Price Sensitivities & Quasirandom Numbers

Deniz Kenan Kılıç & Dilek Aydoğan

April 28, 2016

Outline

Outline

Outline

Outline

- **Greeks** are calculated to check the effect of small changes in input parameters on the value of an option.
- Measuring the impact of price changes ‘delta(Δ)’ and ‘gamma(Γ)’:$\frac{\partial}{\partial S_1(t)}X(t)$, $\frac{\partial^2}{\partial S_1(t)^2}X(t)$
- The decreasing time to maturity ‘theta(Θ)’:$\frac{\partial}{\partial t}X(t)$
- Measuring for the consequence of possible errors in the input parameters volatility and interest rate by ‘vega’ and ‘rho(ρ)’:$\frac{\partial}{\partial \sigma}X(t)$ and $\frac{\partial}{\partial r}X(t)$
- When delta is large, the price of the derivative is sensitive to small changes in the price of the underlying security.
- When gamma is small, the change in delta is small. This sensitivity measure is important for deciding how much to adjust a hedge position.
- Theta is usually very small or negative since the value of an option tends to drop as it approaches maturity.
- When vega is large, the security is sensitive to small changes in volatility.

- The most popular **Greeks** are obtained from the Black-Scholes formula. For European call option:

$$\Delta_{call} = \Phi(d_1(t)) = \Phi\left(\frac{ln(S_1(t)/K) + \left(r + \frac{1}{2}\sigma^2\right)(T-t)}{\sigma\sqrt{T-t}}\right).$$

- For options on multiple underlyings, we have **Greeks** for each partial derivative w.r.t. one of the underlyings.
- Following methods are for obtaining the option price sensitivities.

Finite difference method

- A differentiable function can locally be approximated by a linear function. (Fundamental theorem of calculus)
- From tangent line, $f'(x)$ numerically is $\frac{f(x+h)-f(x)}{h}$ for a nonzero value of h. As h is small, one talks of **finite difference**.
- For $X(t) = X(t; s(t), r, \sigma)$, the **forward difference** is

$$\Delta = \frac{\partial X(t)}{\partial s} \approx \tilde{\Delta}_{for} := \frac{X(t; S(t)+h, r, \sigma) - X(t; S(t), r, \sigma)}{h}$$

- Or the **central difference**

$$\Delta \approx \tilde{\Delta}_{for} := \frac{X(t; S(t)+h, r, \sigma) - X(t; S(t)-h, r, \sigma)}{2h}$$

Consider a function that is differentiable up to the third order with bounded derivatives. Then a Taylor series expansion in x

$$f(x+h) = f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + \frac{1}{6}f'''(x)h^3 + o(h^3),$$

$$f(x-h) = f(x) - f'(x)h + \frac{1}{2}f''(x)h^2 - \frac{1}{6}f'''(x)h^3 + o(h^3).$$

Using above equations we get the estimates for the bias of the forward and the central differences as

$$B_{for}(f) = \left|\frac{f(x+h)-f(x)}{h} - f'(x)\right| + O(h),$$

$$B_{cen}(f) = \left|\frac{f(x+h)-f(x-h)}{2h} - f'(x)\right| + O(h^2).$$

Some notes on FDM

- Consider simulated paths $S^i(\theta), i = 1, \ldots, NSim$ depending on the model parameter θ, then MC estimator

$$\widehat{Greek}_\theta^{FD} := \frac{1}{N} \sum_{i=1}^{N} \frac{X\left(S^i(\theta + h)\right) - X\left(S^i(\theta - h)\right)}{2h}.$$

- If $X(.)$ is monotonic in S then, we could hope to have a small variance if we use the same random numbers (i.e. the same simulated Brownian paths) for the computation of both expectations. This principle is called the **principle of common random numbers** (or **path recycling**) and is also sometimes considered explicitly as a method for variance reduction.
- Note that the stepsize h cannot be chosen too small, as then round-off errors might dominate the computations. (See J$\ddot{a}$ckel, P.(2003))

Some notes on FDM

- For second order derivative such as gamma, we have

$$\frac{\frac{f(t+h)-f(t)}{h}-\frac{f(t)-f(t-h)}{h}}{h}=\frac{f(t+h)-2f(t)+f(t-h)}{h^2}.$$

Advantages and disadvantages of finite differences:

- To apply the FDM we do not need any further information on the model like the payoff or the nature of the model parameter θ.
- In addition to the Monte Carlo error and the possible discretization error for simulating the stock price paths, a third source of error enters the computations due to bias caused by finite difference.
- Furthermore, it is not possible to handle discontinuous payoffs.

The pathwise differentiation method

- The PWM requires additional information on the payoff and the parameter set θ. The method is not applicable to discontinuous payoff functions.
- In general method is expression involving the derivative of the payoff and the derivative of the simulated path such that $\frac{\partial}{\partial\theta}\mathbb{E}(B) = \mathbb{E}\left(f'(S(T))\frac{\partial S(T)}{\partial\theta}\right)$ where $B = f(S(T))$ is the payoff of an option.
- If above representation is valid then the corresponding pathwise differentiation Monte Carlo estimator

$$\widehat{Greek}_{\theta}^{PW} := \frac{1}{N}\sum_{i=1}^{N} f'\left(S^{(i)}(T)\right)\frac{\partial S^{(i)}(T)}{\partial\theta}$$

is unbiased(i.e. bias is equal to 0).

The pathwise differentiation method

For the Black-Scholes model we have;

$$dS(t) = rS(t)dt + \sigma S(t)dW(t),$$

and explicit solution

$$S(T) = S(0)e^{(r-\frac{1}{2}\sigma^2)T+\sigma\sqrt{T}Z}.$$

where $S(0)$ is spot value, $Z \sim \mathbb{N}(0,1)$, r is interest rate, σ is volatility, T is expiration date, K is strike price.
Consider the pathwise estimator with respect to the implied volatility σ_{BS}, the Vega of the option is

$$Greek_{\sigma_{BS}}^{PW} = \exp(-rT)\left(-\sigma T + \sqrt{T}Z\right)S(T)1_{\{S(T)>K\}}.$$

Matlab code for implementing the Vega for geometric Brownian motion for European Call and Put options

```
function optval = PW_Vega_CallPut(S,Z,K,C,r,sigma,T)
% S = NSim x 1 matrix of simulated path
% K = Strike price
% C = 1 -> Call; C = 0 -> Put
Indicator = S(:,end);

if(C==1)
    Indicator(Indicator<=K) = 0;
    Indicator(Indicator>K) = 1;
    optval =exp(-r*T) * ...
        mean(S(:,end).*Indicator.*(-sigma*T+sqrt(T)*Z));
else
    Indicator(Indicator>=K) = 0;
    Indicator(Indicator<K) = 1;
    optval =exp(-r*T) * ...
        mean(S(:,end).*Indicator.*(-sigma*T+sqrt(T)*Z));
end
end
```

The likelihood ratio method

We use the differentiability of the **density** of the stock price with respect to this parameter. For the density of the stock price $g(.)$(the dependence of the expectation on a parameter θ related to the stock price is summarized) at time T, we have $\mathbb{E}(f(S(T))) = \int f(S)g(S)dS$. After interchanging the differentiation w.r.t. the parameter θ with the integration, we get

$$\frac{\partial}{\partial\theta}\mathbb{E}(f(S(T))) = \int f(S)\frac{\partial}{\partial\theta}g(S)dS = \int f(S)\frac{\frac{\partial}{\partial\theta}g(S)}{g(S)}g(S)dS,$$

with definition of the likelihood ratio (or score function) as

$$w(S;\theta) := \frac{\frac{\partial}{\partial\theta}g(S)}{g(S)} = \frac{\partial ln(g(S))}{\partial\theta}$$

Then by the law of large numbers (the average of the results obtained from a large number of trials should be close to the expected value, and will tend to become closer as more trials are performed), the likelihood ratio estimator is an unbiased and strongly consistent estimator for the desired derivative as

$$I_{LR}(\theta;N) := \frac{1}{N}\sum_{i=1}^{N} f\left(S^{(i)}(T)\right) w\left(S^{(i)}(T);\theta\right).(iii)$$

Advantages of the likelihood ratio method

- **Comparison to finite difference methods:** Finite difference method has extra errors. (i.e. the discretization error of the derivative is a second source for errors) On the other hand, the likelihood ratio does not need a discretization of a derivative.
- **Comparison to the pathwise differentiation method:** No differentiability or continuity assumption for the payoff function $(f(S(T)))$ of the option is needed. Therefore discontinuous payoffs such as digital options can be considered.

Matlab code for calculation of the delta and gamma using the likelihood ratio method

```
%Parameters initialization
function rep = EurLik
r=0.1; %Risk-free rate
k=100; %Strike price
sigma=0.3; %Stock return volatility
delta=0.03; %Dividend yield
T=1; %Maturity
S0=80; %Initial stock price

NStep = floor(T*365.25); %Number of time step
Delta = 1/365.25; %Length of a step
NTraj = 100; %Number of paths

%Simulation of paths
dW = zeros(2*NTraj,1); S = S0*ones(2*NTraj,1);
Temp =zeros(NTraj,1); dW = cat(1,Temp, -Temp);
%End of parameters initialization

%Storing the simulation values
for i=1:NStep
Temp = sqrt(Delta)*randn(NTraj,1);
dW = cat(1,Temp,-Temp);
S = S + S.*((r-delta)*Delta + sigma*dW);
end
%End of storing the simulation values %End of Simulation of paths

%Computation of the values used in the estimators formulae
d = (log(S/S0) - (r - delta - 0.5*sigma^2)*T)/(sigma*sqrt(T));
```

Matlab code for calculation of the delta and gamma using the finite difference method

```
S = 80; % Spot
K = 100; % Strike
T = 1; % Maturity
N = 100; % Number of sample paths
sigma = .3; %Volatility
r = .1; % Interest rate

z = randn(N,1); % For each sample path, draw a random number to simulate terminal price of underlying
d_S = 1/365.25; % Size of increment/decrement to calculate delta and gamma

payoff = max(S*exp((r-sigma^2/2)*T+sigma*sqrt(T)*z)-K,0); % Option payoff with spot = S
payoff_Sp = max((S+d_S)*exp((r-sigma^2/2)*T+sigma*sqrt(T)*z)-K,0); % Option payoff with spot = S + d_S
payoff_Sm = max((S-d_S)*exp((r-sigma^2/2)*T+sigma*sqrt(T)*z)-K,0); % Option payoff with spot = S - d_S

price = mean(exp(-r*T)*payoff); % Price of option with spot = S
price_Sp = mean(exp(-r*T)*payoff_Sp); % Price of option with spot = S + d_S
price_Sm = mean(exp(-r*T)*payoff_Sm); % Price of option with spot = S - d_S

% Delta
delta = (price_Sp - price_Sm)/(2*d_S)
[CallVal, PutVal] = blsdelta(80, 100, 0.10, 1, 0.30, 0.03);
% Gamma
gamma = (price_Sp - 2*price + price_Sm)/(d_S^2)
blsgamma(80, 100, 0.10, 1, 0.30, 0.03);
```

Comparison of Finite Difference and Likelihood Ratio Results for Delta and Gamma

Table: Estimates for Δ and Γ of a European Call

Greek	Method N	100	10,000	100,000	Exact value
Δ	Finite Difference	0.4219	0.4081	0.3951	0.3486
	Likelihood ratio	0.2262	0.3489	0.3475	
Γ	Finite Difference	0	0.0386	0.0221	0.0151
	Likelihood ratio	0.0051	0.0151	0.0151	

Results

- For discontinuous payoffs, finite difference method and pathwise differentiation method are not available. To this end we motivated the likelihood ratio method.
- For Γ calculation finite difference method gives bad results. Since it starts to oscillate when coefficient of second derivative is very small.
- Since the functions we deal with in financial engineering are often non-differentiable or even discontinuous option payoffs, this may raise some trouble. On the contrary, in the likelihood ratio approach, the parameter is associated with the probability density of the underlying random variables.

Outline

Introduction

- The quasirandom sequence of numbers looks like random numbers because it looks unpredictable. However, they are deterministic alternative to random sequences for use in Monte Carlo methods.
- Main aim in QMC method is solving the problem of numerical integration over the unit hypercube.
- Points are chosen as elements of a low-discrepancy sequence for quasi-Monte Carlo method. (For MC points are chosen randomly distributed.)
- Quasirandom sequences are used to represent uniformly distributed random numbers.

Discrepancy

Definition (Discrepancy)

The discrepancy of a finite set $\Phi_t = \left\{x_i \in [0,1)^t, i = 1, \ldots, n\right\}$ of t-dimensional points is defined as

$$D(x_1, \ldots, x_n) := \sup_{A \in \mathcal{A}} \left| \frac{\Phi_t \cap A}{n} - volume(A) \right|$$

where $\mathcal{A} = \left\{ \prod_{j=1}^{t} [a_j, b_j) \,|0 \leq a_j < b_j \leq 1 \right\}$

- Discrepancy is a measure of deviation from uniformity of a sequence of points in unit hypercube D $(D = [0,1]^s)$. (i.e. how we cover the volume by distinct x_i's.)

Discrepancy

- Uniformity of sequence is measured by its discrepancy, which is the error in representation of the volume of subsets of the unit hypercube by the fraction of points in the subsets.
- The star discrepancy $D^*\left(x_1, \ldots, x_n\right)$ restricts the set of rectangles $\mathcal{A}$ to the ones with vertices $a_j = 0, j = 1, \ldots, t$.
- $D^* \leq D \leq 2^d D^*$

Koksma-Hlawka Inequality

Theorem

For smooth functions f with finite variation V(f) in the sense of Hardy and Krause we have

$$\left| \int_{[0,1)^t} f(x)dx - \frac{1}{n}\sum_{i=1}^{n} f(x_i) \right| \leq V(f)D^*(x_1, \ldots, x_n).$$

- Approximation is bounded by product of two elements. (measure of smoothness and uniformity)
- Filling sample area "efficiently" and has lower discrepancy than straight pseudo-random number set.

Halton Sequences

Algorithm of one-dimensional Halton sequences (Van-der-Corput sequences) can be given as;

- Select a prime number b as base.
- For the j-th number in the Halton sequence write $j \in \mathbb{N}$ in base b, $j = \sum_{i=0}^{\infty} a_i b^i$, where $a_i \in \{0, \ldots, b-1\}$.
- Obtain the Halton grid number as $H_j = \sum_{i=0}^{\infty} a_i b^{-i-1} \in [0, 1)$.

Halton Sequences

Algorithm of t-dimensional Halton sequences can be given as;

- Select t different prime numbers $b_k, k = 1, \ldots, t$ as bases, generally the first t primes.
- For the j-th point in the Halton sequence write $j \in \mathbb{N}$ in each base b_k, $j = \sum_{i=0}^{\infty} a_{k,i} b_k^i$, where $a_{k,i} \in \{0, \ldots, b_k - 1\}$.
- Obtain the t-dimensional Halton grid point as $H_j = (h_1, \ldots, h_t)$ with $h_k = \sum_{i=0}^{\infty} a_{k,i} b_k^{-i-1}, k = 1, \ldots, t$.

High Dimensional Halton Sequences

Table: Halton sequences for first 3 dimensions

	Dim=1 (Base 2)	Dim=2 (Base 3)	Dim=3 (Base 5)
j=1	1/2	1/3	1/5
j=2	1/4	2/3	2/5
j=3	3/4	1/9	3/5
j=4	1/8	4/9	4/5
j=5	5/8	7/9	1/25
j=6	3/8	2/9	6/25
j=7	7/8	5/9	11/25
j=8	1/16	8/9	16/25

R Results

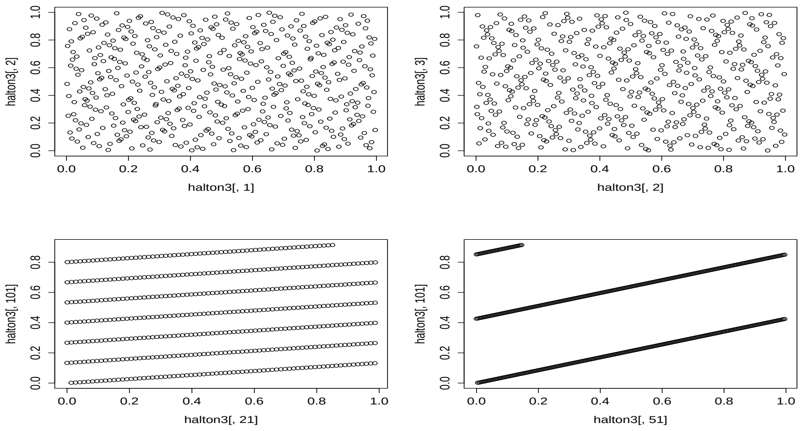

Figure: Halton Sequences with 500 Observations and Related Dimensions

Sobol Sequences

- The high dimension Halton sequences exhibit long cycle lengths due to the large prime number base. (Ex. in dim. 55 base is 257 as 55th prime)
- Sobol sequences use only base 2.
- Compared to Halton, the Sobol appears more irregular in higher dimensions and covers the area better.
- The algorithm of Sobol sequences is similar to the Halton algorithm but an important difference is the generator matrix V, which mixes the bits.
- Properties of V:
    - The elements of it are equal to 0 or 1.
    - Its columns are the binary expansions of a set of direction numbers $v_1, \cdots, v_r$, where r is the number of terms in the binary expansion of k.
    - This matrix will be upper triangular.

The Algorithm of Sobol Sequences

- Select a primitive polynomial

$$x^q + c_1 x^{q-1} + \cdots + c_{q-1} x + 1$$

  where $c_i \in \{0, 1\}$

- This polynomial defines a recurrence relation:

$$m_j = 2c_1 m_{j-1} \oplus 2^2 c_2 m_{j-2} \oplus \cdots \oplus 2^{q-1} c_{q-1} m_{j-q+1} \oplus 2^q m_{j-q} \oplus m_{j-q}.$$

- Specify the initial values $m_1, \cdots, m_q$.
- By using m_j's, define the direction numbers as follows:

$$v_j = m_j / 2^j.$$

The Algorithm of Sobol Sequences (Cont.)

- Construct the corresponding generator matrix.
- Now, let $a(k) = (a_0(k), \cdots, a_{r-1}(k))^T$ denote the vector of coefficients of the binary representation of k:

$$k = a_0(k) + 2a_1(k) + \ldots + 2^{r-1}a_{r-1}(k).$$

- And, let

$$\begin{pmatrix} y_1(k) \\ y_2(k) \\ \vdots \\ y_r(k) \end{pmatrix} = V \begin{pmatrix} a_0(k) \\ a_1(k) \\ \vdots \\ a_{r-1}(k) \end{pmatrix} mod2$$

- Then, the kth point in the sequence is computed as follows:

$$x_k = \frac{y_1(k)}{2} + \frac{y_2(k)}{4} + \cdots + \frac{y_r(k)}{2^r}$$

Example

Consider the primitive polynomial

$$x^3 + x^2 + 1$$

with degree q=3.
Observe that,

$$c_1 = 1, c_{q-1} = c_2 = 0$$

Then,

$$m_j = 2m_{j-1} \oplus 2^3 m_{j-3} \oplus m_{j-3}$$

Suppose we initialize it with $m_1 = 1$, $m_2 = 3$, $m_3 = 3$. Then, the next two elements are,

$$\begin{aligned} m_4 &= (2 \cdot 3) \oplus (8 \cdot 1) \oplus 1 \\ &= 0110 \oplus 1000 \oplus 0001 \\ &= 1111 \\ &= 15 \end{aligned}$$

Example (Cont.)

$$\begin{aligned} m_5 &= (2 \cdot 15) \oplus (8 \cdot 3) \oplus 3 \\ &= 11110 \oplus 11000 \oplus 00011 \\ &= 00101 \\ &= 5 \end{aligned}$$

Now, v_j values can be calculated by dividing m_j by 2^j and the results are as follows:

$$v_1 = 0.1, v_2 = 0.11, v_3 = 0.011, v_4 = 0.1111, v_5 = 0.00101,$$

and the corresponding generator matrix is $V = \begin{pmatrix} 1 & 1 & 0 & 1 & 0 \\ 0 & 1 & 1 & 1 & 0 \\ 0 & 0 & 1 & 1 & 1 \\ 0 & 0 & 0 & 1 & 0 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix}$.

Example (Cont.)

For each k find the vector $a(k)$ of binary coefficients of k and multiply it by the matrix V to get the x_k.
To illustrate let's compute x_3:

$$3 = 1 \cdot 2^0 + 1 \cdot 2^1 + 0 \cdot 2^2 + 0 \cdot 2^3 + 0 \cdot 2^4$$

$$a(3) = (1, 1, 0, 0, 0)$$

$$\begin{pmatrix} y_1(3) \\ y_2(3) \\ y_3(3) \\ y_4(3) \\ y_5(3) \end{pmatrix} = V \begin{pmatrix} 1 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix}$$

Example (Cont.)

$$x_3 = \frac{y_1(3)}{2} + \frac{y_2(3)}{2^2} + \cdots + \frac{y_5(3)}{2^5} = 1/4$$

And, the the last three points that can be generated with a 5×5 matrix which are $k = 29, 30, 31$ have the values $7/32$, $15/32$ and $31/32$, respectively.

R Results

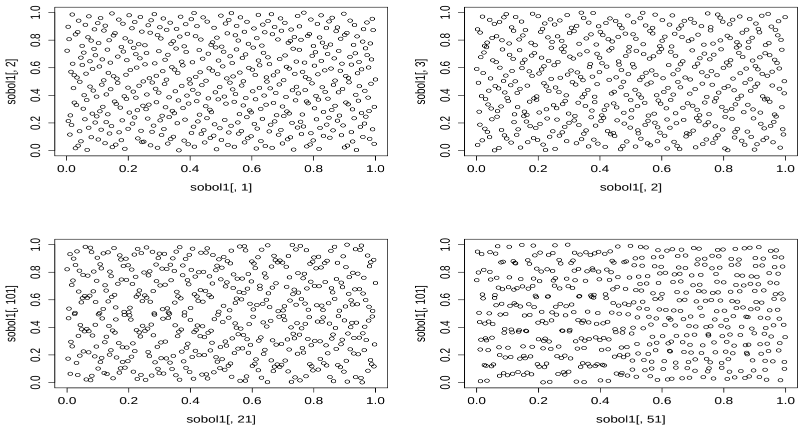

Figure: Sobol Sequences with 500 Observations and Related Dimensions

Summary

- The integral is approximated using a well-chosen sequence of points.
- In most cases QMC give better point estimates for a similar computational effort compared with regular Monte Carlo.
- Quasi-random numbers result in faster convergence.
- Hybrid methods or variance reduction techniques can be used in order to improve accuracy.

Outline

Bibliography I

F.D. Rouah.
The Heston Model and Its Extensions in Matlab and C#
John Wiley & Sons Publication, 2013.

H.T. Huynh and V.S. Lai and I. Soumare.
Stochastic Simulation and Applications in Finance with Matlab Programs
John Wiley & Sons Publication, 2008.

J. Kienitz and D. Wetterau.
Financial Modelling, Theory, Implementation and Practice with Matlab Source
John Wiley & Sons Publication, 2012.

Bibliography II

P. Brandimarte.
Handbook in Monte Carlo Simulation, Applications in Financial Engineering, Risk Management and Economics
John Wiley & Sons Publication, 2014.

P. Brandimarte.
Numerical Methods in Finance and Economics A Matlab-Based Introduction
John Wiley & Sons Publication, 2006.

P. Jackel.
Monte Carlo Methods in Finance
John Wiley & Sons Publication, 2003.