Algorithms and Relevant Formulas
from R. Seydel: Tools for Computational Finance (Springer)

preliminary version 24 Jan 2006

Black–Scholes–Merton World

relevant variables and notations:

t	current time, $0 \le t \le T$
T	expiration time, maturity
$r > 0$	risk-free interest rate
$S,\ S_t$	spot price, current price per share of stock/asset/underlying
σ	annual volatility
K	strike, exercise price per share
$V(S,t)$	value of an option at time t and underlying price S
V_{C}	value of a call option
V_{P}	value of a put option

r and σ are assumed constant.

Black–Scholes equation for a European-style standard options

$$\frac{\partial V}{\partial t} + \frac{1}{2}\sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} + rS\frac{\partial V}{\partial S} - rV = 0$$

Black–Scholes formula

The Black–Scholes equation has a closed-form solution, the Black–Scholes formula. This formula is written in terms of the time to maturity τ,

$$\tau := T - t \ ,$$

which leads to

$$\begin{aligned}
d_1(S,\tau;K,r,\sigma) &:= \frac{1}{\sigma\sqrt{\tau}}\left\{\log\frac{S}{K} + \left(r + \frac{\sigma^2}{2}\right)\tau\right\} \\
d_2(S,\tau;K,r,\sigma) &:= \frac{1}{\sigma\sqrt{\tau}}\left\{\log\frac{S}{K} + \left(r - \frac{\sigma^2}{2}\right)\tau\right\} \\
V_{\mathrm{P}}^{\mathrm{eur}}(S,\tau;K,r,\sigma) &= -SF(-d_1) + Ke^{-r\tau}F(-d_2) \\
V_{\mathrm{C}}^{\mathrm{eur}}(S,\tau;K,r,\sigma) &= SF(d_1) - Ke^{-r\tau}F(d_2)
\end{aligned}$$

(dividend-free case). F denotes the cumulated standard normal distribution function.

Distribution Function of the Standard Normal Distribution

$$f(x) := \frac{1}{\sqrt{2\pi}} \exp\left(-\frac{x^2}{2}\right)$$

$$F(x) := \int_{-\infty}^{x} f(t)\ dt$$

Define

$$z := \frac{1}{1 + 0.2316419x}$$

and the coefficients

$$\begin{array}{llll} a_1 = & 0.319381530 & a_4 = & -1.821255978 \\ a_2 = & -0.356563782 & a_5 = & 1.330274429 \\ a_3 = & 1.781477937. & & \end{array}$$

Then

$$F(x) = 1 - f(x)\left(a_1 z + a_2 z^2 + a_3 z^3 + a_4 z^4 + a_5 z^5\right) + \varepsilon(x)\ ,$$

for $0 \leq x < \infty$ with an absolute error ε bounded by

$$|\varepsilon(x)| < 7.5 * 10^{-8}$$

(see [Abramowitz, Stegun, 1968]). Hence we have the approximating formula

$$F(x) \approx 1 - f(x)z((((a_5 z + a_4)z + a_3)z + a_2)z + a_1)\ ,$$

which requires 17 arithmetic operations and the evaluation of the exponential function to obtain an accuracy of about 7 decimals. For $x < 0$ apply $F(x) = 1 - F(-x)$.

Binomial Method

$$\boxed{\begin{aligned} \beta &:= \frac{1}{2}(e^{-r\Delta t} + e^{(r+\sigma^2)\Delta t}) \\ u &= \beta + \sqrt{\beta^2 - 1} \\ d &= 1/u = \beta - \sqrt{\beta^2 - 1} \\ p &= \frac{e^{r\Delta t} - d}{u - d} \end{aligned}} \tag{1.11}$$

Call: $V(S(t_M), t_M) = \max\{S(t_M) - K, 0\}$, hence:

$$V_{jM} := (S_{jM} - K)^+ \tag{1.12C}$$

Put: $V(S(t_M), t_M) = \max\{K - S(t_M), 0\}$, hence:

$$V_{jM} := (K - S_{jM})^+ \tag{1.12P}$$

European option:

$$V_{ji} = e^{-r\Delta t} \cdot (pV_{j+1,i+1} + (1-p)V_{j,i+1}) \,. \tag{1.13}$$

American Call:

$$V_{ji} = \max\left\{(S_{ji} - K)^+,\ e^{-r\Delta t} \cdot (pV_{j+1,i+1} + (1-p)V_{j,i+1})\right\} \tag{1.14C}$$

American Put:

$$V_{ji} = \max\left\{(K - S_{ji})^+,\ e^{-r\Delta t} \cdot (pV_{j+1,i+1} + (1-p)V_{j,i+1})\right\} \tag{1.14P}$$

Algorithm: binomial method

$$\boxed{\begin{aligned} \textit{Input:}\ & r,\ \sigma,\ S = S_0,\ T,\ K,\ \text{choice of put or call,} \\ & \text{European or American},\ M \\ \textit{calculate:}\ & \Delta t := T/M,\ u,\ d,\ p \text{ from (1.11)} \\ & S_{00} := S_0 \\ & S_{jM} = S_{00} u^j d^{M-j},\ j = 0, 1, ..., M \\ & \text{(for American options, also } S_{ji} = S_{00} u^j d^{i-j} \\ & \ \text{for } 0 < i < M,\ j = 0, 1, ..., i) \\ & V_{jM} \text{ from (1.12)} \\ & V_{ji} \text{ for } i < M \begin{cases} \text{from (1.13) for European options} \\ \text{from (1.14) for American options} \end{cases} \\ \textit{Output:}\ & V_{00} \text{ is the approximation } V_0^{(M)} \text{ to } V(S_0, 0) \end{aligned}}$$

Stochastic Processes

Algorithm: simulation of a Wiener process

$$\begin{array}{|l|}\hline
\textit{Start:}\quad t_0 = 0,\ W_0 = 0;\ \Delta t \\
\textit{loop}\quad j = 1, 2, \ldots : \\
\qquad t_j = t_{j-1} + \Delta t \\
\qquad \text{draw}\quad Z \sim \mathcal{N}(0,1) \\
\qquad W_j = W_{j-1} + Z\sqrt{\Delta t} \\ \hline
\end{array}$$

Definition: Itô stochastic differential equation

An Itô stochastic differential equation is

$$dX_t = a(X_t, t)dt + b(X_t, t)dW_t; \tag{1.31a}$$

this together with $X_{t_0} = X_0$ is a symbolic short form of the integral equation

$$X_t = X_{t_0} + \int_{t_0}^{t} a(X_s, s)ds + \int_{t_0}^{t} b(X_s, s)dW_s. \tag{1.31b}$$

Algorithm: Euler discretization of an SDE

Approximations y_j to X_{t_j} are calculated by

$$\begin{array}{|l|}\hline
\textit{Start:}\quad t_0,\ y_0 = X_0,\ \Delta t,\ W_0 = 0. \\
\textit{loop}\quad j = 0, 1, 2, \ldots \\
\qquad t_{j+1} = t_j + \Delta t \\
\qquad \Delta W = Z\sqrt{\Delta t}\ \text{with}\quad Z \sim \mathcal{N}(0,1) \\
\qquad y_{j+1} = y_j + a(y_j, t_j)\Delta t + b(y_j, t_j)\Delta W \\ \hline
\end{array}$$

Model: geometric Brownian motion, GBM

$$\boxed{dS_t = \mu S_t\, dt + \sigma S_t\, dW_t} \tag{1.33, GBM}$$

Itô Lemma

Suppose X_t follows an Itô process (1.31), $dX_t = a(X_t, t)dt + b(X_t, t)dW_t$, and let $g(x, t)$ be a $\mathcal{C}^{2,1}$-smooth function (continuous $\frac{\partial g}{\partial x}$, $\frac{\partial^2 g}{\partial x^2}$, $\frac{\partial g}{\partial t}$). Then $Y_t := g(X_t, t)$ follows an Itô process with the *same* Wiener process W_t:

$$dY_t = \left(\frac{\partial g}{\partial x} a + \frac{\partial g}{\partial t} + \frac{1}{2} \frac{\partial^2 g}{\partial x^2} b^2 \right) dt + \frac{\partial g}{\partial x} b\ dW_t \tag{1.43}$$

where the derivatives of g as well as the coefficient functions a and b in general depend on the arguments (X_t, t).

Random Numbers

Algorithm: linear congruential generator

$$\boxed{\begin{array}{l}\text{Choose } N_0. \\ \text{For } i = 1, 2, \ldots \text{ calculate} \\ N_i = (aN_{i-1} + b) \mod M\end{array}} \tag{2.1}$$

$U_i := N_i/M$ should be uniformly distributed, $U \sim \mathcal{U}[0,1]$. An alternative:

Algorithm: Fibonacci generator

$$\boxed{\begin{array}{rl}\textit{Repeat:} & \zeta := U_i - U_j \\ & \text{if } \zeta < 0, \text{ set } \zeta := \zeta + 1 \\ & U_i := \zeta \\ & i := i - 1 \\ & j := j - 1 \\ & \text{if } i = 0, \text{ set } i := 17 \\ & \text{if } j = 0, \text{ set } j := 17\end{array}}$$

Initialization: Set $i = 17$, $j = 5$, and calculate $U_1, \ldots, U_{17}$ with a congruential generator, for instance with $M = 714025$, $a = 1366$, $b = 150889$. Set the seed $N_0 =$ your favorite dream number, possibly inspired by the system clock of your computer.

Algorithm: Box-Muller (creates $Z \sim \mathcal{N}(0,1)$)

$$\boxed{\begin{array}{l}(1) \text{ generate } U_1 \sim \mathcal{U}[0,1] \text{ and } U_2 \sim \mathcal{U}[0,1]. \\ (2)\ \theta := 2\pi U_2, \quad \rho := \sqrt{-2\log U_1} \\ (3)\ Z_1 := \rho\cos\theta \text{ is a normal variate} \\ \qquad (\text{as well as } Z_2 := \rho\sin\theta).\end{array}}$$

Algorithm: polar method (creates $Z \sim \mathcal{N}(0,1)$)

(1) *Repeat:* generate $U_1, U_2 \sim \mathcal{U}[0,1]$; $V_1 := 2U_1 - 1$, $V_2 := 2U_2 - 1$, *until* $W := V_1^2 + V_2^2 < 1$. (2) $Z_1 := V_1 \sqrt{-2\log(W)/W}$ $Z_2 := V_2 \sqrt{-2\log(W)/W}$ are both standard normal variates.

Algorithm: correlated random variable with expectation μ and covariance Σ

(1) Calculate the Cholesky decomposition $AA = \Sigma$ (2) Calculate $Z \sim \mathcal{N}(0, I)$ componentwise by $Z_i \sim \mathcal{N}(0,1)$, $i = 1, ..., n$, for instance, with Marsaglia's polar algorithm (3) $\mu + AZ$ has the desired distribution $\sim \mathcal{N}(\mu, \Sigma)$

Monte Carlo Simulation

Algorithm: Milstein integration of SDEs

Start: $t_0 = 0,\ y_0 = X_0,\ W_0 = 0,\ \Delta t = T/m$

loop $j = 0, 1, 2, ..., m-1$:

$t_{j+1} = t_j + \Delta t$

Calculate the values $a(y_j),\ b(y_j),\ b'(y_j)$

$\Delta W = Z\sqrt{\Delta t}$ with $Z \sim \mathcal{N}(0,1)$

$$y_{j+1} = y_j + a\Delta t + b\Delta W + \frac{1}{2} bb' \cdot ((\Delta W)^2 - \Delta t)$$

Algorithm: Monte Carlo simulation of European options

(1) *For* $k = 1, ..., N$: Choose a seed and integrate the SDE of the underlying model, here

$$dS = rS\,dt + \sigma S\,dW$$

for $0 \leq t \leq T$; let the final result be $(S_T)_k$.

(2) By evaluating the payoff function one obtains the values

$$(V(S_T, T))_k := V((S_T)_k, T), \quad k = 1, ..., N.$$

(3) An estimate of the risk-neutral expectation is

$$\widehat{\mathsf{E}}(V(S_T, T)) := \frac{1}{N} \sum_{k=1}^{N} (V(S_T, T))_k .$$

(4) The discounted variable

$$\widehat{V} := e^{-rT} \widehat{\mathsf{E}}(V(S_T, T))$$

is a random variable with $\mathsf{E}(\widehat{V}) = V(S_0, 0)$.

PDE methods

With a continuous dividend flow δ and SDE $dS = (\mu - \delta)S\,dt + \sigma S\,dW$ the corresponding Black-Scholes equation for $V(S,t)$ is

$$\frac{\partial V}{\partial t} + \frac{\sigma^2}{2} S^2 \frac{\partial^2 V}{\partial S^2} + (r-\delta) S \frac{\partial V}{\partial S} - rV = 0\,. \tag{4.1}$$

This equation is equivalent to the PDE equation

$$\frac{\partial y}{\partial \tau} = \frac{\partial^2 y}{\partial x^2} \tag{4.2}$$

for $y(x,\tau)$ with $0 \le \tau$, $x \in \mathbb{R}$. This equivalence can be proved by means of the transformations

$$\boxed{\begin{aligned} &S = Ke^x, \quad t = T - \frac{2\tau}{\sigma^2}, \quad q := \frac{2r}{\sigma^2}, \quad q_\delta := \frac{2(r-\delta)}{\sigma^2}, \\ &V(S,t) = V\left(Ke^x, T - \tfrac{2\tau}{\sigma^2}\right) =: v(x,\tau) \quad \text{and} \\ &v(x,\tau) =: K \exp\left\{-\tfrac{1}{2}(q_\delta - 1)x - \left(\tfrac{1}{4}(q_\delta - 1)^2 + q\right)\tau\right\} y(x,\tau). \end{aligned}} \tag{4.3}$$

The payoff is

$$\text{call:} \quad y(x,0) = \max\left\{e^{\frac{x}{2}(q_\delta+1)} - e^{\frac{x}{2}(q_\delta-1)},\ 0\right\} \tag{4.4C}$$

$$\text{put:} \quad y(x,0) = \max\left\{e^{\frac{x}{2}(q_\delta-1)} - e^{\frac{x}{2}(q_\delta+1)},\ 0\right\} \tag{4.4P}$$

auxiliary function:

$$\text{put:}\ \ g(x,\tau) := \exp\{\tfrac{1}{4}((q_\delta-1)^2 + 4q)\tau\} \max\{e^{\frac{1}{2}(q_\delta-1)x} - e^{\frac{1}{2}(q_\delta+1)x}, 0\}$$

$$\text{call } (\delta > 0): \quad g(x,\tau) := \exp\{\tfrac{1}{4}((q_\delta-1)^2 + 4q)\tau\} \max\{e^{\frac{1}{2}(q_\delta+1)x} - e^{\frac{1}{2}(q_\delta-1)x}, 0\}$$

finite-fifference discretization

notations for the grid are

$\tau_\nu := \nu \cdot \Delta\tau$ for $\nu = 0, 1, ..., \nu_{\max}$
$x_i := a + i\Delta x$ for $i = 0, 1, ..., m$
$y_{i\nu} := y(x_i, \tau_\nu)$,
$w_{i\nu}$ approximation to $y_{i\nu}$.
For each time level ν, the $w_{i\nu}$ are collected into a vector

$$w^{(\nu)} := (w_{1\nu}, ..., w_{m-1,\nu})^{tr}$$

Crank–Nicolson framework
choice of method: $\theta = \frac{1}{2}$ for Crank–Nicolson (alternative: $\theta = 1$ for backward-difference method)

$$\lambda := \frac{\Delta\tau}{\Delta x^2}$$

$$\begin{aligned}
&b_{i\nu} := w_{i\nu} + \lambda(1-\theta)(w_{i+1,\nu} - 2w_{i\nu} + w_{i-1,\nu})\ ,\ i = 2, \ldots, m-2 \\
&b_{1\nu} = w_{1\nu} + \lambda(1-\theta)(w_{2\nu} - 2w_{1\nu} + g_{0\nu}) + \lambda\theta g_{0,\nu+1} \\
&b_{m-1,\nu} = w_{m-1,\nu} + \lambda(1-\theta)(g_{m\nu} - 2w_{m-1,\nu} + w_{m-2,\nu}) + \lambda\theta g_{m,\nu+1} \\
&b^{(\nu)} := (b_{1\nu}, ..., b_{m-1,\nu})^{tr} \\
&w^{(\nu)} := (w_{1\nu}, ..., w_{m-1,\nu})^{tr} \\
&g^{(\nu)} := (g_{1\nu}, ..., g_{m-1,\nu})^{tr}
\end{aligned}$$

and

$$A := \begin{pmatrix} 1+2\lambda\theta & -\lambda\theta & & & 0 \\ -\lambda\theta & \ddots & \ddots & & \\ & \ddots & \ddots & \ddots & \\ & & \ddots & \ddots & \ddots \\ 0 & & & \ddots & \ddots \end{pmatrix} \in \mathbb{R}^{(m-1)\times(m-1)} \tag{4.30}$$

Algorithm: computation of American options

For $\nu = 0, 1, ..., \nu_{\max} - 1$:
Calculate the vectors $g := g^{(\nu+1)}$,
$b := b^{(\nu)}$ and $b := b^{(\nu)}$
Calculate the vector w as solution of the problem

$$Aw - b \geq 0, \quad w \geq g, \quad (Aw-b)^{tr}(w-g) = 0. \tag{4.32}$$

$w^{(\nu+1)} := w$

solution of (4.32)

Solve $Aw = b$ such that the side condition
$w \geq g$ is obeyed componentwise.

can be performed by a suitable elimination, using a backward/forward approach in case of a put, and a forward/backward approach in case of a call.

an approximation at S, t for $S > \overline{S}_\text{f}$ with $\tau = T - t$ is given by

$$\overline{V} := \alpha V_\text{P}^\text{eur}(S, \tau; Ke^{r\tau}) + (1-\alpha) V_\text{P}^\text{eur}(S, \tau; K) \ . \tag{4.41}$$

$$\begin{aligned} &\alpha := \left(\frac{r\tau}{a_0 r\tau + a_1} \right)^\beta \quad , \quad \beta := \frac{\ln(S/S_\text{f})}{\ln(K/S_\text{f})} \ , \\ &a_0 = 3.9649 \quad , \quad a_1 = 0.032325 \ . \end{aligned} \tag{4.42}$$

$$\overline{S}_\text{f} := K \left(\frac{2r}{\sigma^2 + 2r} \right)^\gamma \ , \tag{4.43}$$

$$\begin{aligned} &\gamma := \frac{\sigma^2 \tau}{b_0 \sigma^2 \tau + b_1} \ , \\ &b_0 = 1.04083 \quad , \quad b_1 = 0.00963 \ . \end{aligned} \tag{4.44}$$

Algorithm: interpolation method

> For given S, τ, K, r, σ evaluate $\gamma, \overline{S}_\text{f}, \beta$ with $\overline{S}_\text{f}$, and α .
> Evaluate the Black-Scholes formula for V_P^eur
> for the arguments in (4.41).
> Then $\overline{V}$ from (4.41) is an approximation to V_P^am for $S > \overline{S}_\text{f}$.

Algorithm: quadratic approximation

> For given S, τ, K, r, σ evaluate $q = \frac{2r}{\sigma^2}$, $H = 1 - e^{-r\tau}$
>
> and $\lambda := -\frac{1}{2} \left\{ (q-1) + \sqrt{(q-1)^2 + \frac{4q}{H}} \right\}$.
>
> Solve
>
> $$S_\text{f} F(d_1(S_\text{f})) \left[1 - \frac{1}{\lambda} \right] + K e^{-r\tau} \left[1 - F(d_2(S_\text{f})) \right] - K = 0$$
>
> iteratively for S_f. (This involves a sub-algorithm,
> from which $F(d_1(S_\text{f}))$ should be saved.)
> Evaluate $V_\text{P}^\text{eur}(S, \tau)$ using the Black-Scholes formula. Then
>
> $$\overline{V} := V_\text{P}^\text{eur}(S, \tau) - \frac{1}{\lambda} S_\text{f} F(d_1(S_\text{f})) \left(\frac{S}{S_\text{f}} \right)^\lambda$$
>
> is the approximation for $S > S_\text{f}$,
> and $\overline{V} = K - S$ for $S \leq S_\text{f}$.