Numerical Methods for Finance

Dr Robert Nürnberg

This course introduces the major numerical methods needed for quantitative work in finance. To this avail, the course will strike a balance between a general survey of significant numerical methods anyone working in a quantitative field should know, and a detailed study of some numerical methods specific to financial mathematics. In the first part the course will cover e.g.

linear and nonlinear equations, interpolation and optimization,

while the second part introduces e.g.

binomial and trinomial methods, finite difference methods, Monte-Carlo simulation, random number generators, option pricing and hedging.

The assessment consists of 80% an exam and 20% a project.
1. References

6. Hull (2005), *Options, Futures, and Other Derivatives.*
8. Press et al. (1992), *Numerical Recipes in C.* (online)
2. Preliminaries

1. Algorithms.

An *algorithm* is a set of instructions to construct an approximate solution to a mathematical problem.

A basic requirement for an algorithm is that the error can be made as small as we like. Usually, the higher the accuracy we demand, the greater is the amount of computation required.

An algorithm is *convergent* if it produces a sequence of values which converge to the desired solution of the problem.

**Example:** Given $c > 1$ and $\varepsilon > 0$, use the bisection method to seek an approximation to $\sqrt{c}$ with error not greater than $\varepsilon$. 
Example
Find \( x = \sqrt{c} \), \( c > 1 \) constant.

Answer
\[
x = \sqrt{c} \iff x^2 = c \iff f(x) := x^2 - c = 0
\]
\[
\Rightarrow f(1) = 1 - c < 0 \quad \text{and} \quad f(c) = c^2 - c > 0
\]
\[
\Rightarrow \exists \, \bar{x} \in (1, c) \quad s.t. \quad f(\bar{x}) = 0
\]
\[
f'(x) = 2x > 0 \quad \Rightarrow \quad f \text{ monotonically increasing} \quad \Rightarrow \quad \bar{x} \text{ is unique.}
\]

Denote \( I_n := [a_n, b_n] \) with \( I_0 = [a_0, b_0] = [1, c] \). Let \( x_n := \frac{a_n + b_n}{2} \).

(i) If \( f(x_n) = 0 \) then \( \bar{x} = x_n \).

(ii) If \( f(a_n) f(x_n) > 0 \) then \( \bar{x} \in (x_n, b_n), \) let \( a_{n+1} := x_n, \) \( b_{n+1} := b_n \).

(iii) If \( f(a_n) f(x_n) < 0 \) then \( \bar{x} \in (a_n, x_n), \) let \( a_{n+1} := a_n, \) \( b_{n+1} := x_n \).
Length of $I_n$:
$$m(I_n) = \frac{1}{2} m(I_{n-1}) = \cdots = \frac{1}{2^n} m(I_0) = \frac{c-1}{2^n}$$

**Algorithm**
Algorithm stops if $m(I_n) < \varepsilon$ and let $x^* := x_n$.

*Error as small as we like?*

\[
\bar{x}, x^* \in I_n
\]

\[\Rightarrow \text{error } |x^* - \bar{x}| = |x_n - \bar{x}| \leq m(I_n) \to 0 \text{ as } n \to \infty. \quad \checkmark
\]

*Convergence?*

\[I_0 \supset I_1 \supset \cdots \supset I_n \supset \cdots
\]

\[\Rightarrow \exists! \bar{x} \in \bigcap_{n=0}^{\infty} I_n, \quad f(\bar{x}) = 0, \text{ i.e. } \bar{x} = \sqrt{c}. \quad \checkmark
\]

**Implementation:**
No need to define $I_n = [a_n, b_n]$. It is sufficient to store only 3 points throughout.
Suppose $\bar{x} \in (a, b)$, define $x := \frac{a+b}{2}$.
If $\bar{x} \in (a, x)$ let $a := a$, $b := x$, otherwise let $a := x$, $b := b$. 

2. **Errors.**

There are various errors in computed solutions, such as

- *discretization error* (discrete approximation to continuous systems),
- *truncation error* (termination of an infinite process), and
- *rounding error* (finite digit limitation in computer arithmetic).

If $a$ is a number and $\tilde{a}$ is an approximation to $a$, then

the **absolute error** is $|a - \tilde{a}|$ and

the **relative error** is $\frac{|a - \tilde{a}|}{|a|}$ provided $a \neq 0$.

**Example**: Discuss sources of errors in deriving the numerical solution of the nonlinear differential equation $x' = f(x)$ on the interval $[a, b]$ with initial condition $x(a) = x_0$. 
Example

_discretization error_

\[ x' = f(x) \quad \text{[differential equation]} \]
\[ \frac{x(t + h) - x(t)}{h} = f(x(t)) \quad \text{[difference equation]} \]

\[ \text{DE} = \left| \frac{x(t + h) - x(t)}{h} - x'(t) \right| \]

_truncation error_

\[ \lim_{n \to \infty} x_n = x, \quad \text{approximate } x \text{ with } x_N, \ N \text{ a large number.} \]

\[ \text{TE} = |x - x_N| \]

_rounding error_

We cannot express \( x \) exactly, due to finite digit limitation. We get \( \hat{x} \) instead.

\[ \text{RE} = |x - \hat{x}| \]

Total error = DE + TE + RE.
3. **Well/Ill Conditioned Problems.**

A problem is *well-conditioned* (or *ill-conditioned*) if every small perturbation of the data results in a small (or large) change in the solution.

**Example:** Show that the solution to equations $x + y = 2$ and $x + 1.01y = 2.01$ is ill-conditioned.

**Exercise:** Show that the following problems are ill-conditioned:

(a) solution to the differential equation $x'' - 10x' - 11x = 0$ with initial conditions $x(0) = 1$ and $x'(0) = -1$,

(b) value of $q(x) = x^2 + x - 1150$ if $x$ is near a root.
Example

\[
\begin{align*}
\begin{cases}
x + y &= 2 \\
x + 1.01 \ y &= 2.01
\end{cases} & \Rightarrow \\
\begin{cases}
x &= 1 \\
y &= 1
\end{cases}
\end{align*}
\]

Change 2.01 to 2.02:

\[
\begin{align*}
\begin{cases}
x + y &= 2 \\
x + 1.01 \ y &= 2.02
\end{cases} & \Rightarrow \\
\begin{cases}
x &= 0 \\
y &= 2
\end{cases}
\end{align*}
\]

I.e. 0.5\% change in data produces 100\% change in solution: \textit{ill-conditioned}!

[reason: \det \begin{pmatrix} 1 & 1 \\ 1 & 1.01 \end{pmatrix} = 0.01 \Rightarrow \text{nearly singular}]
4. Taylor Polynomials.

Suppose \( f, f', \ldots, f^{(n)} \) are continuous on \([a, b]\) and \( f^{(n+1)} \) exists on \((a, b)\). Let \( x_0 \in [a, b] \). Then for every \( x \in [a, b] \), there exists a \( \xi \) between \( x_0 \) and \( x \) with

\[
f(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_0)}{k!} (x - x_0)^k + R_n(x)
\]

where \( R_n(x) = \frac{f^{(n+1)}(\xi)}{(n + 1)!} (x - x_0)^{n+1} \) is the remainder.

[Equivalently: \( R_n(x) = \int_{x_0}^{x} \frac{f^{(n+1)}(t)}{n!} (x - t)^n \, dt \).]

Examples:

- \( \exp(x) = \sum_{k=0}^{\infty} \frac{x^k}{k!} \)
- \( \sin(x) = \sum_{k=0}^{\infty} \frac{(-1)^k x^{2k+1}}{(2k + 1)!} \)
5. Gradient and Hessian Matrix.

Assume \( f : \mathbb{R}^n \rightarrow \mathbb{R} \).

The gradient of \( f \) at a point \( x \), written as \( \nabla f(x) \), is a column vector in \( \mathbb{R}^n \) with \( i \)th component \( \frac{\partial f}{\partial x_i}(x) \).

The Hessian matrix of \( f \) at \( x \), written as \( \nabla^2 f(x) \), is an \( n \times n \) matrix with \((i, j)\)th component \( \frac{\partial^2 f}{\partial x_i \partial x_j}(x) \). [As \( \frac{\partial^2 f}{\partial x_i \partial x_j} = \frac{\partial^2 f}{\partial x_j \partial x_i} \), \( \nabla^2 f(x) \) is symmetric.]

Examples:

- \( f(x) = a^T x, \ a \in \mathbb{R}^n \quad \Rightarrow \quad \nabla f = a, \ \nabla^2 f = 0 \)
- \( f(x) = \frac{1}{2} x^T A x, \ A \text{ symmetric} \quad \Rightarrow \quad \nabla f(x) = A x, \ \nabla^2 f = A \)
- \( f(x) = \exp(\frac{1}{2} x^T A x), \ A \text{ symmetric} \)
  \( \Rightarrow \quad \nabla f(x) = \exp(\frac{1}{2} x^T A x) A x, \)
  \( \nabla^2 f(x) = \exp(\frac{1}{2} x^T A x) A x x^T A + \exp(\frac{1}{2} x^T A x) A \)
6. **Taylor’s Theorem.**

Suppose that \( f : \mathbb{R}^n \to \mathbb{R} \) is continuously differentiable and that \( p \in \mathbb{R}^n \). Then we have

\[
f(x + p) = f(x) + \nabla f(x + tp)^T p,
\]

for some \( t \in (0, 1) \).

Moreover, if \( f \) is twice continuously differentiable, we have

\[
\nabla f(x + p) = \nabla f(x) + \int_0^1 \nabla^2 f(x + tp) p \, dt,
\]

and

\[
f(x + p) = f(x) + \nabla f(x)^T p + \frac{1}{2} p^T \nabla^2 f(x + tp) p,
\]

for some \( t \in (0, 1) \).

An $n \times n$ matrix $A = (a_{ij})$ is positive definite if it is symmetric (i.e. $A^T = A$) and $x^T A x > 0$ for all $x \in \mathbb{R}^n \setminus \{0\}$. [I.e. $x^T A x \geq 0$ with “=” only if $x = 0$.]

The following statements are equivalent:

(a) $A$ is a positive definite matrix,

(b) all eigenvalues of $A$ are positive,

(c) all leading principal minors of $A$ are positive.

The leading principal minors of $A$ are the determinants $\Delta_k$, $k = 1, 2, \ldots, n$, defined by

$$
\Delta_1 = \det[a_{11}], \quad \Delta_2 = \det \begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix}, \quad \ldots, \quad \Delta_n = \det A.
$$

A matrix $A$ is symmetric and positive semi-definite, if $x^T A x \geq 0$ for all $x \in \mathbb{R}^n$.

Exercise.

Show that any positive definite matrix $A$ has only positive diagonal entries.

A set $S \subset \mathbb{R}^n$ is a *convex set* if the straight line segment connecting any two points in $S$ lies entirely inside $S$, i.e., for any two points $x, y \in S$ we have

$$\alpha x + (1 - \alpha) y \in S \quad \forall \alpha \in [0, 1].$$

A function $f : D \to \mathbb{R}$ is a *convex function* if its domain $D \subset \mathbb{R}^n$ is a convex set and if for any two points $x, y \in D$ we have

$$f(\alpha x + (1 - \alpha) y) \leq \alpha f(x) + (1 - \alpha) f(y) \quad \forall \alpha \in [0, 1].$$

**Exercise.**

Let $D \subset \mathbb{R}^n$ be a convex, open set.

(a) If $f : D \to \mathbb{R}$ is continuously differentiable, then $f$ is convex if and only if

$$f(y) \geq f(x) + \nabla f(x)^T(y - x) \quad \forall x, y \in D.$$

(b) If $f$ is twice continuously differentiable, then $f$ is convex if and only if $\nabla^2 f(x)$ is positive semi-definite for all $x$ in the domain.
Exercise 2.8.
(a) “⇒”

As $f$ is convex we have for any $x, y$ in the convex set $D$ that

$$f(\alpha y + (1 - \alpha) x) \leq \alpha f(y) + (1 - \alpha) f(x) \quad \forall \alpha \in [0, 1].$$

Hence

$$f(y) \geq \frac{f(x + \alpha (y - x)) - f(x)}{\alpha} + f(x).$$

Letting $\alpha \to 0$ yields $f(y) \geq f(x) + \nabla f(x)^T (y - x)$.

“⇐”

For any $x_1, x_2 \in D$ and $\lambda \in [0, 1]$ let $x := \lambda x_1 + (1 - \lambda) x_2 \in D$ and $y := x_1$.

On noting that $y - x = x_1 - \lambda x_1 - (1 - \lambda) x_2 = (1 - \lambda) (x_1 - x_2)$ we have that

$$f(x_1) = f(y) \geq f(x) + \nabla f(x)^T (y - x) = f(x) + (1 - \lambda) \nabla f(x)^T (x_1 - x_2). \quad (\dagger)$$

Similarly, letting $x := \lambda x_1 + (1 - \lambda) x_2$ and $y := x_2$ gives, on noting that $y - x = \lambda (x_2 - x_1)$, that

$$f(x_2) \geq f(x) + \lambda \nabla f(x)^T (x_2 - x_1). \quad (\ddagger)$$
Combining $\lambda \cdot (†) + (1 - \lambda) \cdot (‡)$ gives

$$\lambda f(x_1) + (1 - \lambda) f(x_2) \geq f(x) = f(\lambda x_1 + (1 - \lambda) x_2) \quad \Rightarrow \quad f \text{ is convex.}$$

(b) “$\Leftarrow$”

For any $x, x_0 \in D$ use Taylor’s theorem at $x_0$:

$$f(x) = f(x_0) + \nabla f(x_0)^T (x - x_0) + \frac{1}{2} (x - x_0)^T \nabla^2 f(x_0 + \theta (x - x_0)) (x - x_0) \quad \theta \in (0, 1)$$

As $\nabla^2 f$ is positive semi-definite, this immediately gives

$$f(x) \geq f(x_0) + \nabla f(x_0)^T (x - x_0) \quad \Rightarrow \quad f \text{ is convex.}$$

“$\Rightarrow$”

Assume $\nabla^2 f$ is not positive semi-definite in the domain $D$. Then there exists $x_0 \in D$ and $\hat{x} \in \mathbb{R}^n$ s.t. $\hat{x}^T \nabla^2 f(x_0) \hat{x} < 0$. As $D$ is open we can find $x_1 := x_0 + \alpha \hat{x} \in D$, for $\alpha > 0$ sufficiently small. Hence $(x_1 - x_0)^T \nabla^2 f(x_0) (x_1 - x_0) < 0$. For $x_1$ sufficiently close to $x_0$ the continuity of $\nabla^2 f$ gives $(x_1 - x_0)^T \nabla^2 f(x_0 + \theta (x_1 - x_0)) (x_1 - x_0) < 0$ for all $\theta \in (0, 1)$. Taylor’s theorem then yields $f(x_1) < f(x_0) + \nabla f(x_0)^T (x_1 - x_0)$. This contradicts $f$ being convex, see (a).

A vector norm on \( \mathbb{R}^n \) is a function, \( \| \cdot \| \), from \( \mathbb{R}^n \) into \( \mathbb{R} \) with the following properties:

(i) \( \| x \| \geq 0 \) for all \( x \in \mathbb{R}^n \) and \( \| x \| = 0 \) if and only if \( x = 0 \).

(ii) \( \| \alpha x \| = |\alpha| \| x \| \) for all \( \alpha \in \mathbb{R} \) and \( x \in \mathbb{R}^n \).

(iii) \( \| x + y \| \leq \| x \| + \| y \| \) for all \( x, y \in \mathbb{R}^n \).

Common vector norms are the \( l_1 \), \( l_2 \) (Euclidean), and \( l_\infty \)-norms:

\[
\| x \|_1 = \sum_{i=1}^{n} |x_i|, \quad \| x \|_2 = \left\{ \sum_{i=1}^{n} x_i^2 \right\}^{1/2}, \quad \| x \|_\infty = \max_{1 \leq i \leq n} |x_i|.
\]

Exercise.

(a) Prove that \( \| \cdot \|_1 \), \( \| \cdot \|_2 \) and \( \| \cdot \|_\infty \) are norms.

(b) Given a symmetric positive definite matrix \( A \), prove that

\[
\| x \|_A := \sqrt{x^T A x}
\]

is a norm.
Example.

Draw graphs defined by $\|x\|_1 \leq 1$, $\|x\|_2 \leq 1$, $\|x\|_\infty \leq 1$ when $n = 2$.

Exercise. Prove that for all $x, y \in \mathbb{R}^n$ we have

$$(a) \quad \sum_{i=1}^{n} |x_i y_i| \leq \|x\|_2 \|y\|_2 \quad \text{[Scharz inequality]}$$

and

$$(b) \quad \frac{1}{\sqrt{n}} \|x\|_2 \leq \|x\|_\infty \leq \|x\|_1 \leq \sqrt{n} \|x\|_2.$$
10. **Spectral Radius.**

The *spectral radius* of a matrix $A \in \mathbb{R}^{n \times n}$ is defined by $\rho(A) = \max_{1 \leq i \leq n} |\lambda_i|$, where $\lambda_1, \ldots, \lambda_n$ are all the eigenvalues of $A$.

11. **Matrix Norms.**

For an $n \times n$ matrix $A$, the *natural matrix norm* $\|A\|$ for a given vector norm $\| \cdot \|$ is defined by

$$\|A\| = \max_{\|x\|=1} \|Ax\|.$$ 

The common matrix norms are

$$\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}|, \quad \|A\|_2 = \sqrt{\rho(A^T A)}, \quad \|A\|_\infty = \max_{1 \leq i \leq n} \sum_{j=1}^{n} |a_{ij}|.$$ 

**Exercise:** Compute $\|A\|_1$, $\|A\|_\infty$, and $\|A\|_2$ for $A = \begin{bmatrix} 1 & 1 & 0 \\ 1 & 2 & 1 \\ -1 & 1 & 2 \end{bmatrix}$.

(*Answer:* $\|A\|_1 = \|A\|_\infty = 4$ and $\|A\|_2 = \sqrt{7 + \sqrt{7}} \approx 3.1$.)

A sequence of vectors \( \{x^{(k)}\} \subset \mathbb{R}^n \) is said to converge to a vector \( x \in \mathbb{R}^n \) if \( \|x^{(k)} - x\| \to 0 \) as \( k \to \infty \) for an arbitrary vector norm \( \| \cdot \| \). This is equivalent to the componentwise convergence, i.e., \( x_i^{(k)} \to x_i \) as \( k \to \infty \), \( i = 1, \ldots, n \).

A square matrix \( A \in \mathbb{R}^{n \times n} \) is said to be convergent if \( \|A^k\| \to 0 \) as \( k \to \infty \), which is equivalent to \( (A^k)_{ij} \to 0 \) as \( k \to \infty \) for all \( i, j \).

The following statements are equivalent:

(i) \( A \) is a convergent matrix,

(ii) \( \rho(A) < 1 \),

(iii) \( \lim_{k \to \infty} A^k x = 0 \), for every \( x \in \mathbb{R}^n \).

Exercise. Show that \( A \) is convergent, where

\[
A = \begin{bmatrix}
1/2 & 0 \\
1/4 & 1/2
\end{bmatrix}.
\]
3. Algebraic Equations

1. Decomposition Methods for Linear Equations.

A matrix $A \in \mathbb{R}^{n \times n}$ is said to have $LU$ decomposition if $A = LU$ where $L \in \mathbb{R}^{n \times n}$ is a lower triangular matrix ($l_{ij} = 0$ if $1 \leq i < j \leq n$) and $U \in \mathbb{R}^{n \times n}$ is an upper triangular matrix ($u_{ij} = 0$ if $1 \leq j < i \leq n$).

The decomposition is unique if one assumes e.g. $l_{ii} = 1$ for $1 \leq i \leq n$.

$$L = \begin{pmatrix} l_{11} \\ l_{21} & l_{22} \\ l_{31} & l_{32} & l_{33} \\ \vdots & \ddots \\ l_{n1} & l_{n2} & l_{n3} & \cdots & l_{nn} \end{pmatrix}, \quad U = \begin{pmatrix} u_{11} & u_{12} & u_{13} & \cdots & u_{1n} \\ u_{22} & u_{23} & \cdots & u_{2n} \\ \vdots & \ddots & \ddots & \vdots \\ u_{n-1,n-1} & u_{n-1,n} & \cdots & u_{nn} \end{pmatrix}$$
In general, the diagonal elements of either $L$ or $U$ are given and the remaining elements of the matrices are determined by directly comparing two sides of the equation. The linear system $Ax = b$ is then equivalent to $Ly = b$ and $Ux = y$.

**Exercise.**

Show that the solution to $Ly = b$ is

$$y_1 = b_1/l_{11}, \quad y_i = (b_i - \sum_{k=1}^{i-1} l_{ik} y_k)/l_{ii}, \quad i = 2, \ldots, n$$

(forward substitution) and the solution to $Ux = y$ is

$$x_n = y_n/u_{nn}, \quad x_i = (y_i - \sum_{k=i+1}^{n} u_{ik} x_k)/u_{ii}, \quad i = n - 1, \ldots, 1$$

(backward substitution).
2. Crout Algorithm.

Exercise.

Let $A$ be tridiagonal, i.e. $a_{ij} = 0$ if $|i - j| > 1$ ($a_{ij} = 0$ except perhaps $a_{i-1,i}$, $a_{ii}$ and $a_{i,i+1}$), and strictly diagonally dominant ($|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ holds for $i = 1, \ldots, n$). Show that $A$ can be factorized as $A = LU$ where $l_{ii} = 1$ for $i = 1, \ldots, n$, $u_{11} = a_{11}$, and

\[
\begin{align*}
    u_{i,i+1} &= a_{i,i+1} \\
    l_{i+1,i} &= a_{i+1,i} / u_{ii} \\
    u_{i+1,i+1} &= a_{i+1,i+1} - l_{i+1,i} u_{i,i+1}
\end{align*}
\]

for $i = 1, \ldots, n - 1$. [Note: $L$ and $U$ are tridiagonal.]

C++ Exercise: Write a program to solve a tridiagonal and strictly diagonally dominant linear equation $Ax = b$ by the Crout algorithm. The input are the number of variables $n$, the matrix $A$, and the vector $b$. The output is the solution $x$. 
Exercise 3.2.

$u_{11} = a_{11}$ and $u_{i,i+1} = a_{i,i+1}$, $l_{i+1,i} = a_{i+1,i}/u_{ii}$, $u_{i+1,i+1} = a_{i+1,i+1} - l_{i+1,i} u_{i,i+1}$, for $i = 1, \ldots, n - 1$, can easily be shown.

It remains to show that $u_{ii} \neq 0$ for $i = 1, \ldots, n$. We proceed by induction to show that

$$|u_{ii}| > |a_{i,i+1}|,$$

where for convenience we define $a_{n,n+1} := 0$.

- $i = 1$: $|u_{11}| = |a_{11}| > |a_{1,2}|$ ✓
- $i \mapsto i + 1$:
  $$|u_{i+1,i+1}| = \left|a_{i+1,i+1} - \frac{a_{i+1,i} a_{i,i+1}}{u_{ii}}\right|$$
  $$\geq |a_{i+1,i+1}| - |a_{i+1,i}| \frac{|a_{i,i+1}|}{|u_{ii}|} \geq |a_{i+1,i+1}| - |a_{i+1,i}| |a_{i+1,i+2}|$$ ✓

Overall we have that $|u_{ii}| > 0$ and so the Crout algorithm is well defined. Moreover, $\det(A) = \det(L) \det(U) = \det(U) = \prod_{i=1}^{n} u_{ii} \neq 0$. 

25
3. **Choleski Algorithm.**

**Exercise.**

Let $A$ be a positive definite matrix. Show that $A$ can be factorized as $A = LL^T$ where $L$ is a lower triangular matrix.

(i) Compute 1st column:

\[
l_{11} = \sqrt{a_{11}}, \quad l_{i1} = a_{i1}/l_{11}, \quad i = 2, \ldots, n.\]

(ii) For $j = 2, \ldots, n - 1$ compute $j$th column:

\[
l_{jj} = (a_{jj} - \sum_{k=1}^{j-1} l_{jk}^2)^{\frac{1}{2}} \]

\[
l_{ij} = (a_{ij} - \sum_{k=1}^{j-1} l_{ik} l_{jk})/l_{jj}, \quad i = j + 1, \ldots, n.\]
(iii) Compute $n$th column:

$$l_{nn} = \left(a_{nn} - \sum_{k=1}^{n-1} l_{nk}^2\right)^{\frac{1}{2}}.$$
4. **Iterative Methods for Linear Equations.**

Split \( A \) into \( A = M + N \) with \( M \) nonsingular and convert the equation \( A \boldsymbol{x} = \boldsymbol{b} \) into an equivalent equation \( \boldsymbol{x} = \boldsymbol{C} \boldsymbol{x} + \boldsymbol{d} \) with \( \boldsymbol{C} = -M^{-1} N \) and \( \boldsymbol{d} = M^{-1} \boldsymbol{b} \).

Choose an initial vector \( \boldsymbol{x}^{(0)} \) and then generate a sequence of vectors by

\[
\boldsymbol{x}^{(k)} = \boldsymbol{C} \boldsymbol{x}^{(k-1)} + \boldsymbol{d}, \quad k = 1, 2, \ldots
\]

The resulting sequence converges to the solution of \( A \boldsymbol{x} = \boldsymbol{b} \), for an arbitrary initial vector \( \boldsymbol{x}^{(0)} \), if and only if \( \rho(C) < 1 \).

The objective is to choose \( M \) such that \( M^{-1} \) is easy to compute and \( \rho(C) < 1 \).

The iteration stops if \( \| \boldsymbol{x}^{(k)} - \boldsymbol{x}^{(k-1)} \| < \varepsilon \).

[Note: In practice one solves \( M \boldsymbol{x}^{(k)} = -N \boldsymbol{x}^{(k-1)} + \boldsymbol{b} \), for \( k = 1, 2, \ldots \).]
Claim.
The iteration $x^{(k)} = C x^{(k-1)} + d$ is convergent if and only if $\rho(C) < 1$.

Proof.
Define $e^{(k)} := x^{(k)} - x$, the error of the $k$th iterate. Then

$$e^{(k)} = C x^{(k-1)} + d - (C x + d) = C (x^{(k-1)} - x) = C e^{(k-1)} = C^2 e^{(k-2)} = \ldots C^k e^{(0)},$$

where $e^{(0)} = x^{(0)} - x$ is an arbitrary vector.

Assume $C$ is similar to the diagonal matrix $\Lambda = \text{diag}(\lambda_1, \ldots, \lambda_n)$, where $\lambda_i$ are the eigenvalues of $C$.

\[ \Rightarrow \exists X \text{ nonsingular s.t. } C = X \Lambda X^{-1} \]

\[ \Rightarrow e^{(k)} = C^k e^{(0)} = X \Lambda^k X^{-1} e^{(0)} = X \begin{pmatrix} \lambda_1^k & & \\ & \ddots & \\ & & \lambda_n^k \end{pmatrix} X^{-1} e^{(0)} \to 0 \text{ as } k \to \infty \]

\[ \iff |\lambda_i| < 1 \quad \forall \ i = 1, \ldots, n \]

\[ \iff \rho(C) < 1. \]
5. Jacobi Algorithm.

**Exercise:** Let $M = D$ and $N = L + U$ ($L$ strict lower triangular part of $A$, $D$ diagonal, $U$ strict upper triangular part). Show that the $i$th component at the $k$th iteration is

$$x_i^{(k)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k-1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right)$$

for $i = 1, \ldots, n$.


**Exercise:** Let $M = D + L$ and $N = U$. Show that the $i$th component at the $k$th iteration is

$$x_i^{(k)} = \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right)$$

for $i = 1, \ldots, n$. 

30
7. SOR Algorithm.

Exercise.

Let $M = \frac{1}{\omega} D + L$ and $N = U + (1 - \frac{1}{\omega}) D$ where $0 < \omega < 2$. Show that the $i$th component at the $k$th iteration is

$$x_i^{(k)} = (1 - \omega) x_i^{(k-1)} + \omega \frac{1}{a_{ii}} \left( b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k-1)} \right)$$

for $i = 1, \ldots, n$.

C++ Exercise: Write a program to solve a diagonally dominant linear equation $Ax = b$ by the SOR algorithm. The input are the number of variables $n$, the matrix $A$, the vector $b$, the initial vector $x^0$, the relaxation parameter $\omega$, and tolerance $\varepsilon$. The output is the number of iterations $k$ and the approximate solution $x^k$. 
8. **Special Matrices.**

If $A$ is strictly diagonally dominant, then Jacobi and Gauss–Seidel converge for any initial vector $x^{(0)}$. In addition, SOR converges for $\omega \in (0, 1]$.

If $A$ is positive definite and $0 < \omega < 2$, then the SOR method converges for any initial vector $x^{(0)}$.

If $A$ is positive definite and tridiagonal, then $\rho(C_{GS}) = [\rho(C_{J})]^2 < 1$ and the optimal choice of $\omega$ for the SOR method is $\omega = \frac{2}{1 + \sqrt{1 - \rho(C_{GS})}} \in [1, 2)$. With this choice of $\omega$, $\rho(C_{SOR}) = \omega - 1 \leq \rho(C_{GS})$.

**Exercise.**

Find the optimal $\omega$ for the SOR method for the matrix

$$A = \begin{bmatrix} 4 & 3 & 0 \\ 3 & 4 & -1 \\ 0 & -1 & 4 \end{bmatrix}.$$  

(*Answer: $\omega \approx 1.24$.*)
9. **Condition Numbers.**

The *condition number* of a nonsingular matrix $A$ relative to a norm $\| \cdot \|$ is defined by

$$
\kappa(A) = \| A \| \cdot \| A^{-1} \|.
$$

Note that $\kappa(A) \geq \| A A^{-1} \| = \| I \| = \max_{\| x \| = 1} \| x \| = 1$.

A matrix $A$ is *well-conditioned* if $\kappa(A)$ is close to one and is *ill-conditioned* if $\kappa(A)$ is much larger than one.

Suppose $\| \delta A \| < \frac{1}{\| A^{-1} \|}$. Then the solution $\tilde{x}$ to $(A + \delta A) \tilde{x} = b + \delta b$ approximates the solution $x$ of $Ax = b$ with error estimate

$$
\frac{\| x - \tilde{x} \|}{\| x \|} \leq \frac{\kappa(A)}{1 - \| \delta A \| \| A^{-1} \|} \left( \frac{\| \delta b \|}{\| b \|} + \frac{\| \delta A \|}{\| A \|} \right).
$$

In particular, if $\delta A = 0$ (no perturbation to matrix $A$) then

$$
\frac{\| x - \tilde{x} \|}{\| x \|} \leq \kappa(A) \frac{\| \delta b \|}{\| b \|}.
$$
Example.
Consider Example 1.3.

\[
A = \begin{pmatrix}
1 & 1 \\
1 & 1.01
\end{pmatrix}
\]

\[
\Rightarrow A^{-1} = \frac{1}{\det A} \begin{pmatrix}
1.01 & -1 \\
-1 & 1
\end{pmatrix} = \frac{1}{0.01} \begin{pmatrix}
1.01 & -1 \\
-1 & 1
\end{pmatrix} = \begin{pmatrix}
101 & -100 \\
-100 & 100
\end{pmatrix}
\]

Recall

\[
\|A\|_1 = \max_{1 \leq j \leq n} \sum_{i=1}^{n} |a_{ij}|
\]

Hence

\[
\|A\|_1 = \max(2, 2.01) = 2.01 , \quad \|A^{-1}\|_1 = \max(201, 200) = 201.
\]

\[
\Rightarrow \kappa_1(A) = \|A\|_1 \cdot \|A^{-1}\|_1 = 404.01 \gg 1 \quad (\text{ill-conditioned!})
\]

Similarly \( \kappa_{\infty} = 404.01 \) and \( \kappa_2 = \rho(A) \rho(A^{-1}) = \frac{\lambda_{\text{max}}}{\lambda_{\text{min}}} = 402.0075. \)
10. **Hilbert Matrix.**

An $n \times n$ Hilbert matrix $H_n = (h_{ij})$ is defined by $h_{ij} = 1/(i + j - 1)$ for $i, j = 1, 2, \ldots, n$.

Hilbert matrices are notoriously ill-conditioned and $\kappa(H_n) \to \infty$ very rapidly as $n \to \infty$.

$$H_n = \begin{pmatrix}
1 & 1 & \cdots & \frac{1}{n} \\
\frac{1}{2} & 1 & \cdots & \frac{1}{n-1} \\
\frac{1}{3} & \frac{1}{2} & \cdots & \frac{1}{n-2} \\
\vdots & \vdots & & \vdots \\
\frac{1}{n} & \frac{1}{n+1} & \cdots & \frac{1}{2n-1}
\end{pmatrix}$$

**Exercise.**

Compute the condition numbers $\kappa_1(H_3)$ and $\kappa_1(H_6)$. 
(Answer: $\kappa_1(H_3) = 748$ and $\kappa_1(H_6) = 2.9 \times 10^6$.)
11. **Fixed Point Method for Nonlinear Equations.**

A function $g : \mathbb{R} \to \mathbb{R}$ has a *fixed point* $\bar{x}$ if

$$g(\bar{x}) = \bar{x}.$$

A function $g$ is a *contraction mapping* on $[a, b]$ if $g : [a, b] \to [a, b]$ and

$$|g'(x)| \leq L < 1, \quad \forall x \in (a, b)$$

where $L$ is a constant.

**Exercise.**

Assume $g$ is a contraction mapping on $[a, b]$. Prove that $g$ has a unique fixed point $\bar{x}$ in $[a, b]$, and for any $x_0 \in [a, b]$, the sequence defined by

$$x_{n+1} = g(x_n), \quad n \geq 0,$$

converges to $\bar{x}$. The algorithm is called *fixed point iteration*. 
Exercise 3.11.

Existence:
Define \( h(x) = x - g(x) \) on \([a,b]\). Then \( h(a) = a - g(a) \leq 0 \) and \( h(b) = b - g(b) \geq 0 \).
As \( h \) is continuous, \( \exists \ c \in [a,b] \) s.t. \( h(c) = 0 \). I.e. \( c = g(c) \). ✓

Uniqueness:
Suppose \( p, q \in [a,b] \) are two fixed points. Then

\[
|p - q| = |g(p) - g(q)| = |g'(\alpha) (p - q)| \leq L |p - q|
\]

\[
\Rightarrow (1 - L) |p - q| \leq 0 \Rightarrow |p - q| \leq 0 \Rightarrow p = q. \quad ✓
\]

Convergence:

\[
|x_n - \bar{x}| = |g(x_{n-1}) - g(\bar{x})| = |g'(\alpha) (x_{n-1} - \bar{x})| \\
\leq L |x_{n-1} - \bar{x}| \leq ... \leq L^n |x_0 - \bar{x}| \to 0 \text{ as } n \to \infty .
\]

Hence

\[
x_n \to \bar{x} \text{ as } n \to \infty . \quad ✓
\]
12. **Newton Method for Nonlinear Equations.**

Assume that \( f \in C^1([a, b]) \), \( f(\bar{x}) = 0 \) (\( \bar{x} \) is a root or zero) and \( f'(\bar{x}) \neq 0 \).

The *Newton method* can be used to find the root \( \bar{x} \) by generating a sequence \( \{x_n\} \) satisfying

\[
x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}, \quad n = 0, 1, \ldots
\]

provided \( f'(x_n) \neq 0 \) for all \( n \).

The sequence \( x_n \) converges to the root \( \bar{x} \) as long as the initial point \( x_0 \) is sufficiently close to \( \bar{x} \).

The algorithm stops if \( |x_{n+1} - x_n| < \varepsilon \), a prescribed error tolerance, and \( x_{n+1} \) is taken as an approximation to \( \bar{x} \).

**Geometric Interpretation:**

Tangent line at \( (x_n, f(x_n)) \) is

\[
Y = f(x_n) + f'(x_n)(X - x_n).
\]
Setting $Y = 0$ yields $x_{n+1} := X = x_n - \frac{f(x_n)}{f'(x_n)}$. 
13. **Choice of Initial Point.**

Suppose \( f \in C^2([a, b]) \) and \( f(\bar{x}) = 0 \) with \( f'(\bar{x}) \neq 0 \). Then there exists \( \delta > 0 \) such that the Newton method generates a sequence \( x_n \) converging to \( \bar{x} \) for any initial point \( x_0 \in [\bar{x} - \delta, \bar{x} + \delta] \) (\( x_0 \) can only be chosen locally).

However, if \( f \) satisfies the following additional conditions:

1. \( f(a)f(b) < 0 \),
2. \( f'' \) does not change sign on \([a, b]\),
3. tangent lines to the curve \( y = f(x) \) at both \( a \) and \( b \) cut the \( x \)-axis within \([a, b]\); i.e. \( a - \frac{f(a)}{f'(a)}, b - \frac{f(b)}{f'(b)} \in [a, b] \)

then \( f(x) = 0 \) has a unique root \( \bar{x} \) in \([a, b]\) and Newton method converges to \( \bar{x} \) for any initial point \( x_0 \in [a, b] \) (\( x_0 \) can be chosen globally).

**Example.**

Use the Newton method to compute \( x = \sqrt{c}, c > 1 \), and show that the initial point can be any point in \([1, c]\).
Example.

Find $x = \sqrt{c}$, $c > 1$.

**Answer.**

$x$ is root of $f(x) := x^2 - c$.

Newton: $x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)} = x_n - \frac{x_n^2 - c}{2x_n} = \frac{1}{2} \left( x_n + \frac{c}{x_n} \right)$, $n \geq 0$.

**How to choose $x_0$?**

Check the 3 conditions on $[1, c]$.

1. $f(1) = 1 - c < 0$, $f(c) = c^2 - c > 0$. $\Rightarrow f(1) f(c) < 0$ $\checkmark$

2. $f'' = 2$ $\checkmark$

3. Tangent line at 1: $Y = f(1) + f'(1) (X - 1) = 1 - c + 2(X - 1)$

   Let $Y = 0$, then $X = 1 + \frac{c - 1}{2} \in (1, c)$. $\checkmark$

   Tangent line at $c$: $Y = f(c) + f'(c) (X - c) = c^2 - c + 2c (X - c)$

   Let $Y = 0$, then $X = c - \frac{c - 1}{2} \in (1, c)$. $\checkmark$
⇒ Newton convergence for any $x_0 \in [1, c]$. 
Numerical Example.

Find $\sqrt{7}$.  
(From calculator: $\sqrt{7} = 2.6457513$.)

Newton converges for all $x_0 \in [1, 7]$. Choose $x_0 = 4$.  

$$x_1 = \frac{1}{2} \left( x_0 + \frac{7}{x_0} \right) = 2.875$$

$$x_2 = 2.6548913$$

$$x_3 = 2.6457670$$

$$x_4 = 2.6457513$$

Comparison to bisection method with $I_0 = [1, 7]$:

$$I_1 = [1, 4]$$

$$I_2 = [2.5, 4]$$

$$I_3 = [2.5, 3.25]$$

$$I_4 = [2.5, 2.875]$$

$$\vdots$$

$$I_{25} = [2.6457512, 2.6457513]$$
14. **Pitfalls.**

Here are some difficulties which may be encountered with the Newton method:

1. \( \{x_n\} \) may wander around and not converge (there are only complex roots to the equation),

2. initial approximation \( x_0 \) is too far away from the desired root and \( \{x_n\} \) converges to some other root (this usually happens when \( f'(x_0) \) is small),

3. \( \{x_n\} \) may diverge to \(+\infty\) (the function \( f \) is positive and monotonically decreasing on an unbounded interval), and

4. \( \{x_n\} \) may repeat (cycling).
15. **Rate of Convergence.**

Suppose \( \{x_n\} \) is a sequence that converges to \( \bar{x} \).

The convergence is said to be **linear** if there is a constant \( r \in (0, 1) \) such that

\[
\frac{|x_{n+1} - \bar{x}|}{|x_n - \bar{x}|} \leq r, \quad \text{for all } n \text{ sufficiently large.}
\]

The convergence is said to be **superlinear** if

\[
\lim_{n \to \infty} \frac{|x_{n+1} - \bar{x}|}{|x_n - \bar{x}|} = 0.
\]

In particular, the convergence is said to be **quadratic** if

\[
\frac{|x_{n+1} - \bar{x}|}{|x_n - \bar{x}|^2} \leq M, \quad \text{for all } n \text{ sufficiently large.}
\]

where \( M \) is a positive constant, not necessarily less than 1.

**Example.** \( x_n = \bar{x} + 0.5^n \) linear, \( x_n = \bar{x} + 0.5^{2^n} \) quadratic.

**Example.** Show that the Newton method converges quadratically.
Example.

Define \( g(x) = x - \frac{f(x)}{f'(x)} \). Then the Newton method is given by

\[ x_{n+1} = g(x_n). \]

Moreover, \( f(\bar{x}) = 0 \) and \( f'(\bar{x}) \neq 0 \) imply that

\[
\begin{align*}
g(\bar{x}) &= \bar{x}, \\
g'(\bar{x}) &= 1 - \frac{(f')^2 - f f''}{(f')^2} (\bar{x}) = f(\bar{x}) \frac{f''(\bar{x})}{(f'(\bar{x}))^2} = 0, \\
g''(\bar{x}) &= \frac{f''(\bar{x})}{f'(\bar{x})}.
\end{align*}
\]

Assuming that \( x_n \to \bar{x} \) we have that

\[
\frac{|x_{n+1} - \bar{x}|}{|x_n - \bar{x}|^2} = \frac{|g(x_n) - g(\bar{x})|}{|x_n - \bar{x}|^2} \stackrel{\text{Taylor}}{=} \frac{|g'(\bar{x}) (x_n - \bar{x}) + \frac{1}{2} g''(\eta_n) (x_n - \bar{x})^2|}{|x_n - \bar{x}|^2}
\]

\[
= \frac{1}{2} |g''(\eta_n)| \to \frac{1}{2} |g''(\bar{x})| =: \lambda \quad \text{as } n \to \infty.
\]
Hence $|x_{n+1} - \bar{x}| \approx \lambda |x_n - \bar{x}|^2 \Rightarrow\text{ quadratic convergence rate.}$
4. Interpolations

1. Polynomial Approximation.

For any continuous function $f$ defined on an interval $[a, b]$, there exist polynomials $P$ that can be as “close” to the given function as desired.

Taylor polynomials agree closely with a given function at a specific point, but they concentrate their accuracy only near that point.

A good polynomial needs to provide a relatively accurate approximation over an entire interval.
2. Interpolating Polynomial – Lagrange Form.

Suppose $x_i \in [a, b], i = 0, 1, \ldots, n$, are pairwise distinct mesh points in $[a, b]$. The Lagrange polynomial $p$ is a polynomial of degree $\leq n$ such that

$$p(x_i) = f(x_i), \quad i = 0, 1, \ldots, n.$$ 

$p$ can be constructed explicitly as

$$p(x) = \sum_{i=0}^{n} L_i(x) f(x_i)$$

where $L_i$ is a polynomial of degree $n$ satisfying

$$L_i(x_j) = 0, \quad j \neq i, \quad L_i(x_i) = 1.$$ 

This results in

$$L_i(x) = \prod_{j \neq i} \left( \frac{x - x_j}{x_i - x_j} \right) \quad i = 0, 1, \ldots, n.$$ 

$p$ is called linear interpolation if $n = 1$ and quadratic interpolation if $n = 2$. 

50
Exercise.

Find the Lagrange polynomial \( p \) for the following points \((x, f(x))\): \((1, 0)\), \((-1, -3)\), and \((2, -4)\). Assume that a new point \((0, 2)\) is observed, and construct a Lagrange polynomial to incorporate this new information in it.

Error formula.

Suppose \( f \) is \( n + 1 \) times differentiable on \([a, b]\). Then it holds that

\[
f(x) = p(x) + \frac{f^{(n+1)}(\xi)}{(n + 1)!}(x - x_0) \cdots (x - x_n),
\]

where \( \xi = \xi(x) \) lies in \((a, b)\).

Proof.

Define \( g(x) = f(x) - p(x) + \lambda \prod_{j=0}^{n} (x - x_j) \), where \( \lambda \in \mathbb{R} \) is a constant. Clearly, \( g(x_j) = 0 \) for \( j = 0, \ldots, n \). To estimate the error at \( x = \alpha \notin \{x_0, \ldots, x_n\} \), choose \( \lambda \) such that \( g(\alpha) = 0 \).
Hence

\[ g(x) = f(x) - p(x) - (f(\alpha) - p(\alpha)) \prod_{j=0}^{n} \left( \frac{x - x_j}{\alpha - x_j} \right). \]

\[ \Rightarrow \quad g \text{ has at least } n + 2 \text{ zeros: } x_0, \ldots, x_n, \alpha. \]

Mean Value Theorem yields that

\[ g' \quad \text{ has at least } n + 1 \text{ zeros} \]

\[ : \]

\[ g^{(n+1)} \quad \text{ has at least } 1 \text{ zero, say } \xi \]

Moreover, as \( p \) is polynomial of degree \( \leq n \), it holds that \( p^{(n+1)} = 0 \).
Hence

\[ 0 = g^{(n+1)}(\xi) = f^{(n+1)}(\xi) - (f(\alpha) - p(\alpha)) \frac{(n + 1)!}{\prod_{j=0}^{n}(\alpha - x_j)} \]

\[ \Rightarrow \quad \text{Error} = f(\alpha) - p(\alpha) = \frac{f^{(n+1)}(\xi)}{(n + 1)!} \prod_{j=0}^{n}(\alpha - x_j). \]
3. **Trapezoid Rule.**

We can use linear interpolation \((n = 1, x_0 = a, x_1 = b)\) to approximate \(f(x)\) on \([a, b]\) and then compute \(\int_a^b f(x) \, dx\) to get the *trapezoid rule:*

\[
\int_a^b f(x) \, dx \approx \frac{1}{2} (b - a) [f(a) + f(b)].
\]

If we partition \([a, b]\) into \(n\) equal subintervals with mesh points \(x_i = a + ih, i = 0, \ldots, n\), and step size \(h = (b - a)/n\), we can derive the *composite trapezoid rule:*

\[
\int_a^b f(x) \, dx \approx \frac{h}{2} [f(x_0) + 2 \sum_{i=1}^{n-1} f(x_i) + f(x_n)].
\]

The approximation error is in the order of \(O(h^2)\) if \(|f''|\) is bounded.
Use linear interpolation \((n = 1, x_0 = a, x_1 = b)\) to approximate \(f(x)\) on \([a, b]\) and then compute \(\int_a^b f(x) \, dx\).

**Answer.**

The linear interpolating polynomial is \(p(x) = f(a) L_0(x) + f(b) L_1(x)\), where

\[
L_0(x) = \frac{x - b}{a - b}, \quad \text{and} \quad L_1(x) = \frac{x - a}{b - a}.
\]

\[
\Rightarrow \int_a^b f(x) \, dx \approx \int_a^b p(x) \, dx = f(a) \int_a^b \frac{x - b}{a - b} \, dx + f(b) \int_a^b \frac{x - a}{b - a} \, dx
\]

\[
= f(a) \frac{1}{a - b} \left[ x - b \right]^b_a + f(b) \frac{1}{b - a} \left[ x - a \right]^b_a
\]

\[
= f(a) \frac{1}{a - b} \left[ -(a - b)^2 \right]_a^b + f(b) \frac{1}{b - a} \left[ (b - a)^2 \right]_a^b
\]

\[
= \frac{b - a}{2} (f(a) + f(b)) \quad \leftarrow \quad \text{Trapezoid Rule}
\]

[Of course, one could have arrived at this formula with a simple geometric argument.]
**Error Analysis.**

Let $f(x) = p(x) + E(x)$, where $E(x) = \frac{f''(\xi)}{2} (x - a) (x - b)$ with $\xi \in (a, b)$. Assume that $|f''| \leq M$ is bounded. Then

\[
\left| \int_{a}^{b} E(x) \, dx \right| \leq \int_{a}^{b} |E(x)| \, dx \leq \frac{M}{2} \int_{a}^{b} (x - a) (b - x) \, dx
\]

\[
= \frac{M}{2} \int_{a}^{b} (x - a) [(b - a) - (x - a)] \, dx
\]

\[
= \frac{M}{2} \int_{a}^{b} [-(x - a)^2 + (b - a) (x - a)] \, dx
\]

\[
= \frac{M}{2} \left[ -\frac{1}{3} (b - a)^3 + \frac{1}{2} (b - a)^3 \right] \, dx
\]

\[
= \frac{M}{12} (b - a)^3.
\]
The composite formula can be obtained by considering the partitioning of \([a, b]\) into
\[a = x_0 < x_1 < \ldots < x_{n-1} < x_n = b,\] where \(x_i = a + i h\) with \(h := \frac{b - a}{n} \).

\[
\int_a^b f(x) \, dx = \sum_{i=0}^{n-1} \int_{x_i}^{x_{i+1}} f(x) \, dx \approx \sum_{i=0}^{n-1} \frac{x_{i+1} - x_i}{2} (f(x_i) + f(x_{i+1}))
\]
\[= \sum_{i=0}^{n-1} \frac{h}{2} (f(x_i) + f(x_{i+1}))
\]
\[= h \left[ \frac{1}{2} f(a) + f(x_1) + \ldots + f(x_{n-1}) + \frac{1}{2} f(b) \right].
\]

Error analysis then yields that
\[
\text{Error} \leq \frac{M}{12} h^3 n = \frac{M (b - a)}{12} h^2 = O(h^2).
\]
4. **Simpson’s Rule.**

**Exercise.**

Use quadratic interpolation \((n = 2, x_0 = a, x_1 = \frac{a+b}{2}, x_2 = b)\) to approximate \(f(x)\) on \([a, b]\) and then compute \(\int_a^b f(x) \, dx\) to get the Simpson’s rule:

\[
\int_a^b f(x) \, dx \approx \frac{1}{6} (b - a) \left[ f(a) + 4 f\left(\frac{a+b}{2}\right) + f(b) \right].
\]

Derive the composite Simpson’s rule:

\[
\int_a^b f(x) \, dx \approx \frac{h}{3} \left[ f(x_0) + 2 \sum_{i=2}^{n/2} f(x_{2i-2}) + 4 \sum_{i=1}^{n/2} f(x_{2i-1}) + f(x_n) \right],
\]

where \(n\) is an even number and \(x_i\) and \(h\) are chosen as in the composite trapezoid rule.

[One can show that the approximation error is in the order of \(O(h^4)\) if \(|f^{(4)}|\) is bounded.]
5. **Newton–Cotes Formula.**

Suppose \( x_0, \ldots, x_n \) are mesh points in \([a, b]\), usually mesh points are equally spaced and \( x_0 = a, \ x_n = b \), then integral can be approximated by the *Newton–Cotes formula*:

\[
\int_a^b f(x) \, dx \approx \sum_{i=0}^{n} A_i f(x_i)
\]

where parameters \( A_i \) are determined in such a way that the integral is exact for all polynomials of degree \( \leq n \).

[Note: \( n+1 \) unknowns \( A_i \) and \( n+1 \) coefficients for polynomial of degree \( n \).] **Exercise.**

Use Newton–Cotes formula to derive the trapezoid rule and the Simpson’s rule. Prove that if \( f \) is \( n+1 \) times differentiable and \(|f^{(n+1)}| \leq M\) on \([a, b]\) then

\[
| \int_a^b f(x) \, dx - \sum_{i=0}^{n} A_i f(x_i) | \leq \frac{M}{(n+1)!} \int_a^b \prod_{i=0}^{n} |x - x_i| \, dx.
\]
Exercise 4.5.

We have that
\[ \int_a^b q(x) \, dx = \sum_{i=0}^n A_i q(x_i) \quad \text{for all polynomials } q \text{ of degree } \leq n. \]

Let \( q(x) = L_j(x) \), where \( L_j \) is the \( j \)th Lagrange polynomial for the data points \( x_0, x_1, \ldots, x_n \).

I.e. \( L_j \) is of degree \( n \) and satisfies \( L_j(x_i) = \delta_{ij} = \begin{cases} 1 & i = j \\ 0 & i \neq j \end{cases} \). Now

\[ \int_a^b L_j \, dx = \sum_{i=0}^n A_i L_j(x_i) = A_j \]

\[ \Rightarrow \quad \int_a^b f(x) \, dx \approx \sum_{i=0}^n A_i f(x_i) = \sum_{i=0}^n f(x_i) \int_a^b L_i(x) \, dx \]

\[ = \int_a^b \sum_{i=0}^n f(x_i) L_i(x) \, dx = \int_a^b p(x) \, dx, \]

where \( p(x) \) is the interpolating Lagrange polynomial to \( f \). Hence we find trapezoid
\( (n = 1) \) and Simpson’s rule \((n = 2, \text{ with } x_1 = \frac{a+b}{2})\).

The Lagrange polynomial has the error term

\[
 f(x) = p(x) + E(x), \quad E(x) := \frac{f^{(n+1)}(\xi)}{(n + 1)!} (x - x_0) \cdots (x - x_n),
\]

where \( \xi = \xi(x) \) lies in \((a, b)\). Hence

\[
 \left| \int_a^b f(x) \, dx - \int_a^b p(x) \, dx \right| = \left| \int_a^b E(x) \, dx \right| \leq \int_a^b |E(x)| \, dx
\]

\[
 \leq \frac{M}{(n + 1)!} \int_a^b \prod_{i=0}^n |x - x_i| \, dx.
\]
6. **Ordinary Differential Equations.**

An *initial value problem* for an ODE has the form

\[ x'(t) = f(t, x(t)), \quad a \leq t \leq b \quad \text{and} \quad x(a) = x_0. \]  

(1)

(1) is equivalent to the integral equation:

\[ x(t) = x_0 + \int_a^t f(s, x(s)) \, ds, \quad a \leq t \leq b. \]  

(2)

To solve (2) numerically we divide \([a, b]\) into subintervals with mesh points \(t_i = a + ih, \quad i = 0, \ldots, n\), and step size \(h = (b - a)/n\). (2) implies

\[ x(t_{i+1}) = x(t_i) + \int_{t_i}^{t_{i+1}} f(s, x(s)) \, ds, \quad i = 0, \ldots, n - 1. \]
(a) If we approximate \( f(s, x(s)) \) on \([t_i, t_{i+1}]\) by \( f(t_i, x(t_i)) \), we get the **Euler (explicit) method** for equation (1):

\[
w_{i+1} = w_i + hf(t_i, w_i), \quad w_0 = x_0.
\]

We have \( x(t_{i+1}) \approx w_{i+1} \) if \( h \) is sufficiently small.

[Taylor: \( x(t_{i+1}) = x(t_i) + x'(t_i) h + O(h^2) = x(t_i) + f(t_i, x(t_i)) h + O(h^2) \).]

(b) If we approximate \( f(s, x(s)) \) on \([t_i, t_{i+1}]\) by linear interpolation with points \((t_i, f(t_i, x(t_i)))\) and \((t_{i+1}, f(t_{i+1}, x(t_{i+1})))\), we get the **trapezoidal (implicit) method** for equation (1):

\[
w_{i+1} = w_i + \frac{h}{2} [f(t_i, w_i) + f(t_{i+1}, w_{i+1})], \quad w_0 = x_0.
\]

(c) If we combine the Euler method with the trapezoidal method, we get the **modified Euler (explicit) method** (or Runge–Kutta 2nd order method):

\[
w_{i+1} = w_i + \frac{h}{2} [f(t_i, w_i) + f(t_{i+1}, w_i + hf(t_i, w_i))], \quad w_0 = x_0.
\]
7. Divided Differences.

Suppose a function $f$ and $(n + 1)$ distinct points $x_0, x_1, \ldots, x_n$ are given. Divided differences of $f$ can be expressed in a table format as follows:

<table>
<thead>
<tr>
<th>$x_k$</th>
<th>0DD</th>
<th>1DD</th>
<th>2DD</th>
<th>3DD</th>
<th>...</th>
</tr>
</thead>
<tbody>
<tr>
<td>$x_0$</td>
<td>$f[x_0]$</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_1$</td>
<td>$f[x_1]$</td>
<td>$f[x_0, x_1]$</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_2$</td>
<td>$f[x_2]$</td>
<td>$f[x_1, x_2]$</td>
<td>$f[x_0, x_1, x_2]$</td>
<td></td>
<td></td>
</tr>
<tr>
<td>$x_3$</td>
<td>$f[x_3]$</td>
<td>$f[x_2, x_3]$</td>
<td>$f[x_1, x_2, x_3]$</td>
<td>$f[x_0, x_1, x_2, x_3]$</td>
<td></td>
</tr>
<tr>
<td>...</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td>...</td>
</tr>
</tbody>
</table>
where \( f[x_i] = f(x_i) \)

\[
f[x_i, x_{i+1}] = \frac{f[x_{i+1}] - f[x_i]}{x_{i+1} - x_i}
\]

\[
f[x_i, x_{i+1}, \ldots, x_{i+k}] = \frac{f[x_{i+1}, x_{i+2}, \ldots, x_{i+k}] - f[x_i, x_{i+1}, \ldots, x_{i+k-1}]}{x_{i+k} - x_i}
\]

\[
f[x_1, \ldots, x_n] = \frac{f[x_2, \ldots, x_n] - f[x_1, \ldots, x_{n-1}]}{x_n - x_1}
\]
8. **Interpolating Polynomial – Newton Form.**

One drawback of Lagrange polynomials is that there is no recursive relationship between $P_{n-1}$ and $P_n$, which implies that each polynomial has to be constructed individually. Hence, in practice one uses the *Newton polynomials*.

The Newton interpolating polynomial $P_n$ of degree $n$ that agrees with $f$ at the points $x_0, x_1, \ldots, x_n$ is given by

$$P_n(x) = f[x_0] + \sum_{k=1}^{n} f[x_0, x_1, \ldots, x_k] \prod_{i=0}^{k-1} (x - x_i).$$

Note that $P_n$ can be computed recursively using the relation

$$P_n(x) = P_{n-1}(x) + f[x_0, x_1, \ldots, x_n](x - x_0)(x - x_1) \cdots (x - x_{n-1}).$$

[Note that $f[x_0, x_1, \ldots, x_k]$ can be found on the diagonal of the DD table.]

**Exercise.**

Suppose values $(x, y)$ are given as $(1, -2), (-2, -56), (0, -2), (3, 4), (-1, -16),$ and $(7, 376)$. Is there a cubic polynomial that takes these values? (Answer: $2x^3 - 7x^2 +$
$5x - 2.$
Exercise 4.8.

Data points: (1, −2), (−2, −56), (0, −2), (3, 4), (−1, −16), (7, 376).

<table>
<thead>
<tr>
<th>$x_k$</th>
<th>0DD</th>
<th>1DD</th>
<th>2DD</th>
<th>3DD</th>
<th>4DD</th>
<th>5DD</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>−2</td>
<td></td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>−2</td>
<td>−56</td>
<td>18</td>
<td></td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>0</td>
<td>−2</td>
<td>27</td>
<td>−9</td>
<td></td>
<td></td>
<td></td>
</tr>
<tr>
<td>3</td>
<td>4</td>
<td>2</td>
<td>−5</td>
<td>2</td>
<td></td>
<td></td>
</tr>
<tr>
<td>−1</td>
<td>−16</td>
<td>5</td>
<td>−3</td>
<td>2</td>
<td>0</td>
<td></td>
</tr>
<tr>
<td>7</td>
<td>376</td>
<td>49</td>
<td>11</td>
<td>2</td>
<td>0</td>
<td>0</td>
</tr>
</tbody>
</table>

Newton polynomial:

$$p(x) = -2 + 18(x - 1) - 9(x - 1)(x + 2) + 2(x - 1)(x + 2)x$$

$$= 2x^3 - 7x^2 + 5x - 2.$$  

[Note: We could have stopped at the row for $x_3 = 3$ and checked whether $p_3$ goes through the remaining data points.]
9. **Piecewise Polynomial Approximations.**

Another drawback of interpolating polynomials is that $P_n$ tends to oscillate widely when $n$ is large, which implies that $P_n(x)$ may be a poor approximation to $f(x)$ if $x$ is not close to the interpolating points.

If an interval $[a, b]$ is divided into a set of subintervals $[x_i, x_{i+1}]$, $i = 0, 1, \ldots, n-1$, and on each subinterval a different polynomial is constructed to approximate a function $f$, such an approximation is called *spline*.

The simplest spline is the linear spline $P$ which approximates the function $f$ on the interval $[x_i, x_{i+1}]$, $i = 0, 1, \ldots, n - 1$, with $P$ agreeing with $f$ at $x_i$ and $x_{i+1}$.

Linear splines are easy to construct but are not smooth at points $x_1, x_2, \ldots, x_{n-1}$. 
10. **Natural Cubic Splines.**

Given a function $f$ defined on $[a, b]$ and a set of points $a = x_0 < x_1 < \cdots < x_n = b$, a function $S$ is called a *natural cubic spline* if there exist $n$ cubic polynomials $S_i$ such that:

(a) $S(x) = S_i(x)$ for $x$ in $[x_i, x_{i+1}]$ and $i = 0, 1, \ldots, n - 1$;

(b) $S_i(x_i) = f(x_i)$ and $S_i(x_{i+1}) = f(x_{i+1})$ for $i = 0, 1, \ldots, n - 1$;

(c) $S'_{i+1}(x_{i+1}) = S''_i(x_{i+1})$ for $i = 0, 1, \ldots, n - 2$;

(d) $S'''_{i+1}(x_{i+1}) = S'''_i(x_{i+1})$ for $i = 0, 1, \ldots, n - 2$;

(e) $S''(x_0) = S''(x_n) = 0$. 
Natural Cubic Splines.

\[ S_i \quad S_{i+1} \]

\[ x_i \quad x_{i+1} \quad x_{i+2} \]

(a) \(4n\) parameters
(b) \(2n\) equations
(c) \(n - 1\) equations
(d) \(n - 1\) equations
(e) 2 equations

\[ \begin{cases} 4n \text{ equations} \\ 4n \text{ equations} \end{cases} \]
Example. Assume $S$ is a natural cubic spline that interpolates $f \in C^2([a, b])$ at the nodes $a = x_0 < x_1 < \cdots < x_n = b$. We have the following smoothness property of cubic splines:

$$\int_a^b [S''(x)]^2 \, dx \leq \int_a^b [f''(x)]^2 \, dx.$$ 

In fact, it even holds that

$$\int_a^b [S''(x)]^2 \, dx = \min_{g \in \mathcal{G}} \int_a^b [g''(x)]^2 \, dx,$$

where $\mathcal{G} := \{g \in C^2([a, b]) : g(x_i) = f(x_i) \quad i = 0, 1, \ldots, n\}$.

Exercise: Determine the parameters $a$ to $h$ so that $S(x)$ is a natural cubic spline, where

$S(x) = ax^3 + bx^2 + cx + d$ for $x \in [-1, 0]$ and

$S(x) = ex^3 + fx^2 + gx + h$ for $x \in [0, 1]$

with interpolation conditions $S(-1) = 1$, $S(0) = 2$, and $S(1) = -1$.

(Answer: $a = -1, b = -3, c = -1, d = 2, e = 1, f = -3, g = -1, h = 2$.)

Denote
\[ c_i = S''(x_i), \quad i = 0, 1, \ldots, n. \]

Then \( c_0 = c_n = 0. \)

Since \( S_i \) is a cubic function on \([x_i, x_{i+1}]\), we know that \( S_i'' \) is a linear function on \([x_i, x_{i+1}]\). Hence it can be written as
\[
S_i''(x) = c_i \frac{x_{i+1} - x}{h_i} + c_{i+1} \frac{x - x_i}{h_i}
\]
where \( h_i := x_{i+1} - x_i \).
Exercise.

Show that $S_i$ is given by

$$S_i(x) = \frac{c_i}{6 h_i} (x_{i+1} - x)^3 + \frac{c_{i+1}}{6 h_i} (x - x_i)^3 + p_i (x_{i+1} - x) + q_i (x - x_i),$$

where

$$p_i = \left( \frac{f(x_i)}{h_i} - \frac{c_i h_i}{6} \right), \quad q_i = \left( \frac{f(x_{i+1})}{h_i} - \frac{c_{i+1} h_i}{6} \right)$$

and $c_1, \ldots, c_{n-1}$ satisfy the linear equations:

$$h_{i-1} c_{i-1} + 2 (h_{i-1} + h_i) c_i + h_i c_{i+1} = u_i,$$

where

$$u_i = 6 (d_i - d_{i-1}), \quad d_i = \frac{f(x_{i+1}) - f(x_i)}{h_i}$$

for $i = 1, 2, \ldots, n - 1$.

**C++ Exercise:** Write a program to construct a natural cubic spline. The inputs are the number of points $n + 1$ and all the points $(x_i, y_i), i = 0, 1, \ldots, n$. The output is a natural cubic spline expressed in terms of the functions $S_i$ defined on the intervals $[x_i, x_{i+1}]$ for $i = 0, 1, \ldots, n - 1$. 
5. Basic Probability Theory

1. CDF and PDF.

Let $(\Omega, \mathcal{F}, P)$ be a probability space, $X$ be a random variable. The cumulative distribution function (cdf) $F$ of $X$ is defined by

$$F(x) = P(X \leq x), \quad x \in \mathbb{R}.$$ 

$F$ is an increasing right-continuous function satisfying

$$F(-\infty) = 0, \quad F(+\infty) = 1.$$ 

If $F$ is absolutely continuous then $X$ has a probability density function (pdf) $f$ defined by

$$f(x) = F'(x), \quad x \in \mathbb{R}.$$ 

$F$ can be recovered from $f$ by the relation

$$F(x) = \int_{-\infty}^{x} f(u) \, du.$$

A random variable $X$ has a normal distribution with parameters $\mu$ and $\sigma^2$, written $X \sim N(\mu, \sigma^2)$, if $X$ has the pdf

$$
\phi(x) = \frac{1}{\sigma \sqrt{2\pi}} e^{-\frac{(x-\mu)^2}{2\sigma^2}}
$$

for $x \in \mathbb{R}$.

If $\mu = 0$ and $\sigma^2 = 1$ then $X$ is called a standard normal random variable and its cdf is usually written as

$$
\Phi(x) = \int_{-\infty}^{x} \frac{1}{\sqrt{2\pi}} e^{-\frac{u^2}{2}} \, du.
$$

If $X \sim N(\mu, \sigma^2)$ then the characteristic function (Fourier transform) of $X$ is given by

$$
c(s) = E(e^{isX}) = e^{i\mu t - \frac{\sigma^2 s^2}{2}},
$$

where $i = \sqrt{-1}$. The moment generating function (mgf) is given by

$$
m(s) = E(e^{sX}) = e^{\mu t + \frac{\sigma^2 s^2}{2}}.
$$
3. **Approximation of Normal CDF.**

It is suggested that the standard normal cdf $\Phi(x)$ can be approximated by a “polynomial” $\tilde{\Phi}(x)$ as follows:

$$\tilde{\Phi}(x) := 1 - \Phi'(x) (a_1 k + a_2 k^2 + a_3 k^3 + a_4 k^4 + a_5 k^5)$$

when $x \geq 0$ and $\tilde{\Phi}(x) := 1 - \tilde{\Phi}(-x)$ when $x < 0$.

The parameters are given by $k = \frac{1}{1+\gamma x}$, $\gamma = 0.2316419$, $a_1 = 0.319381530$, $a_2 = -0.356563782$, $a_3 = 1.781477937$, $a_4 = -1.821255978$, and $a_5 = 1.330274429$. This approximation has a maximum absolute error less than $7.5 \times 10^{-8}$ for all $x$.

**C++ Exercise:** Write a program to compute $\Phi(x)$ with (3) and compare the result with that derived with the composite Simpson Rule (see Exercise 4.4). The input is a variable $x$ and the number of partitions $n$ over the interval $[0, x]$. The output is the results from the two methods and their error.
4. Lognormal Random Variable.

Let \( Y = e^X \) and \( X \) be a \( N(\mu, \sigma^2) \) random variable. Then \( Y \) is a lognormal random variable.

**Exercise:** Show that

\[
E(Y) = e^{\mu + \frac{1}{2} \sigma^2}, \quad E(Y^2) = e^{2\mu + 2\sigma^2}.
\]

5. An Important Formula in Pricing European Options.

If \( V \) is lognormally distributed and the standard deviation of ln \( V \) is \( s \) then \(^1\)

\[
E(\max(V - K, 0)) = E(V) \Phi(d_1) - K \Phi(d_2)
\]

where

\[
d_1 = \frac{1}{s} \ln \frac{E(V)}{K} + \frac{s}{2} \quad \text{and} \quad d_2 = d_1 - s.
\]

\(^1K\) will be later denoted by \( X \). We use a different notation here to avoid an abuse of notation, as \( K \) is not a random variable.
\[ E(V - K)^+ = E(V) \Phi(d_1) - K \Phi(d_2), \quad d_1 = \frac{1}{s} \ln \frac{E(V)}{K} + \frac{s}{2} \quad \text{and} \quad d_2 = d_1 - s. \]

**Proof.**

Let \( g \) be the pdf of \( V \). Then

\[
E(V - K)^+ = \int_{-\infty}^{\infty} (v - K)^+ g(v) \, dv = \int_{K}^{\infty} (v - K) g(v) \, dv.
\]

As \( V \) is lognormal, \( \ln V \) is normal \( \sim N(m, s^2) \), where \( m = \ln(E(V)) - \frac{1}{2} s^2 \).

Let \( Y := \frac{\ln V - m}{s} \), i.e. \( V = e^{m+sY} \). Then \( Y \sim N(0, 1) \) with pdf: \( \phi(y) = \frac{1}{\sqrt{2\pi}} e^{-\frac{y^2}{2}} \).

\[
E(V - K)^+ = E(e^{m+sY} - K)^+ = \int_{\ln K - m}^{\infty} (e^{m+sY} - K) \phi(y) \, dy
\]

\[
= \int_{\ln K - m}^{\infty} e^{m+sY} \phi(y) \, dy - K \int_{\ln K - m}^{\infty} \phi(y) \, dy
\]

\[
= I_1 - K I_2.
\]
\[
I_1 = \int_{\ln K - m/s}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-y^2/2 + m + s y} \, dy \\
= \int_{\ln K - m/s}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-(y-s)^2/2 + m + s^2/2} \, dy \\
= e^{m + s^2/2} \int_{\ln K - m/s}^{\infty} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} \, dy \\
= e^{m + s^2/2} \left[ 1 - \Phi \left( \frac{\ln K - m}{s} - s \right) \right] = e^{m + s^2/2} \Phi \left( \frac{-\ln K - m}{s} + s \right) \\
= e^{\ln E(V)} \Phi \left( \frac{-\ln K + \ln E(V) - s^2/2}{s} + s \right) \\
= E(V) \Phi \left( \frac{1}{s} \ln \frac{E(V)}{K} + s/2 \right) = E(V) \Phi(d_1),
\]

on recalling \( m = \ln(E(V)) - \frac{1}{2} s^2 \).
Similarly

\[ I_2 = 1 - \Phi \left( \frac{\ln K - m}{s} \right) = \Phi \left( - \frac{\ln K - m}{s} \right) = \Phi (d_1 - s) = \Phi (d_2). \]
6. **Correlated Random Variables.**

Assume $X = (X_1, \ldots, X_n)$ is an $n$-vector of random variables.

The mean of $X$ is an $n$-vector $\mu = (E(X_1), \ldots, E(X_n))$.

The covariance of $X$ is an $n \times n$-matrix $\Sigma$ with components

$$\Sigma_{ij} = (\text{Cov}X)_{ij} = E((X_i - \mu_i)(X_j - \mu_j)).$$

The variance of $X_i$ is given by $\sigma_i^2 = \Sigma_{ii}$ and the correlation between $X_i$ and $X_j$ is given by $\rho_{ij} = \frac{\Sigma_{ij}}{\sigma_i \sigma_j}$.

$X$ is called a multi-dimensional normal vector, written as $X \sim N(\mu, \Sigma)$, if $X$ has pdf

$$f(x) = \frac{1}{(2\pi)^{n/2}} \frac{1}{\det \Sigma}^{1/2} \exp \left\{ -\frac{1}{2} (x - \mu)^T \Sigma^{-1} (x - \mu) \right\}$$

where $\Sigma$ is a symmetric positive definite matrix.
7. Convergence.

Let \( \{X_n\} \) be a sequence of random variables. There are four types of convergence concepts associated with \( \{X_n\} \):

- **Almost sure convergence**, written \( X_n \xrightarrow{\text{a.s.}} X \), if there exists a null set \( N \) such that for all \( \omega \in \Omega \setminus N \) one has

\[
X_n(\omega) \to X(\omega), \quad n \to \infty.
\]

- **Convergence in probability**, written \( X_n \xrightarrow{\text{P}} X \), if for every \( \varepsilon > 0 \) one has

\[
P(|X_n - X| > \varepsilon) \to 0, \quad n \to \infty.
\]

- **Convergence in norm**, written \( X_n \xrightarrow{\text{L}^p} X \), if \( X_n, X \in L^p \) and

\[
E|X_n - X|^p \to 0, \quad n \to \infty.
\]

- **Convergence in distribution**, written \( X_n \xrightarrow{D} X \), if for any \( x \) at which \( P(X \leq x) \) is continuous one has

\[
P(X_n \leq x) \to P(X \leq x), \quad n \to \infty.
\]
8. **Strong Law of Large Numbers.**

Let \( \{X_n\} \) be independent, identically distributed (iid) random variables with finite expectation \( E(X_1) = \mu \). Then

\[
\frac{Z_n}{n} \xrightarrow{a.s.} \mu
\]

where \( Z_n = X_1 + \cdots + X_n \).

9. **Central Limit Theorem.** Let \( \{X_n\} \) be iid random variables with finite expectation \( \mu \) and finite variance \( \sigma^2 > 0 \). For each \( n \), let

\[
Z_n = X_1 + \cdots + X_n.
\]

Then

\[
\frac{Z_n - \mu}{\frac{\sigma}{\sqrt{n}}} = \frac{Z_n - n\mu}{\sqrt{n}\sigma} \xrightarrow{D} Z
\]

where \( Z \) is a \( N(0,1) \) random variable, i.e.,

\[
P\left( \frac{Z_n - n\mu}{\sqrt{n}\sigma} \leq z \right) \rightarrow \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{z} e^{-\frac{x^2}{2}} \, dx, \quad n \to \infty.
\]
10. **Lindeberg–Feller Central Limit Theorem.**

Suppose $X$ is a triangular array of random variables, i.e.,

$$X = \{X_1^n, X_2^n, \ldots, X_{k(n)}^n : n \in \{1, 2, \ldots\}\}, \text{ with } k(n) \to \infty \text{ as } n \to \infty,$$

such that, for each $n$, $X_1^n, \ldots, X_{k(n)}^n$ are independently distributed and are bounded in absolute value by a constant $y_n$ with $y_n \to 0$. Let

$$Z_n = X_1^n + \cdots + X_{k(n)}^n.$$

If $E(Z_n) \to \mu$ and $\text{var}(Z_n) \to \sigma^2 > 0$, then $Z_n$ converges in distribution to a normally distributed random variable with mean $\mu$ and variance $\sigma^2$.

Note: Lindeberg–Feller implies the Central Limit Theorem (5.9).
If $X_1, X_2, \ldots$ are iid with expectation $\mu$ and variance $\sigma^2$, then define

$$X_i^n := \frac{X_i - \mu}{\sqrt{n} \sigma}, \quad i = 1, 2, \ldots, k(n) := n.$$ 

For each $n$, $X_1^n, \ldots, X_{k(n)}^n$ are independent and

$$E(X_i^n) = \frac{E(X_i) - \mu}{\sqrt{n} \sigma} = \frac{\mu - \mu}{\sqrt{n} \sigma} = 0,$$

$$\text{Var}(X_i^n) = \frac{1}{n \sigma^2} \text{Var}X_i - \mu = \frac{1}{n \sigma^2} \sigma^2 = \frac{1}{n}.$$ 

Let $Z_n = X_1^n + \cdots + X_{k(n)}^n = (\sum_{i=1}^n X_i - n \mu) \frac{1}{\sqrt{n \sigma}}$, then

$$E(Z_n) = \sum_{i=1}^{k(n)} E(X_i^n) = 0,$$

$$\text{Var}(Z_n) = \sum_{i=1}^{k(n)} \text{Var}(X_i^n) = 1.$$ 

Hence, by Lindeberg–Feller,

$$Z_n \xrightarrow{D} Z \sim N(0, 1).$$
6. Optimization

1. Unconstrained Optimization.

Given $f : \mathbb{R}^n \rightarrow \mathbb{R}$.

Minimize $f(x)$ over $x \in \mathbb{R}^n$.

$f$ has a local minimum at a point $\bar{x}$ if $f(\bar{x}) \leq f(x)$ for all $x$ near $\bar{x}$, i.e.

$$\exists \varepsilon > 0 \quad s.t. \quad f(\bar{x}) \leq f(x) \quad \forall x : \|x - \bar{x}\| < \varepsilon .$$

$f$ has a global minimum at $\bar{x}$ if

$$f(\bar{x}) \leq f(x) \quad \forall x \in \mathbb{R}^n .$$
2. Optimality Conditions.

- **First order necessary conditions:**
  Suppose that $f$ has a local minimum at $\bar{x}$ and that $f$ is continuously differentiable in an open neighbourhood of $\bar{x}$. Then $\nabla f(\bar{x}) = 0$. ($\bar{x}$ is called a *stationary point*.)

- **Second order sufficient Conditions:**
  Suppose that $f$ is twice continuously differentiable in an open neighbourhood of $\bar{x}$ and that $\nabla f(\bar{x}) = 0$ and $\nabla^2 f(\bar{x})$ is positive definite. Then $\bar{x}$ is a strict local minimizer of $f$.

**Example:** Show that $f = (2x_1^2 - x_2)(x_1^2 - 2x_2)$ has a minimum at $(0, 0)$ along any straight line passing through the origin, but $f$ has no minimum at $(0, 0)$.

**Exercise:** Find the minimum solution of

$$f(x_1, x_2) = 2x_1^2 + x_1x_2 + x_2^2 - x_1 - 3x_2.$$ \hspace{1cm} (4)

*(Answer. $\frac{1}{7}(-1, 11).$)*
Sufficient Condition.

Taylor gives for any \( d \in \mathbb{R}^n \):

\[
f(\bar{x} + d) = f(\bar{x}) + \nabla f(\bar{x})^T d + \frac{1}{2} d^T \nabla^2 f(\bar{x} + \lambda d) d \quad \lambda \in (0, 1).
\]

If \( \bar{x} \) is not strict local minimizer, then

\[
\exists \{x_k\} \subset \mathbb{R}^n \setminus \{\bar{x}\} : \quad x_k \to \bar{x} \quad \text{s.t.} \quad f(x_k) \leq f(\bar{x}).
\]

Define \( d_k := \frac{x_k - \bar{x}}{\|x_k - \bar{x}\|} \). Then \( \|d_k\| = 1 \) and there exists a subsequence \( \{d_{k_j}\} \) such that \( d_{k_j} \to d_\star \) as \( j \to \infty \) and \( \|d_\star\| = 1 \). W.l.o.g. we assume \( d_k \to d_\star \) as \( k \to \infty \).

\[
f(\bar{x}) \geq f(x_k) = f(\bar{x} + \|x_k - \bar{x}\| d_k)
\]

\[
= f(\bar{x}) + \|x_k - \bar{x}\| \nabla f(\bar{x})^T d_k + \frac{1}{2} \|x_k - \bar{x}\|^2 d_k^T \nabla^2 f(\bar{x} + \lambda_k \|x_k - \bar{x}\| d_k) d_k
\]

\[
= f(\bar{x}) + \frac{1}{2} \|x_k - \bar{x}\|^2 d_k^T \nabla^2 f(\bar{x} + \lambda_k \|x_k - \bar{x}\| d_k) d_k.
\]

Hence \( d_k^T \nabla^2 f(\bar{x} + \lambda_k \|x_k - \bar{x}\| d_k) d_k \leq 0 \), and on letting \( k \to \infty \)

\[
d_k^T \nabla^2 f(\bar{x}) d_\star \leq 0.
\]

As \( d_\star \neq 0 \), this is a contradiction to \( \nabla^2 f(\bar{x}) \) being symmetric positive definite. Hence \( \bar{x} \) is a strict local minimizer.
Example 6.2. Show that \( f = (2x_1^2 - x_2)(x_1^2 - 2x_2) \) has a minimum at \((0, 0)\) along any straight line passing through the origin, but \( f \) has no minimum at \((0, 0)\).

Answer.

Straight line through \((0, 0)\): \( x_2 = \alpha x_1, \alpha \in \mathbb{R} \) fixed.

\[
g(r) := f(r, \alpha r) = (2r^2 - \alpha r)(r^2 - 2\alpha r)
\]

\[
g'(r) = 8r^3 - 15\alpha r^2 + 4\alpha^2 r,
\]

\[
g''(r) = 24r^2 - 30\alpha r + 4\alpha^2
\]

\[
\Rightarrow g'(0) = 0 \quad \text{and} \quad g''(0) = 4\alpha^2 > 0.
\]

Hence \( r = 0 \) is a minimizer for \( g \iff (0, 0) \) is a minimizer for \( f \) along any straight line.

Now let \( (x_1^k, x_2^k) = (\frac{1}{k}, \frac{1}{k^2}) \rightarrow (0, 0) \) as \( k \rightarrow \infty \). Then

\[
f(x_1^k, x_2^k) = -\frac{1}{k^2} \cdot \frac{1}{k^2} < 0 = f(0, 0) \quad \forall k.
\]

Hence \((0, 0)\) is not a minimizer for \( f \).

[Note: \( \nabla f(0, 0) = 0 \), but \( \nabla^2 f(0, 0) = \begin{pmatrix} 0 & 0 \\ 0 & 4 \end{pmatrix} \).]

**Exercise.** When $f$ is convex, any local minimizer $\bar{x}$ is a global minimizer of $f$. If in addition $f$ is differentiable, then any stationary point $\bar{x}$ is a global minimizer of $f$.

(*Hint.* Use a contradiction argument.)
Exercise 6.3.
When $f$ is convex, any local minimizer $\bar{x}$ is a global minimizer of $f$.

Proof.
Suppose $\bar{x}$ is a local minimizer, but not a global minimizer. Then

$$\exists \, \tilde{x} \quad \text{s.t.} \quad f(\tilde{x}) < f(\bar{x}).$$

Since $f$ is convex, we have that

$$f(\lambda \tilde{x} + (1 - \lambda) \bar{x}) \leq \lambda f(\tilde{x}) + (1 - \lambda) f(\bar{x})$$
$$< \lambda f(\bar{x}) + (1 - \lambda) f(\bar{x}) = f(\bar{x}) \quad \forall \lambda \in (0, 1].$$

Let $x_\lambda := \lambda \tilde{x} + (1 - \lambda) \bar{x}$. Then

$$x_\lambda \to \bar{x} \quad \text{and} \quad f(x_\lambda) < f(\bar{x}) \quad \text{as} \lambda \to 0.$$

This is a contradiction to $\bar{x}$ being a local minimizer.
Hence $\bar{x}$ is a global minimizer for $f$. 

92
4. **Line Search.**

The basic procedure to solve numerically an unconstrained problem (minimize $f(x)$ over $x \in \mathbb{R}^n$) is as follows.

(i) Choose an initial point $x^0 \in \mathbb{R}^n$ and an initial search direction $d^0 \in \mathbb{R}^n$ and set $k = 0$.

(ii) Choose a step size $\alpha_k$ and define a new point $x^{k+1} = x^k + \alpha_k d^k$. Check if the stopping criterion is satisfied ($\|\nabla f(x^{k+1})\| < \varepsilon$?). If yes, $x^{k+1}$ is the optimal solution, stop. If no, go to (iii).

(iii) Choose a new search direction $d^{k+1}$ (*descent direction*) and set $k = k + 1$. Go to (ii).

The essential and most difficult part in any search algorithm is to choose a descent direction $d^k$ and a step size $\alpha_k$ with good convergence and stability properties.
5. **Steepest Descent Method.**

\( f \) is differentiable.

Choose \( d^k = -g^k \), where \( g^k = \nabla f(x^k) \), and choose \( \alpha_k \) s.t.

\[
    f(x^k + \alpha_k d^k) = \min_{\alpha \in \mathbb{R}} f(x^k + \alpha d^k).
\]

Note that the successive descent directions are orthogonal to each other, i.e. \( (g^k)^T g^{k+1} = 0 \), and the convergence for some functions may be very slow, called *zigzagging*.

**Exercise.**

Use the steepest descent (SD) method to solve (4) with the initial point \( x^0 = (1, 1) \).

\( \text{(Answer: First three iterations give } x^1 = (0, 1), x^2 = (0, \frac{3}{2}), \text{ and } x^3 = (-\frac{1}{8}, \frac{3}{2}). \)
**Steepest Descent.**

Taylor gives:

\[
    f(x^k + \alpha d_k) = f(x^k) + \alpha \nabla f(x^k)^T d^k + O(\alpha^2).
\]

As

\[
    \nabla f(x^k)^T d^k = \|\nabla f(x^k)\| \|d^k\| \cos \theta^k,
\]

with \(\theta^k\) the angle between \(d^k\) and \(\nabla f(x^k)\), we see that \(d^k\) is a descent direction if \(\cos \theta^k < 0\). The descent is steepest when \(\theta^k = \pi \iff \cos \theta^k = -1\).

**Zigzagging.**

\(\alpha_k\) is minimizer of \(\phi(\alpha) := f(x^k + \alpha d^k)\) with \(d^k = -g^k\).

Hence

\[
    0 = \phi'(\alpha_k) = \nabla f(x_k + \alpha_k d^k)^T d^k = \nabla f(x^{k+1})^T (-g^k) = -(g^{k+1})^T g^k.
\]

Hence \(d^{k+1} \perp d^k\), which leads to zigzagging.
Exercise 6.5.
Use the SD method to solve (4) with the initial point $x^0 = (1, 1)$. [min: $\frac{1}{7}(-1, 11).$]

Answer.

$\nabla f = (4x_1 + x_2 - 1, 2x_2 + x_1 - 3)$.

Iteration 0: $d^0 = -\nabla f(x^0) = -(4, 0) \neq (0, 0)$.

$\phi(\alpha) = f(x^0 + \alpha d^0) = f(1 - 4\alpha, 1) = 2(1 - 4\alpha)^2 - 2$

minimum point at $\alpha_0 = \frac{1}{4}$

$\Rightarrow \quad x^1 = x^0 + \alpha_0 d^0 = (0, 1),$

d^1 = -\nabla f(x^1) = -(0, -1) = (0, 1) \neq (0, 0)$.

Iteration 1: $x^2 = (0, \frac{3}{2}), \; d^2 = (-\frac{1}{2}, 0)$.

Iteration 2: $x^3 = (-\frac{1}{8}, \frac{3}{2}), \; d^3 = (0, \frac{1}{8})$. 

\[96\]
6. **Newton Method.**

   \( f \) is twice differentiable.

   Choose \( d^k = -[H^k]^{-1}g^k \), where \( H^k = \nabla^2 f(x^k) \).

   Set \( x^{k+1} = x^k + d^k \).

   If \( H^k \) is positive definite then \( d^k \) is a descent direction.

   The main drawback of the Newton method is that it requires the computation of \( \nabla^2 f(x^k) \) and its inverse, which can be difficult and time-consuming.

   **Exercise.**

   Use the Newton method to solve (4) with \( x^0 = (1, 1) \).

   \((Answer: \text{ First iteration gives } x^1 = \frac{1}{i}(-1, 11).)\)
Newton Method.

Taylor gives

\[ f(x^k + d) \approx f(x^k) + d^T \nabla f(x^k) + \frac{1}{2} d^T \nabla^2 f(x^k) d =: m(d) \]

\[ \min_d m(d) \Rightarrow \nabla m(d) = 0 \]

\[ \Rightarrow \nabla f(x^k) + \nabla^2 f(x^k) d = 0. \]

Hence choose \( d^k = -[\nabla^2 f(x^k)]^{-1} \nabla f(x^k) = -[H^k]^{-1} g^k. \)

If \( H^k \) is positive definite, then so is \( (H^k)^{-1} \), and we get

\[ (d^k)^T g^k = -(g^k)^T (H^k)^{-1} g^k \leq -\sigma_k \|g^k\|^2 < 0 \]

for some \( \sigma_k > 0. \)

Hence \( d^k \) is a descent direction.

[Aside: The Newton method for \( \min_x f(x) \) is equivalent to the Newton method for finding a root of the system of nonlinear equations \( \nabla f(x) = 0. \)]
**Exercise 6.6.**

Use the Newton method to minimize

\[ f(x_1, x_2) = 2x_1^2 + x_1x_2 + x_2^2 - x_1 - 3x_2 \]

with \( x^0 = (1, 1)^T \).

**Answer.**

\[ \nabla f = \begin{pmatrix} 4x_1 + x_2 - 1 \\ 2x_2 + x_1 - 3 \end{pmatrix}, \quad H := \nabla^2 f = \begin{pmatrix} 4 & 1 \\ 1 & 2 \end{pmatrix}. \]

\[ H^{-1} = \frac{1}{\det H} \begin{pmatrix} 2 & -1 \\ -1 & 4 \end{pmatrix} = \frac{1}{7} \begin{pmatrix} 2 & -1 \\ -1 & 4 \end{pmatrix}. \]

Iteration 0: \( x^0 = (1, 1)^T, \nabla f(x^0) = (4, 0)^T \).

\[ x^1 = x^0 - [H^0]^{-1} \nabla f(x^0) = \begin{pmatrix} 1 \\ 1 \end{pmatrix} - \frac{1}{7} \begin{pmatrix} 2 & -1 \\ -1 & 4 \end{pmatrix} \begin{pmatrix} 4 \\ 0 \end{pmatrix} = \frac{1}{7} \begin{pmatrix} -1 \\ 11 \end{pmatrix}. \]

\[ \Rightarrow \nabla f(x^1) = (0, 0)^T \text{ and } H \text{ positive definite}. \]

\[ \Rightarrow x^1 \text{ is minimum point}. \]
7. Choice of Stepsize.

In computing the step size $\alpha_k$ we face a tradeoff. We would like to choose $\alpha_k$ to give a substantial reduction of $f$, but at the same time we do not want to spend too much time making the choice. The ideal choice would be the global minimizer of the univariate function $\phi : \mathbb{R} \to \mathbb{R}$ defined by

$$\phi(\alpha) = f(x^k + \alpha d^k), \quad \alpha > 0,$$

but in general, it is too expensive to identify this value.

A common strategy is to perform an inexact line search to identify a step size that achieves adequate reductions in $f$ with minimum cost.

$\alpha$ is normally chosen to satisfy the Wolfe conditions:

$$f(x^k + \alpha_k d^k) \leq f(x^k) + c_1 \alpha_k (g^k)^T d^k$$  \hspace{1cm} (5)

$$\nabla f(x^k + \alpha_k d^k)^T d^k \geq c_2 (g^k)^T d^k,$$  \hspace{1cm} (6)

with $0 < c_1 < c_2 < 1$. (5) is called the sufficient decrease condition, and (6) is the curvature condition.
Choice of Stepsize.

The simple condition

\[ f(x^k + \alpha_k d^k) < f(x^k) \]  

is not appropriate, as it may not lead to a sufficient reduction.

Example: \( f(x) = (x - 1)^2 - 1 \). So \( \min f(x) = -1 \), but we can choose \( x^k \) satisfying \((\dagger)\) such that \( f(x^k) = \frac{1}{k} \to 0 \).

Note that the sufficient decrease condition (5)

\[ \phi(\alpha) = f(x^k + \alpha d^k) \leq \ell(\alpha) := f(x^k) + c_1 \alpha (g^k)^T d^k \]

yields acceptable regions for \( \alpha \). Here \( \phi(\alpha) < \ell(\alpha) \) for small \( \alpha > 0 \), as \((g^k)^T d^k < 0\) for descent directions.

The curvature condition (6) is equivalent to

\[ \phi'(\alpha) \geq c_2 \phi'(0) \quad [ > \phi'(0) ] \]

i.e. a condition on the desired slope and so rules out unacceptably short steps \( \alpha \). In practice \( c_1 = 10^{-4} \) and \( c_2 = 0.9 \).

An algorithm is said to be *globally convergent* if

\[
\lim_{k \to \infty} \|g^k\| = 0.
\]

It can be shown that if the step sizes satisfy the Wolfe conditions

* then the steepest descent method is globally convergent,
* so is the Newton method provided the Hessian matrices \(\nabla^2 f(x^k)\) have a bounded condition number and are positive definite.

**Exercise.** Show that the steepest descent method is globally convergent if the following conditions hold

(a) \(\alpha_k\) satisfies the Wolfe conditions,

(b) \(f(x) \geq M\quad \forall x \in \mathbb{R}^n\),

(c) \(f \in C^1\) and \(\nabla f\) is Lipschitz: \(\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\|\quad \forall x, y \in \mathbb{R}^n\).
[Hint: Show that $\sum_{k=0}^{\infty} \|g^k\|^2 < \infty$.]
Exercise 6.8.

Assume that $d^k$ is a descent direction, i.e. $(g^k)^T d^k < 0$, where $g^k := \nabla f(x^k)$. Then if

1. $\alpha_k$ satisfies the Wolfe conditions,

2. $f(x) \geq M \quad \forall \ x \in \mathbb{R}^n$,

3. $f \in C^1$ and $\nabla f$ is Lipschitz, i.e. $\|\nabla f(x) - \nabla f(y)\| \leq L \|x - y\| \quad \forall \ x, y \in \mathbb{R}^n$,

it holds that $\sum_{k=0}^{\infty} \cos^2 \theta^k \|g^k\|^2 < \infty$, where $\cos \theta^k := \frac{(g^k)^T d^k}{\|g^k\| \|d^k\|}$.

[Note: SD method is special case with $\cos^2 \theta^k = 1. \quad \Rightarrow \lim_{k \to \infty} \|g^k\| = 0.$]

Proof.

Wolfe condition (6) \quad \Rightarrow \quad (g^{k+1})^T d^k \geq c_2 (g^k)^T d_k

\Rightarrow \quad (g^{k+1} - g^k)^T d^k \geq (c_2 - 1) (g^k)^T d_k. \quad (\dagger)
The Lipschitz condition yields that
\[
(g^{k+1} - g^k)^T d^k \leq \|g^{k+1} - g^k\| \|d^k\| = \|\nabla f(x^{k+1}) - \nabla f(x^k)\| \|d^k\|
\leq L \|x^{k+1} - x^k\| \|d_k\| = \alpha_k L \|d^k\|^2. \tag{†}
\]

Combining (†) and (‡) gives \( \alpha_k \geq \frac{c_2 - 1}{L} \frac{(g^k)^T d^k}{\|d^k\|^2} \), and hence
\[
\alpha_k (g^k)^T d^k \leq \frac{c_2 - 1}{L} \frac{[(g^k)^T d^k]^2}{\|d^k\|^2}
\]

Together with Wolfe condition (5) we get
\[
f(x^{k+1}) \leq f(x^k) + c_1 \frac{c_2 - 1}{L} \frac{[(g^k)^T d^k]^2}{\|d^k\|^2} = f(x^k) - c \cos^2 \theta^k \|g^k\|^2,
\]
where $c := c_1 \frac{1-c_2}{L^2} > 0$. 

$$f(x^{k+1}) \leq f(x^k) - c \cos^2 \theta^k \|g^k\|^2 \leq f(x^0) - c \sum_{j=0}^{k} \cos^2 \theta^j \|g^j\|^2$$

$$\Rightarrow \sum_{j=0}^{k} \cos^2 \theta^j \|g^j\|^2 \leq \frac{1}{c} (f(x^0) - M) \quad \forall k \quad \Rightarrow \sum_{j=0}^{\infty} \cos^2 \theta^j \|g^j\|^2 < \infty.$$
9. **Popular Search Methods.**

In practice the steepest descent method and the Newton method are rarely used due to the slow convergence rate and the difficulty in computing Hessian matrices, respectively.

The popular search methods are

- the *conjugate gradient method* (variation of SD method with superlinear convergence) and
- the *quasi-Newton method* (variation of Newton method without computation of Hessian matrices).

There are some efficient algorithms based on the *trust-region* approach. See Fletcher (2000) for details.
10. **Constrained Optimization.**

Minimize $f(x)$ over $x \in \mathbb{R}^n$ subject to

the equality constraints

$$h_i(x) = 0, \quad i = 1, \ldots, l,$$

and the inequality constraints

$$g_j(x) \leq 0, \quad j = 1, \ldots, m.$$

Assume that all functions involved are differentiable.
11. **Linear Programming.**

The problem is to minimize

\[ z = c_1 x_1 + \cdots + c_n x_n \]

subject to

\[ a_{i1} x_1 + \cdots + a_{in} x_n \geq b_i, \quad i = 1, \ldots, m, \]

and

\[ x_1, \ldots, x_n \geq 0. \]

LPs can be easily and efficiently solved with the simplex algorithm or the interior point method.

MS-Excel has a good in-built LP solver capable of solving problems up to 200 variables. MATLAB with optimization toolbox also provides a good LP solver.
12. **Graphic Method.**

If an LP problem has only two decision variables \((x_1, x_2)\), then it can be solved by the graphic method as follows:

- First draw the feasible region from the given constraints and a contour line of the objective function,
- then, on establishing the increasing direction perpendicular to the contour line, find the optimal point on the boundary of the feasible region,
- then find two linear equations which define that point,
- and finally solve the two equations to obtain the optimal point.

**Exercise.** Use the graphic method to solve the LP:

minimize \(z = -3x_1 - 2x_2\)
subject to \(x_1 + x_2 \leq 80, 2x_1 + x_2 \leq 100, x_1 \leq 40,\) and \(x_1, x_2 \geq 0.\)

(Answer. \(x_1 = 20, x_2 = 60.\))
13. **Quadratic Programming.**

Minimize

$$x^T Q x + c^T x$$

subject to

$$A x \leq b \quad \text{and} \quad x \geq 0,$$

where $Q$ is an $n \times n$ symmetric positive definite matrix, $A$ is an $n \times m$ matrix, $x, c \in \mathbb{R}^n$, $b \in \mathbb{R}^m$.

To solve a QP problem, one

- first derives a set of equations from the Kuhn–Tucker conditions, and
- then applies the Wolfe algorithm or the Lemke algorithm to find the optimal solution.

The MS-Excel solver is capable of solving reasonably sized QP problems, similarly for MATLAB.

\[
\min f(x) \text{ over } x \in \mathbb{R}^n \text{ s.t. } h_i(x) = 0, \ i = 1, \ldots, l; \ g_j(x) \leq 0, \ j = 1, \ldots, m.
\]

Assume that \( \bar{x} \) is an optimal solution.

Under some regularity conditions, called the constraint qualifications, there exist two vectors \( \bar{u} = (\bar{u}_1, \ldots, \bar{u}_l) \) and \( \bar{v} = (\bar{v}_1, \ldots, \bar{v}_m) \), called the Lagrange multipliers, such that the following set of conditions is satisfied:

\[
L_{x_k}(\bar{x}, \bar{u}, \bar{v}) = 0, \ \ k = 1, \ldots, n
\]
\[
h_i(\bar{x}) = 0, \ \ i = 1, \ldots, l
\]
\[
g_j(\bar{x}) \leq 0, \ \ j = 1, \ldots, m
\]
\[
\bar{v}_j g_j(\bar{x}) = 0, \ \ \bar{v}_j \geq 0, \ \ j = 1, \ldots, m
\]

where

\[
L(x, u, v) = f(x) + \sum_{i=1}^{l} u_i h_i(x) + \sum_{j=1}^{m} v_j g_j(x)
\]

is called the Lagrange function or Lagrangian.
Furthermore, if $f : \mathbb{R}^n \rightarrow \mathbb{R}$ and $h_i, g_j : \mathbb{R}^n \rightarrow \mathbb{R}$ are convex, then $\bar{x}$ is an optimal solution if and only if $(\bar{x}, \bar{u}, \bar{v})$ satisfies the Kuhn–Tucker conditions. This holds in particular, when $f$ is convex and $h_i, g_j$ are linear.

**Example.**

Find the minimum solution to the function $x_2 - x_1$ subject to $x_1^2 + x_2^2 \leq 1$.

**Exercise.**

Find the minimum solution to the function $x_1^2 + x_2^2 - 2x_1 - 4x_2$ subject to $x_1 + 2x_2 \leq 2$ and $x_2 \geq 0$.

*(Answer. $(x_1, x_2) = \frac{1}{5} (2, 4)$).*
**Interpretation of Kuhn–Tucker conditions**

Assume that no equality constraints are present. If $\bar{x}$ is an interior point, i.e. no constraints are active, then we recover the usual optimality condition: $\nabla f(\bar{x}) = 0$.

Now assume that $\bar{x}$ lies on the boundary of the feasible set and let $g_{jk}$ be the active constraints at $\bar{x}$. Then a necessary condition for optimality is that we cannot find a descent direction for $f$ at $\bar{x}$ that is also a feasible direction. Such a vector cannot exist, if

$$-\nabla f(\bar{x}) = \sum_k \bar{v}_{jk} \nabla g_{jk}(\bar{x}) \quad \text{with} \quad \bar{v}_{jk} \geq 0.$$  (†)

This is because, if $d \in \mathbb{R}^n$ is a descent direction, then $\nabla f(\bar{x})^T d < 0$ and $\sum_k \bar{v}_{jk} \nabla g_{jk}(\bar{x})^T d > 0$.

So there must exist a $j_k$, such that $\nabla g_{jk}(\bar{x})^T d > 0$. But that means that $d$ is an ascent direction for $g_{jk}$, and as $g_{jk}$ is active at $\bar{x}$, it is not a feasible direction.

If we require $\bar{v}_j g_j(\bar{x}) = 0$ for all inactive constraints, then we can re-write (†) as $0 =$
\[ \nabla f(\bar{x}) + \sum_{j=1}^{m} \bar{v}_j \nabla g_j(\bar{x}) \] with \( \bar{v}_j \geq 0 \). These are the KT conditions.
Application of Kuhn–Tucker: LP Duality

Let $b \in \mathbb{R}^m$, $c \in \mathbb{R}^n$ and $A \in \mathbb{R}^{n \times m}$.

$$\min c^T x \quad \text{s.t.} \quad Ax \geq b, \quad x \geq 0. \quad (P)$$

Equivalent to $$\min c^T x \quad \text{s.t.} \quad b - Ax \leq 0, \quad -x \leq 0.$$ 

Lagrangian: $L = c^T x + v^T (b - Ax) + y^T (-x)$.

Hence, $\bar{x}$ is the solution, if there exist $\bar{v}$ and $\bar{y}$ such that

$$\nabla L = c - A^T \bar{v} - \bar{y} = 0 \quad \Rightarrow \quad \bar{y} = c - A^T \bar{v},$$

KT conditions: $$\bar{v}^T (b - A \bar{x}) = 0, \quad \bar{y}^T (-\bar{x}) = 0,$$

$$\bar{v}, \bar{y} \geq 0, \quad b - A \bar{x} \leq 0, \quad -\bar{x} \leq 0.$$
Eliminate $\bar{y}$ to find $\bar{v}, \bar{x}$:

\[
\begin{align*}
A \bar{x} & \geq b, \quad \bar{x} \geq 0 \quad \text{feasible region: primal} \\
A^T \bar{v} & \leq c, \quad \bar{v} \geq 0 \quad \text{feasible region: dual} \\
\bar{v}^T (b - A \bar{x}) &= 0 \\
\bar{x}^T (c - A^T \bar{v}) &= 0
\end{align*}
\]

\[\Rightarrow \quad \bar{x}^T c = \bar{x}^T A^T \bar{v} = \bar{v}^T b \]

Hence $\bar{v} \in \mathbb{R}^m$ solves the dual:

\[
\max \ b^T v \quad \text{s.t.} \quad A^T v \leq c, \quad v \geq 0 .
\] (D)
Here we have used that

\[
c^T \bar{x} = \min_{x \geq 0, \ A x \geq b} c^T x
\]

\[
\geq \min_{x \geq 0} \max_{v \geq 0} c^T x + v^T (b - A x)
\]

\[
= \max_{v \geq 0} \min_{x \geq 0} c^T x + v^T (b - A x)
\]

\[
= \max_{v \geq 0} \min_{x \geq 0} v^T b + x^T (c - A^T v)
\]

\[
\geq \max_{v \geq 0, \ A^T v \leq c} v^T b
\]

\[
\geq \bar{v}^T b = c^T \bar{x}
\]
Find the minimum solution to the function $x_2 - x_1$ subject to $x_1^2 + x_2^2 \leq 1$.

Answer.
$L = x_2 - x_1 + v (x_1^2 + x_2^2 - 1)$, so the KT conditions become

\begin{align*}
L_{x_1} &= -1 + 2v x_1 = 0 \quad (1) \\
L_{x_2} &= 1 + 2v x_2 = 0 \quad (2) \\
x_1^2 + x_2^2 &\leq 1 \quad (3) \\
v (x_1^2 + x_2^2 - 1) &= 0, \quad v \geq 0 \quad (4)
\end{align*}

(1) $\Rightarrow$ $v > 0$ and hence $x_1 = \frac{1}{2v}, \ x_2 = \frac{-1}{2v}$.

Plugging this into (4) yields $\frac{2}{4v^2} = 1$ and hence

$$
\bar{v} = \frac{1}{\sqrt{2}} \quad \Rightarrow \quad \bar{x}_1 = \frac{1}{\sqrt{2}}, \ \bar{x}_2 = \frac{-1}{\sqrt{2}};
$$

with the optimal value being $\bar{z} = -\sqrt{2}$. 
Example.

\[ \min x_1 \text{ s.t. } x_2 - x_1^3 \leq 0, \ x_1 \leq 1, \ x_2 \geq 0. \]

Since \( x_1^3 \geq x_2 \geq 0 \) we have \( x_1 \geq 0 \) and hence \( \bar{x} = (0, 0) \) is the unique minimizer.

Lagrangian: \( L = x_1 + v_1 (x_2 - x_1^3) + v_2 (x_1 - 1) + v_3 (-x_2). \)

KT conditions for a feasible point \( x \):

\[
\nabla L = \begin{pmatrix}
1 - 3v_1 x_1^2 + v_2 \\
v_1 - v_3
\end{pmatrix} = 0
\]

(1)

\[
v_1 (x_2 - x_1^3) = 0, \quad v_2 (x_1 - 1) = 0, \quad v_3 (-x_2) = 0
\]

(2)

\[
v_1, \ v_2, \ v_3 \geq 0
\]

(3)

Check KT conditions at \( \bar{x} = (0, 0) \):

(1) \( \Rightarrow \ v_2 = -1 < 0 \) impossible!

KT condition is not satisfied, since the constraint qualifications do not hold.

Here \( g_1 = x_2 - x_1^3 \) and \( g_3 = -x_2 \) are active at \( (0, 0) \), and \( \nabla g_1 = (0, 1) \), \( \nabla g_3 = (0, -1) \). Hence \( \nabla g_1 \) and \( \nabla g_3 \) are not linearly independent!
Exercise 6.14
Find the minimum solution to the function $x_1^2 + x_2^2 - 2x_1 - 4x_2$ subject to $x_1 + 2x_2 \leq 2$ and $x_2 \geq 0$.

Answer.
Lagrangian: $L = x_1^2 + x_2^2 - 2x_1 - 4x_2 + v_1(x_1 + 2x_2 - 2) + v_2(-x_2)$
KT conditions:

\[ \nabla L = \begin{pmatrix} 2x_1 - 2 + v_1 \\ 2x_2 - 4 + 2v_1 - v_2 \end{pmatrix} = 0 \quad (1) \]

\[ v_1(x_1 + 2x_2 - 2) = 0, \quad v_2x_2 = 0, \quad v_1, v_2 \geq 0 \quad (2) \]

\[ x_1 + 2x_2 \leq 2, \quad x_2 \geq 0 \quad (3) \]

If $x_1 + 2x_2 - 2 < 0$ then $v_1 = 0 \Rightarrow x_1 = 1, x_2 = 2 + \frac{1}{2}v_2 \geq 2$. Hence from (2), $v_2 = 0$, and so $x_1 = 1, x_2 = 2$. But that contradicts (3), and so it must hold that $x_1 + 2x_2 - 2 = 0$.

If $x_2 > 0$, then $v_2 = 0 \Rightarrow$ solving (1) together with $x_1 + 2x_2 - 2 = 0$ gives $x_1 = \frac{2}{5}$, $x_2 = \frac{4}{5}$; and $v_1 = \frac{6}{5}, v_2 = 0$. \hfill √
If $x_2 = 0$ then $x_1 = 2 - 2x_2 = 2$. But then from (1), $v_1 = -2 < 0$. Impossible.
7. Lattice (Tree) Methods


A call (or put) option is a contract that gives the holder the right to buy (or sell) a prescribed asset (underlying asset) by a certain date $T$ (expiration date) for a predetermined price $X$ (exercise price).

A European option can only be exercised on the expiration date while an American option can be exercised at any time prior to the expiration date.

The other party to the holder of the option contract is called the writer.

The holder and the writer are said to be in long and short positions of the contract, respectively.

The terminal payoffs of a European call (or put) option is $(S(T) - X)^+ := \max(S(T) - X, 0)$ (or $(X - S(T))^+$), where $S(T)$ is the underlying asset price at time $T$. 

2. Random Walk Model and Assumption.

Assume $S(t)$ and $V(t)$ are the asset price and the option price at time $t$, respectively, and the current time is $t$ and current asset price is $S$, i.e. $S(t) = S$.

After one period of time $\Delta t$, the asset price $S(t + \Delta t)$ is either $uS$ ("up" state) with probability $q$ or $dS$ ("down" state) with probability $1 - q$, where $q$ is a real (subjective) probability.

To avoid riskless arbitrage opportunities, we must have

$$u > R > d$$

where $R := e^{r \Delta t}$ and $r$ is the riskless interest rate.

Here $r$ is the riskless interest rate that allows for unlimited borrowing or lending at the same rate $r$.

Investing $1$ at time $t$ yields a value (return) at time $t + \Delta t$ of $\$R = \$(e^{r \Delta t})$. (continuous compounding).
Continuous compounding

Saving an amount $B$ at time $t$, yields with interest rate $r$ at time $t + \Delta t$ without compounding: $(1 + r \Delta t) B$.

Compounding the interest $n$ times yields

$$\left(1 + \frac{r \Delta t}{n}\right)^n B \rightarrow e^{r \Delta t} B \text{ as } n \rightarrow \infty.$$ 

Example:

Consider $\Delta t = 1$ and $r = 0.05$.

Then $R = e^r = 1.0512711$, equivalent to an AER of 5.127%.

<table>
<thead>
<tr>
<th>$n$</th>
<th>$(1 + \frac{r}{n})^n$</th>
<th>AER</th>
</tr>
</thead>
<tbody>
<tr>
<td>1</td>
<td>1.05</td>
<td>5 %</td>
</tr>
<tr>
<td>4</td>
<td>1.05094534</td>
<td>5.095 %</td>
</tr>
<tr>
<td>12</td>
<td>1.05116190</td>
<td>5.116 %</td>
</tr>
<tr>
<td>52</td>
<td>1.05124584</td>
<td>5.125 %</td>
</tr>
<tr>
<td>365</td>
<td>1.05126750</td>
<td>5.127 %</td>
</tr>
</tbody>
</table>
**No Arbitrage.**  \( u > R > d \)

If \( R \geq u \), then short sell \( \alpha > 0 \) units of the asset, deposit \( \alpha S \) in the bank; giving a portfolio value at time \( t \) of:

\[
\pi(t) = 0
\]

and at time \( t + \Delta t \) of

\[
\pi(t + \Delta t) = R(\alpha S') - \alpha S(t + \Delta t) \\
\geq R(\alpha S') - \alpha (u S) \geq 0.
\]

Moreover, in the “down” state (with probability \( 1 - q > 0 \)) the value is

\[
\pi(t + \Delta t) = R(\alpha S') - \alpha (d S') > 0.
\]

Hence we can make a riskless profit with positive probability.

No arbitrage implies \( u > R \).

A similar argument yields that \( d < R \).
3. **Replicating Portfolio.**

We form a portfolio consisting of

α units of underlying asset (α > 0 buying, α < 0 short selling) and
cash amount B in riskless cash bond (B > 0 lending, B < 0 borrowing).

If we choose

\[ \alpha = \frac{V(uS, t + \Delta t) - V(dS, t + \Delta t)}{uS - dS}, \]
\[ B = \frac{uV(dS, t + \Delta t) - dV(uS, t + \Delta t)}{uR - dR}, \]

we have replicated the payoffs of the option at time \( t + \Delta t \), no matter how the asset price changes.
Replicating Portfolio.

Value of portfolio at time $t$: $\pi(t) = \alpha S + B$, and at time $t + \Delta t$:

$$\pi(t + \Delta t) = \alpha S(t + \Delta t) + R B = \begin{cases} \alpha u S + R B =: \pi_u^\Delta t & \text{w. prob. } q \\ \alpha d S + R B =: \pi_d^\Delta t & \text{w. prob. } 1 - q \end{cases}$$

Call option value at time $t + \Delta t$ (expiration time):

$$C(t + \Delta t) = \begin{cases} (u S - X)^+ =: C_u^\Delta t & \text{w. prob. } q \\ (d S - X)^+ =: C_d^\Delta t & \text{w. prob. } 1 - q \end{cases}$$

To replicate the call option, we must have $\pi_u^\Delta t = C_u^\Delta t$ and $\pi_d^\Delta t = C_d^\Delta t$. Hence

$$\begin{align*}
\begin{cases}
\alpha u S + R B = C_u^\Delta t \\
\alpha d S + R B = C_d^\Delta t
\end{cases}
\end{align*} \iff
\begin{align*}
\begin{cases}
\alpha = \frac{C_u^\Delta t - C_d^\Delta t}{u S - d S} \\
B = \frac{u C_d^\Delta t - d C_u^\Delta t}{u R - d R}
\end{cases}
\end{align*}$$

This gives the fair price for the call option at time $t$: $\pi(t) = \alpha S + B$.  

128
Replicating Portfolio.

The value of the call option at time $t$ is:

$$C(t) = \pi(t) = \alpha S + B = \frac{C^\Delta_t - C_d^\Delta_t}{u S - d S} S + \frac{u C^\Delta_t - d C_u^\Delta_t}{u R - d R} = \frac{1}{R} \left[ \frac{R - d}{u - d} C_u^\Delta t + \frac{u - R}{u - d} C_d^\Delta t \right] = \frac{1}{R} \left[ p C_u^\Delta t + (1 - p) C_d^\Delta t \right],$$

where

$$p = \frac{R - d}{u - d}.$$

No arbitrage argument: If $C(t) < \pi(t)$, then buy call option at price $C(t)$ and sell portfolio at $\pi(t)$ (that is short sell $\alpha$ units of asset and lend $B$ amounts of cash to the bank). This gives an instantaneous profit of $\pi(t) - C(t) > 0$. And we know that at time $t + \Delta t$, the payoff from our call option will exactly compensate for the value of the portfolio, as $C(t + \Delta t) = \pi(t + \Delta t)$.

A similar argument for the case $C(t) > \pi(t)$ yields that $C(t) = \pi(t)$. 
4. Risk Neutral Option Price.

The current option price is given by

\[ V(S, t) = \frac{1}{R} \left( p V(uS, t + \Delta t) + (1 - p) V(dS, t + \Delta t) \right), \]  

(7)

where \( p = \frac{R - d}{u - d} \). Note that

1. the real probability \( q \) does not appear in the option pricing formula,
2. \( p \) is a probability \((0 < p < 1)\) since \( d < R < u \), and
3. \( p (uS) + (1 - p) (dS) = RS \), so \( p \) is called the risk neutral probability and the process of finding \( V \) is called the risk neutral valuation.
5. **Riskless Hedging Principle.**

We can also derive the option pricing formula (7) as follows:

form a portfolio with a long position of one unit of option and a short position of $\alpha$ units of underlying asset.

By choosing an appropriate $\alpha$ we can ensure such a portfolio is riskless.

**Exercise.**

Find $\alpha$ (called *delta*) and derive the pricing formula (7).

In a risk neutral continuous time model, the asset price $S$ is assumed to follow a lognormal process.

(as discussed in detail in Stochastic Processes I, omitted here).

Over a period $[t, t + \tau]$ the asset price $S(t + \tau)$ can be expressed in terms of $S(t) = S$ as

$$S(t + \tau) = S e^{(r - \frac{1}{2} \sigma^2) \tau + \sigma \sqrt{\tau} Z}$$

where $r$ is the riskless interest rate, $\sigma$ the volatility, and $Z \sim N(0, 1)$ a standard normal random variable.

$S(t + \tau)$ has the first moment $Se^{r \tau}$ and the second moment $S^2 e^{(2r + \sigma^2) \tau}$. 
7. Relations Between $u$, $d$ and $p$.

By equating the first and second moments of the asset price $S(t + \Delta t)$ in both continuous and discrete time models, we obtain the relation

\[
S \left( p u + (1 - p) d \right) = S e^{r \Delta t},
\]

\[
S^2 \left( p u^2 + (1 - p) d^2 \right) = S^2 e^{(2r+\sigma^2) \Delta t},
\]

or equivalently,

\[
pu + (1 - p)d = e^{r \Delta t}, \tag{8}
\]

\[
pu^2 + (1 - p)d^2 = e^{(2r+\sigma^2) \Delta t}. \tag{9}
\]

There are two equations and three unknowns $u, d, p$. An extra condition is needed to uniquely determine a solution.

[Note that (8) implies $p = \frac{R - d}{u - d}$, where $R = e^{r \Delta t}$ as before.]
8. **Cox–Ross–Rubinstein Model.**

The extra condition is

\[ u d = 1. \]  \hspace{1cm} (10)

The solutions to (8), (9), (10) are

\[ u = \frac{1}{2R} \left( \hat{\sigma}^2 + 1 + \sqrt{\left( \hat{\sigma}^2 + 1 \right)^2 - 4R^2} \right), \]

\[ d = \frac{1}{2R} \left( \hat{\sigma}^2 + 1 - \sqrt{\left( \hat{\sigma}^2 + 1 \right)^2 - 4R^2} \right), \]

where

\[ \hat{\sigma}^2 = e^{(2r+\sigma^2)\Delta t}. \]

If the higher order term \( O((\Delta t)^{\frac{3}{2}}) \) is ignored in \( u, d \), then

\[ u = e^{\sigma \sqrt{\Delta t}}, \quad d = e^{-\sigma \sqrt{\Delta t}}, \quad p = \frac{R - d}{u - d}. \]

These are the parameters chosen by Cox–Ross–Rubinstein for their model. Note that with this choice (9) is satisfied up to \( O((\Delta t)^2) \).
Proof.

\[
\hat{\sigma}^2 = e^{(2r + \sigma^2) \Delta t}
\]

\[
= p (u^2 - d^2) + d^2
\]

\[
= \frac{R - d}{u - d} (u + d) (u - d) + d^2
\]

\[
= Ru - d u + R d
\]

\[
= Ru - 1 + \frac{R}{u}
\]

\[\Rightarrow \quad Ru^2 - (1 + \hat{\sigma}^2) u + R = 0.\]

As \(u > d\), we get the unique solutions

\[
u_{\text{true}} = \frac{1 + \hat{\sigma}^2 + \sqrt{(1 + \hat{\sigma}^2)^2 - 4 R^2}}{2 R} = \frac{1 + \hat{\sigma}^2}{2 R} + \sqrt{\left(\frac{1 + \hat{\sigma}^2}{2 R}\right)^2 - 1}
\]

and

\[
d_{\text{true}} = \frac{1}{u} = \frac{1 + \hat{\sigma}^2 - \sqrt{(1 + \hat{\sigma}^2)^2 - 4 R^2}}{2 R} = \frac{1 + \hat{\sigma}^2}{2 R} - \sqrt{\left(\frac{1 + \hat{\sigma}^2}{2 R}\right)^2 - 1}.
\]
Moreover

\[
\frac{1 + \hat{\sigma}^2}{2 R} = \frac{1}{2} \left( e^{(2 r + \sigma^2) \Delta t} + 1 \right) e^{-r \Delta t}
\]

\[
= \frac{1}{2} (2 + (2 r + \sigma^2) \Delta t + O((\Delta t)^2)) \left( 1 - r \Delta t + O((\Delta t)^2) \right)
\]

\[
= \frac{1}{2} (2 + \sigma^2 \Delta t + O((\Delta t)^2))
\]

\[
= 1 + \frac{1}{2} \sigma^2 \Delta t + O((\Delta t)^2),
\]

and

\[
\sqrt{\left( \frac{1 + \hat{\sigma}^2}{2 R} \right)^2} - 1 = \sqrt{\left( 1 + \frac{1}{2} \sigma^2 \Delta t + O((\Delta t)^2) \right)^2} - 1
\]

\[
= \sqrt{1 + \sigma^2 \Delta t + O((\Delta t)^2) - 1}
\]

\[
= \sigma \sqrt{\Delta t \sqrt{1 + O(\Delta t)}}
\]

\[
= \sigma \sqrt{\Delta t (1 + O(\Delta t))}
\]

\[
= \sigma \sqrt{\Delta t} + O((\Delta t)^{3/2}).
\]

\[
\Rightarrow \quad u_{\text{true}} = 1 + \frac{1}{2} \sigma^2 \Delta t + \sigma \sqrt{\Delta t} + O((\Delta t)^{3/2}).
\]
The Cox–Ross–Rubinstein (CRR) model uses
\[ u = e^{\sigma \sqrt{\Delta t}} = 1 + \sigma \sqrt{\Delta t} + \frac{1}{2} \sigma^2 \Delta t + O((\Delta t)^{3/2}). \]

Hence the first three terms in the Taylor series of the CRR value \( u \) and the true value \( u_{\text{true}} \) match. So
\[ u = u_{\text{true}} + O((\Delta t)^{3/2}). \]

Estimating the error in (9) by this choice of \( u \) gives
\[
\text{Error} = pu^2 + (1 - p) d^2 - e^{(2r+\sigma^2) \Delta t} \\
= R(u + d) - 1 - e^{(2r+\sigma^2) \Delta t} \\
= e^{r \Delta t} \left(e^{\sigma \sqrt{\Delta t}} + e^{-\sigma \sqrt{\Delta t}}\right) - 1 - e^{(2r+\sigma^2) \Delta t} \\
= (1 + r \Delta t + O((\Delta t)^2)) \left(2 + \sigma^2 \Delta t + O((\Delta t)^2)\right) - 1 \\
- \left(1 + (2r + \sigma^2) \Delta t + O((\Delta t)^2)\right) \\
= O((\Delta t)^2).
\]
9. **Jarrow–Rudd Model.**

The extra condition is

\[ p = \frac{1}{2}. \]

(11)

**Exercise.**

Show that the solutions to (8), (9), (11) are

\[ u = R \left( 1 + \sqrt{e^{\sigma^2 \Delta t} - 1} \right), \]
\[ d = R \left( 1 - \sqrt{e^{\sigma^2 \Delta t} - 1} \right). \]

Show that if \( O((\Delta t)^{3/2}) \) is ignored then

\[ u = e^{(r - \frac{\sigma^2}{2}) \Delta t + \sigma \sqrt{\Delta t}}, \]
\[ d = e^{(r - \frac{\sigma^2}{2}) \Delta t - \sigma \sqrt{\Delta t}}. \]

(These are the parameters chosen by Jarrow–Rudd for their model.)

Also show that (8) and (9) are satisfied up to \( O((\Delta t)^2) \).
10. **Tian Model.**

The extra condition is

\[ p u^3 + (1 - p) d^3 = e^{3r \Delta t + 3 \sigma^2 \Delta t}. \]  

(12)

**Exercise.**

Show that the solutions to (8), (9), (12) are

\[ u = \frac{RQ}{2} \left( Q + 1 + \sqrt{Q^2 + 2Q - 3} \right), \]
\[ d = \frac{RQ}{2} \left( Q + 1 - \sqrt{Q^2 + 2Q - 3} \right), \]
\[ p = \frac{R - d}{u - d}, \]

where \( R = e^{r \Delta t} \) and \( Q = e^{\sigma^2 \Delta t}. \)

Also show that if \( O((\Delta t)^{\frac{3}{2}}) \) is ignored, then

\[ p = \frac{1}{2} - \frac{3}{4} \sigma \sqrt{\Delta t}. \]
(Note that $u d = R^2 Q^2$ instead of $u d = 1$ and that the binomial tree loses its symmetry about $S$ whenever $u d \neq 1$.)
11. **Black–Scholes Equation (BSE).**

As the time interval \( \Delta t \) tends to zero, the one period option pricing formula, (7) with (8) and (9), tends to the Black–Scholes Equation

\[
\frac{\partial V}{\partial t} + r S \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} - r V = 0. \tag{13}
\]

**Proof.**

Two variable Taylor expansion at \((S, t)\) gives

\[
V(uS, t + \Delta t) = V + V_S (uS - S) + V_t \Delta t \\
+ \frac{1}{2} \left( V_{SS} (uS - S)^2 + 2 V_{St} (uS - S) \Delta t + V_{tt} (\Delta t)^2 \right) \\
+ \text{higher order terms}.
\]

Similarly,

\[
V(dS, t + \Delta t) = V + V_S (dS - S) + V_t \Delta t \\
+ \frac{1}{2} \left( V_{SS} (dS - S)^2 + 2 V_{St} (dS - S) \Delta t + V_{tt} (\Delta t)^2 \right) \\
+ \text{higher order terms}.
\]
Substituting into the right hand side of (7) gives

\[ V = \{ V + S V_S [p (u - 1) + (1 - p) (d - 1)] + V_t \Delta t \\
+ \frac{1}{2} [S^2 V_{SS} (p (u - 1)^2 + (1 - p) (d - 1)^2) \\
+ 2 S V_{St} (p (u - 1) + (1 - p) (d - 1)) \Delta t \\
+ V_{tt} (\Delta t)^2] + \text{higher order terms} \} e^{-r \Delta t}. \]

Now it follows from (8) that

\[ p (u - 1) + (1 - p) (d - 1) = pu + (1 - p) d - 1 = e^{r \Delta t} - 1 = r \Delta t + O((\Delta t)^2). \]

Similarly, (8) and (9) give that

\[ p (u - 1)^2 + (1 - p) (d - 1)^2 = pu^2 + (1 - p) d^2 - 2 (pu + (1 - p) d) + 1 \\
= e^{(2r+\sigma^2) \Delta t} - 2 e^{r \Delta t} + 1 = \sigma^2 \Delta t + O((\Delta t)^2). \]
So, for the RHS of (7) we get

\[ V = \{ V + SV_S (r \Delta t + O((\Delta t)^2)) + V_t \Delta t \\
+ \frac{1}{2} \left[ S^2 V_{SS} (\sigma^2 \Delta t + O((\Delta t)^2)) + 2 V_{St} (r \Delta t + O((\Delta t)^2)) \Delta t \right] \\
+ O((\Delta t)^2) \} e^{-r \Delta t} \]

\[ = \{ V + (r SV_S + V_t + \frac{1}{2} \sigma^2 S^2 V_{SS}) \Delta t + O((\Delta t)^2) \} (1 - r \Delta t + O((\Delta t)^2)) \]

\[ = V + (-r V + r SV_S + V_t + \frac{1}{2} \sigma^2 S^2 V_{SS}) \Delta t + O((\Delta t)^2). \]

Cancelling \( V \) and dividing by \( \Delta t \) yields

\[-r V + r SV_S + V_t + \frac{1}{2} \sigma^2 S^2 V_{SS} + O(\Delta t) = 0.\]

Ignoring the \( O(\Delta t) \) term, we get the BSE.

\[ \Rightarrow \quad \text{The binomial model approximates BSE to 1}\text{st order accuracy.} \]
12. **n-Period Option Price Formula.**

For a multiplicative \( n \)-period binomial process, the call value is

\[
C = R^{-n} \sum_{j=0}^{n} \binom{n}{j} p^j (1 - p)^{n-j} \max(u^j d^{n-j} S - X, 0),
\]

(14)

where \( \binom{n}{j} = \frac{n!}{j!(n-j)!} \) is the binomial coefficient.

Define \( k \) to be the smallest non-negative integer such that \( u^k d^{n-k} S \geq X \).

The call pricing formula can be simplified as

\[
C = S \Psi(n, k, p') - X R^{-n} \Psi(n, k, p),
\]

(15)

where \( p' = \frac{u p}{R} \) and \( \Psi \) is the *complementary binomial distribution function* defined by

\[
\Psi(n, k, p) = \sum_{j=k}^{n} \binom{n}{j} p^j (1 - p)^{n-j}.
\]
Asset:

\[ S \rightarrow uS \]
\[ S \rightarrow dS \]

\[ u^2S \]
\[ d^2S \]

recombined tree

Call:

\[ C_{uu}^{2\Delta t} = (u^2S - X)^+ \]
\[ C_{ud}^{2\Delta t} = (udS - X)^+ \]
\[ C_{dd}^{2\Delta t} = (d^2S - X)^+ \]

↑
↑
↑

\[ t \]
\[ t + \Delta t \]
\[ t + 2\Delta t \]
Call price at time $t + \Delta t$:

$$C^{\Delta t}_u = \frac{1}{R} \left( p C_{u u}^{2 \Delta t} + (1 - p) C_{u d}^{2 \Delta t} \right), \quad C^{\Delta t}_d = \frac{1}{R} \left( p C_{u d}^{2 \Delta t} + (1 - p) C_{d d}^{2 \Delta t} \right).$$

Call price at time $t$:

$$C = \frac{1}{R} \left( p C^t_u + (1 - p) C^t_d \right) = \frac{1}{R^2} \left( p^2 C_{u u}^{2 \Delta t} + 2 p (1 - p) C_{u d}^{2 \Delta t} + (1 - p)^2 C_{d d}^{2 \Delta t} \right).$$

**Proof** of (14) by induction:

Assume (14) holds for $n \leq k - 1$, then for $n = k$, there are $k - 1$ periods between time $t + \Delta t$ and $t + k \Delta t$. So

$$C^{\Delta t}_u = \frac{1}{R^{k-1}} \sum_{j=0}^{k-1} \binom{k-1}{j} p^j (1 - p)^{k-1-j} \left( u^j d^{k-1-j} (u S - X) \right)^+$$

$$= \frac{1}{R^{k-1}} \sum_{j=1}^{k} \binom{k-1}{j-1} p^{j-1} (1 - p)^{k-j} \left( u^j d^{k-j} S - X \right)^+. $$
\[
\Rightarrow \quad p \ C_{u}^{\Delta t} = \frac{1}{R^{k-1}} \sum_{j=1}^{k} \binom{k-1}{j-1} p^{j} (1 - p)^{k-j} (u^{j} d^{k-j} S - X)^{+},
\]

and similarly

\[
(1 - p) \ C_{d}^{\Delta t} = \frac{1}{R^{k-1}} \sum_{j=0}^{k-1} \binom{k-1}{j} p^{j} (1 - p)^{k-j} (u^{j} d^{k-j} S - X)^{+}.
\]

Combining gives

\[
C = \frac{1}{R} \left( p \ C_{u}^{\Delta t} + (1 - p) \ C_{d}^{\Delta t} \right)
= \frac{1}{R^{k}} \left[ p^{k} (u^{k} S - X)^{+} + (1 - p)^{k} (d^{k} S - X)^{+} \right.
+ \sum_{j=1}^{k-1} \left\{ \binom{k-1}{j-1} + \binom{k-1}{j} \right\} p^{j} (1 - p)^{k-j} (u^{j} d^{k-j} S - X)^{+} \left. \right]\]

\[
= \binom{k}{j}
\]

This proves (14).
Let $k$ be the smallest non-negative integer such that

$$u^k d^{n-k}S \geq X \iff \left(\frac{u}{d}\right)^k \geq \frac{X}{S d^n} \iff k \geq \ln \frac{X}{S d^n} / \ln \frac{u}{d}.$$ 

Then

$$(u^j d^{n-j}S - X)^+ = \begin{cases} 0 & j < k \\ u^j d^{n-j}S - X & j \geq k \end{cases}.$$ 

Hence

$$C = \frac{1}{R^n} \sum_{j=k}^{n} \binom{n}{j} p^j (1-p)^{n-j} (u^j d^{n-j}S - X)$$

$$= S \sum_{j=k}^{n} \binom{n}{j} \left[ \frac{p u}{R} \right]^j \left[ \frac{(1-p) d}{R} \right]^{n-j} - X R^{-n} \sum_{j=k}^{n} \binom{n}{j} p^j (1-p)^{n-j}.$$ 

On setting $p' = \frac{p u}{R}$, it holds that

$$1 - p' = 1 - \frac{R - d}{u - d} \frac{u}{R} = -\frac{R d + u d}{R (u - d)} = \frac{u - R d}{u - d} = \frac{(1-p) d}{R}.$$ 

This proves (15).

As the time interval $\Delta t$ tends to zero, the $n$-period call price formula (15) tends to the Black–Scholes call option pricing formula

$$c = S \Phi(d_1) - X e^{-r(T-t)} \Phi(d_2)$$

(16)

where

$$d_1 = \frac{\ln \frac{S}{X} + (r + \frac{\sigma^2}{2})(T-t)}{\sigma \sqrt{T-t}}$$

and

$$d_2 = \frac{\ln \frac{S}{X} + (r - \frac{\sigma^2}{2})(T-t)}{\sigma \sqrt{T-t}} \equiv d_1 - \sigma \sqrt{T-t}.$$ 

Proof.

The $n$-period call price (15) is given by

$$C = S \Psi(n, k, p') - X R^{-n} \Psi(n, k, p) = S \Psi(n, k, p') - X e^{-r \cdot n \Delta t} e^{-r(T-t)} \Psi(n, k, p).$$
Hence we need to prove
\[
\Psi(n, k, p') \rightarrow \Phi(d_1) \quad \text{and} \quad \Psi(n, k, p) \rightarrow \Phi(d_2) \quad \text{as} \quad \Delta t \rightarrow 0.
\]
Let $j$ be the binomial RV of the number of upward moves in $n$ periods. Then

$$S_n = S u^j d^{n-j} = S \left( \frac{u}{d} \right)^j d^n,$$

where $S_n$ is the asset price at time $T = t + n \Delta t$.

$$\Rightarrow \quad \ln \frac{S_n}{S} = j \ln \frac{u}{d} + n \ln d = j \ln u + (n - j) \ln d,$$

$$E(j) = np, \quad \text{Var}(j) = np (1 - p),$$

$$E \left( \ln \frac{S_n}{S} \right) = np \ln \frac{u}{d} + n \ln d, \quad \text{Var} \left( \ln \frac{S_n}{S} \right) = np (1 - p) \left( \ln \frac{u}{d} \right)^2.$$  

Moreover,

$$1 - \Psi(n, k, p) = P(j \leq k - 1) = P \left( \ln \frac{S_n}{S} \leq (k - 1) \ln \frac{u}{d} + n \ln d \right).$$

As $k$ is the smallest integer such that $k \geq \ln \frac{X}{S d^n}/\ln \frac{u}{d}$, there exists an $\alpha \in (0, 1]$ such that

$$k - 1 = \ln \frac{X}{S d^n}/\ln \frac{u}{d} - \alpha,$$
and so

\[ 1 - \Psi(n, k, p) = P \left( \ln \frac{S_n}{S} \leq \ln \frac{X}{S} - \alpha \ln \frac{u}{d} \right). \]
Define

\[ X^n_i := \begin{cases} 
    \ln u & \text{with prob. } p \\
    \ln d & \text{with prob } 1 - p 
\end{cases}, \quad i = 1, \ldots, n, \]

and \( Z_n := X^n_1 + \ldots + X^n_n \).

Then, for each \( n \), \( \{X^n_1, \ldots, X^n_n\} \) are independent and \( Z_n \sim \ln \frac{S_n}{S} \). Hence

\[ 1 - \Psi(n, k, p) = P(Z_n \leq \beta_n), \quad \beta_n := \ln \frac{X}{S} - \alpha \ln \frac{u}{d}. \]

Assuming e.g. the CRR model, we get

\[ E(Z_n) = np \ln u + n(1 - p) \ln d \to \mu (T - t), \quad \text{where } \mu := r - \frac{\sigma^2}{2}, \]

\[ \text{Var}(Z_n) = np (1 - p) \ln^2 \frac{u}{d} \to \sigma^2 (T - t), \]

\[ \beta_n = \ln \frac{X}{S} - \alpha \ln \frac{u}{d} \to \ln \frac{X}{S}, \]

and \( |X^n_i| \leq |\sigma \sqrt{\Delta t}| = \frac{\sigma \sqrt{T - t}}{\sqrt{n}} \equiv y_n \to 0 \quad \text{as } n \to \infty. \)
Hence Theorem 5.10. implies that

$$Z_n \xrightarrow{D} Z \sim N(\mu (T - t), \sigma^2 (T - t)) .$$
So, as $n \to \infty$

\[ 1 - \Psi(n, k, p) = P(Z_n \leq \beta_n) \to \frac{1}{\sqrt{2\pi \sigma^2 (T - t)}} \int_{-\infty}^{\ln \frac{X}{S}} e^{-\frac{(s - \mu (T - t))^2}{2\sigma^2 (T - t)}} \, ds. \]

On letting

\[ y := \frac{s - \mu (T - t)}{\sigma \sqrt{T - t}}, \]

we have

\[ 1 - \Psi(n, k, p) \to \frac{1}{\sqrt{2\pi}} \int_{-\infty}^{\ln \frac{X}{S} - \mu (T - t)} e^{-\frac{y^2}{2}} \, dy = \Phi \left( \frac{\ln \frac{X}{S} - \mu (T - t)}{\sigma \sqrt{T - t}} \right). \]

Hence

\[ \Psi(n, k, p) \to 1 - \Phi \left( \frac{\ln \frac{X}{S} - \mu (T - t)}{\sigma \sqrt{T - t}} \right) = \Phi \left( - \frac{\ln \frac{X}{S} - \mu (T - t)}{\sigma \sqrt{T - t}} \right) \]

\[ = \Phi \left( \frac{\ln \frac{S}{X} + (r - \frac{\sigma^2}{2})(T - t)}{\sigma \sqrt{T - t}} \right) = \Phi(d_2). \]

Similarly one can show $\Psi(n, k, p') \to \Phi(d_1)$.

Choose the number of time steps that is required to march down until the expiration time \( T \) to be \( n \), so that \( \Delta t = \frac{T - t}{n} \).

At time \( t_i = t + i \Delta t \) there are \( i + 1 \) nodes.

Denote these nodes as \((i, j)\) with \( i = 0, 1, \ldots, n \) and \( j = 0, 1, \ldots, i \), where the first component \( i \) refers to time \( t_i \) and the second component \( j \) refers to the asset price \( S u^j d^{i-j} \).

Compute the option prices \( V^n_j = V(S u^j d^{n-j}, T), j = 0, 1, \ldots, n \), at expiration time \( t_n = T \) over \( n + 1 \) nodes.

Then go backwards one step and compute option prices at time \( t_{n-1} = T - \Delta t \) using the one period binomial option pricing formula over \( n \) nodes.

Continue until reaching the current time \( t_0 = t \).
The recursive formula is

\[ V^i_j = \frac{1}{R} \left( p V^{i+1}_{j+1} + (1 - p) V^{i+1}_j \right), \quad j = 0, \ldots, i \text{ and } i = n - 1, \ldots, 0, \]

where

\[ p = \frac{R - d}{u - d}. \]

The value \( V^0_0 = V(S, t) \) is the option price at the current time \( t \).

**C++ Exercise:** Write a program to implement the Cox–Ross–Rubinstein model to price options with a non-dividend paying underlying asset. The inputs are the asset price \( S \), the strike price \( X \), the riskless interest rate \( r \), the current time \( t \) and the maturity time \( T \), the volatility \( \sigma \), and the number of steps \( n \). The outputs are the European call and put option prices.
15. **Greeks.**

Assume the current time is $t = 0$. Some common sensitivity measures of a European call option price $c$ (the rate of change of the call price with respect to the underlying factors) are *delta*, *gamma*, *vega*, *rho*, *theta*, defined by

$$
\delta = \frac{\partial c}{\partial S}, \quad \gamma = \frac{\partial^2 c}{\partial S^2}, \quad v = \frac{\partial c}{\partial \sigma}, \quad \rho = \frac{\partial c}{\partial r}, \quad \theta = -\frac{\partial c}{\partial T}.
$$

**Exercise.**

Derive the following relations from the call price (16).

$$
\begin{align*}
\delta &= \Phi(d_1), \\
\gamma &= \frac{\Phi'(d_1)}{S \sigma \sqrt{T}}, \\
v &= S \sqrt{T} \Phi'(d_1), \\
\rho &= X T e^{-rT} \Phi(d_2), \\
\theta &= -\frac{S \Phi'(d_1) \sigma}{2 \sqrt{T}} - r X e^{-rT} \Phi(d_2).
\end{align*}
$$
16. **Dynamic Hedging.**

Greeks can be used to hedge an option position against changes of underlying factors. For example, to hedge a short position of a European call against changes of the asset price, one should keep a long position of the underlying asset with $\delta$ units. This strategy requires continuous trading and is prohibitively expensive if there are any transaction costs.
17. Dividends.

(a) If the asset pays a continuous dividend yield at a rate $q$, then use $r - q$ instead of $r$ in building the asset price tree, but still use $r$ in discounting the option values in the backward calculation.

In the risk-neutral model, the asset has a rate of return of $r - q$. So

(8) becomes \[ p u + (1 - p) d = e^{(r - q) \Delta t}, \]
(9) becomes \[ p u^2 + (1 - p) d^2 = e^{(2r - 2q + \sigma^2) \Delta t}. \]

However, the discount factor \(\frac{1}{R}\) remains the same:

\[
\frac{1}{R} = e^{-r \Delta t}. \]
(b) If the asset pays a discrete proportional dividend $\beta S$ between $t_{i-1}$ and $t_i$, then the asset price at node $(i, j)$ drops to $(1 - \beta) u^j d^{i-j} S$ instead of the usual $u^j d^{i-j} S$. The tree is still recombined.
(c) If the asset pays a discrete cash dividend $D$ between $t_{i-1}$ and $t_i$, then the asset price at node $(i, j)$ drops to $u^j d^{i-j} S - D$ instead of the usual $u^j d^{i-j} S$. There are $i + 1$ new recombined trees emanating from the nodes at time $t_i$, but the original tree is no longer recombined.
Suppose a cash dividend $D$ is paid between $t_{i-1}$ and $t_i$, then at time $t_{i+m}$ there are $(i + 1) (m + 1)$ nodes, instead of $i + m + 1$ in a recombined tree.

If there are several cash dividends, then the number of nodes is much larger than in a normal tree.

**Example.** Assume there is a cash dividend $D$ between periods 1 and 2. Then the possible asset prices at time $n = 2$ are $\{u^2 S - D, u d S - D, d^2 S - D\}$. At $n = 3$ there are 6 nodes: $u \cdot \{u^2 S - D, u d S - D, d^2 S - D\}$ and $d \cdot \{u^2 S - D, u d S - D, d^2 S - D\}$, instead of 4.

[Reason: $d (u^2 S - D) \neq u (u d S - D)$].
\[ u^2 \left( u^2 S - D \right) \]

\[ u^2 S - D \]

\[ u d S - D \]

\[ d^2 S - D \]

\[ d^2 \left( d^2 S - D \right) \]
18. **Trinomial Model.**

Suppose the asset price is \( S \) at time \( t \). At time \( t + \Delta t \) the asset price can be \( uS \) (up-state) with probability \( p_u \), \( mS \) (middle-state) with probability \( p_m \), and \( dS \) (down-state) with probability \( p_d \). Then the risk neutral one period option pricing formula is

\[
V(S, t) = \frac{1}{R} \left( p_u V(uS, t + \Delta t) + p_m V(mS, t + \Delta t) + p_d V(dS, t + \Delta t) \right).
\]

The trinomial tree approach is equivalent to the explicit finite difference method. By equating the first and second moments of the asset price \( S(t + \Delta t) \) in both continuous and discrete models, we obtain the following equations:

\[
\begin{align*}
p_u + p_m + p_d &= 1, \\
p_u u + p_m m + p_d d &= e^{r \Delta t}, \\
p_u u^2 + p_m m^2 + p_d d^2 &= e^{(2r + \sigma^2) \Delta t}.
\end{align*}
\]

There are three equations and six variables \( u, m, d \) and \( p_u, p_m, p_d \). We need three more equations to uniquely determine these variables.
19. **Boyle Model.**

The three additional conditions are

\[ m = 1, \quad u d = 1, \quad u = e^{\lambda \sigma \sqrt{\Delta t}}, \]

where \( \lambda \) is a free parameter.

**Exercise.**

Show that the risk-neutral probabilities are given by

\[ p_u = \frac{(W - R) u - (R - 1)}{(u - 1) (u^2 - 1)}, \quad p_m = 1 - p_u - p_d, \]
\[ p_d = \frac{(W - R) u^2 - (R - 1) u^3}{(u - 1) (u^2 - 1)}, \]

where \( R = e^{r \Delta t} \) and \( W = e^{(2r + \sigma^2) \Delta t} \).

Note that if \( \lambda = 1 \), which corresponds to the choice of \( u \) as in the CRR model, certain sets of parameters can lead to \( p_m < 0 \). To rectify this, choose \( \lambda > 1 \).
Boyle claimed that if $p_u \approx p_m \approx p_d \approx \frac{1}{3}$, then the trinomial scheme with 5 steps is comparable to the CRR binomial scheme with 20 steps.
20. **Hull–White Model.**

This is the same as the Boyle model with $\lambda = \sqrt{3}$.

**Exercise.**

Show that the risk-neutral probabilities are

$$
\begin{align*}
p_d &= \frac{1}{6} - \sqrt{\frac{\Delta t}{12\sigma^2}} \left( r - \frac{1}{2} \sigma^2 \right), \\
p_m &= \frac{2}{3}, \\
p_u &= \frac{1}{6} + \sqrt{\frac{\Delta t}{12\sigma^2}} \left( r - \frac{1}{2} \sigma^2 \right),
\end{align*}
$$

if terms of order $O(\Delta t)$ are ignored.
21. **Kamrad–Ritchken Model.**

If $S$ follows a lognormal process, then we can write $\ln S(t + \Delta t) = \ln S(t) + Z$, where $Z$ is a normal random variable with mean $(r - \frac{1}{2}\sigma^2)\Delta t$ and variance $\sigma^2 \Delta t$.

KR suggested to approximate $Z$ by a discrete random variable $Z^a$ as follows: $Z^a = \Delta x$ with probability $p_u$, $0$ with probability $p_m$, and $-\Delta x$ with probability $p_d$, where $\Delta x = \lambda \sigma \sqrt{\Delta t}$ and $\lambda \geq 1$.

The corresponding $u, m, d$ in the trinomial tree are $u = e^{\Delta x}$, $m = 1$, $d = e^{-\Delta x}$.

By omitting the higher order term $O((\Delta t)^2)$, they showed that the risk-neutral probabilities are

$$p_u = \frac{1}{2\lambda^2} + \frac{1}{2\lambda \sigma} \left( r - \frac{1}{2}\sigma^2 \right) \sqrt{\Delta t},$$

$$p_m = 1 - \frac{1}{\lambda^2},$$

$$p_d = \frac{1}{2\lambda^2} - \frac{1}{2\lambda \sigma} \left( r - \frac{1}{2}\sigma^2 \right) \sqrt{\Delta t}.$$  

Note that if $\lambda = 1$ then $p_m = 0$ and the trinomial scheme is reduced to the binomial
KR claimed that if $p_u = p_m = p_d = \frac{1}{3}$, then the trinomial scheme with 15 steps is comparable to the binomial scheme ($p_m = 0$) with 55 steps. They also discussed the trinomial scheme with two correlated state variables.
\[ Z^a = \begin{cases} 
\Delta x & \text{with prob. } p_u \\
0 & \text{with prob. } p_m \\
-\Delta x & \text{with prob. } p_d 
\end{cases} \]

and the equations become

\[ p_u + p_m + p_d = 1, \quad (1) \]

\[ E(Z^a) = \Delta x (p_u - p_d) = (r - \frac{1}{2} \sigma^2) \Delta t, \quad (2) \]

\[ \text{Var}(Z^a) = (\Delta x)^2 (p_u + p_d) - \underbrace{(\Delta x)^2 (p_u - p_d)^2}_{O((\Delta t)^2)} = \sigma^2 \Delta t. \]

Dropping the \( O((\Delta t)^2) \) term leads to

\[ (\Delta x)^2 (p_u + p_d) = \sigma^2 \Delta t. \quad (3) \]
Hence

\[ p_u = \frac{1}{2} \left[ \Delta t \left( \frac{\sigma^2}{(\Delta x)^2} + \frac{\Delta t}{\Delta x} \left( r - \frac{1}{2} \sigma^2 \right) \right) \right], \quad p_d = \frac{1}{2} \left[ \Delta t \left( \frac{\sigma^2}{(\Delta x)^2} - \frac{\Delta t}{\Delta x} \left( r - \frac{1}{2} \sigma^2 \right) \right) \right]. \]

\[ \Rightarrow \quad p_u = \frac{1}{2} \left[ \frac{1}{\lambda^2} + \frac{1}{\lambda \sigma} \left( r - \frac{1}{2} \sigma^2 \right) \sqrt{\Delta t} \right], \quad p_d = \frac{1}{2} \left[ \frac{1}{\lambda^2} - \frac{1}{\lambda \sigma} \left( r - \frac{1}{2} \sigma^2 \right) \sqrt{\Delta t} \right]. \]
8. Finite Difference Methods

1. Diffusion Equations of One State Variable.

\[
\frac{\partial u}{\partial t} = c^2 \frac{\partial^2 u}{\partial x^2}, \quad (x, t) \in D, 
\]

where \( t \) is a time variable, \( x \) is a state variable, and \( u(x, t) \) is an unknown function satisfying the equation.

To find a well-defined solution, we need to impose the initial condition

\[
u(x, 0) = u_0(x)
\]

and, if \( D = [a, b] \times [0, \infty) \), the boundary conditions

\[
u(a, t) = g_a(t) \quad \text{and} \quad u(b, t) = g_b(t),
\]

where \( u_0, g_a, g_b \) are continuous functions.
If \( D = (-\infty, \infty) \times (0, \infty) \), we need to impose the boundary conditions

\[
\lim_{|x| \to \infty} u(x, t) e^{-a x^2} = 0 \quad \text{for any } a > 0. \tag{20}
\]

(20) implies \( u(x, t) \) does not grow too fast as \( |x| \to \infty \).

The diffusion equation (17) with the initial condition (18) and the boundary conditions (19) is well-posed, i.e. there exists a unique solution that depends continuously on \( u_0 \), \( g_a \) and \( g_b \).
2. Grid Points.

To find a numerical solution to equation (17) with finite difference methods, we first need to define a set of grid points in the domain $D$ as follows:

Choose a state step size $\Delta x = \frac{b-a}{N}$ ($N$ is an integer) and a time step size $\Delta t$, draw a set of horizontal and vertical lines across $D$, and get all intersection points $(x_j, t_n)$, or simply $(j, n)$,

where $x_j = a + j \Delta x$, $j = 0, \ldots, N$, and $t_n = n \Delta t$, $n = 0, 1, \ldots$.

If $D = [a, b] \times [0, T]$ then choose $\Delta t = \frac{T}{M}$ ($M$ is an integer) and $t_n = n \Delta t$, $n = 0, \ldots, M$. 
\[ t_M = T \]
\[ t_0 = 0 \]
\[ x_0 = a \quad x_1 \quad x_2 \quad \ldots \quad x_N = b \]
3. **Finite Differences.**

The partial derivatives

\[ u_x := \frac{\partial u}{\partial x} \quad \text{and} \quad u_{xx} := \frac{\partial^2 u}{\partial x^2} \]

are always approximated by central difference quotients, i.e.

\[ u_x \approx \frac{u_{j+1}^n - u_{j-1}^n}{2 \Delta x} \quad \text{and} \quad u_{xx} \approx \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} \quad (21) \]

at a grid point \((j, n)\). Here \(u_j^n = u(x_j, t_n)\).

Depending on how \(u_t\) is approximated, we have three basic schemes: explicit, implicit, and Crank–Nicolson schemes.
4. Explicit Scheme.

If $u_t$ is approximated by a forward difference quotient

$$u_t \approx \frac{u_j^{n+1} - u_j^n}{\Delta t}$$

at $(j, n)$,

then the corresponding difference equation to (17) at grid point $(j, n)$ is

$$w_j^{n+1} = \lambda w_{j+1}^n + (1 - 2\lambda) w_j^n + \lambda w_{j-1}^n,$$  \hspace{1cm} (22)

where

$$\lambda = c^2 \frac{\Delta t}{(\Delta x)^2}.$$  

The initial condition is $w_j^0 = u_0(x_j)$, $j = 0, \ldots, N$, and

the boundary conditions are $w_0^n = g_a(t_n)$ and $w_N^n = g_b(t_n)$, $n = 0, 1, \ldots$.

The difference equations (22), $j = 1, \ldots, N - 1$, can be solved explicitly.
5. **Implicit Scheme.**

If $u_t$ is approximated by a backward difference quotient

$$u_t \approx \frac{u_j^{n+1} - u_j^n}{\Delta t}$$

at $(j, n + 1)$,

then the corresponding difference equation to (17) at grid point $(j, n + 1)$ is

$$-\lambda w_{j+1}^{n+1} + (1 + 2 \lambda) w_j^{n+1} - \lambda w_{j-1}^{n+1} = w_j^n. \quad (23)$$

The difference equations (23), $j = 1, \ldots, N - 1$, together with the initial and boundary conditions as before, can be solved using the Crout algorithm or the SOR algorithm.
Explicit Method.

\[
\frac{w_{j+1}^n - w_j^n}{\Delta t} = c^2 \frac{w_{j+1}^{n+1} - 2w_j^n + w_{j+1}^n}{(\Delta x)^2}
\]

Letting \( \lambda := c^2 \frac{\Delta t}{(\Delta x)^2} \) gives (22).

Implicit Method.

\[
\frac{w_{j+1}^n - w_j^n}{\Delta t} = c^2 \frac{w_{j+1}^{n+1} - 2w_j^{n+1} + w_{j+1}^{n+1}}{(\Delta x)^2}
\]

Letting \( \lambda := c^2 \frac{\Delta t}{(\Delta x)^2} \) gives (23).

In matrix form

\[
\begin{pmatrix}
1 + 2\lambda & -\lambda \\
-\lambda & 1 + 2\lambda & -\lambda \\
& \ddots & \ddots \\
& & -\lambda & 1 + 2\lambda
\end{pmatrix}
\begin{pmatrix}
w_1^n \\
w_2^n \\
\vdots \\
w_{N-1}^n
\end{pmatrix} = \begin{pmatrix}
b_1^n \\
b_2^n \\
\vdots \\
b_{N-1}^n
\end{pmatrix}
\]
The matrix is tridiagonal and diagonally dominant. \( \Rightarrow \) Crout / SOR.

The Crank–Nicolson scheme is the average of the explicit scheme at \((j, n)\) and the implicit scheme at \((j, n + 1)\).

The resulting difference equation is

\[
-\frac{\lambda}{2} w_{j-1}^{n+1} + (1 + \lambda) w_{j}^{n+1} - \frac{\lambda}{2} w_{j+1}^{n+1} = \frac{\lambda}{2} w_{j-1}^{n} + (1 - \lambda) w_{j}^{n} + \frac{\lambda}{2} w_{j+1}^{n}.
\]  

(24)

The difference equations (24), \(j = 1, \ldots, N - 1\), together with the initial and boundary conditions as before, can be solved using Crout algorithm or SOR algorithm.
Crank–Nicolson.

\( \frac{1}{2} \left[ (†) + (‡) \right] \) gives

\[
\frac{w_{j}^{n+1} - w_{j}^{n}}{\Delta t} = \frac{1}{2} c^2 \frac{w_{j-1}^{n} - 2 w_{j}^{n} + w_{j+1}^{n}}{(\Delta x)^2} + \frac{1}{2} c^2 \frac{w_{j-1}^{n+1} - 2 w_{j}^{n+1} + w_{j+1}^{n+1}}{(\Delta x)^2}
\]

Letting \( \mu := \frac{1}{2} \frac{c^2 \Delta t}{(\Delta x)^2} = \frac{\lambda}{2} \) gives

\[
- \mu w_{j+1}^{n+1} + (1 + 2 \mu) w_{j}^{n+1} - \mu w_{j-1}^{n+1} = \hat{w}_{j}^{n+1}
\]

and

\( \hat{w}_{j}^{n+1} = \mu w_{j+1}^{n} + (1 - 2 \mu) w_{j}^{n} + \mu w_{j-1}^{n} \).

This can be interpreted as

\( \hat{w}_{j}^{n+1} \quad \text{— predictor \quad (explicit method)} \)

\( w_{j}^{n+1} \quad \text{— corrector \quad (implicit method)} \)
7. Local Truncation Errors.

These are measures of the error by which the exact solution of a differential equation does not satisfy the difference equation at the grid points and are obtained by substituting the exact solution of the continuous problem into the numerical scheme. A necessary condition for the convergence of the numerical solutions to the continuous solution is that the local truncation error tends to zero as the step size goes to zero. In this case the method is said to be consistent.

It can be shown that all three methods are consistent.

The explicit and implicit schemes have local truncation errors $O(\Delta t, (\Delta x)^2)$, while that of the Crank–Nicolson scheme is $O((\Delta t)^2, (\Delta x)^2)$. 
Local Truncation Error.

For the explicit scheme we get for the LTE at \((j, n)\)

\[
E^n_j = \frac{u(x_j, t_{n+1}) - u(x_j, t_n)}{\Delta t} - c^2 \frac{u(x_{j-1}, t_n) - 2u(x_j, t_n) + u(x_{j+1}, t_n)}{(\Delta x)^2}.
\]

With the help of a Taylor expansion at \((x_j, t_n)\) we find that

\[
\frac{u(x_j, t_{n+1}) - u(x_j, t_n)}{\Delta t} = u_t(x_j, t_n) + O(\Delta t),
\]

\[
\frac{u(x_{j-1}, t_n) - 2u(x_j, t_n) + u(x_{j+1}, t_n)}{(\Delta x)^2} = u_{xx}(x_j, t_n) + O((\Delta x)^2).
\]

Hence

\[
E^n_j = u_t(x_j, t_n) - c^2 u_{xx}(x_j, t_n) + O(\Delta t) + O((\Delta x)^2) = 0.
\]
8. **Numerical Stability.**

Consistency is only a necessary but not a sufficient condition for convergence. Roundoff errors incurred during calculations may lead to a blow up of the solution or erode the whole computation.

A scheme is *stable* if roundoff errors are not amplified in the calculations. The *Fourier method* can be used to check if a scheme is stable.

Assume that a numerical scheme admits a solution of the form

\[ v_j^n = a^{(n)}(\omega) e^{i j \omega \Delta x}, \quad (25) \]

where \( \omega \) is the wave number and \( i = \sqrt{-1} \).
Define

\[ G(\omega) = \frac{a^{(n+1)}(\omega)}{a^{(n)}(\omega)}, \]

where \( G(\omega) \) is an amplification factor, which governs the growth of the Fourier component \( a(\omega) \).

The von Neumann stability condition is given by

\[ |G(\omega)| \leq 1 \]

for \( 0 \leq \omega \Delta x \leq \pi \).

It can be shown that the explicit scheme is stable if and only if \( \lambda \leq \frac{1}{2} \), called conditionally stable, and the implicit and Crank–Nicolson schemes are stable for any values of \( \lambda \), called unconditionally stable.
**Stability Analysis.**

For the explicit scheme we get on substituting (25) into (22) that

\[
a^{(n+1)}(\omega) e^{i j \omega \Delta x} = \lambda a^{(n)}(\omega) e^{i (j+1) \omega \Delta x} + (1 - 2 \lambda) a^{(n)}(\omega) e^{i j \omega \Delta x} + \lambda a^{(n)}(\omega) e^{i (j-1) \omega \Delta x}
\]

\[
\implies G(\omega) = \frac{a^{(n+1)}(\omega)}{a^{(n)}(\omega)} = \lambda e^{i \omega \Delta x} + (1 - 2 \lambda) + \lambda e^{-i \omega \Delta x}.
\]

The von Neumann stability condition then is

\[
|G(\omega)| \leq 1 \iff |\lambda e^{i \omega \Delta x} + (1 - 2 \lambda) + \lambda e^{-i \omega \Delta x}| \leq 1
\]

\[
\iff |(1 - 2 \lambda) + 2 \lambda \cos(\omega \Delta x)| \leq 1
\]

\[
\iff |1 - 4 \lambda \sin^2\left(\frac{\omega \Delta x}{2}\right)| \leq 1 \quad [\cos 2 \alpha = 1 - 2 \sin^2 \alpha]
\]

\[
\iff 0 \leq 4 \lambda \sin^2\left(\frac{\omega \Delta x}{2}\right) \leq 2
\]

\[
\iff 0 \leq \lambda \leq \frac{1}{2 \sin^2\left(\frac{\omega \Delta x}{2}\right)}
\]

for all \(0 \leq \omega \Delta x \leq \pi\).
This is equivalent to $0 \leq \lambda \leq \frac{1}{2}$. 
Remark.
The explicit method is stable, if and only if
\[ \Delta t \leq \frac{(\Delta x)^2}{2c^2}. \] (†)

(†) is a strong restriction on the time step size \( \Delta t \). If \( \Delta x \) is reduced to \( \frac{1}{2} \Delta x \), then \( \Delta t \) must be reduced to \( \frac{1}{4} \Delta t \).
So the total computational work increases by a factor 8.

Example.
\[ u_t = u_{xx} \quad (x, t) \in [0, 1] \times [0, 1] \]
Take \( \Delta x = 0.01 \). Then
\[ \lambda \leq \frac{1}{2} \quad \Rightarrow \quad \Delta t \leq 0.00005 \]
I.e. the number of grid points is equal to
\[ \frac{1}{\Delta x} \frac{1}{\Delta t} = 100 \times 20,000 = 2 \times 10^6. \]
Remark.

In vector notation, the explicit scheme can be written as

\[ w^{n+1} = A w^n + b^n, \]

where \( w^n = (w^n_1, \ldots, w^n_{N-1})^T \in \mathbb{R}^{N-1} \) and

\[
A = \begin{pmatrix}
1 - 2\lambda & \lambda \\
\lambda & 1 - 2\lambda & \lambda \\
& \ddots & \ddots \\
\lambda & 1 - 2\lambda & & \\
\end{pmatrix} \in \mathbb{R}^{(N-1) \times (N-1)}, \quad \quad \quad b^n = \begin{pmatrix}
\lambda w^n_0 \\
0 \\
\vdots \\
0 \\
\lambda w^n_{N-1} \\
\end{pmatrix} \in \mathbb{R}^{N-1}.
\]

For the implicit method we get

\[ B w^{n+1} = w^n + b^{n+1}, \] where

\[
B = \begin{pmatrix}
1 + 2\lambda & -\lambda \\
-\lambda & 1 + 2\lambda & -\lambda \\
& \ddots & \ddots \\
& & -\lambda & 1 + 2\lambda \\
\end{pmatrix}.
\]
Remark.
Forward diffusion equation: \( u_t - c^2 u_{xx} = 0 \quad t \geq 0 \).
Backward diffusion equation:
\[
\begin{align*}
    u_t + c^2 u_{xx} &= 0 & t \leq T \\
    u(x, T) &= u_T(x) & \forall x \\
    u(a, t) &= g_a(t), \quad u(b, t) = g_b(t) & \forall t
\end{align*}
\]
[Note: We could use the transformation \( v(x, t) := u(x, T - t) \) in order to transform this into a standard forward diffusion problem.]

We can solve the backward diffusion equation directly by starting at \( t = T \) and solving “backwards”, i.e. given \( w^{n+1} \), find \( w^n \).

Implicit: \( w^{n+1} = \tilde{A} w^n + \tilde{b}^n \)  
Explicit: \( \tilde{B} w^{n+1} = w^n + \tilde{b}^n \)

The von Neumann stability condition for the backward problem then becomes
\[
|\tilde{G}(\omega)| = \left| \frac{a^n(\omega)}{a^{n+1}(\omega)} \right| \leq 1.
\]
Stability of the Binomial Model.

The binomial model is an explicit method for a backward equation.

\[ V^n_j = \frac{1}{R} \left( p V_{j+1}^{n+1} + (1 - p) V_{j-1}^{n+1} \right) = \frac{1}{R} \left( p V_{j+1}^{n+1} + 0 V_j^{n+1} + (1 - p) V_{j-1}^{n+1} \right) \]

for \( j = -n, -n + 2, \ldots, n - 2, n \) and \( n = N - 1, \ldots, 1, 0 \).

Here the initial values \( V_{-N}^N, V_{-N+2}^N, \ldots, V_{N-2}^N, V_N^N \) are given.

Now let \( V^n_j = a^{(n)}(\omega) e^{i j \omega \Delta x} \), then

\[ a^{(n)}(\omega) e^{i j \omega \Delta x} = \frac{1}{R} \left( p a^{(n+1)}(\omega) e^{i(j+1) \omega \Delta x} + (1 - p) a^{(n+1)}(\omega) e^{i(j-1) \omega \Delta x} \right) \]

\[ \Rightarrow \tilde{G}(\omega) = \left( p e^{i \omega \Delta x} + (1 - p) e^{-i \omega \Delta x} \right) e^{-r \Delta t} \]

\[ = \left( \cos(\omega \Delta x) + \underbrace{2p - 1}_{= q} i \sin(\omega \Delta x) \right) e^{-r \Delta t} \]

\[ \Rightarrow |\tilde{G}(\omega)|^2 = (\cos^2(\omega \Delta x) + q^2 \sin^2(\omega \Delta x)) e^{-2r \Delta t} \]

\[ = (1 + (q^2 - 1) \sin^2(\omega \Delta x)) e^{-2r \Delta t} \]

\[ \leq e^{-2r \Delta t} \leq 1 \]

if \( q^2 \leq 1 \iff -1 \leq q \leq 1 \iff p \in [0, 1] \). Hence the binomial model is stable.
Stability of the CRR Model.

We know that the binomial model is stable if $p \in (0, 1)$.

For the CRR model we have that

$$u = e^{\sigma \sqrt{\Delta t}}, \quad d = e^{-\sigma \sqrt{\Delta t}}, \quad p = \frac{R - d}{u - d},$$

so $p \in (0, 1)$ is equivalent to $u > R > d$.

Clearly, for $\Delta t$ small, we can ensure that

$$e^{\sigma \sqrt{\Delta t}} > e^{r \Delta t}. $$

Hence the CRR model is stable, if $\Delta t$ is sufficiently small, i.e. if $\Delta t < \frac{\sigma^2}{r^2}$.

Alternatively, one can argue (less rigorously) as follows. Since $\Delta x = uS - S = S(e^{\sigma \sqrt{\Delta t}} - 1) \approx S \sigma \sqrt{\Delta t}$ and as the BSE can be written as

$$u_t + \frac{1}{2} \sigma^2 S^2 u_{SS} + \ldots = 0,$$

194
it follows that

\[ \lambda = c^2 \frac{\Delta t}{(\Delta x)^2} = \frac{1}{2} \sigma^2 S^2 \frac{\Delta t}{S^2 \sigma^2 \Delta t} = \frac{1}{2} \Rightarrow \text{CRR is stable.} \]
9. **Simplification of the BSE.**

Assume $V(S, t)$ is the price of a European option at time $t$.

Then $V$ satisfies the Black–Scholes equation (13) with appropriate initial and boundary conditions.

Define

$$\tau = T - t, \quad x = \ln S, \quad w(\tau, x) = e^{\alpha x + \beta \tau} V(t, S),$$

where $\alpha$ and $\beta$ are parameters.

Then the Black–Scholes equation can be transformed into a basic diffusion equation:

$$\frac{\partial w}{\partial \tau} = \frac{1}{2} \sigma^2 \frac{\partial^2 w}{\partial x^2}$$

with a new set of initial and boundary conditions.

Finite difference methods can be used to solve the corresponding difference equations and hence to derive option values at grid points.
Transformation of the BSE.

Consider a call option.

Let $\tau = T - t$ be the remaining time to maturity. Set $u(x, \tau) = V(x, t)$. Then $\frac{\partial u}{\partial \tau} = -\frac{\partial V}{\partial t}$ and the BSE (13) is equivalent to

$$u_\tau = \frac{1}{2} \sigma^2 S^2 u_{SS} + r S u_S - r u,$$

(†)

$$u(S, 0) = V(S, T) = (S - X)^+,$$

(IC)

$$u(0, \tau) = V(0, t) = 0, \quad u(S, \tau) = V(S, t) \approx S \quad \text{as } S \to \infty.$$  

(BC)

Let $x = \ln S$ ($\Longleftrightarrow S = e^x$). Set $\tilde{u}(x, \tau) = u(S, \tau)$. Then

$$\tilde{u}_x = u_S e^x = S u_S, \quad \tilde{u}_{xx} = S u_S + S^2 u_{SS}$$

and (†) becomes

$$\tilde{u}_\tau = \frac{1}{2} \sigma^2 \tilde{u}_{xx} + (r - \frac{1}{2} \sigma^2) \tilde{u}_x - r \tilde{u},$$

(‡)

$$\tilde{u}(x, 0) = u(S, 0) = (e^x - X)^+,$$

(IC)

$$\tilde{u}(x, \tau) = u(0, \tau) = 0 \quad \text{as } x \to -\infty, \quad \tilde{u}(x, \tau) = u(e^x, \tau) \approx e^x \quad \text{as } x \to \infty.$$  

(BC)
Note that the growth condition (20), \( \lim_{|x| \to \infty} \tilde{u}(x, \tau) e^{-a x^2} = 0 \) for any \( a > 0 \), is satisfied. Hence (‡) is well defined.

Let \( w(x, \tau) = e^{\alpha x + \beta \tau} \tilde{u}(x, \tau) \iff \tilde{u}(x, \tau) = e^{-\alpha x - \beta \tau} w(x, \tau) =: C w(x, \tau) \). Then

\[
\begin{align*}
\tilde{u}_\tau &= C (-\beta w + w_\tau) \\
\tilde{u}_x &= C (-\alpha w + w_x) \\
\tilde{u}_{xx} &= C (-\alpha (-\alpha w + w_x) + (-\alpha w_x + w_{xx})) = C (\alpha^2 w - 2 \alpha w_x + w_{xx}).
\end{align*}
\]

So (‡) is equivalent to

\[
C (-\beta w + w_\tau) = \frac{1}{2} \sigma^2 C (\alpha^2 w - 2 \alpha w_x + w_{xx}) + (r - \frac{1}{2} \sigma^2) C (-\alpha w + w_x) - r C w.
\]

In order to cancel the \( w \) and \( w_x \) terms we need to have

\[
\begin{aligned}
-\beta &= \frac{1}{2} \sigma^2 \alpha^2 - (r - \frac{1}{2} \sigma^2) \alpha - r, \\
0 &= \frac{1}{2} \sigma^2 (-2 \alpha) + r - \frac{1}{2} \sigma^2.
\end{aligned}
\]

\[
\begin{aligned}
\alpha &= \frac{1}{\sigma^2} (r - \frac{1}{2} \sigma^2), \\
\beta &= \frac{1}{2 \sigma^2} (r - \frac{1}{2} \sigma^2)^2 + r.
\end{aligned}
\]
With this choice of $\alpha$ and $\beta$ the equation (‡) is equivalent to

$$w_\tau = \frac{1}{2} \sigma^2 w_{xx},$$  \hspace{1cm} (‡)

$$w(x, 0) = e^{\alpha x} \tilde{u}(x, 0) = e^{\alpha x} (e^x - X)^+, \hspace{1cm} (IC)$$

$$w(x, \tau) = 0 \text{ as } x \to -\infty, \quad w(x, \tau) \approx e^{\alpha x + \beta \tau} e^x \text{ as } x \to \infty. \hspace{1cm} (BC)$$

Note that the growth condition (20) is satisfied. Hence (‡) is well defined.

**Implementation.**

1. Choose a truncated interval $[a, b]$ to approximate $(-\infty, \infty)$.

   $$e^{-8} = 0.0003, \quad e^{8} = 2981 \quad \Rightarrow \quad [a, b] = [-8, 8] \text{ serves all practical purposes.}$$

2. Choose integers $N, M$ to get the step sizes $\Delta x = \frac{b-a}{N}$ and $\Delta \tau = \frac{T-t}{M}$.

   Grid points $(x_j, \tau_n)$:

   $$x_j = a + j \Delta x, \quad j = 0, 1, \ldots, N \quad \text{and} \quad \tau_n = n \Delta \tau, \quad n = 0, 1, \ldots, M.$$  

   Note: $x_0, x_N$ and $\tau_0$ represent the boundary of the grid with known values.
3. Solve (\#) with
\[ w(x, 0) = e^{\alpha x} (e^x - X)^+, \quad \text{(IC)} \]
\[ w(a, \tau) = 0, \quad w(b, \tau) = \begin{cases} e^{(\alpha+1)b + \beta \tau} & \text{or} \\ e^{\alpha b} (e^b - X) e^{\beta \tau} & \text{(a better choice)} \end{cases}. \quad \text{(BC)} \]

Note: If the explicit method is used, \( N \) and \( M \) need to be chosen such that
\[
\frac{1}{2} \sigma^2 \frac{\Delta \tau}{(\Delta x)^2} \leq \frac{1}{2} \iff M \geq \frac{\sigma^2 (T - t)}{(b - a)^2} N^2.
\]

If the implicit or Crank–Nicolson scheme is used, there are no restrictions on \( N, M \). Use Crout or SOR to solve.

4. Assume \( w(x_j, \tau_M), j = 0, 1, \ldots, N \) are the solutions from step 3, then the call option price at time \( t \) is
\[
V(S_j, t) = e^{-\alpha x_j - \beta (T - t)} w(x_j, \underbrace{T - t}_{\tau_M}) \quad j = 0, 1, \ldots, N,
\]
where \( S_j = e^{x_j} \) and \( T - t \equiv \tau_M \).
Note: The $S_j$ are not equally spaced.
10. **Direct Discretization of the BSE.**

**Exercise:** Apply the Crank–Nicolson scheme directly to the BSE (13), i.e. there is no transformation of variables, and write out the resulting difference equations and do a stability analysis.

**C++ Exercise:** Write a program to solve the BSE (13) using the result of the previous exercise and the Crout algorithm. The inputs are the interest rate $r$, the volatility $\sigma$, the current time $t$, the expiry time $T$, the strike price $X$, the maximum price $S_{\text{max}}$, the number of intervals $N$ in $[0, S_{\text{max}}]$, and the number of subintervals $M$ in $[t, T]$. The output are the asset prices $S_j$, $j = 0, 1, \ldots, N$, at time $t$, and their corresponding European call and put prices (with the same strike price $X$).

Assume that the asset prices $S_j$ and option values $V_j$, $j = 0, 1, \ldots, N$, are known at time $t$.

The sensitivities of $V$ at $S_j$, $j = 1, \ldots, N - 1$, are computed as follows:

$$
\delta_j = \frac{\partial V}{\partial S}|_{S=S_j} \approx \frac{V_{j+1} - V_{j-1}}{S_{j+1} - S_{j-1}},
$$

which is $\frac{V_{j+1} - V_{j-1}}{2 \Delta S}$, if $S$ is equally spaced.

$$
\gamma_j = \frac{\partial^2 V}{\partial S^2}|_{S=S_j} \approx \frac{V_{j+1}-V_j}{S_{j+1}-S_j} - \frac{V_j-V_{j-1}}{S_j-S_{j-1}},
$$

which is $\frac{V_{j+1} - 2V_j + V_{j-1}}{(\Delta S)^2}$, if $S$ is equally spaced.
12. **Diffusion Equations of Two State Variables.**

\[
\frac{\partial u}{\partial t} = \alpha^2 \left( \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right), \quad (x, y, t) \in [a, b] \times [c, d] \times [0, \infty). \tag{26}
\]

The initial conditions are

\[ u(x, y, 0) = u_0(x, y) \quad \forall (x, y) \in [a, b] \times [c, d], \]

and the boundary conditions are

\[ u(a, y, t) = g_a(y, t), \quad u(b, y, t) = g_b(y, t) \quad \forall y \in [c, d], \ t \geq 0, \]

and

\[ u(x, c, t) = g_c(x, t), \quad u(x, d, t) = g_d(x, t) \quad \forall x \in [a, b], \ t \geq 0. \]

Here we assume that all the functions involved are consistent, in the sense that they have the same value at common points, e.g. \( g_a(c, t) = g_c(a, t) \) for all \( t \geq 0 \).
13. **Grid Points.**

\((x_i, y_j, t_n),\) where

\[x_i = a + i \Delta x, \quad \Delta x = \frac{b - a}{I}, \quad i = 0, \ldots, I,\]

\[y_j = c + j \Delta y, \quad \Delta y = \frac{d - c}{J}, \quad j = 0, \ldots, J,\]

\[t_n = n \Delta t, \quad \Delta t = \frac{T}{N}, \quad n = 0, \ldots, N\]

and \(I, J, N\) are integers.

Recalling the finite differences (21), we have

\[u_{xx} \approx \frac{u_{i+1,j}^n - 2u_{i,j}^n + u_{i-1,j}^n}{(\Delta x)^2}\]

\[\text{and} \quad u_{yy} \approx \frac{u_{i,j+1}^n - 2u_{i,j}^n + u_{i,j-1}^n}{(\Delta y)^2}.\]

at a grid point \((i, j, n).\)
Depending on how $u_t$ is approximated, we have three basic schemes: explicit, implicit, and Crank–Nicolson schemes.
14. **Explicit Scheme.**

If $u_t$ is approximated by a forward difference quotient

$$ u_t \approx \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} $$

at $(i,j,n)$,

then the corresponding difference equation at grid point $(i,j,n)$ is

$$ w_{i,j}^{n+1} = (1 - 2\lambda - 2\mu) w_{i,j}^n + \lambda w_{i+1,j}^n + \lambda w_{i-1,j}^n + \mu w_{i,j+1}^n + \mu w_{i,j-1}^n \quad (27) $$

for $i = 1, \ldots, I - 1$ and $j = 1, \ldots, J - 1$, where

$$ \lambda = \alpha^2 \frac{\Delta t}{(\Delta x)^2} \quad \text{and} \quad \mu = \alpha^2 \frac{\Delta t}{(\Delta y)^2}. $$

(27) can be solved explicitly. It has local truncation error $O(\Delta t, (\Delta x)^2, (\Delta y)^2)$, but is only conditionally stable.
15. **Implicit Scheme.**

If \( u_t \) is approximated by a backward difference quotient \( u_t \approx \frac{u_{i,j}^{n+1} - u_{i,j}^n}{\Delta t} \) at \((i, j, n + 1)\), then the difference equation at grid point \((i, j, n + 1)\) is

\[
(1 + 2\lambda + 2\mu) w_{i,j}^{n+1} - \lambda w_{i+1,j}^{n+1} - \lambda w_{i-1,j}^{n+1} - \mu w_{i,j+1}^{n+1} - \mu w_{i,j-1}^{n+1} = w_{i,j}^n
\]  

(28)

for \( i = 1, \ldots, I - 1 \) and \( j = 1, \ldots, J - 1 \).

For fixed \( n \), there are \((I - 1)(J - 1)\) unknowns and equations. (28) can be solved by relabeling the grid points and using the SOR algorithm.

(28) is unconditionally stable with local truncation error \( O(\Delta t, (\Delta x)^2, (\Delta y)^2) \), but is more difficult to solve, as it is no longer tridiagonal, so the Crout algorithm cannot be applied.

16. **Crank–Nicolson Scheme.**

It is the average of the explicit scheme at \((i, j, n)\) and the implicit scheme at \((i, j, n + 1)\). It is similar to the implicit scheme but with the improved local truncation error \( O((\Delta t)^2, (\Delta x)^2, (\Delta y)^2) \).
Solving the Implicit Scheme.

\[
(1 + 2\lambda + 2\mu) w^{n+1}_{i,j} - \lambda w^{n+1}_{i+1,j} - \lambda w^{n+1}_{i-1,j} - \mu w^{n+1}_{i,j+1} - \mu w^{n+1}_{i,j-1} = w^n_{i,j}
\]

With SOR for \( \omega \in (0, 2) \).

For each \( n = 0, 1, \ldots, N \)

1. Set \( w^{n+1,0} := w^n \) and

   fill in the boundary conditions \( w^{n+1,0}_{0,j} , w^{n+1,0}_{i,j} , w^{n+1,0}_{i,0} , w^{n+1,0}_{i,J} \) for all \( i, j \).

2. For \( k = 0, 1, \ldots \)

   For \( i = 1, \ldots I - 1, j = 1, \ldots, J - 1 \)

   \[
   \hat{w}_{i,j}^{k+1} = \frac{1}{1+2\lambda+2\mu} \left( w^{n+1}_{i,j} + \lambda w^{n+1}_{i+1,j} + \lambda w^{n+1}_{i-1,j} + \mu w^{n+1}_{i,j+1} + \mu w^{n+1}_{i,j-1} \right)
   \]

   \[
   w^{n+1,k+1}_{i,j} = (1 - \omega) w^{n+1,k}_{i,j} + \omega \hat{w}_{i,j}^{k+1}
   \]

   until \( \| w^{n+1,k+1}_{i,j} - w^{n+1,k}_{i,j} \| < \varepsilon \).

3. Set \( w^{n+1} = w^{n+1,k+1} \).
With Block Jacobi/Gauss–Seidel.

Denote \( \tilde{w}_j = \left( \begin{array}{c} w_{1,j} \\ w_{2,j} \\ \vdots \\ w_{I-1,j} \end{array} \right) \in \mathbb{R}^{I-1}, \ j = 1, \ldots, J - 1, \ w = \left( \begin{array}{c} \tilde{w}_1 \\ \tilde{w}_2 \\ \vdots \\ \tilde{w}_{J-1} \end{array} \right) \in \mathbb{R}^{(I-1)(J-1)}. \)
On setting $c = 1 + 2\lambda + 2\mu$, we have from (28) for $j$ fixed

\[
\begin{pmatrix}
  c & -\lambda \\
  -\lambda & c & -\lambda \\
  & \ddots & \ddots & \ddots \\
  -\lambda & c
\end{pmatrix}
\begin{pmatrix}
  w_{1,j} \\
  w_{2,j} \\
  \vdots \\
  w_{I-1,j}
\end{pmatrix}
+ \begin{pmatrix}
  -\mu \\
  & \ddots & \ddots & \ddots \\
  & & -\mu
\end{pmatrix}
\begin{pmatrix}
  w_{1,j+1} \\
  w_{2,j+1} \\
  \vdots \\
  w_{I-1,j+1}
\end{pmatrix}
+ \begin{pmatrix}
  -\mu \\
  & \ddots & \ddots & \ddots \\
  & & -\mu
\end{pmatrix}
\begin{pmatrix}
  w_{1,j-1} \\
  w_{2,j-1} \\
  \vdots \\
  w_{I-1,j-1}
\end{pmatrix}

= \begin{pmatrix}
  w_{1,j} + \lambda w_{0,j}^{n+1} \\
  w_{2,j}^{n} \\
  \vdots \\
  w_{I-2,j}^{n} \\
  w_{I-1,j} + \lambda w_{I,j}^{n+1}
\end{pmatrix}
\iff
\begin{pmatrix}
  A \tilde{w}_j^{n+1} + B \tilde{w}_{j+1}^{n+1} + B \tilde{w}_{j-1}^{n+1} = d_j^n
\end{pmatrix}
Rewriting

\[ A \tilde{w}_{j+1}^{n+1} + B \tilde{w}_{j+1}^{n+1} + B \tilde{w}_{j-1}^{n+1} = d_j^n \quad j = 1, \ldots, J - 1 \]

as

\[
\begin{pmatrix}
A & B \\
B & A & B \\
\vdots & \vdots & \ddots \\
B & A & B \\
B & A \\
\end{pmatrix}
\begin{pmatrix}
\tilde{w}_1^{n+1} \\
\tilde{w}_2^{n+1} \\
\vdots \\
\tilde{w}_{J-1}^{n+1} \\
\end{pmatrix}
= \begin{pmatrix}
d_1^n - B \tilde{w}_0^{n+1} \\
d_2^n \\
\vdots \\
d_{J-2}^n \\
d_{J-1}^n - B \tilde{w}_J^{n+1} \\
\end{pmatrix} =: \begin{pmatrix}
\tilde{d}_1^n \\
\tilde{d}_2^n \\
\vdots \\
\tilde{d}_{J-2}^n \\
\tilde{d}_{J-1}^n \\
\end{pmatrix},
\]

where \( \tilde{w}_0^{n+1} \) and \( \tilde{w}_J^{n+1} \) represent boundary points, leads to the following Block Jacobi
iteration: For $k = 0, 1, \ldots$

\[
A \tilde{w}^{n+1,k+1}_1 = -B \tilde{w}^{n+1,k}_2 + \tilde{d}^n_1
\]

\[
A \tilde{w}^{n+1,k+1}_2 = -B \tilde{w}^{n+1,k}_1 - B \tilde{w}^{n+1,k}_3 + d^n_2
\]

\vdots

\[
A \tilde{w}^{n+1,k+1}_{J-2} = -B \tilde{w}^{n+1,k}_J - B \tilde{w}^{n+1,k}_{J-1} + d^n_{J-2}
\]

\[
A \tilde{w}^{n+1,k+1}_{J-1} = -B \tilde{w}^{n+1,k}_{J-2} + \tilde{d}^n_{J-1}
\]

Similarly, the Block Gauss–Seidel iteration is given by:

For $k = 0, 1, \ldots$

\[
A \tilde{w}^{n+1,k+1}_1 = -B \tilde{w}^{n+1,k}_2 + \tilde{d}^n_1
\]

\[
A \tilde{w}^{n+1,k+1}_2 = -B \tilde{w}^{n+1,k}_1 - B \tilde{w}^{n+1,k}_3 + d^n_2
\]

\vdots

\[
A \tilde{w}^{n+1,k+1}_{J-2} = -B \tilde{w}^{n+1,k}_J - B \tilde{w}^{n+1,k}_{J-1} + d^n_{J-2}
\]

\[
A \tilde{w}^{n+1,k+1}_{J-1} = -B \tilde{w}^{n+1,k}_{J-2} + \tilde{d}^n_{J-1}
\]
In each case, use the Crout algorithm to solve for $\tilde{w}_{j}^{n+1,k+1}, j = 1, \ldots, J - 1$.

---

**Note on Stability.**

Recall that in 1d a scheme was stable if $|G(\omega)| = \left| \frac{a^{(n+1)}(\omega)}{a^{(n)}(\omega)} \right| \leq 1$, where $v_j^n = a^{(n)}(\omega) e^{i j \omega \Delta x}$.

In 2d, this is adapted to

\[ v_{i,j}^n = a^{(n)}(\omega) e^{\sqrt{-1} i \omega \Delta x + \sqrt{-1} j \omega \Delta y}. \]
17. **Alternating Direction Implicit (ADI) Method.**

An alternative finite difference method is the ADI scheme, which is unconditionally stable while the difference equations are still tridiagonal and diagonally dominant. The ADI algorithm can be used to efficiently solve the Black–Scholes two asset pricing equation:

\[
V_t + \frac{1}{2} \sigma_1^2 S_1 V S_1 + \frac{1}{2} \sigma_2^2 S_2 V S_2 + \rho \sigma_1 \sigma_2 S_1 S_2 V + r S_1 V + r S_2 V - r V = 0.
\]

(29)

See Clewlow and Strickland (1998) for details on how to transform the Black–Scholes equation (29) into the basic diffusion equation (26) and then to solve it with the ADI scheme.
ADI scheme

Implicit method at \((i, j, n + 1)\):

\[
\frac{w_{i,j}^{n+1} - w_{i,j}^n}{\Delta t} = \alpha^2 \frac{w_{i+1,j}^n - 2w_{i,j}^n + w_{i-1,j}^n}{(\Delta x)^2} + \alpha^2 \frac{w_{i,j+1}^{n+1} - 2w_{i,j}^{n+1} + w_{i,j-1}^{n+1}}{(\Delta y)^2}
\]

approx. \(u_{xx}\) using \((i, j, n)\) data

Implicit method at \((i, j, n + 2)\):

\[
\frac{w_{i,j}^{n+2} - w_{i,j}^{n+1}}{\Delta t} = \alpha^2 \frac{w_{i+1,j}^{n+2} - 2w_{i,j}^{n+2} + w_{i-1,j}^{n+2}}{(\Delta x)^2} + \alpha^2 \frac{w_{i,j+1}^{n+1} - 2w_{i,j}^{n+1} + w_{i,j-1}^{n+1}}{(\Delta y)^2}
\]

approx. \(u_{yy}\) using \((i, j, n + 1)\) data

We can write the two equations as follows:

\[
-\mu w_{i,j+1}^{n+1} + (1 + 2\mu) w_{i,j}^{n+1} - \mu w_{i,j-1}^{n+1} = \lambda w_{i+1,j}^n + (1 - 2\lambda) w_{i,j}^n + \lambda w_{i-1,j}^n \quad (\dagger)
\]

\[
-\lambda w_{i+1,j}^{n+2} + (1 + 2\lambda) w_{i,j}^{n+2} - \lambda w_{i-1,j}^{n+2} = \mu w_{i,j+1}^{n+1} + (1 - 2\mu) w_{i,j}^{n+1} + \mu w_{i,j-1}^{n+1} \quad (\ddagger)
\]
To solve (†), fix \( i = 1, \ldots, I - 1 \) and solve a tridiagonal system to get \( w_{i,j}^{n+1} \) for \( j = 1, \ldots, J - 1 \).

This can be done with e.g. the Crout algorithm.

To solve (‡), fix \( j = 1, \ldots, J - 1 \) and solve a tridiagonal system to get \( w_{i,j}^{n+2} \) for \( i = 1, \ldots, I - 1 \).

Currently the method works on the interval \([t_n, t_{n+2}]\) and has features of an explicit method. In order to obtain an (unconditionally stable) implicit method, we need to adapt the method so that it works on the interval \([t_n, t_{n+1}]\) and hence gives values \( w_{i,j}^n \) for all \( n = 1, \ldots, N \).

Introduce the intermediate time point \( n + \frac{1}{2} \). Then (†) generates \( w_{i,j}^{n+\frac{1}{2}} \) (not used) and (‡) generates \( w_{i,j}^{n+1} \).

\[
-\frac{\mu}{2} w_{i,j+1}^{n+\frac{1}{2}} + (1 + \mu) w_{i,j}^{n+\frac{1}{2}} - \frac{\mu}{2} w_{i,j-1}^{n+\frac{1}{2}} = \frac{\lambda}{2} w_{i+1,j}^n + (1 - \lambda) w_{i,j}^n + \frac{\lambda}{2} w_{i-1,j}^n \quad (†)
\]

\[
-\frac{\lambda}{2} w_{i+1,j}^{n+1} + (1 + \lambda) w_{i,j}^{n+1} - \frac{\lambda}{2} w_{i,j+1}^{n+1} = \frac{\mu}{2} w_{i,j+1}^{n+\frac{1}{2}} + (1 - \mu) w_{i,j}^{n+\frac{1}{2}} + \frac{\mu}{2} w_{i,j-1}^{n+\frac{1}{2}} \quad (‡)
\]
9. Simulation

1. Uniform Random Number Generators.

In order to use simulation techniques, we first need to generate independent samples from some given distribution functions.

The simplest and the most important distribution in simulation is the uniform distribution $U[0, 1]$.

Note that if $X$ is a $U[0, 1]$ random variable, then $Y = a + (b - a)X$ is a $U[a, b]$ random variable.

We focus on how to generate $U[0, 1]$ random variables.

**Examples.**

- 2 outcomes, $p_1 = p_2 = \frac{1}{2}$ (coin)
- 6 outcomes, $p_1 = \ldots = p_6 = \frac{1}{6}$ (dice)
- 3 outcomes, $p_1 = p_2 = 0.49$, $p_3 = 0.05$ (roulette wheel)
2. Linear Congruential Generator.

A common technique is to generate a sequence of integers \( n_i \), defined recursively by

\[
n_i = (a n_{i-1}) \mod m
\]

for \( i = 1, 2, \ldots, N \), where \( n_0 (\neq 0) \) is called the seed, \( a > 0 \) and \( m > 0 \) are integers such that \( a \) and \( m \) have no common factors.

(30) generates a sequence of numbers in the set \( \{1, 2, \ldots, m - 1\} \).

Note that \( n_i \) are periodic with period \( \leq m - 1 \), this is because there are not \( m \) different \( n_i \) and two in \( \{n_0, \ldots, n_{m-1}\} \) must be equal: \( n_i = n_{i+p} \) with \( p \leq m - 1 \).

If the period is \( m - 1 \) then (30) is said to have full period.

The condition of full period is that \( m \) is a prime, \( a^{m-1} - 1 \) is divisible by \( m \), and \( a^j - 1 \) is not divisible by \( m \) for \( j = 1, \ldots, m - 2 \).

**Example.** \( n_0 = 35, a = 13, m = 100 \).

Then the sequence is \( \{35, 55, 15, 95, 35, \ldots\} \). So \( p = 4 \).
3. **Pseudo–Uniform Random Numbers.**

If we define

$$x_i = \frac{n_i}{m}$$

then $x_i$ is a sequence of numbers in the interval $(0, 1)$.

If (30) has full period then these $x_i$’s are called pseudo-$U[0, 1]$ random numbers.

In view of the periodic property, the number $m$ should be as large as possible, because a small set of numbers makes the outcome easier to predict – a contrast to randomness.

The main drawback of linear congruential generators is that consecutive points obtained lie on parallel hyperplanes, which implies that the unit cube cannot be uniformly filled with these points.

For example, if $a = 6$ and $m = 11$, then the ten distinct points generated lie on just two parallel lines in the unit square.
4. **Choice of Parameters.**

A good choice of \( a \) and \( m \) is given by \( a = 16807 \) and \( m = 2147483647 = 2^{31} - 1 \).

The seed \( n_0 \) can be any positive integer and can be chosen manually.

This allows us to repeatedly generate the same set of numbers, which may be useful when we want to compare different simulation techniques.

In general, we let the computer choose \( n_0 \) for us, a common choice is the computer’s internal clock.

For details on the implementation of LCG, see Press et al. (1992).

For state of the art random number generators (linear and nonlinear), see the pLab website at [http://random.mat.sbg.ac.at/](http://random.mat.sbg.ac.at/). It includes extensive information on random number generation, as well as links to free software in a variety of computing languages.
5. Normal Random Number Generators

Once we have generated a set of pseudo-$U[0, 1]$ random numbers, we can generate pseudo-$N(0, 1)$ random numbers.

Again there are several methods to generate a sequence of independent $N(0, 1)$ random numbers.

Note that if $X$ is a $N(0, 1)$ random variable, then $Y = \mu + \sigma X$ is a $N(\mu, \sigma^2)$ random variable.

This method is based on the Central Limit Theorem.

We know that if $X_i$ are independent identically distributed random variables, with finite mean $\mu$ and finite variance $\sigma^2$, then

$$Z_n = \frac{\sum_{i=1}^{n} X_i - n \mu}{\sigma \sqrt{n}}$$

(31)

converges in distribution to a $N(0, 1)$ random variable, i.e.

$$P(Z_n \leq x) \to \Phi(x), \quad n \to \infty.$$  

If $X$ is $U[0, 1]$ distributed then its mean is $\frac{1}{2}$ and its variance is $\frac{1}{12}$.

If we choose $n = 12$ then (31) is simplified to

$$Z_n = \sum_{i=1}^{12} X_i - 6.$$  

(32)
Note that $Z_n$ generated in this way is only an approximate normal random number. This is due to the fact that the Central Limit Theorem only gives convergence in distribution, not almost surely, and $n$ should tend to infinity. In practice there are hardly any differences between pseudo-$N(0, 1)$ random numbers generated by (32) and those generated by other techniques. The disadvantage is that we need to generate 12 uniform random numbers to generate 1 normal random number, and this does not seem efficient.
7. **Box–Muller Method.**

This is a direct method as follows: Generate two independent \( U[0, 1] \) random numbers \( X_1 \) and \( X_2 \), define

\[
Z = h(X)
\]

where \( X = (X_1, X_2) \) and \( Z = (Z_1, Z_2) \) are \( \mathbb{R}^2 \) vectors and \( h : [0, 1]^2 \to \mathbb{R}^2 \) is a vector function defined by

\[
(z_1, z_2) = h(x_1, x_2) = \left( \sqrt{-2 \ln x_1 \cos(2 \pi x_2)}, \sqrt{-2 \ln x_1 \sin(2 \pi x_2)} \right).
\]

Then \( Z_1 \) and \( Z_2 \) are two independent \( N(0, 1) \) random numbers.
This result can be easily proved with the transformation method. Specifically, we can find the inverse function $h^{-1}$ by

$$(x_1, x_2) = h^{-1}(z_1, z_2) = \left( e^{-\frac{1}{2} (z_1^2 + z_2^2)}, \frac{1}{2\pi} \arctan \frac{z_2}{z_1} \right).$$

The absolute value of the determinant of the Jacobian matrix is

$$\left| \frac{\partial (x_1, x_2)}{\partial (z_1, z_2)} \right| = \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} z_1^2} \cdot \frac{1}{\sqrt{2\pi}} e^{-\frac{1}{2} z_2^2}. \quad (33)$$

From the transformation theorem of random variables and (33) we know that if $X_1$ and $X_2$ are independent $U[0, 1]$ random variables, then $Z_1$ and $Z_2$ are independent $N(0, 1)$ random variables.
8. **Correlated Normal Distributions.**

Assume $X \sim N(\mu, \Sigma)$, where $\Sigma \in \mathbb{R}^{n \times n}$ is symmetric positive definite.

To generate a normal vector $X$ do the following:

(a) Calculate the Cholesky decomposition $\Sigma = LL^T$.

(b) Generate $n$ independent $N(0, 1)$ random numbers $Z_i$ and let $Z = (Z_1, \ldots, Z_n)$.

(c) Set $X = \mu + L Z$.

**Example.**

To generate 2 correlated $N(\mu, \sigma^2)$ RVs with correlation coefficient $\rho$, let

$$(Y_1, Y_2) = (Z_1, \rho Z_1 + \sqrt{1 - \rho^2} Z_2)$$

and then set

$$(X_1, X_2) = (\mu + \sigma Y_1, \mu + \sigma Y_2).$$
9. **Inverse Transformation Method.**

To generate a random variable \( X \) with known cdf \( F(x) \), the inverse transform method sets

\[
X = F^{-1}(U),
\]

where \( U \) is a \( U[0, 1] \) random variable (i.e. \( X \) satisfies \( F(X) = U \sim U[0, 1] \)).

The inverse of \( F \) is well defined if \( F \) is strictly increasing and continuous, otherwise we set

\[
F^{-1}(u) = \inf\{x : F(x) \geq u\}.
\]

**Exercise.**

Use the inversion method to generate \( X \) from a *Rayleigh distribution*, i.e.

\[
F(x) = 1 - \exp\left(\frac{-x^2}{2 \sigma^2}\right).
\]
Inverse Transformation Method.

\[ X = F^{-1}(U) \]

\[ \Rightarrow \quad P(X \leq x) = P(F^{-1}(U) \leq x) \]

\[ = P(U \leq F(x)) \]

\[ = \int_0^{F(x)} 1 \, du \]

\[ = F(x) \]

\[ \Rightarrow \quad X \sim F \]

Examples.

- Exponential with parameter \( a > 0 \): \( F(x) = 1 - e^{-ax}, \ x \geq 0. \)

Let \( U \sim U[0, 1] \) and

\[ F(X) = U \iff 1 - e^{-aX} = U \]

\[ \iff X = -\frac{1}{a} \ln(1 - U) = -\frac{1}{a} \ln U. \]
Hence $X \sim \text{Exp}(a)$.

- Arcsin law: Let $B_t$ be a Brownian motion on the time interval $[0, 1]$. Let $T = \arg \max_{0 \leq t \leq 1} B_t$. Then, for any $t \in [0, 1]$ we have that
  
  $$P(T \leq t) = \frac{2}{\pi} \arcsin \sqrt{t} =: F(t).$$

Similarly, let $L = \sup\{t \leq 1 : B_t = 0\}$. Then, for any $s \in [0, 1]$ we have that
  
  $$P(L \leq s) = \frac{2}{\pi} \arcsin \sqrt{s} = F(s).$$

How to generate from this distribution?

Let $U \sim U[0, 1]$ and

$$F(X) = U \iff \frac{2}{\pi} \arcsin \sqrt{X} = U$$

$$\iff X = \left(\sin \frac{\pi U}{2}\right)^2 = \frac{1}{2} - \frac{1}{2} \cos(\pi U).$$

Hence $X \sim F$. 

230
10. **Acceptance-Rejection Method.**

Assume $X$ has pdf $f(x)$ on a set $S$, $g$ is another pdf on $S$ from which we know how to generate samples, and $f$ is dominated by $g$ on $S$, i.e. there exists a constant $c$ such that

$$f(x) \leq c g(x) \quad \forall x \in S.$$ 

The acceptance-rejection method generates a sample $X$ from $g$ and a sample $U$ from $U[0, 1]$, and accepts $X$ as a sample from $f$ if $U \leq \frac{f(X)}{c g(X)}$ and rejects $X$ otherwise, and repeats the process.

**Exercise.**

Suppose the pdf $f$ is defined over a finite interval $[a, b]$ and is bounded by $M$. Use the acceptance-rejection method to generate $X$ from $f$. 

---

231
Acceptance-Rejection Method.

Let $Y$ be a sample returned by the algorithm. Then $Y$ has the same distribution of $X$ conditional on $U \leq \frac{f(X)}{c g(X)}$.

So for any measurable subset $A \subset S$, we have

\[ P(Y \in A) = P(X \in A \mid U \leq \frac{f(X)}{c g(X)}) = \frac{P(X \in A \& U \leq \frac{f(X)}{c g(X)})}{P(U \leq \frac{f(X)}{c g(X)})}. \]

Now, as $U \sim U[0, 1],$

\[ P(U \leq \frac{f(X)}{c g(X)}) = \int_S P(U \leq \frac{f(x)}{c g(x)}) g(x) \, dx = \int_S \frac{f(x)}{c g(x)} g(x) \, dx = \frac{1}{c} \int_S f(x) \, dx = \frac{1}{c}, \]

which yields

\[ P(Y \in A) = c P(X \in A \& U \leq \frac{f(X)}{c g(X)}) = c \int_A \frac{f(x)}{c g(x)} g(x) \, dx = \int_A f(x) \, dx. \]

As $A$ was chosen arbitrarily, we have that

\[ Y \sim f. \]
11. **Monte Carlo Method for Option Pricing.**

A wide class of derivative pricing problems come down to the evaluation of the following expectation

\[ E[f(Z(T; t, z))] , \]

where \( Z \) denotes the stochastic process that describes the price evolution of one or more underlying financial variables such as asset prices and interest rates, under the respective risk neutral probability distributions.

The process \( Z \) has the initial value \( z \) at time \( t \).

The function \( f \) specifies the value of the derivative at the expiration time \( T \).

Monte Carlo simulation is a powerful and versatile technique for estimating the expected value of a random variable.

**Examples.**

- \( E[e^{-r(T-t)} (S_T - X)^+] \), where \( S_T = S_t e^{(r-\frac{1}{2}\sigma^2)(T-t)+\sigma\sqrt{T-t}Z} , Z \sim N(0, 1) \).
- \( E[e^{-r(T-t)} (\max_{t \leq \tau \leq T} S_\tau - X)^+] \), where \( S_\tau \) as before.

(a) Generate sample paths of the underlying state variables (asset prices, interest rates, etc.) according to risk neutral probability distributions.
(b) For each simulated sample path, evaluate discounted cash flows of the derivative.
(c) Take the sample average of the discounted cash flows over all sample paths.

C++ Exercise: Write a program to simulate paths of stock prices $S$ satisfying the discretized SDE:

$$\Delta S = r S \Delta t + \sigma S \sqrt{\Delta t} Z$$

where $Z \sim N(0, 1)$. The inputs are the initial asset price $S$, the time horizon $T$, the number of partitions $n$ (time-step $\Delta t = \frac{T}{n}$), the interest rate $r$, the volatility $\sigma$, and the number of simulations $M$. The output is a graph of paths of the stock price (or the data needed to generate the graph).
13. **Main Advantage and Drawback.**

With the Monte Carlo approach it is easy to price complicated terminal payoff function such as path-dependent options.

Practitioners often use the brute force Monte Carlo simulation to study newly invented options.

However, the method requires a large number of simulation trials to achieve a high level of accuracy, which makes it less competitive compared to the binomial and finite difference methods, when analytic properties of an option pricing model are better known and formulated.

[Note: The CLT tells us that the Monte Carlo estimate is correct to order $O\left(\frac{1}{\sqrt{M}}\right)$. So to increase the accuracy by a factor of 10, we need to compute 100 times more paths.]
14. **Computation Efficiency.**

Suppose $W_{\text{total}}$ is the total amount of computational work units available to generate an estimate of the value of the option $V$. Assume there are two methods for generating MC estimates, requiring $W_1$ and $W_2$ units of computational work for each run. Assume $W_{\text{total}}$ is divisible by both $W_1$ and $W_2$. Denote by $V_1^i$ and $V_2^i$ the samples for the estimator of $V$ using method 1 and 2, and $\sigma_1$ and $\sigma_2$ their standard deviation. The sample means for estimating $V$ using $W_{\text{total}}$ amount of work are, respectively,

$$
\frac{1}{N_1} \sum_{i=1}^{N_1} V_1^i \quad \text{and} \quad \frac{1}{N_2} \sum_{i=1}^{N_2} V_2^i
$$

where $N_i = \frac{W_{\text{total}}}{W_i}$, $i = 1, 2$. The law of large numbers tells us that the above two estimators are approximately normal with mean $V$ and variances

$$
\frac{\sigma_1^2}{N_1} \quad \text{and} \quad \frac{\sigma_2^2}{N_2}.
$$

Hence, method 1 is preferred over method 2 provided that $\sigma_1^2 W_1 \leq \sigma_2^2 W_2$. 

236
15. **Antithetic Variate Method.**

Suppose \( \{ \varepsilon_i \} \) are independent samples from \( N(0, 1) \) for the simulation of asset prices, so that

\[
S_T^i = S_t e^{(r - \sigma^2/2)(T-t)+\sigma \sqrt{T-t} \varepsilon_i}
\]

for \( i = 1, \ldots, M \), where \( M \) is the total number of simulation runs. An unbiased estimator of a European call option price is given by

\[
\hat{c} = \frac{1}{M} \sum_{i=1}^{M} c^i = \frac{1}{M} \sum_{i=1}^{M} e^{-r(T-t)} \max(S_T^i - X, 0).
\]

We observe that \( \{-\varepsilon_i\} \) are also independent samples from \( N(0, 1) \), and therefore the simulated price

\[
\tilde{S}_T^i = S_t e^{(r - \sigma^2/2)(T-t)-\sigma \sqrt{T-t} \varepsilon_i}
\]

for \( i = 1, \ldots, M \), is a valid sample of the terminal asset price.
A new unbiased estimator is given by

\[ \tilde{c} = \frac{1}{M} \sum_{i=1}^{M} \tilde{c}^i = \frac{1}{M} \sum_{i=1}^{M} e^{-r(T-t)} \max(\tilde{S}_T^i - X, 0). \]

Normally, we expect \( c^i \) and \( \tilde{c}^i \) to be negatively correlated, i.e. if one estimate overshoots the true value, the other estimate undershoots the true value.

The antithetic variate estimate is defined to be

\[ \bar{c} = \frac{\hat{c} + \tilde{c}}{2}. \]

The antithetic variate method improves the computation efficiency provided that \( \text{cov}(c^i, \tilde{c}^i) \leq 0. \)

[Reason: \( \text{Var}(X + Y) = \text{Var}(X) + \text{Var}(Y) + 2 \text{cov}(X, Y). \)

\[ \Rightarrow \quad \text{Var}(\bar{c}) = \frac{1}{2} \text{Var}(\hat{c}) + \frac{1}{2} \text{cov}(\hat{c}, \tilde{c}) \]
16. **Control Variate Method.**

Suppose A and B are two similar options and the analytic price formula for option B is available.

Let $V_A$ and $\hat{V}_A$ be the true and estimated values of option A, $V_B$ and $\hat{V}_B$ are similar notations for option B.

The control variate estimate of option A is defined to be

$$\tilde{V}_A = \hat{V}_A + (V_B - \hat{V}_B),$$

where the error $V_B - \hat{V}_B$ is used as an adjustment in the estimation of $V_A$.

The control variate method reduces the variance of the estimator of $V_A$ when the options A and B are strongly (positively) correlated.
The general control variate estimate of option A is defined to be

$$\tilde{V}_A^\beta = \hat{V}_A + \beta (V_B - \hat{V}_B),$$

where $\beta \in \mathbb{R}$ is a parameter.

The minimum variance of $\tilde{V}_A^\beta$ is achieved when

$$\beta^* = \frac{\text{cov}(\hat{V}_A, \hat{V}_B)}{\text{Var}(\hat{V}_B)}.$$

Unfortunately, $\text{cov}(\hat{V}_A, \hat{V}_B)$ is in general not available.

One may estimate $\beta^*$ using the regression technique from the simulated option values $V_A^i$ and $V_B^i$, $i = 1, \ldots, M$.

Note: $E(X + \beta (E(Y) - Y)) = E(X) + \beta (E(Y) - E(Y)) = E(X)$ and

$$\text{Var}(X + \beta (E(Y) - Y)) = \text{Var}(X - \beta Y) = \text{Var}(X) + \beta^2 \text{Var}(Y) - 2 \beta \text{cov}(X, Y).$$
17. **Other Variance Reduction Procedures.**

Other methods include

- *importance sampling* that modifies distribution functions to make sampling more efficient,

- *stratified sampling* that divides the distribution regions and takes samples from these regions according to their probabilities,

- *moment matching* that adjusts the samples such that their moments are the same as those of distribution functions,

- *low-discrepancy sequence* that leads to estimating error proportional to $\frac{1}{M}$ rather than $\frac{1}{\sqrt{M}}$, where $M$ is the sample size.

For details of these methods see Hull (2005) or Glasserman (2004).
18. **Pricing European Options.**

If the payoff is a function of the underlying asset at one specific time, then we can simplify the above Monte Carlo procedure.

For example, a European call option has a payoff \( \max(S(T) - X, 0) \) at expiry.

If we assume that \( S \) follows a lognormal process, then

\[
S(T) = S_0 e^{(r - \frac{1}{2} \sigma^2) T + \sigma \sqrt{T} Z}
\]  

(34)

where \( Z \) is a standard \( N(0, 1) \) random variable.

To value the call option, we only need to simulate \( Z \) to get samples of \( S(T) \).

There is no need to simulate the whole path of \( S \).

The computational work load will be considerably reduced.

Of course, if we want to price path-dependent European options, we will have to simulate the whole path of the asset process \( S \).
**C++ Exercise**: Write a program to price European call and put options using the Monte Carlo simulation with the antithetic variate method. The standard normal random variables can be generated by first generating $U[0, 1]$ random variables and then using the Box–Muller method. The terminal asset prices can be generated by (34). The inputs are the current asset price $S_0$, the exercise price $X$, the volatility $\sigma$, the interest rate $r$, the exercise time $T$, and the number of simulations $M$. The outputs are European call and put prices at time $t = 0$ (same strike price).
10. American Option Pricing

1. Formulation.

A general class of American option pricing problems can be formulated by specifying a Markov process \( \{X(t), 0 \leq t \leq T\} \) representing relevant financial variables such as underlying asset prices, an option payoff \( h(X(t)) \) at time \( t \), an instantaneous short rate process \( \{r(t), 0 \leq t \leq T\} \), and a class of admissible stopping times \( \mathcal{T} \) with values in \([0, T]\).

The American option pricing problem is to find the optimal expected discounted payoff

\[
\sup_{\tau \in \mathcal{T}} E[e^{-\int_0^\tau r(u) \, du} h(X(\tau))].
\]

It is implicit that the expectation is taken with respect to the risk-neutral measure. In this course we assume that the short rate \( r(t) = r \), a non-negative constant for \( 0 \leq t \leq T \).
For example, if the option can only be exercised at times $0 < t_1 < t_2 < \cdots < t_m = T$ (this type of option is often called the Bermudan option), then the value of an American put can be written as

$$\sup_{i=1,\ldots,m} E[e^{-r t_i} (K - S_i)^+]$$

where $K$ is the exercise price, $S_i$ the underlying asset price $S(t_i)$, and $r$ the risk-free interest rate.
American Option
In general, the value of an American option is higher than that of the corresponding European option.
However, an American call for a non-dividend paying asset has the same value as the European call.
This follows from the fact that in this case the optimal exercise time is always the expiration time $T$, as can be seen from the following argument.
Suppose the holder wants to exercise the call at time $t < T$, when $S(t) > X$. Exercising would give a profit of $S(t) - X$. Instead, one could keep the option and sell the asset short at time $t$, then purchase the asset at time $T$ by either
(a) exercising the option at time $t = T$ or
(b) buying at the market price at time $T$.
Hence the holder gained an amount $S(t) > X$ at time $t$ and paid out $\min(S(T), X) \leq X$ at time $T$. This is better than just $S(t) - X$ at time $t$.  

246
2. **Dynamic Programming Formulation.**

Let $h_i$ denote the payoff function for exercise at time $t_i$, $V_i(x)$ the value of the option at $t_i$ given $X_i = x$, assuming the option has not previously been exercised.

We are interested in $V_0(X_0)$.

This value is determined recursively as follows:

\[
V_m(x) = h_m(x),
\]

\[
V_{i-1}(x) = \max\{h_{i-1}(x), C_{i-1}(x)\}, \quad i = m, \ldots, 1,
\]

where $h_i(x)$ is the immediate exercise value at time $t_i$,

\[
C_{i-1}(x) = \mathbb{E}[D_i V_i(X_i) \mid X_{i-1} = x]
\]

is the expected discounted continuation value at time $t_{i-1}$, and

\[
D_i = e^{-r(t_i-t_{i-1})}
\]

is the discount factor over $[t_{i-1}, t_i]$. 

3. **Stopping Rule.**

The optimal stopping time $\tau$ is determined by

$$
\tau = \min\{t_i : h_i(X_i) \geq C_i(X_i), \ i = 1, \ldots, m\}.
$$

If $\hat{C}_i(x)$ is the approximation to $C_i(x)$, then the sub-optimal stopping time $\hat{\tau}$ is determined by

$$
\hat{\tau} = \min\{t_i : h_i(X_i) \geq \hat{C}_i(X_i), \ i = 1, \ldots, m\}.
$$

At the terminal exercise time $t_m = T$, the American option value at node $j$ is given by $V_j^m = h_j^m$, $j = 0, 1, \ldots, m$, where $h_j^i = h(S_j^i)$ is the intrinsic value at $(i, j)$, e.g. $h(s) = (K - s)^+$. At time $t_i$, $i = m - 1, \ldots, 0$, the option value at node $j$ is given by

$$V_j^i = \max \left( h_j^i, \frac{1}{R} [p V_{j+1}^{i+1} + (1 - p) V_{j+1}^{i+1}] \right).$$

(35)

Dynamic programming is used to find the option value at time $t_0$. In (35) we have $p = \frac{R - d}{u - d}$ and the values of $p, u, d$ are determined as usual.
5. **Finite Difference Method.**

The dynamic programming approach cannot be applied with the implicit or Crank–Nicolson scheme.

Suppose the difference equation with an implicit scheme has the form

\[ a_{j-1} V_{j-1} + a_j V_j + a_{j+1} V_{j+1} = d_j \]  

(36)

for \( j = 1, \ldots, N - 1 \), where the superscript “\( n + 1 \)” is omitted for brevity and \( d_j \) represents the known quantities.

Recall the SOR algorithm to solve (36) is given by

\[ V_j^{(k)} = V_j^{(k-1)} + \frac{\omega}{a_j} \left( d_j - a_{j-1} V_{j-1}^{(k)} - a_j V_j^{(k-1)} - a_{j+1} V_{j+1}^{(k-1)} \right) \]

for \( j = 1, \ldots, N - 1 \) and \( k = 1, 2, \ldots \).
Let e.g. \( h_j = (S_j^{n+1} - K)^+ \) be the intrinsic value of the American option at node \((j, n + 1)\).

The solution procedure to find \( V_j^{n+1} \) is then

\[
V_j^{(k)} = \max \left( h_j, V_j^{(k-1)} + \frac{\omega}{a_j} (d_j - a_{j-1} V_{j-1}^{(k)} - a_j V_j^{(k-1)} - a_{j+1} V_{j+1}^{(k-1)}) \right)
\]

(37)

for \( j = 1, \ldots, N - 1 \) and \( k = 1, 2, \ldots \).

The procedure (37) is called the \textit{projected SOR scheme}.
6. **Random Tree Method.**

This is a Monte Carlo method based on simulating a tree of paths of the underlying Markov chain $X_0, X_1, \ldots, X_m$.

Fix a branching parameter $b \geq 2$.

From the initial state $X_0$ simulate $b$ independent successor states $X_1^1, \ldots, X_1^b$ all having the law of $X_1$.

From each $X_1^i$ simulate $b$ independent successors $X_2^{i1}, \ldots, X_2^{ib}$ from the conditional law of $X_2$ given $X_1 = X_1^i$.

From each $X_2^{i1}$ generate $b$ successors $X_3^{i1i21}, \ldots, X_3^{i1i2b}$, and so on.

A generic node in the tree at time step $i$ is denoted by $X_i^{j_1j_2\cdots j_i}$.

At the $m$th time step there are $b^m$ nodes and the computational cost has exponential growth.

(e.g. if $b = 5$ and $m = 5$, there are 3125 nodes;
if $b = 5$ and $m = 10$, there are about 10 million nodes.)

Write $\hat{V}_{i}^{j_{1}j_{2}\cdots j_{i}}$ for the value of the high estimator at node $X_{i}^{j_{1}j_{2}\cdots j_{i}}$.

At the terminal nodes we set

$$\hat{V}_{m}^{j_{1}j_{2}\cdots j_{m}} = h_{m}(X_{m}^{j_{1}j_{2}\cdots j_{m}}).$$

Working backwards we get

$$\hat{V}_{i}^{j_{1}j_{2}\cdots j_{i}} = \max \left\{ h_{i}(X_{i}^{j_{1}j_{2}\cdots j_{i}}), \frac{1}{b} \sum_{j=1}^{b} D_{i+1} \hat{V}_{i+1}^{j_{1}j_{2}\cdots j_{i+1}} \right\}.$$ 

$\hat{V}_{0}$ is biased high in the sense that

$$E[\hat{V}_{0}] \geq V_{0}(X_{0})$$

and converges in probability and in norm to the true value $V_{0}(X_{0})$ as $b \to \infty$. 
8. **Low Estimator.**

Write $\hat{v}_{j_1 j_2 \cdots j_i}$ for the value of the low estimator at node $X_{i}^{j_1 j_2 \cdots j_i}$.

At the terminal nodes we set

$$\hat{v}_{m}^{j_1 j_2 \cdots j_m} = h_m(X_{m}^{j_1 j_2 \cdots j_m}).$$

Working backwards, for $k = 1, \ldots, b$, we set

$$\hat{v}_{i, k}^{j_1 j_2 \cdots j_i} = \begin{cases} h_i(X_{i}^{j_1 j_2 \cdots j_i}) & \text{if } \frac{1}{b-1} \sum_{j=1, j \neq k}^{b} D_{i+1} \hat{v}_{i+1}^{j_1 j_2 \cdots j_i} \leq h_i(X_{i}^{j_1 j_2 \cdots j_i}) \\ D_{i+1} \hat{v}_{i+1}^{j_1 j_2 \cdots j_i} & \text{otherwise} \end{cases}$$

we then set

$$\hat{v}_{i}^{j_1 j_2 \cdots j_i} = \frac{1}{b} \sum_{k=1}^{b} \hat{v}_{i, k}^{j_1 j_2 \cdots j_i}.$$ 

$\hat{v}_0$ is biased low and converges in probability and in norm to $V_0(X_0)$ as $b \to \infty$. 

254
Example. High Estimator.

Exercise price $K = 100$, interest rate $r = 0$.
Asset price (call price)
9. **Confidence Interval.**

Let $\bar{V}_0(M)$ denote the sample mean of the $M$ replications of $\hat{V}_0$, $s_V(M)$ the sample standard deviation, and $z_{\frac{\delta}{2}}$ the $1 - \frac{\delta}{2}$ quantile of the normal distribution. Then

$$\bar{V}_0(M) \pm z_{\frac{\delta}{2}} \frac{s_V(M)}{\sqrt{M}}$$

provides an asymptotically valid (for large $M$) $1 - \delta$ confidence interval for $E[\hat{V}_0]$. Similarly, we can get the confidence interval for $E[\hat{v}_0]$.

The interval

$$\left(\bar{v}_0(M) - z_{\frac{\delta}{2}} \frac{s_v(M)}{\sqrt{M}}, \bar{v}_0(M) + z_{\frac{\delta}{2}} \frac{s_v(M)}{\sqrt{M}}\right)$$

contains the unknown value $V_0(X_0)$ with probability of at least $1 - \delta$. 
10. **Regression Method.**

Assume that the continuation value can be expressed as

$$E(V_{i+1}(X_{i+1}) \mid X_i = x) = \sum_{r=1}^{n} \beta_{ir} \psi_r(x) = \beta_i^T \psi(x),$$  \hspace{1cm} (38)

where $\beta_i = (\beta_{i1}, \ldots, \beta_{in})^T$, $\psi(x) = (\psi_1(x), \ldots, \psi_n(x))^T$, and $\psi_r$, $r = 1, \ldots, n$, are basis functions (e.g. polynomials $1, x, x^2, \ldots$). Then the vector $\beta_i$ is given by

$$\beta_i = B_{\psi}^{-1} b_{\psi V},$$

where $B_{\psi} = E[\psi(X_i) \psi(X_i)^T]$ is an $n \times n$ matrix (assumed to be non-singular), $b_{\psi V} = E[\psi(X_i) V_{i+1}(X_{i+1})]$ is an $n$ column vector, and the expectation is over the joint distribution of $(X_i, X_{i+1})$. 

257
The regression method can be used to estimate $\beta_i$ as follows:

- generate $b$ independent paths $(X_{1j}, \ldots, X_{mj})$, $j = 1, \ldots, b$, and
- suppose that the values $V_{i+1}(X_{i+1,j})$ are known, then

$$\hat{\beta}_i = \hat{B}_\psi^{-1} \hat{b}_\psi V,$$

where $\hat{B}_\psi$ is an $n \times n$ matrix with $qr$ entry

$$\frac{1}{b} \sum_{j=1}^{b} \psi_q(X_{ij}) \psi_r(X_{ij})$$

and $\hat{b}_\psi V$ is an $n$ vector with $r$th entry

$$\frac{1}{b} \sum_{j=1}^{b} \psi_r(X_{ij}) V_{i+1}(X_{i+1,j}).$$

However, $V_{i+1}$ is unknown and must be replaced by some estimated value $\hat{V}_{i+1}$. 
11. **Pricing Algorithm.**

(a) Generate $b$ independent paths $\{X_{0j}, X_{1j}, \ldots, X_{mj}\}$, $j = 1, \ldots, b$, of the Markov chain, where $X_{0j} = X_0$, $j = 1, \ldots, b$.

(b) Set $\hat{V}_{mj} = h_m(X_{mj})$, $j = 1, \ldots, b$.

(c) For $i = m - 1, \ldots, 0$, and given estimated values $\hat{V}_{i+1,j}$, $j = 1, \ldots, b$, use the regression method to calculate $\hat{\beta}_i$ and set $\hat{V}_{ij} = \max\{h_i(X_{ij}), D_{i+1} \hat{\beta}_i^T \psi(X_{ij})\}$.

(d) Set $\hat{V}_0 = \frac{1}{b} \sum_{j=1}^{b} \hat{V}_{0j}$.

The regression algorithm is biased high and $\hat{V}_0$ converges to $V_0(X_0)$ as $b \to \infty$ if the relation (38) holds for all $i = 0, \ldots, m - 1$. 
12. **Longstaff–Schwartz Algorithm.**

Same as 10.11, except in computing $\hat{V}_{ij}$:

$$\hat{V}_{ij} = \begin{cases} h_i(X_{ij}) & \text{if } D_{i+1} \hat{\beta}_i^T \psi(X_{ij}) \leq h_i(X_{ij}) \\ D_{i+1} \hat{V}_{i+1,j} & \text{otherwise} \end{cases}$$

The LS algorithm is biased low and $\hat{V}_0$ converges to $V_0(X_0)$ as $b \to \infty$ if the relation (38) holds for all $i = 0, \ldots, m - 1$.

13. **Other Methods.** Parametric approximation, state-space partitioning, stochastic mesh method, duality algorithm, and obstacle (free boundary-value) problem formulation.

(See Glasserman (2004) and Seydel (2006) for details.)
11. Exotic Option Pricing

1. Introduction.

Derivatives with more complicated payoffs than the standard European or American calls and puts are called *exotic options*, many of which are so called *path-dependent options*.

Some common exotic options are

- **Asian option**: Average price call and put, average strike call and put. Arithmetic or geometric average can be taken.
- **Barrier option**: Up/down-and-in/out call/put. Double barrier, partial barrier and other time dependent effects possible.
- **Range forward contract**: It is a portfolio of a long call with a higher strike price and a short put with a lower strike price.
- **Bermudan option**: It is a nonstandard American option in which early exercise is restricted to certain fixed times.
• **Compound option**: It is an option on an option. There are four basic forms: call-on-call, call-on-put, put-on-call, put-on-put.

• **Chooser option**: The option holders have the right to choose whether the option is a call or a put.

• **Binary (digital) option**: The terminal payoff is a fixed amount of cash, if the underlying asset price is above a strike price, and it pays nothing otherwise.

• **Lookback option**: The terminal payoff depends on the maximum or minimum realized asset price over the life of the option.
Examples.

*Range forward contract:* Long call and short put

Assume you have a long position in a call with strike price $X_2$ and a short position in a put with strike price $X_1 < X_2$. Then the terminal payoff is given by

$$ (S - X_2)^+ - (X_1 - S)^+ = \begin{cases} S - X_2 & S > X_2 \\ 0 & X_1 \leq S \leq X_2 \\ S - X_1 & S < X_1 \end{cases} $$

Buy this option, if you do not think that the asset price will go down.

⇒ You can generate cash from a price increase.
Option price = Call price - Put price
**Compound option:** Call on call option

\[ \tilde{C}(T_2) = (S(T_2) - X_2)^+ \]

\[ \tilde{C}(T_1) \equiv \tilde{C}(S(T_1), X_2, T_1, \ldots) \]

\[ C(t) = ? \]

\[ C(T_1) = (\tilde{C}(T_1) - X_1)^+ \]

**Binary (digital) option:**

\[
\begin{align*}
\text{Price} = & \ E[e^{-rT}Q1_{s_T \geq X}] = e^{-rT}Q \ E[1_{s_T \geq X}] = e^{-rT}Q \ P(S_T \geq X).
\end{align*}
\]

The price can be easily computed, since \( S_T \) is lognormally distributed.
Lookback Option

Let

\[ m = \min_{t \leq u \leq T} S(u) \quad \text{and} \quad M = \max_{t \leq u \leq T} S(u). \]

- (Floating strike) Call: \((S_T - m)^+\).
- (Floating strike) Put: \((M - S_T)^+\).
- Fixed strike Call: \((M - X)^+\).
- Fixed strike Put: \((X - m)^+\).

266
2. **Barrier Options.**

The existence of options at the expiration date depends on whether the underlying asset prices have crossed certain values (barriers).

There are four basic forms: down-and-out, up-and-out, down-and-in, up-and-in.

If there is only one barrier then we can derive analytic formulas for standard European barrier options.

However, there are many situations (two barriers, American options, etc.), in which we have to rely on numerical procedures to find the option values.

The standard binomial and trinomial schemes can be used for this purpose, but their convergence is very slow, because the barriers assumed by the tree is different from the true barrier.
Barriers assumed by binomial trees.
The usual tree calculations implicitly assume that the outer barrier is the true barrier.
Barriers assumed by trinomial trees.
3. **Positioning Nodes on Barriers.**

Assume that there are two barriers $H_1$ and $H_2$ with $H_1 < H_2$.

Assume that the trinomial scheme is used for the option pricing with $m = 1$ and $d = \frac{1}{u}$.

Choose $u$ such that nodes lie on both barriers.

$u$ must satisfy $H_2 = H_1 u^N$ (and hence $H_1 = H_2 d^N$), or equivalently

$$\ln H_2 = \ln H_1 + N \ln u,$$

with $N$ an integer.

It is known that $u = e^{\sigma \sqrt{3 \Delta t}}$ is a good choice in the standard trinomial tree, recall the Hull–White model (7.20).
A good choice of $N$ is therefore

$$N = \text{int} \left[ \frac{\ln H_2 - \ln H_1}{\sigma \sqrt{3 \Delta t}} + 0.5 \right]$$

and $u$ is determined by

$$u = \exp \left( \frac{1}{N} \ln \frac{H_2}{H_1} \right).$$

Normally, the trinomial tree is constructed so that the central node is the initial asset price $S$. In this case, the asset price at the first node is the initial asset price. After that, we choose the central node of the tree (beginning with the first period) to be $H_1 u^M$, where $M$ is an integer that makes this quantity as close as possible to $S$, i.e.

$$M = \text{int} \left[ \frac{\ln S - \ln H_1}{\ln u} + 0.5 \right].$$
Tree with nodes lying on each of two barriers.
4. **Adaptive Mesh Model.**

Computational efficiency can be greatly improved if one projects a high resolution tree onto a low resolution tree in order to achieve a more detailed modelling of the asset price in some regions of the tree.

E.g. to price a standard American option, it is useful to have a high resolution tree near its maturity around the strike price.

To price a barrier option it is useful to have a high resolution tree near its barriers. See Hull (2005) for details.
5. **Asian Options.**

The terminal payoff depends on some form of averaging of the underlying asset prices over a part or the whole of the life of the option.

There are two types of terminal payoffs:

- \( \max(A - X, 0) \) (*average price call*) and
- \( \max(S_T - A, 0) \) (*average strike call*),

where \( A \) is the average of asset prices, and there are many different ways of computing \( A \).

From now on we focus on pricing average price calls, the results for average strike calls can be obtained similarly.

We also assume that the asset price \( S \) follows a lognormal process in a risk-neutral world.
6. **Continuous Arithmetic Average.**

The average $A$ is computed as

$$ A = \frac{1}{T} I(T), $$

where the function $I(t)$ is defined by

$$ I(t) = \int_0^t S(u) \, du. $$

Assume that $V(S, I, t)$ is the option value at time $t$. Then $V$ satisfies the following diffusion equation:

$$ \frac{\partial V}{\partial t} + rS \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} - rV + S \frac{\partial V}{\partial I} = 0 \quad (39) $$

with suitable boundary and terminal conditions.

If finite difference methods are used to solve (39), then the schemes are prone to serious oscillations and implicit schemes have poor performance, due to the missing of one diffusion term in this two-dimensional PDE.
7. **Continuous Geometric Average.**

The average $A$ is computed as

$$A = \exp \left( \frac{1}{T} I(T) \right),$$

where the function $I(t)$ is defined by

$$I(t) = \int_0^t \ln S(u) \, du.$$  

The option value $V$ satisfies the following diffusion equation:

$$\frac{\partial V}{\partial t} + r S \frac{\partial V}{\partial S} + \frac{1}{2} \sigma^2 S^2 \frac{\partial^2 V}{\partial S^2} - r V + \ln S \frac{\partial V}{\partial I} = 0.$$  \hspace{1cm} (40)

**Exercise.** Show that if we define a new variable

$$y = \frac{I + (T - t) \ln S}{T}$$

and seek a solution of the form $V(S, I, t) = F(y, t)$, then the equation (40) can be reduced to a one-state-variable PDE:

$$\frac{\partial F}{\partial t} + \frac{1}{2} \left( \frac{\sigma (T - t)}{T} \right)^2 \frac{\partial^2 F}{\partial y^2} + \left( r - \frac{1}{2} \sigma^2 \right) \left( \frac{T - t}{T} \right) \frac{\partial F}{\partial y} - r F = 0.$$
8. **Discrete Geometric Average: Case** \( t \geq T_0 \).

Assume that the averaging period is \([T_0, T]\) and that the sampling points are \( t_i = T_0 + i \Delta t \), \( i = 1, 2, \ldots, n \), \( \Delta t = \frac{T - T_0}{n} \).

The average \( A \) is computed as

\[
A = \left( \prod_{i=1}^{n} S(t_i) \right)^{\frac{1}{n}}.
\]

Assume

\[
t \geq T_0,
\]

i.e. the current time is already within the averaging period.

Assume

\[
t = t_k + \alpha \Delta t
\]

for some integer \( k \): \( 0 \leq k \leq n - 1 \) and \( 0 \leq \alpha < 1 \).
Then \( \ln A \) is a normal random variable with mean \( \ln \tilde{S}(t) + \mu_A \) and variance \( \sigma_A^2 \), where

\[
\tilde{S}(t) = \left[ S(t_1) \cdots S(t_k) \right]^{\frac{1}{n}} S(t)^{\frac{n-k}{n}}
\]

\[
\mu_A = \left( r - \frac{\sigma^2}{2} \right) (T - T_0) \left( \frac{n - k}{n^2} (1 - \alpha) + \frac{(n - k - 1)(n - k)}{2n^2} \right)
\]

\[
\sigma_A^2 = \sigma^2 (T - T_0) \left( \frac{(n - k)^2}{n^3} (1 - \alpha) + \frac{(n - k - 1)(n - k)(2n - 2k - 1)}{6n^3} \right)
\]

The European call price at time \( t \) is

\[
c(S, t) = e^{-r(T-t)} \left( \tilde{S}(t) e^{\mu_A + \frac{1}{2} \sigma_A^2} \Phi(d_1) - X \Phi(d_2) \right)
\]

where

\[
d_1 = \frac{1}{\sigma_A} \left( \ln \frac{\tilde{S}(t)}{X} + \mu_A \right) + \sigma_A
\]

and

\[
d_2 = d_1 - \sigma_A.
\]
Proof.

\[ t = t_k + \alpha \Delta t, \quad \Delta t = \frac{T-T_0}{n}, \quad 0 \leq k \leq n - 1, \quad 0 \leq \alpha < 1 \]

\[ A = [S(t_1) \cdots S(t_n)]^{\frac{1}{n}} = \left[ \frac{S(t_n)}{S(t_{n-1})} \frac{S^2(t_{n-1})}{S^2(t_{n-2})} \cdots \frac{S^{n-k}(t_{k+1})}{S^{n-k}(t_{k})} \right]^{\frac{1}{n}} \]

\[ = \left[ R_n R_{n-1}^{2} \cdots R_{k+2}^{n-k-1} R_{k}^{n-k} \right]^{\frac{1}{n}} \tilde{S}(t) \]

\[ \Rightarrow \ln A = \ln \tilde{S}(t) + \frac{1}{n} [\ln R_n + 2 \ln R_{n-1} + \ldots + (n - k - 1) \ln R_{k+2} + (n - k) \ln R_t] \]

Since \( S \) is a lognormal process \( (dS = r S \, dt + \sigma S \, dW) \), we have that \( \ln R_n, \ldots, \ln R_{k+2}, \ln R_t \) are independent normally distributed and

\[ \ln R_n, \ldots, \ln R_{k+2} \sim N(\mu \, \Delta t, \sigma^2 \, \Delta t), \]

\[ \ln R_t \sim N(\mu (t_{k+1} - t), \sigma^2 (t_{k+1} - t)) = N(\mu (1 - \alpha) \, \Delta t, \sigma^2 (1 - \alpha) \, \Delta t), \]

where \( \mu = r - \frac{1}{2} \sigma^2 \).
Hence ln \( A \) is normal with mean

\[
E[\ln A] = E[\ln \tilde{S}(t)] + \frac{1}{n} E[\ln R_n + 2 \ln R_{n-1} + \ldots + (n-k-1) \ln R_{k+2} + (n-k) \ln R_t]
\]

\[
= \ln \tilde{S}(t) + \frac{1}{n} [\mu \Delta t + 2 \mu \Delta t + \ldots + (n-k-1) \mu \Delta t + (n-k) \mu (1-\alpha) \Delta t]
\]

\[
= \ln \tilde{S}(t) + \frac{1}{n} \mu \Delta t [1 + 2 + \ldots + (n-k-1) + (n-k) (1-\alpha)]
\]

\[
= \ln \tilde{S}(t) + \mu \Delta t \left[ \frac{(n-k-1)(n-k)}{2n} + \frac{n-k}{n} (1-\alpha) \right] = \ln \tilde{S}(t) + \mu_A.
\]

Moreover, the variance of \( \ln A \) is given by

\[
\text{Var}(\ln A) = \frac{1}{n^2} \left[ \text{Var}(\ln R_n) + \ldots + (n-k-1)^2 \text{Var}(\ln R_{k+2}) + (n-k)^2 \text{Var}(\ln R_t) \right]
\]

\[
= \frac{1}{n^2} \sigma^2 \Delta t \left[ 1^2 + 2^2 + \ldots + (n-k-1)^2 + (n-k-1)^2 (1-\alpha) \right]
\]

\[
= \sigma^2 \Delta t \left[ \frac{(n-k-1)(n-k)(2n-2k-1)}{6n^2} + \frac{(n-k)^2}{n^2} (1-\alpha) \right] = \sigma_A^2.
\]

[Used: \( \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \) and \( \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6} \).]
As $A = e^{\ln A}$, $A$ is lognormally distributed and the important formula in 5.5. tells us that

$$E[(A - X)^+] = E(A) \Phi(d_1) - X \Phi(d_2),$$

where $d_1 = \frac{1}{s} \ln \frac{E(A)}{X} + \frac{s}{2}$ and $d_2 = d_1 - s$, with $s^2 = \text{Var}(\ln A)$.

Now

$$\ln A \sim N(\ln \tilde{S}(t) + \mu_A, \sigma_A^2)$$

and so, on recalling 5.4.,

$$E(A) = \tilde{S}(t) e^{\mu_A + \frac{1}{2} \sigma_A^2}.$$

Combining gives

$$E[(A - X)^+] = \tilde{S}(t) e^{\mu_A + \frac{1}{2} \sigma_A^2} \Phi(d_1) - X \Phi(d_2),$$

where

$$d_1 = \frac{1}{\sigma_A} \left[ \ln \frac{\tilde{S}(t)}{X} + \mu_A + \frac{1}{2} \sigma_A^2 \right] + \frac{1}{2} \sigma_A = \frac{1}{\sigma_A} \left[ \ln \frac{\tilde{S}(t)}{X} + \mu_A \right] + \sigma_A$$
and $d_2 = d_1 - \sigma_A$.

Finally, the Asian (European) call price is $e^{-r(T-t)} E[(A - X)^+]$. 
Aside: \[ \sum_{i=1}^{n} i = \frac{n(n+1)}{2} \text{ and } \sum_{i=1}^{n} i^2 = \frac{n(n+1)(2n+1)}{6}. \]

Let \( f(x) = x \) then

\[
\frac{1}{2} = \int_{0}^{1} f(x) \, dx = \sum_{i=1}^{n} \int_{\frac{i-1}{n}}^{\frac{i}{n}} x \, dx = \sum_{i=1}^{n} \frac{1}{2} x^2 \bigg|_{\frac{i-1}{n}}^{\frac{i}{n}} = \sum_{i=1}^{n} \frac{1}{2} \left[ \left( \frac{i}{n} \right)^2 - \left( \frac{i-1}{n} \right)^2 \right]
\]

\[
= \sum_{i=1}^{n} \frac{1}{2} \left[ \frac{2i}{n^2} - \frac{1}{n^2} \right] = \frac{1}{n^2} \sum_{i=1}^{n} i - \frac{1}{2n^2} \sum_{i=1}^{n} 1 = \frac{1}{n^2} \sum_{i=1}^{n} i - \frac{1}{2n}
\]

Hence

\[
\frac{1}{n^2} \sum_{i=1}^{n} i = \frac{1}{2} + \frac{1}{2n} = \frac{n+1}{2n} \quad \Rightarrow \quad \sum_{i=1}^{n} i = \frac{n(n+1)}{2}.
\]

Similarly, for \( f(x) = x^2 \) we obtain

\[
\frac{1}{3} = \int_{0}^{1} f(x) \, dx = \sum_{i=1}^{n} \int_{\frac{i-1}{n}}^{\frac{i}{n}} x^2 \, dx = \ldots
\]
9. **Discrete Geometric Average: Case** $t < T_0$.

**Exercise.**

Assume

$$t < T_0,$$

i.e. the current time is before the averaging period.

Show that $\ln A$ is a normal random variable with mean $\ln S(t) + \mu_A$ and variance $\sigma_A^2$, where

$$\mu_A = \left( r - \frac{\sigma^2}{2} \right) \left( (T_0 - t) + \frac{n + 1}{2n} (T - T_0) \right)$$

$$\sigma_A^2 = \sigma^2 \left( (T_0 - t) + \frac{(n + 1)(2n + 1)}{6n^2} (T - T_0) \right).$$

The European call price at time $t$ is

$$c(S, t) = e^{-r(T-t)} \left( S e^{\mu_A + \frac{1}{2} \sigma_A^2} \Phi(d_1) - X \Phi(d_2) \right)$$

where $d_1 = \frac{1}{\sigma_A} \left( \ln \frac{S}{X} + \mu_A \right) + \sigma_A$ and $d_2 = d_1 - \sigma_A$. 
10. **Limiting Case** $n \equiv 1$.

If $n = 1$ then $A = S(T)$.

Furthermore,

\[
\mu_A = \left(r - \frac{\sigma^2}{2}\right)(T - t)
\]
\[
\sigma^2_A = \sigma^2(T - t).
\]

The call price reduces to the Black–Scholes call price.
$A = (\prod_{i=1}^1 S(t_i))^1 = S(t_1) = S(T)$, independently from the choice of $T_0$.

Hence choose e.g. $T_0 > t$.

$$\mu_A = (r - \frac{\sigma^2}{2}) [(T_0 - t) + (T - T_0)] = (r - \frac{\sigma^2}{2}) (T - t),$$

$$\sigma_A^2 = \sigma^2 [(T_0 - t) + (T - T_0)] = \sigma^2 (T - t).$$

Call price:

$$c = e^{-r (T-t)} \left( S e^{(r-\frac{\sigma^2}{2}) (T-t) + \frac{1}{2} \sigma^2 (T-t)} \Phi(d_1) - X \Phi(d_2) \right) = S \Phi(d_1) - X e^{-r (T-t)} \Phi(d_2),$$

where

$$d_1 = \frac{1}{\sigma_A} \left( \ln \frac{S}{X} + \mu_A \right) + \sigma_A$$

$$= \frac{1}{\sigma \sqrt{T-t}} \left( \ln \frac{S}{X} + (r - \frac{\sigma^2}{2}) (T - t) \right) + \sigma \sqrt{T-t}$$

$$= \frac{1}{\sigma \sqrt{T-t}} \left( \ln \frac{S}{X} + r (T - t) \right) + \frac{1}{2} \sigma \sqrt{T-t}$$

and $d_2 = d_1 - \sigma_A = \frac{1}{\sigma \sqrt{T-t}} \left( \ln \frac{S}{X} + r (T - t) \right) - \frac{1}{2} \sigma \sqrt{T-t}.$
This is just the Black–Scholes pricing formula (16), see 7.13.
11. **Limiting Case** $n = \infty$.

If $n \to \infty$ then $A \to \exp \left( \frac{1}{T - T_0} \int_{T_0}^{T} \ln S(u) \, du \right)$.

Furthermore, if time $t < T_0$ then

$$
\mu_A \to (r - \frac{\sigma^2}{2}) \left( \frac{1}{2} T + \frac{1}{2} T_0 - t \right),
$$

$$
\sigma^2_A \to \sigma^2 \left( \frac{1}{3} T + \frac{2}{3} T_0 - t \right).
$$

If time $t \geq T_0$ then

$$
\bar{S}(t) \to \left[ S(t) \right]^{\frac{T-t}{T-T_0}} \exp \left( \frac{1}{T - T_0} \int_{T_0}^{t} \ln S(u) \, du \right),
$$

$$
\mu_A \to \left( r - \frac{\sigma^2}{2} \right) \frac{(T - t)^2}{2(T - T_0)},
$$

$$
\sigma^2_A \to \sigma^2 \frac{(T - t)^3}{3(T - T_0)^2}.
$$

The above values, together with the discrete average price formulas, then yield pricing
formulas for the continuous geometric mean.
Proof.

Two cases: $t < T_0$ and $t \geq T_0$. Here we look at the latter.

\[ \mu_A = \left( r - \frac{\sigma^2}{2} \right) (T - T_0) \left( \frac{1}{n} (1 - \frac{k}{n}) (1 - \alpha) + \frac{1}{2} (1 - \frac{k}{n} - \frac{1}{n}) (1 - \frac{k}{n}) \right), \]

\[ \sigma^2_A = \sigma^2 (T - T_0) \left( \frac{1}{n} (1 - \frac{k}{n})^2 (1 - \alpha) + \frac{1}{6} (1 - \frac{k}{n} - \frac{1}{n}) (1 - \frac{k}{n}) (2 - 2\frac{k}{n} - \frac{1}{n}) \right). \]

Since $t = t_k + \alpha \Delta t = T_0 + k \Delta t + \alpha \Delta t = T_0 + k \frac{T-T_0}{n} + \alpha \frac{T-T_0}{n}$, we have that

\[ \frac{t - T_0}{T - T_0} = \frac{k}{n} + \frac{\alpha}{n} \implies \frac{k}{n} \rightarrow \frac{t - T_0}{T - T_0} \iff 1 - \frac{k}{n} \rightarrow \frac{T - t}{T - T_0}. \]

Hence, as $n \rightarrow \infty$, we have that

\[ \mu_A \rightarrow \left( r - \frac{\sigma^2}{2} \right) (T - T_0) \left( \frac{1}{2} \frac{(T-t)^2}{(T-T_0)^2} \right) = \left( r - \frac{\sigma^2}{2} \right) \frac{(T-t)^2}{2 (T-T_0)}, \]

\[ \sigma^2_A \rightarrow \sigma^2 (T - T_0) \left( \frac{1}{3} \frac{(T-t)^3}{(T-T_0)^3} \right) = \sigma^2 \frac{(T-t)^3}{3 (T-T_0)^2}. \]
Moreover,

\[ \tilde{S}(t) = \left[ S(t_1) \cdots S(t_k) \right]^{\frac{1}{n}} S(t)^{\frac{n-k}{n}} \]

\[ \Rightarrow \ln \tilde{S}(t) = \frac{1}{n} \left[ \ln S(t_1) + \ldots + \ln S(t_k) \right] + \frac{n-k}{n} \ln S(t) \]

\[ = \frac{1}{T - T_0} \Delta t \left[ \ln S(t_1) + \ldots + \ln S(t_k) \right] + \left( 1 - \frac{k}{n} \right) \ln S(t) . \]

As \( t = t_k + \alpha \Delta t \rightarrow t_k \) as \( n \rightarrow \infty \), we have

\[ \ln \tilde{S}(t) \rightarrow \frac{1}{T - T_0} \int_{T_0}^{t} \ln S(u) \, du + \frac{T - t}{T - T_0} \ln S(t) \]

\[ = \frac{1}{T - T_0} \int_{T_0}^{t} \ln S(u) \, du + \ln \left( S^{\frac{T-t}{T-T_0}}(t) \right) \]

\[ \Rightarrow \quad \tilde{S}(t) \rightarrow S^{\frac{T-t}{T-T_0}}(t) \exp \left( \frac{1}{T - T_0} \int_{T_0}^{t} \ln S(u) \, du \right) . \]

Note that the average \( A = \left[ S(t_1) \cdots S(t_n) \right]^{\frac{1}{n}} \) converges to

\[ \exp \left( \frac{1}{T - T_0} \int_{T_0}^{t} \ln S(u) \, du \right) \quad \text{as} \; n \rightarrow \infty , \]
which is just the continuous geometric average.
12. **Discrete Arithmetic Average.**

The average $A$ is computed as

$$A = \frac{1}{n} \sum_{i=1}^{n} S(t_i).$$

The option value is difficult to compute because $A$ is not lognormal.

The binomial method can be used but great care must be taken so that the number of nodes does not increase exponentially.

Monte Carlo simulation is possibly the most efficient method.

Since an option with discrete arithmetic average is very similar to that with discrete geometric average, and since the latter has an analytic pricing formula, the control variate method (with the geometric call price as control variate) can be used to reduce the variance of the Monte Carlo simulation.