \end{cases}$$

- We can assume any functional shape we want for s(t). A common simple assumption is to assume it is step-wise constant: s0,1 in 0-1M, s1,2 in 1M-2M, s2,3 in 2M-3M, and the system above is trivially solved
- An alternative assumption would be to assume s(t) is stepwise linear

Implied Volatility vs Local Volatility

$$\int_0^{1/12} \sigma(u)^2\,du = \sigma_{1M}^2(1/12)$$

Local vol

B&S Implied Vol

- B&S implied volatilities feed in the B&S formula. They represent total variance over a period.
- Local volatilities are instantaneous volatilities which appears in the diffusion model.
- Implied volatility are obtained integrating over local volatilities
- Note the no-arbitrage constraint, implied variances must be increasing:

$$0 \leq \int_0^{\frac{2}{12}} \sigma^2(u)\,du = \sigma_{2M}^2 \frac{2}{12} - \sigma_{1M}^2 \frac{1}{12}$$

Variance Reduction

- Many attempts have been made to try and improve Monte Carlo convergence rate
- The most common techniques are:
  - Antithetic Variates
  - Control Variates
  - Path Adjustments for Moment Matching
  - Weighted Monte Carlo
  - Stratified Sampling
  - Importance Sampling
  - Quasi Monte Carlo

Antithetic Variates

- The MC basic estimator for the mean of a function of Gaussian variable ε has the form

$$V_n = \frac{1}{n}\sum_{i=1}^{n} f\left(\varepsilon_i\right), \qquad \varepsilon \sim N\left(0,1\right)$$

- and, because $f(\varepsilon_i)$ are i.i.d. by central limit theorem we know it has distribution

$$V_n \sim N\left(E\left[f\left(\varepsilon\right)\right], Var\left[f\left(\varepsilon\right)\right]/n\right)$$

- Because ε has symmetric distribution this is completely equivalent to

$$V_n = \frac{1}{N}\sum_{i=1}^{N} f\left(-\varepsilon_i\right)$$

Antithetic Variates

- This suggests we could change the estimator to

$$\widetilde{V}_n = \frac{1}{N}\left[\sum_{i=1}^{N} \frac{f(\varepsilon_i)+f(-\varepsilon_i)}{2}\right]$$

- We can see that the estimator is still unbiased (i.e. its expectation is the true mean:

$$E\left[\frac{f(\varepsilon)+f(-\varepsilon)}{2}\right] = \frac{1}{2}\left[E\left[f(\varepsilon)\right]+E\left[f(-\varepsilon)\right]\right] = E\left[f(\varepsilon)\right]$$

- because all terms in the sum are still i.i.d, the central limit theorem still holds and the estimator converges to:

$$\widetilde{V}_n \sim N\left(E\left[f(\varepsilon)\right], Var\left[\frac{f(\varepsilon)+f(-\varepsilon)}{2}\right]/n\right)$$

Antithetic Variates

- Assuming most of the hard computational work happens in the function $f(\varepsilon)$, and that the cost of computing extra PRNs is negligible, the new estimator with n realizations has roughly the same computational cost as the basic estimator with $2n$ realizations
- We comparing the convergence speed of the 2 estimators:

$$Var\left[\tilde{V}_n\right] < Var\left[V_{2n}\right] \quad \Rightarrow \quad \frac{Var\left[\frac{f(\varepsilon)+f(-\varepsilon)}{2}\right]}{n} < \frac{Var\left[f(\varepsilon)\right]}{2n}$$

$$Var\left[\frac{f(\varepsilon)+f(-\varepsilon)}{2}\right] < \frac{1}{2} Var\left[f(\varepsilon)\right]$$

Antithetic Variates

$$Var\left[\frac{f(\varepsilon)+f(-\varepsilon)}{2}\right]=\frac{1}{4}\left(Var[f(\varepsilon)]+2Cov[f(\varepsilon),f(-\varepsilon)]+Var[f(-\varepsilon)]\right)$$

$$=\frac{1}{2}\left(Var[f(\varepsilon)]+Cov[f(\varepsilon),f(-\varepsilon)]\right)$$

$$Var\left[\frac{f(\varepsilon)+f(-\varepsilon)}{2}\right]<\frac{1}{2}Var[f(\varepsilon)] \quad \Rightarrow \quad Cov[f(\varepsilon),f(-\varepsilon)]<0$$

- For the inequality to hold the sign of the covariance term must be negative
- Unfortunately this is not always the case…

Antithetic Variates

- We show two extreme cases:
  - linear: $f(\varepsilon) = a\,\varepsilon + b$
    - $\mathrm{Cov}[f(\varepsilon), f(-\varepsilon)] = -\mathrm{Var}[f(\varepsilon)]$
    - and the variance of the estimator drops to zero (i.e. the integration is exact)
  - symmetric: $f(\varepsilon) = f(-\varepsilon)$, e.g. $f(\varepsilon) = |\,\varepsilon\,|$
    - $\mathrm{Cov}[f(\varepsilon), f(-\varepsilon)] = \mathrm{Var}[f(\varepsilon)]$
    - and the variance of the estimator is double as before

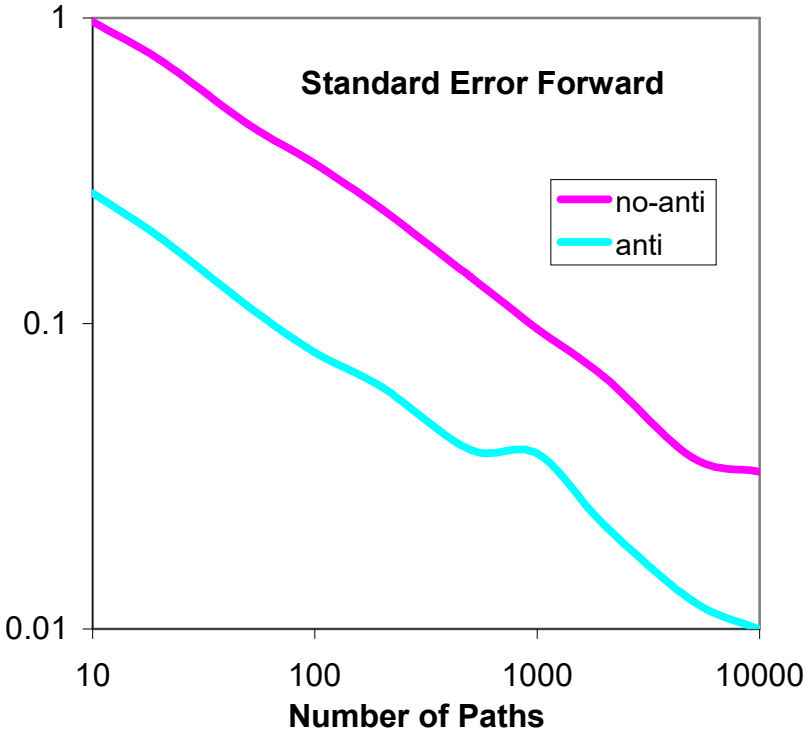
Antithetic Variates Example

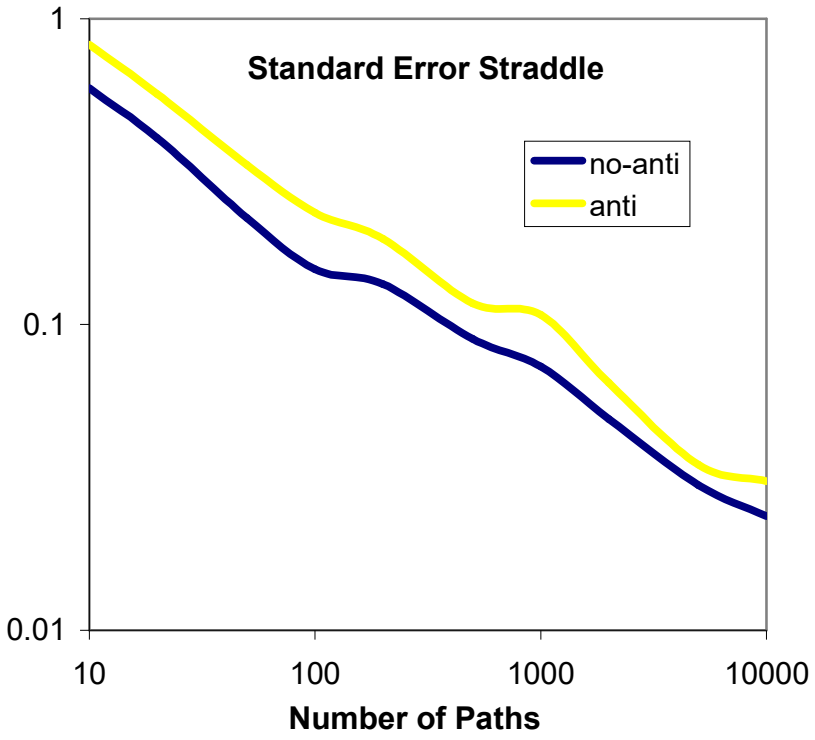
Let's consider the **undiscounted** values of a straddle (symmetric) and a forward (anti-symmetric) with a strike chosen so that we maximize the symmetry in the payoffs

$$dS_t = S_t(\mu\, dt + \sigma\, dW_t), \qquad S_0 = 10, \quad K = S_0 \exp\left(\left(\mu - r - \frac{\sigma^2}{2}\right)T\right)$$

SYMMETRIC	ANTISYMMETRIC
$Payoff = \lvert S_T - K \rvert$	$Payoff = S_T - K$
$V = K\, E\left[\ \lvert e^{\sigma\sqrt{T}\varepsilon} - 1 \rvert\ \right]$	$V = K\left(e^{\sigma\sqrt{T}\varepsilon} - 1\right)$
$\hat{V}_n = \frac{K}{n}\sum_{i=1}^{n} \lvert e^{\sigma\sqrt{T}\varepsilon} - 1 \rvert$	$\hat{V}_n = \frac{K}{n}\sum_{i=1}^{n}\left(e^{\sigma\sqrt{T}\varepsilon} - 1\right)$

Antithetic Variates Example





- But forward and straddle are not really anti-symmetric and symmetric with respect to Normal PRN, because of the exponential function.

Antithetic Variates

- In multi-dimension (multiple risk factors or multiple time steps) the easiest extension of the technique is simply to flip the sign of the entire Gaussian vector of **uncorrelated** N(0,1) needed to generate a path
- This is equivalent to creating a mirror log-path in the hyperplane
- In reality there are other possible ways to do antithetic in multi-dimension (see **antithetic sampling**)
- The theoretical hypothesis for the technique to improve convergence is that: **the payoff must be monotonic with respect to all element of the random vector** (which is rarely true and difficult to verify for complex payoffs)
- Some payoff have both a symmetric and an anti-symmetric component

Antithetic Variates

- In summary:
  - It always save some computational cost for the generation of PRNs, but this might be marginal
  - It aims at reducing errors associated with anti-symmetric components of the payoff function (if there is any)
  - It can add value or be slightly detrimental, depending on shape of the payoff function
  - It is difficult to know in advance if it will be useful or not
  - It is super-easy to implement
- If we do not know anything about the payoff function, given in same cases it can add lot of value and in same cases it is slightly detrimental, usually it is worth using it
- Convergence order is still $O(N^{-½})$, but with a different constant of proportionality in front.
- How to estimate the standard error, as in previous chart?

Antithetic - Implementation

```
function [res, stdErr] = euroCallMCVecAnti( rf, yield, vol, T, spot, strike, nSim )

    % seed generator with an arbitrary seed and use Mersenne Twister
    rng(1347,'twister');

    % nSim must be even
    if (mod(nSim,2)), nSim = nSim +1; end
    nHalf = nSim / 2;

    % for efficiency generate all gaussian random number upfront
    rnds = randn( nHalf, 1 );
    rnds = [rnds; -rnds];  % the 2nd half are antithetic of 1st half

    logStdDev = vol * sqrt( T );
    factor = spot * exp( ( (rf-yield) - 0.5 * vol * vol ) * T );
    k = strike / factor;

    % vectorial operations
    s = exp( logStdDev * rnds);
    payoff = max( s - k, 0.0 );
    pv = exp( -rf * T ) * factor * payoff; % do not aggregate yet, will use for stdev

    % aggregate results
    res = mean(pv);
    % note the different formula standard deviation, because we must keep
    % into account that we are using antitetic, hence the random numbers
    % are not all independent (i.i.d). However, pairs of random numbers
    % and the correspondent antithetic are independent (i.i.d)
    antipvs = (pv(1:nHalf) + pv(1+nHalf:end)) / 2; % aggregate antithetic pairs
    variance = var(antipvs);
    stdErr = sqrt(variance / nHalf); % we have nHalf antitethic samples
```

Empirical Estimation of Standard Error

- When using variance reduction techniques (e.g. antithetic) the formula might get complicate
- An alternative is to split the N paths into M block of B=N/M paths each
- NOTE: possible variance reduction techniques (e.g. antithetic) need to be contained within the blocks
- We compute the MC estimator V_b over each block of size b. This is distributed as N(E[V],x). Note we do not know x, as it depends on the variance reduction technique we are using.
- We estimate the variance of V_n as the empirical variance of V_b rescaled by m, i.e.:

$$E[V] \approx E[V_b] = \frac{1}{m}\sum_{i=1}^{m} V_{b,i}$$

$$Var[V_n] = \frac{1}{m} Var[V_b] \quad \Rightarrow \quad \mathrm{SE}[V_n] = \sqrt{\frac{\frac{1}{m-1}\sum_{i=1}^{m}\left(V_{b,i} - E[V_b]\right)^2}{m}}$$

Control Variates

- Suppose we want to approximate E[$f(\varepsilon)$] via Monte Carlo estimation and **we know exactly** the value to which the Monte carlo estimator converge for another related function E[$g(\varepsilon)$] (inclusive of possible integration errors)
- We can use this information to improve the convergence of the estimator for E[$f(\varepsilon)$] by defining a new estimator:

$$\text{let} \quad \tilde{f} = MC \text{ estimator for } E[f]$$

$$\tilde{g} = MC \text{ estimator for } E[g] \quad \text{(computed in the same simulation)}$$

$$\hat{f} = \tilde{f} - \lambda(\tilde{g} - E[g])$$

$$E[\hat{f}] = E[\tilde{f} - \lambda(\tilde{g} - E[g])] = E[\tilde{f}] - \lambda \underbrace{(E[\tilde{g}] - E[g])}_{=0} = E[f] \quad \textbf{(unbiased)}$$

Control Variates

- As shown, the new estimator in unbiased and its variance is

$$Var\left[\hat{f}\right] = Var\left[\tilde{f} - \lambda(\tilde{g} - E[g])\right]$$
$$= Var\left[\tilde{f}\right] + \lambda^2 Var\left[\tilde{g} - E[g]\right] - 2\lambda Cov\left[\tilde{f}, \tilde{g} - E[g]\right]$$
$$= \frac{Var[f]}{n} + \lambda^2 \frac{Var[g]}{n} - 2\lambda \frac{Cov[f,g]}{n}$$

- which is minimized for

$$\lambda = \frac{Cov[f,g]}{Var[g]}$$

$$Var\left[\hat{f}\right] = \frac{1}{n}\left\{Var[f] + \left(\frac{Cov[f,g]}{Var[g]}\right)^2 Var[g] - 2\frac{Cov[f,g]}{Var[g]} Cov[f,g]\right\}$$
$$= \frac{Var[f]}{n}\left\{1 - \frac{Cov[f,g]^2}{Var[f]Var[g]}\right\} = \frac{Var[f]}{n}\left\{1 - \rho^2\right\}$$

Control Variates

- The higher the correlation between the two functions, the more effective the technique (decent when $|\rho| > 80\%$)
- The covariance can be estimated from the sample of simulated data of f(x) and g(x)
- The technique can be extended to multiple control variates (multiple regressors)
- Examples:
  - For a European call option we could use the forward
  - For a European payoff we could use a combination of put and call at different strikes
  - For an arithmetic average option we could use a geometric average option (the value of a geometric average option is known analytically because the product of lognormal random variables is lognormal, and the dimension of the integral can be reduced)

Control Variates

- This technique is very powerful and can lead to significant improvement in convergence speed, but it has very limited applicability because:
  - requires a portfolio *g* **highly correlated** with the payoff we want to price
  - we must know **exactly** the value to which the Monte Carlo estimator converge (not some theoretical value), otherwise the estimator is **biased**
  - because volatilities are not constant (price distributions are not lognormal) and pricing models are often approximated via discretization schemes, it is extremely **unlikely to know exactly** the value to which the Monte Carlo estimator would converge for E[*g*]
  - Estimation of the covariance via least squares could be tricky because of discontinuities in the payoffs function and collinearity of regressors

Control Variate - Implementation

```
% Monte Carlo with control variates and computation of standard error
% we use E[S] as a control variate for a call option
function [res, stdErr] = euroCallMCVecCV( rf, yield, vol, T, spot, strike, nSim )
    % seed generator and compute gaussian
    rng(1347,'twister');
    rnds = randn( nSim, 1 );

    % useful constants
    logStdDev = vol * sqrt( T );
    drift = ((rf-yield) - 0.5 * vol * vol ) * T;
    fwd = spot * exp((rf-yield) * T); % E[S] theoretical
    df = exp(-rf*T);  % discount factor

    % vectorial operations
    s = spot .* exp( drift + logStdDev * rnds);
    payoff = max( s - strike, 0.0 );  % call option
    cvpayoff = s;                     % E[S] estimated
    c = cov([payoff, cvpayoff]);      % covariance matrix
    lambda = c(2,1) / c(2,2);         % CV coefficient
    cvadjusted = payoff - lambda * (cvpayoff - fwd); % CV adjustment

    % aggregate results
    res = df * mean(cvadjusted);
    stdErr = df * sqrt(var(cvadjusted) / nSim);
```

Stratified Sampling

- The expectation of a function of stochastic variable X can be expressed as:

$$E\left[f(x)\right]=\sum_{j=1}^{k} E\left[f(X)\,|\,X\in A_j\right]p\left(X\in A_j\right)$$

- where A_j are **disjoint** set of values for the stochastic variable X, constructed in such a way that they cover the entire domain of the variable X.
  - X is called the **stratification variable**
  - the set A_j are referred to as **strata**
- Note that the sum of probabilities $p(X\in A_j)$ of each strata must add up to one by construction, because they are disjoint and cover all the domain of X

Stratified Sampling

- If we split the total number of paths n in k arbitrary subsets n_j which we allocate to the strata k, then we have a new MC estimator

$$\hat{V}_n = \sum_{j=1}^{k} p_j \frac{1}{n_j} \sum_{i=1}^{n_j} f\left(X_{i,j}\right)$$

where $X_{i,j}$ belongs to strata j

- This could converge better o worse than the standard estimator depending on how
  - we choose the variable X (could be Uniform PRN or Gaussian PRN or the price itself, or any other function)
  - we chose the strata
  - we allocate the paths to each strata (one popular criteria to allocate paths to each strata is **proportional allocation**)

Stratified Sampling

- The estimator is still unbiased and it is possible to proof that, because the central limit theorem applies to each strata, and the final estimator is a sum of normal variables, it is also normal and converges to the true value
- This method is very powerful, but:
  - complicate to implement
  - difficult to generalize (every implementation is tightly coupled with a specific payoff and model)
  - difficult to define the strata in multiple dimension and to generate samples from multi dimensional conditional distributions
- Here we merely provide a simple example to illustrate the idea

Stratified Sampling

- Consider a call spread K1, K2 in the Black Sholes world
- On the left of K1 the payoff is always zero, while on the right of K2 the payoff is always K2-K1. This suggest that the interesting region where there is some variance is [K1,K2]
- K1 and K2 can be mapped to points in the Unif(0,1) distribution:

$$\text{payoff} = \max\left[S_T - K_1, 0\right] - \max\left[S_T - K_2, 0\right]$$

$$S_T(u_i) = K_i \quad \Rightarrow \quad S_0 \exp\left[\left(\mu - \frac{\sigma^2}{2}\right)T + \sigma\sqrt{T}\,N^{-1}(u_i)\right] = K_i, \qquad u \sim Unif(0,1)$$

$$\Rightarrow \quad u_i = N\left(\frac{1}{\sigma\sqrt{T}}\left[\ln\frac{K_i}{S_0} - \left(\mu - \frac{\sigma^2}{2}\right)T\right]\right)$$

Stratified Sampling

- The payoff offers us a natural subdivision in strata with respect to the uniform random variable *u*:

$$A_1 : u \in [0, u_1] \quad A_2 : u \in [u_1, u_2] \quad A_3 : u \in [u_2, 1]$$
$$p(A_1) = u_1 \quad p(A_2) = u_2 - u_1 \quad p(A_3) = 1 - u_2$$

- We can assign a very small number of paths to A1, because the payoff is zero (even just one), and also to A3 (even just one), because the payoff is constant (there is no variance in the regions).
- Out of n paths, let's say we assign 10% to A1, 80% to A2, and 10% to A3, i.e.

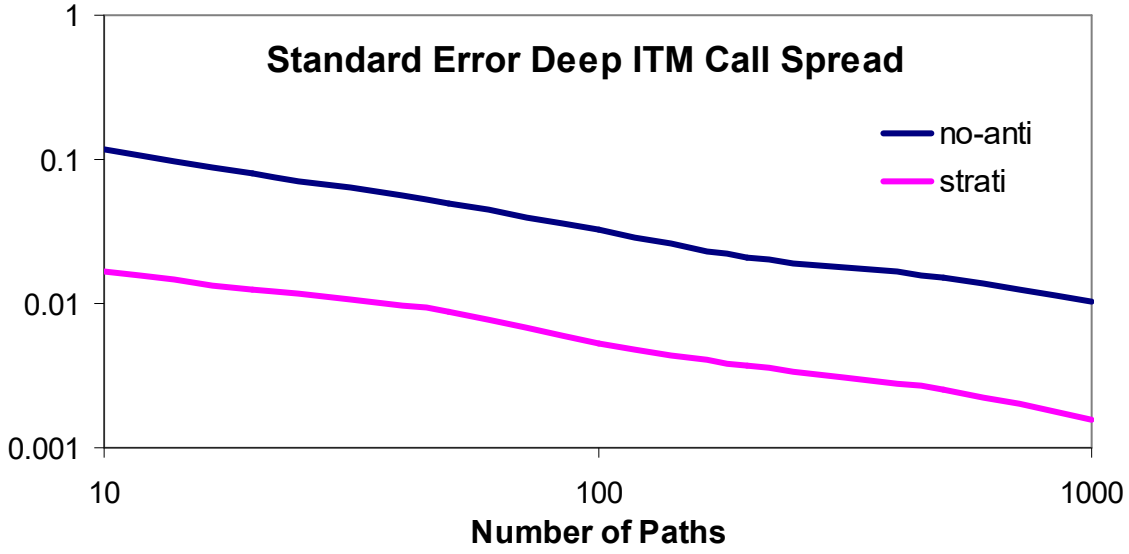
$$n_1 = 10\% n, \quad n_2 = 80\% n, \quad n_3 = 10\% n$$

Stratified Sampling

- The stratified sampling estimator is:

$$\hat{V}_n = \frac{u_1}{n_1}\sum_{i=1}^{n_1}\underbrace{f\left(u_{i,1}\right)}_{0} + \frac{u_2-u_1}{n_2}\sum_{i=1}^{n_2} f\left(u_{i,2}\right) + \frac{1-u_2}{n_3}\sum_{i=1}^{n_3}\underbrace{f\left(u_{i,3}\right)}_{K_2-K_1}$$

$$= \frac{u_2-u_1}{n_2}\sum_{i=1}^{n_2} f\left(u_{i,2}\right) + \left(1-u_2\right)\left(K_2-K_1\right)$$



rf	5%
yield	2%
vol	30%
T	1
Spot	10
K2	5
K2	7

Importance Sampling

- Importance sampling involves a change of probability measure. Instead of drawing X from a distribution with p.d.f. p(X), we draw it from a different distribution with p.d.f. q(X)

$$
\begin{aligned}
E^P\left[f(x)\right] &= \int f(x)\,p(x)\,dx \\
&= \int f(x)\frac{p(x)}{q(x)}\,q(x)\,dx = \int f(x)\,r(x)\,q(x)\,dx \\
&= E^Q\left[f(x)\,r(x)\right]
\end{aligned}
$$

where $r(x) = \dfrac{p(x)}{q(x)}$ is the Radon Nikodym derivative

Importance Sampling

- To improve convergence of the estimator, we would like the variance f(x)r(x) with respect to q(x) to be very small
- In the limit, if f(x)r(x) was constant the variance would be zero and one single path would be sufficient to achieve a perfect estimate
- The intuition is that we want r(x) to be small when f(x) is large and viceversa
- This implies that q(x)>p(x) (more paths) where f(x) is large and q(x)<p(x) (less paths) where f(x) is small

Importance Sampling

- This can be very useful to price payoff which depends on rare events, like deep OTM digital put options

$$\text{payoff} = 1_{S_T < K}, \quad \text{where} \quad K = 0.5\, S_0$$

$$S_T < K \quad \Rightarrow \quad \varepsilon < \bar{\varepsilon} = \frac{1}{\sigma\sqrt{T}}\left[\ln 0.5 - \left(\mu - \frac{\sigma^2}{2}\right)T\right], \qquad \varepsilon \sim N(0,1)$$

$$P\left(S_T < K\right) = P\left(\varepsilon < \bar{\varepsilon}\right) = N\left(\frac{1}{\sigma\sqrt{T}}\left[\ln 0.5 - \left(\mu - \frac{\sigma^2}{2}\right)T\right]\right) \qquad \left(\text{which could be very small}\right)$$

- Not many paths fall in that region and lot of computations go wasted!

Importance Sampling

- To make that rare event more likely we could either shift the normal distribution to the left, or increase its volatility (this last action would have an effect on both tails)
- Let's follow the first approach:

$$E^P\left[1_{S_T<K}\right]=\int_{-\infty}^{\bar{\varepsilon}} 1\, p(\varepsilon)\, d\varepsilon=\int_{-\infty}^{\bar{\varepsilon}} 1 \frac{1}{\sqrt{2\pi}}\exp\left[-\frac{\varepsilon^2}{2}\right] d\varepsilon$$

$$\text{let} \quad q(\varepsilon)=N(m,1), \qquad r(\varepsilon)=\frac{p(\varepsilon)}{q(\varepsilon)}=\frac{\frac{1}{\sqrt{2\pi}}\exp\left[-\frac{\varepsilon^2}{2}\right]}{\frac{1}{\sqrt{2\pi}}\exp\left[-\frac{(\varepsilon-m)^2}{2}\right]}=\exp\left[\frac{m^2-2m\varepsilon}{2}\right]$$

$$E^P\left[1_{S_T<K}\right]=\int_{-\infty}^{\bar{\varepsilon}} 1\exp\left[\frac{m^2-2m\varepsilon}{2}\right]\frac{1}{\sqrt{2\pi}}\exp\left[-\frac{(\varepsilon-m)^2}{2}\right] d\varepsilon = E^Q\left[1_{S_T<K}\, r(\varepsilon)\right]$$

Importance Sampling

- We choose *m* so that this produce a smaller variance

$$Var^P\left[1_{S_T<K}\right]=E^P\left[\left(1_{S_T<K}\right)^2\right]-\left(E^P\left[1_{S_T<K}\right]\right)^2=E^P\left[\left(1_{S_T<K}\right)^2\right]-V^2$$

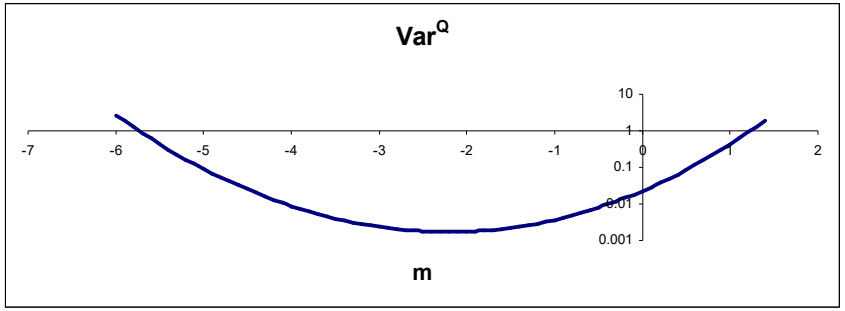
$$Var^Q\left[1_{S_T<K}r(\varepsilon)\right]=E^Q\left[\left(1_{S_T<K}r(\varepsilon)\right)^2\right]-\left(E^Q\left[1_{S_T<K}r(\varepsilon)\right]\right)^2=E^Q\left[\left(1_{S_T<K}r(\varepsilon)\right)^2\right]-V^2$$

$$Var^Q\left[1_{S_T<K}r(\varepsilon)\right]<Var^P\left[1_{S_T<K}\right] \quad \Rightarrow \quad E^Q\left[\left(1_{S_T<K}r(\varepsilon)\right)^2\right] ¿\, E^P\left[\left(1_{S_T<K}\right)^2\right]$$

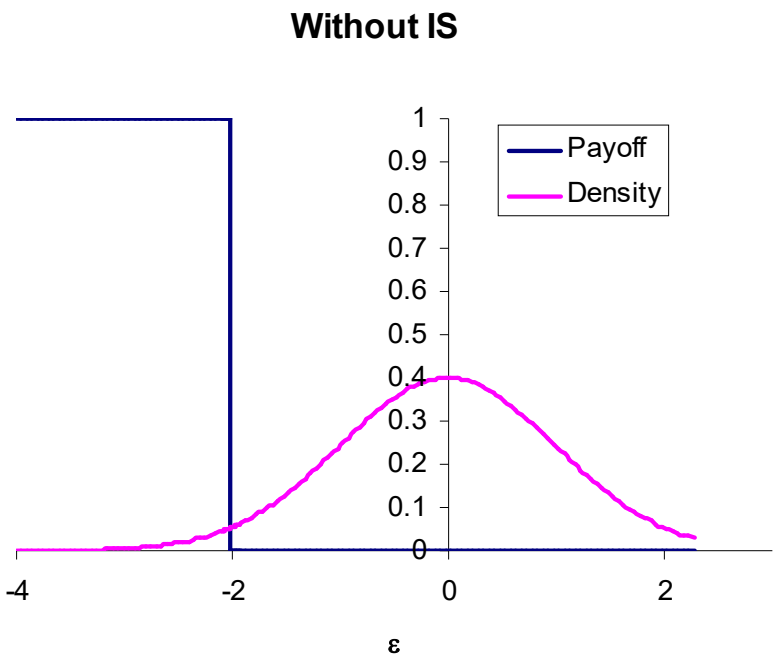
$$\int_{-\infty}^{\bar{\varepsilon}} 1\left(\exp\left[\frac{m^2-2m\varepsilon}{2}\right]\right)^2\frac{1}{\sqrt{2\pi}}\exp\left[-\frac{(\varepsilon-m)^2}{2}\right]d\varepsilon<\int_{-\infty}^{\bar{\varepsilon}} 1\frac{1}{\sqrt{2\pi}}\exp\left[-\frac{\varepsilon^2}{2}\right]d\varepsilon$$

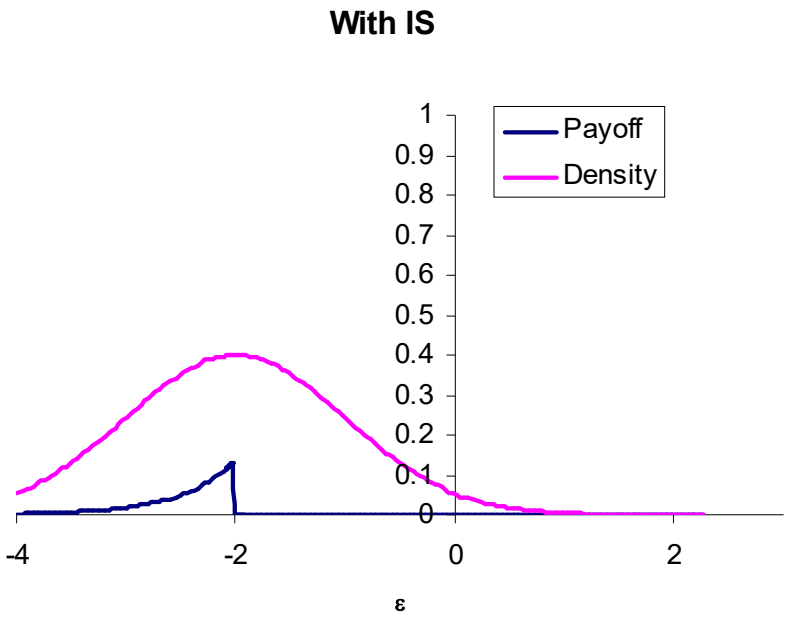
$$\exp\left[m^2\right]N(m+\bar{\varepsilon})<N(\bar{\varepsilon})$$

- if $m=0$ we have an equality
- Var^Q has a minimum point

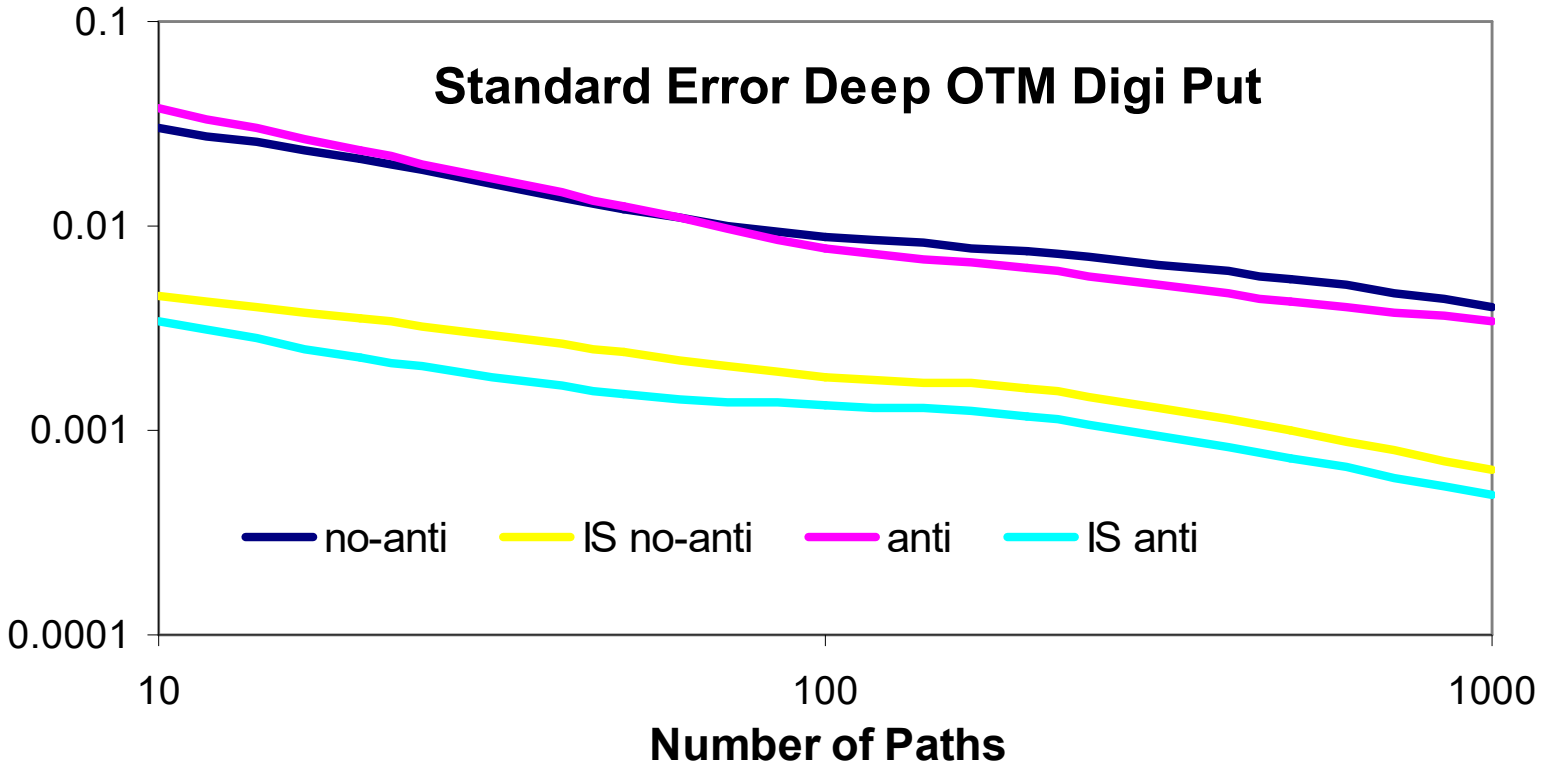


Importance Sampling





Importance Sampling



- Antitethic variates does not help without IS, despite the fact that hypothesis are verified, but it becomes helpful with IS
- That is because the payoff has a large symmetric component, and IS reduces the symmetric component in favor of the anti-symmetric one.
- 10 paths with IS achieve roughly same standard error as 500 paths without IS

Importance Sampling

- It can be very but:
  - Like stratified sampling this technique is usually strictly coupled with one particular payoff, i.e. it is not easily generalizable
  - It is difficult to implement and does not extend easily to multiple dimensions

Control Variates

- Summarizing
  - antithetic variates – generic and easy to implement but limited effectiveness
  - control variates – easy to implement and can be very effective but requires careful choice of control variate in each case, and knowledge of its converged MC estimator
  - stratified sampling - very useful but difficult to generalize
  - importance sampling – very useful for applications with rare events, but needs to be fine-tuned for each application
- Overall, a tradeoff between
  - simplicity and generality
  - efficiency and programming effort on the other

Empirical Estimation of Standard Error

- Based on the central limit theorem the the standard deviation of the MC estimator (**standard error**) can be estimated as $\sigma/n^{1/2}$
- When we manipulate the PRNs, so that they are no longer completely i.i.d., as in the case of the antithetic, this result no longer hold
- Often it is possible to derive new standard error estimators
- A brute force technique consists of assessing empirically the standard error, using the standard deviation sample estimator on results obtained in *M* **independent** simulations
- Note this is very expensive! Can be useful in design phase, but not in a production environment
- We illustrate the idea with an example

Empirical Estimation of Standard Error

- Consider the trivial integration problem: E[u], where u~Unif(0,1)
- We want to estimate the order of convergence of some MC estimator using *M*=30 independent simulations. We want to limit the analysis on convergence to a discrete set of number of paths: $N \in \{ 10, 20, 50, 100 \}$
- If we consider a simulation with 20 paths as the continuation of the simulation with 10 paths, then we can compute a full set of estimators V_n for every *N*, in one single run of the algorithm
- If then we run the next set of simulations without reinitializing the seed of the random number generator, we obtain a new independent sequence of PRNs (equivalent to **skip-ahead** the PRNs used for all previous simulation)

American Exercise

- With Monte Carlo is difficult to price derivatives which depends on American or Bermuda style exercise decisions
- Consider for example an American Put option on a non dividend paying stock which follows a GBM
- There is a function $f(X_t,t)$ such that if at any time $f(X_t,t)<0$ then it is convenient to exercise
- The function $f(X_t,t)$ is the **optimal exercise frontier**

American Exercise

- The optimal exercise frontier is not known in advance.
- When doing tree pricing we discovered on-the-fly at which nodes it was convenient to exercise during the execution of the algorithm
- At every node we were taking

  $V=\max\{ K-S, B [q V^{+} + (1-q) V^{-}] \}$

  $= \max\{$ exercise at this node

  , discounted value if we do not exercise $\}$
- This is natural with backward algorithms (dynamic programming), not with forward algorithms

American Exercise

- In MC, for any path, we do not know the value of the derivative contingent to not exercising, so we are not able to make this decision
- In theory, at every time step, for every MC path, I should launch a new MC simulation to compute the value of the derivative if we do not exercise at this time step
- The amount of computation grows exponentially with the time step and the problems becomes untreateable
- You can think about it as a *N*-nomial non recombining tree, where *N* is the number of paths. The order of growth is N^M, where *M* is the number of time steps

American Exercise

- A Bermudas problem, which has exercise decisions at every interval Δt, has formulation:

let $h(t, S_t)$ value from exercise in the state (t, S_t)

$V_t(S_t)$ value of the derivative in the state (t, S_t),

assuming not exercised before

$D(t)$ discount factor from time t to time $t+\Delta t$

at every time step we need to solve the problem:

$$V_t(S_t) = \max\left[h(t, S_t), D(t)\, E_t\left[V_{t+\Delta t}(S_{t+\Delta t}) \mid S_t\right]\right]$$

Longstaff Schwartz

- A technique developed by Longstaff and Schwartz is commonly used to solve this problem
- First it generates all paths, then it works **backward** in time assigning values to all paths
- The expectation of continuation at every time step is obtained via polynomial regression over the value of the derivative at the next time step

Longstaff Schwartz

1) Generate and store N paths

2) At the last time step T of every path, initialize using the payoff, like in a tree: $V_T\left(S_T\right)=h\left(T,S_T\right)$

3) Estimate the optimal exercise frontier at previous time step $t-\Delta t$ regressing the vector $V_t\left(S_t\right)$ against a polynomial basis of the state variable $S_{t-\Delta t}$ of degree P: $V_t\left(S_t\right)=\sum_{i=0}^{P}\alpha_i\left(S_{t-\Delta t}\right)^i+\varepsilon_t$

$$E\left[V_t\left(S_t\right)|S_t\right]\approx\sum_{i=0}^{P}\alpha_i\left(S_{t-\Delta t}\right)^i$$

4) Compute: $V_{t-\Delta t}\left(S_{t-\Delta t}\right)=\max\left[h\left(t-\Delta t,S_{t-\Delta t}\right),D\left(t-\Delta t\right)\sum_{i=0}^{P}\alpha_i\left(S_{t-\Delta t}\right)^i\right]$

4) If $t>0$ goto step 2

Longstaff Schwartz

- The polynomial basis used in step 3 could be replaced with other functions.
- Need to choose carefully state variables
- In the least squares things can go very wrong in presence discontinuities, for instance if there are lot of zeros (digitals)
- The technique generally **underestimate**, but with small samples it could overestimate due to introducing bias (perfect insight) (could use out of sample)
- Regression is expensive on a high number of paths (out of sample would help here as well)
- Greeks tend to be unstable. Possible mitigations are:
  - freeze frontier
  - freeze decision

Calculating Greeks

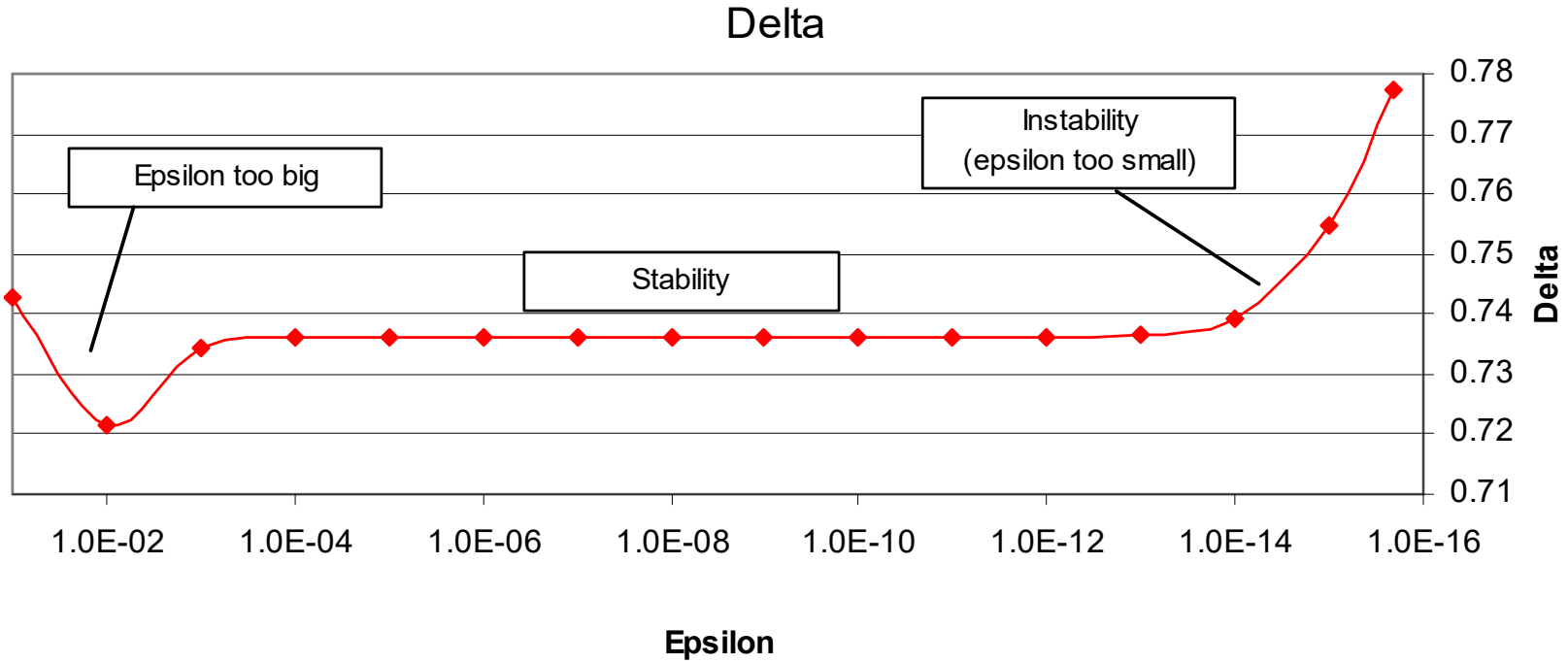
- Indirect methods
  - Require multiple runs of the MC pricing engine
  - Consequently they tend to be computationally expensive
  - Typical example: finite differences
- Direct methods
  - Done in a single run of the pricing engine
  - Usually faster than indirect methods
  - Not always applicable

Greeks – Finite Differences

- Be careful: ε (increment used for finite differences) needs to be small, but not as small as possible
  - ε too big will increase truncation error
  - ε too small will lead to numerical instability, as the propagation of the machine rounding error and Monte Carlo error will kick in, outweighing the benefit of having reduced ε
- There is no golden rule.
- Different greeks (delta, gamma, rho, vega, …) and/or different problems (type of derivative contract, dynamic of underlying, …) may require different values of ε.
- The number of paths also affects the range of allowable values for ε

Greeks – Finite Differences

- Example Delta of a European Call Option on an asset, whose price under the equivalent martingale measure follows a GBM with constant parameters.



- Note Gamma is *much* more unstable!

Greeks – Finite Differences

- Important: you need to **use the same RNs** when valuing $V(S^+)$ and $V(S^-)$
- Suppose the value of the derivative obtained trough MC is $V = V^* + Err$, where V^* is the true value, while Err is the approximation error of the algorithm and depends on the particular sequence of PRNs used.
- Intuitively, if we use the same random numbers, then we can assume Err is approximately the same for $V(S^+)$ and $V(S^-)$, therefore, when we take the difference, it will tend to cancel out

$$V(S^+) - V(S^-) = \left[V^{¿}(S^+) + Err^+\right] - \left[V^{¿}(S^-) + Err^-\right]$$
$$\approx V^{¿}(S^+) - V^{¿}(S^-)$$

- More rigorously, it can be shown that the use of the same RNs in the up and down estimator, minimize the variance of the estimator

Other Uses of Monte Carlo

- In finance there are other uses of Monte Carlo in addition to derivative pricing:
  - **Scenario generation**: generate many market scenario used for what-if analysis
  - **Value at Risk**: establish a confidence interval on the maximum loss associated with a certain position over a certain time horizon
  - **Potential Future Exposure**: estimate the potential future liability a client might have toward us, for credit risk estimation
- In these cases, usually the diffused process are based on subjective assumptions rather than on risk neutral probabilities

Further Readings

- Glasserman, *Monte Carlo Methods in Financial Engineering*
- Kloeden, *A Brief Overview of Numerical Methods for Stochastic Differential Equations*
- Rubinstein, Samorodnitsky, Shaked, *Antithetic Variates, Multivariate Dependence and Simulation of Stochastic Systems*
- Longstaff, Schwartz, *Valuing American Options By Simulation: a Simple Least Squares Approach*
- Prof Giles' lecture notes, *http://people.maths.ox.ac.uk/~gilesm/mc/*
- *Quasi-Monte Carlo Simulation, http://www.puc-rio.br/marco.ind/quasi_mc.html,* (a good resource with Excel VBA examples)
- Niederreiter, *Low-Discrepancy Simulation,* (a 2009 survey on recent developments in Quasi Monte Carlo)