http://www.goddardconsulting.ca/financial-engineering.html#Derivatives

**Option Pricing - Finite Difference Methods**

Finite difference methods (also called finite element methods) are used to price options by approximating the (continuous-time) differential equation that describes how an option price evolves over time by a set of (discrete-time) difference equations. The discrete difference equations may then be solved iteratively to calculate a price for the option.

This tutorial covers the general mathematical concepts behind finite diffence methods. Companion tutorials cover specific aspects of the following three finite difference methods, including links to examples of implementing the methods in MATLAB,

1. [Explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html)
2. [Implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html)
3. [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html)

Finite difference methods are very similar to binomial and trinomial models. The [Binomial Model](http://www.goddardconsulting.ca/option-pricing-binomial-index.html) series of tutorials cover their use in option pricing including examples of implementing serveral versions of the binomial model in MATLAB. Other [Financial Engineering](http://www.goddardconsulting.ca/financial-engineering.html) tutorials may be found on the [Software Tutorials](http://www.goddardconsulting.ca/software-tutorials.html) page.

Each of the finite difference methods has advantages and disadvantages. However, they all involve a similar four step process. The following sections discuss those four steps,

1. [Discretize the appropriate (continuous-time, partial) differential equation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMDiscretization).
2. [Specify a grid of potential current and future prices for the underlying asset](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid).
3. [Calculate the payoff of the option at specfic *boundaries* of the grid of potential underlying prices](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMPayoff).
4. Iteratively determine the option price at all other grid points, including the point for the current time and underlying price (i.e. the option price today). The iteration procedure is differrent depending on whether the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html), [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html)or [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html) is being used and whether there is the possibility of early exercise of the option.

**Discretizing a Differential Equation**

Black, Scholes and Merton showed that a riskless portfolio made up of an asset with value *S* and an option with value *ƒ(t,S)* satisfies the differential equationBlack-Scholes-Merton PDE

Equation 1: Black-Scholes-Merton PDE

[Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMDiscretization) is called a partial differential equation. It is comprised of (partial) derivatives with repsect to time *t* and asset value *S*.

The solution to the Black-Scholes-Merton PDE depends on several factors, including the expected form of *ƒ(t,S)* and *boundary* conditions imposed on the solution. Boundary conditions are specified to reflect the expected payoff of the option at expiry and for minimum and maximum values of *S*. They are discussed in more detail in the [Specifying *Boundaries* Conditions](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMPayoff) section.

There are several ways that the Black-Scholes-Merton PDE can be solved for the unknown value of *ƒ(t,S)*. Black and Scholes developed an exact analytic solution for pricing plain vanilla European Put and Call options. However often an analytic solution is not available. In such instances finite difference methods can be used to calculate approximate solutions for *ƒ(t,S)* that are valid over small discrete time intervals *Δt*.

At the heart of finite difference methods are the approximation of the partial derivatives in the PDE by appropriate difference equations. Depending on the difference equations used then the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html), [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) or [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html) is obtained.

The next sections outline different ways of approximating the partial derivatives of[Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMDiscretization).

**Functions of One Variable**

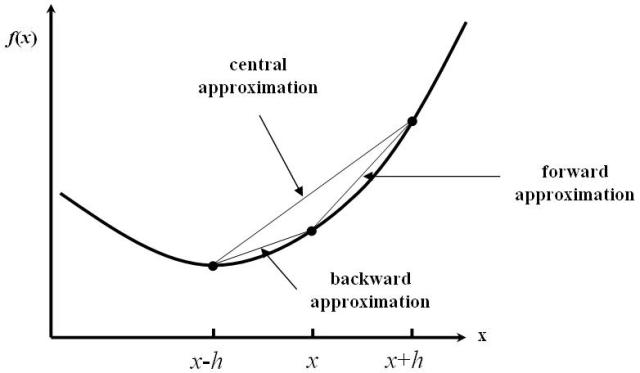
Consider the function of one variable *ƒ(x)* shown in [Figure 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FcnOneVar).

Figure 1: Function of One Variable

Then *ƒ′(x)*, the derivative of the function at *x*, can be approximated in many ways. The most common are called the [forward](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#forwardapprox), [backward](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) and [central](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#centralapprox) approximation, all of which are drawn and indicated on [Figure 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FcnOneVar).

**Forward Approximation**

Consider the Taylor's series expansion for *ƒ(x+h)*,Taylors Series Expansion

Equation 2: Taylor's Series Expansion

Then re-arranging Equation 2 leads to,Forward Approx.

Equation 3: Forward Approximation for *ƒ′(x)*

where *Ο(h)* symbolizes terms *of the order of h*. Assuming the step *h* is small then *Ο(h)*may be ignored and Equation 3 represents an approximation to *ƒ′(x)* at *x*.

**Backward Approximation**

Consider the Taylor's series expansion for *ƒ(x-h)*,Taylors Series Expansion

Equation 4: Taylor's Series Expansion

Then re-arranging Equation 4 leads to,Forward Approx.

Equation 5: Backward Approximation for *ƒ′(x)*

where *Ο(h)* symbolizes terms *of the order of h*. Assuming the step *h* is small then *Ο(h)*may be ignored and Equation 5 represents an approximation to *ƒ′(x)* at *x*.

**Central Approximation**

In addition to the forward and backwards approximation there is also a central approximation to *ƒ′(x)*. This is formed by subtracting [Equation 4](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) from [Equation 2](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#forwardapprox) and rearranging the result to obtain the equation,Central Approx.

Equation 6: Central Approximation for *ƒ′(x)*

Note that the truncation error for the central approximation is *Ο(h2)*, i.e. *of the order of h2*. This implies that it converges to the correct solution faster than either the forward or backward approximations.

**Approximating the *2 nd* Derivative**

In addition to the above first derivative approximations, to discretize the Black-Scholes-Merton PDE of [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMDiscretization) also requires an approxmation for the second (partial) derivative term. This is formed by adding the two Talyor's series approximations of [Equation 2](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#forwardapprox) and [Equation 4](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) and rearranging the result to obtain the equation,Second Deriv. Approx.

Equation 7: Approximation for *f″(x)*

Note that as with the [central approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#centralapprox) for the first derivative, the truncation error for this second derivative approximation is *Ο(h2)*, i.e. *of the order of h2*. This has consequences for the speed of convergence of the finite difference algorithms.

**Functions of Two Variables**

When the function to be approximated is dependent on more than one variable (as is the case with *ƒ(t,S)*) and the PDE being solved contains derivatives with respect to more than one variable (as is the case with the Black-Scholes-Merton PDE of [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMDiscretization)) then the PDE is discretized by using the above approximations derived for functions of one variable while holding all other variables constant.

**Specifying a Grid of Underlying Asset Prices**

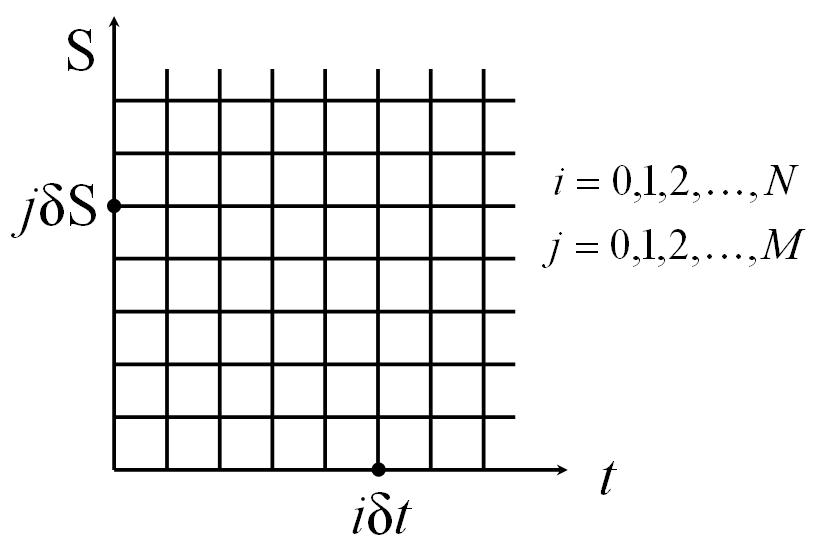
The second step in pricing options using a finite difference method is to create a lattice, or grid, of potential future prices of the underlying asset(s). A typical grid is shown in [Figure 2](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid).

Figure 2: Grid of Time and Underlying Asset Prices

The lattice is generated by dividing the time between today and expiry into *M* equal periods, and the underlying price into *N* equal levels. This leads to a grid with *M+1* time points and *N+1* price levels. The grid is typically chosen so that the current price of the underlying asset lies close to the middle of the *N* equal price levels of the grid.

Note that time is considered to run from zero (today) to *NΔt=T* (at expiry). Alternative derivations may reverse this order and consider time counting down towards expiry.

**Specifying *Boundary* Conditions**

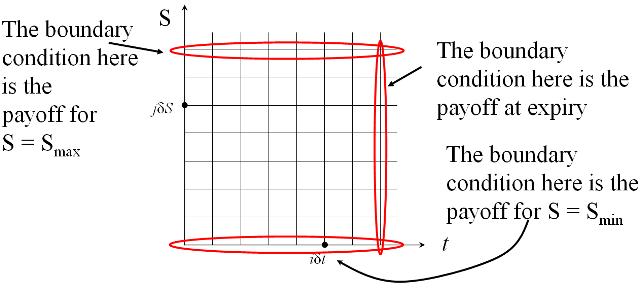
The third step in pricing options using finite difference methods is to calculate the payoff at each node on the boundary of the grid - hence they are called *boundary* conditions. The specific boundary, and the payoff for the option at the boundary, will be different for different types of options and different parameters used in a given option. However, a general picture is given in [Figure 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMPayoff).

Figure 3: Specifying Boundary Conditions

Once the boundary conditions have been specified the interior points can be calulated using an iterative approach. The specifics of the iterative approach is different depending on the finite different method chosen and are discussed in the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html), [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) and [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html) tutorials respectively.

**Option Pricing Using The Explicit Finite Difference Method**

This tutorial discusses the specifics of the explicit finite difference method as it is applied to option pricing. Example code implementing the explicit method in MATLAB and used to price a simple option is given in the [Explicit Method - A MATLAB Implementation](http://www.goddardconsulting.ca/matlab-finite-diff-explicit.html)tutorial.

The [Finite Difference Methods](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html) tutorial covers general mathematical concepts behind finite diffence methods and should be read before this tutorial. Alternative finite difference methods, namely the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) and the [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html), are covered in companion tutorials.

**Discretizing the Black-Scholes-Merton PDE**

For the explicit method the Black-Scholes-Merton partial differential equation,Black-Scholes-Merton PDE

is discretized using the following formulae

* use a [backward approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) for *∂ƒ/∂t*  (Compare this with the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html)where the [forward approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#forwardapprox) is used.)Backwards Approx.
* use a [central approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#centralapprox) for *∂ƒ/∂S*

Central Approx.

* use a [standard approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#secondderivapprox) for *∂2ƒ/∂S2*

Second Order Approx.

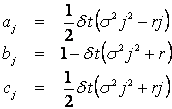
where the indices *i* and *j* represent nodes on the [pricing grid](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid).

Substituting these approximations into the PDE gives,Explicit Formulae Full

which reduces to

Explicit Formulae Reduced

Equation 1: Explicit Finite Difference Equations

where

Equation 2: Explicit Finite Difference Parameters

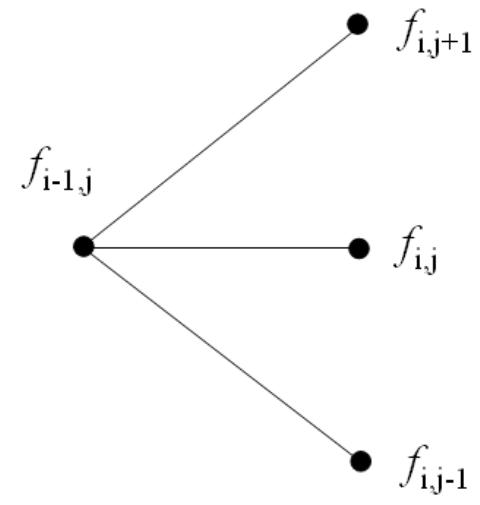
To see why this is called the explicit finite difference scheme consider the following diagram,

Figure 1: Explicit Finite Difference Viewed as a Trinomial Tree

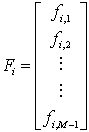
[Figure 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#explicittritree) is a pictorial representation of [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#expliciteqns). They show that given values for *ƒi, j+1*, *ƒi, j* and *ƒi, j-1* then values for *ƒi-1, j* can explicitly (and easily) be calculated.

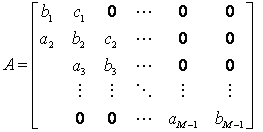
In the option pricing framework, [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#expliciteqns) (and hence [Figure 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#explicittritree)) shows that given the value of the option at the [boundary conditions](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMPayoff) (and most noticably at expiry) then all interior points of the [pricing grid](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid) can be calculated by using a *backwards induction*approach to step backwards through time. That is, given the option payoff at expiry nodes then the prices *δt* before expiry can be calulated, then from those prices the value *2δt* before expiry can be calculated, and working iteratively backwards through time until the option price at grid nodes for *t=0* (i.e. today) can be calulated.

**A Matrix Formulation**

The formulation for the explicit method given in [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#expliciteqns) may be written in the matrix notationExplicit Matrix Eqn

Equation 3: Explicit Finite Difference in Matrix Form

where

and

This matrix notation is used in the [Explicit Method - A MATLAB Implementation](http://www.goddardconsulting.ca/matlab-finite-diff-explicit.html) tutorial.

**Stability and Convergence**

Two important questions to ask about any numerical algorithm are *when is it stable?* and if it's stable then *how fast does it converge?* (An iterative algorithm that is unstable will lead to the calculation of ever increasing numbers that will at some point approach infinity. On the other hand, a stable algorithm will converge to a finite solution. Typically the faster that finite solution is reached the better the algorithm.

From standard results in matrix algebra it is known that a matrix equation of the form given in [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#ExplicitMatrixFormulation) is stable if and only ifExplicit Stability

Equation 4: Explicit Finite Difference Stability Condition

Equation 4 shows the *infinity norm* of the matrix *A*. Heuristically, if the infinity norm of *A*is less than 1 then successive values of *Fi* in [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#ExplicitMatrixFormulation) get smaller and smaller, and hence the algorithm converges, or is stable. (Alternatively if the infinity norm of *A* is greater than 1 then successive values of *Fi* get larger and larger and hence diverge.)

It can be shown that for certain combinations of *ρ*, *σ* and *δt*, (and hence values for *aj*, *bj*and *cj*) the infinity norm of A will be greater than 1. Hence unless the grid size (particularly in the time axis) is chosen appropriately the explicit finite difference method can be unstable, and hence useful for option pricing. (Compare this with both the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) and the [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html) which are both guaranteed to be stable.)

The rate of convergence of the algorithm is directly related to the truncation error introduced when approximating the [partial derivatives](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#ExplicitDiscretization). Hence the explicit method converges at the rates of *Ο(δt)* and *Ο(δS2)*. This is the same convergence rate as the[implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html), but slower than the [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html).

**Pricing American Style Options**

The backwards induction technique used to step the explicit method backwards through time is ideally suited to pricing options that include the possibility of early exercise. At each node, rather than use the value calculated from [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#expliciteqns) (or [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html#ExplicitMatrixFormulation)) that value is compared to the intrinsic value and the maximum of the two if used, i.e.

*ƒi, j* = max(Calculated Value, Intrinsic Value)

**Option Pricing Using The Implicit Finite Difference Method**

This tutorial discusses the specifics of the implicit finite difference method as it is applied to option pricing. Example code implementing the implicit method in MATLAB and used to price a simple option is given in the [Implicit Method - A MATLAB Implementation](http://www.goddardconsulting.ca/matlab-finite-diff-implicit.html)tutorial.

The [Finite Difference Methods](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html) tutorial covers general mathematical concepts behind finite diffence methods and should be read before this tutorial. Alternative finite difference methods, namely the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html) and the [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html), are covered in companion tutorials.

**Discretizing the Black-Scholes-Merton PDE**

For the implicit method the Black-Scholes-Merton partial differential equation,Black-Scholes-Merton PDE

is discretized using the following formulae

* use a [forward approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#forwardapprox) for *∂ƒ/∂t*  (Compare this with the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html)where the [backward approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) is used.)Backwards Approx.
* use a [central approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#centralapprox) for *∂ƒ/∂S*

Central Approx.

* use a [standard approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#secondderivapprox) for *∂2ƒ/∂S2*

Second Order Approx.

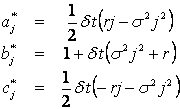
where the indices *i* and *j* represent nodes on the [pricing grid](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid).

Substituting these approximations into the PDE gives,Implicit Formulae Full

which reduces to

Implicit Formulae Reduced

Equation 1: Implicit Finite Difference Equations

where

Equation 2: Implicit Finite Difference Parameters

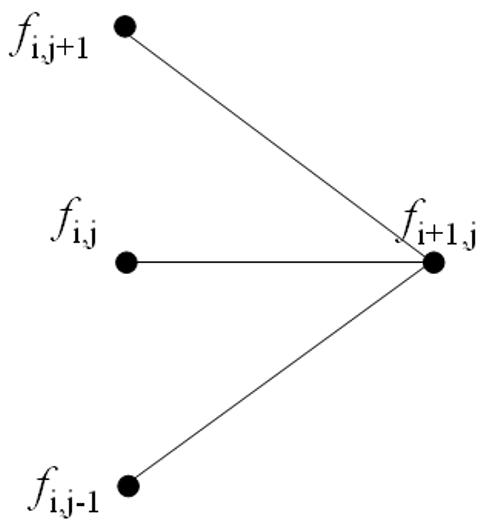
Since the equations are solved working backwards in time, superficially [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#impliciteqns) says that three unknowns must be calculated from only one known value. This is shown pictorially in the following diagram,

Figure 1: Implicit Finite Difference Viewed as a Pseudo-Trinomial Tree

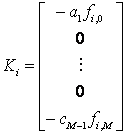
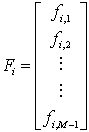
However, when all the equations at a given time point are written simulataneously there are *M-1* equations in *M-1* unknowns. Hence the value for *ƒ* at each node can be calculated uniquely. (This can arguably be seen easier using the matrix notation shown in the following [A Matrix Formulation](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#ImplicitMatrixFormulation) section.)

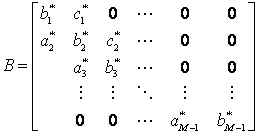
In the option pricing framework, given the option payoff at expiry nodes then the prices *Δt* before expiry can be calulated, then from those prices the value *2Δt* before expiry can be calculated, and working iteratively backwards through time until the option price at grid nodes for *t=0* (i.e. today) can be calulated.

**A Matrix Formulation**

The formulation for the explicit method given in [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#impliciteqns) may be written in the matrix notationImplicit Matrix Eqn

Equation 3: Implicit Finite Difference in Matrix Form

where

and

This matrix notation is used in the [Implicit Method - A MATLAB Implementation](http://www.goddardconsulting.ca/matlab-finite-diff-implicit.html) tutorial.

**Stability and Convergence**

Two important questions to ask about any numerical algorithm are *when is it stable?* and if it's stable then *how fast does it converge?* (An iterative algorithm that is unstable will lead to the calculation of ever increasing numbers that will at some point approach infinity. On the other hand, a stable algorithm will converge to a finite solution. Typically the faster that finite solution is reached the better the algorithm.

From standard results in matrix algebra it is known that a matrix equation of the form given in [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#ImplicitMatrixFormulation) is stable if and only ifImplicit Stability

Equation 4: Implicit Finite Difference Stability Condition

Equation 4 shows the *infinity norm* of the matrix *B -1*. Heuristically, if the infinity norm of *B -1* is less than 1 then successive values of *Fi* in [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#ImplicitMatrixFormulation) get smaller and smaller, and hence the algorithm converges, or is stable. (Alternatively if the infinity norm of *B -1*is greater than 1 then successive values of *Fi* get larger and larger and hence diverge.)

It can be shown that the infinity norm of *B -1* is less than 1 for all values of *ρ*, *σ* and *δt*. Hence the implicit finite difference method is always stable. (Compare this with the[explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html) which can be unstable if *δt* is chosen incorrectly, and the [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html) which is also guaranteed to be stable.)

The rate of convergence of the algorithm is directly related to the truncation error introduced when approximating the [partial derivatives](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#ImplicitDiscretization). Hence the implicit method converges at the rates of *Ο(δt)* and *Ο(δS2)*. This is the same convergence rate as the[explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html), but slower than the [Crank-Nicolson method](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html).

A disadvantage of the implicit method is that it requires the inverse of a matrix (i.e. *B -1*) to be calculated, and the inverse of a matrix is (computationally) an expense operation to perform. Fortunately, for tri-diagonal matrices such as *B* (i.e. matrices that only have non-zero elements down their diagonal and the terms directly above and below their diagonal) fast inversion algorithms are available.

**Pricing American Style Options**

When pricing options that include the possibility of early exercise special care must be taken when solving [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#ImplicitMatrixFormulation) for *Fi*. Taking the inverse of *B* to calculate a value for *Fi*then comparing the calculated values to the intrinsic value of the option and taking the larger value (on an element by element basis) results in incorrect option values. This is because the modified *Fi* will no longer satisfy [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html#ImplicitMatrixFormulation) as correct values must do.

However an iterative approach to calculating the inverse (such as the Guass-Seidel matrix inversion method) may be used successfully.

**Option Pricing Using The Crank-Nicolson Finite Difference Method**

This tutorial discusses the specifics of the Crank-Nicolson finite difference method as it is applied to option pricing. Example code implementing the Crank-Nicolson method in MATLAB and used to price a simple option is given in the [Crank-Nicolson Method - A MATLAB Implementation](http://www.goddardconsulting.ca/matlab-finite-diff-crank-nicolson.html) tutorial.

The [Finite Difference Methods](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html) tutorial covers general mathematical concepts behind finite diffence methods and should be read before this tutorial. Alternative finite difference methods, namely the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) and the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html), are covered in companion tutorials.

**Discretizing the Black-Scholes-Merton PDE**

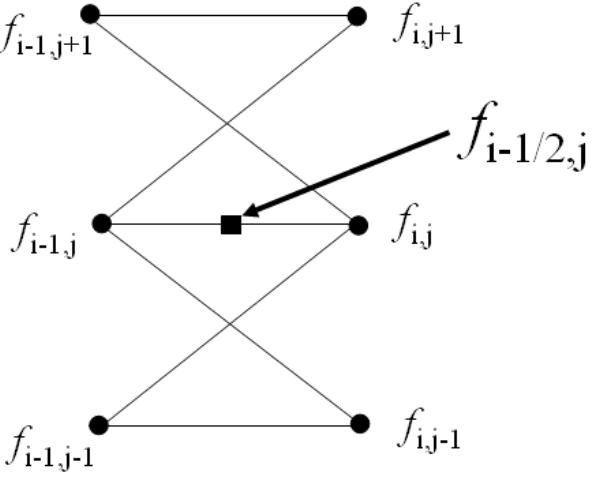
The Crank-Nicolson finite difference method represents an average of the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) and the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html). Consider the grid of points shown in [Figure 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CRDiscretization). This represent a small portion of the general [pricing grid](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid) used in finite difference methods. Indices *i* and *j* represent nodes on the [pricing grid](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid).

Figure 1: Grid of Price Points for the Crank-Nicolson Method

Whereas the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html) prices the node *ƒi-1, j* (the central node in the left hand side) based on the values of *ƒi, j+1*, *ƒi, j* and *ƒi, j-1* (the nodes on the right hand side), and the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) nominally prices the nodes *ƒi-1, j+1*, *ƒi-1, j* and *ƒi-1, j-1* (the three nodes on the left hand side) based on the value of *ƒi, j* (the central node in the right hand side), the Crank-Nicolson method prices all three of the left side nodes based on the values of all three of the right side nodes. Both the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) and the Crank-Nicolson method achieve this by solving a set of *M-1* simultaneous equations.

To derive the Crank-Nicolson difference equations consider the node *ƒi-1/2, j* which lies at the centre of [Figure 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CRDiscretization). (Note that a price will not be calculated for this node. Rather it is used as a mathematical convenience that will not appear in the final equations.)

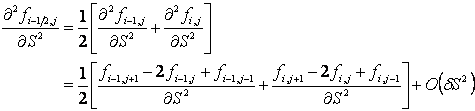
The goal is to discretize the Black-Scholes-Merton partial differential equation,Black-Scholes-Merton PDE

To achieve this (at node *ƒi-1/2, j*) the following approximations are used.

* use a [central approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#centralapprox) for *∂ƒ/∂t*  (Compare this with the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html)where the [backward approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) is used, and the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) where the[backward approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#backwardapprox) is used)Central Approx.
* use a [central approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#centralapprox) for *∂ƒ/∂S*



* use a [standard approximation](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#secondderivapprox) for *∂2ƒ/∂S2*

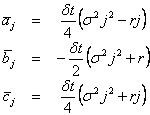


where the indices *i* and *j* represent nodes on the [pricing grid](http://www.goddardconsulting.ca/option-pricing-finite-diff-index.html#FEMGrid).

Substituting these approximations into the Black-Scholes-Merton PDE and collecting like terms this reduces to

Crank-Nicolson Formulae Reduced

Equation 1: Crank-Nicolson Finite Difference Equations

where

Equation 2: Crank-Nicolson Finite Difference Parameters

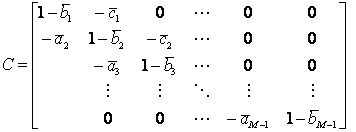
When [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#cneqns) is written for all values of *i* and *j* it leads to a set of *M-1* equations in *M-1* unknowns. Hence the value for *ƒ* at each node can be calculated uniquely by solving this set of simultaneous equations. (This can arguably be seen easier using the matrix notation shown in the following [A Matrix Formulation](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CNMatrixFormulation) section.)

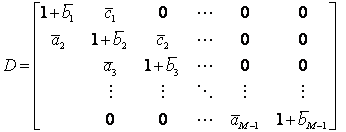
In the option pricing framework, given the option payoff at expiry nodes then the prices *Δt* before expiry can be calulated, then from those prices the value *2Δt* before expiry can be calculated, and working iteratively backwards through time until the option price at grid nodes for *t=0* (i.e. today) can be calulated.

**A Matrix Formulation**

The formulation for the Crank-Nicolson method given in [Equation 1](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#cneqns) may be written in the matrix notationCrank-Nicolson Matrix Eqn

Equation 3: Crank-Nicolson Finite Difference in Matrix Form

where

and

This matrix notation is used in the [Crank-Nicolson Method - A MATLAB Implementation](http://www.goddardconsulting.ca/matlab-finite-diff-crank-nicolson.html)tutorial.

**Stability and Convergence**

Two important questions to ask about any numerical algorithm are *when is it stable?* and if it's stable then *how fast does it converge?* (An iterative algorithm that is unstable will lead to the calculation of ever increasing numbers that will at some point approach infinity. On the other hand, a stable algorithm will converge to a finite solution. Typically the faster that finite solution is reached the better the algorithm.

From standard results in matrix algebra it is known that a matrix equation of the form given in [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CNMatrixFormulation) is stable if and only ifCrank-Nicolson Stability

Equation 4: Crank-Nicolson Finite Difference Stability Condition

Equation 4 shows the *infinity norm* of the product of the matrices *C -1D*. Heuristically, if the infinity norm of *C -1D* is less than 1 then successive values of *Fi* in [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CNMatrixFormulation) get smaller and smaller, and hence the algorithm converges, or is stable. (Alternatively if the infinity norm of *C -1D* is greater than 1 then successive values of *Fi* get larger and larger and hence diverge.)

It can be shown that the infinity norm of *C -1D* is less than 1 for all values of *ρ*, *σ* and *δt*. Hence the Crank-Nicolson finite difference method is always stable. (Compare this with the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html) which can be unstable if *δt* is chosen incorrectly, and the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html) which is also guaranteed to be stable.)

The rate of convergence of the algorithm is directly related to the truncation error introduced when approximating the [partial derivatives](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CRDiscretization). Hence the Crank-Nicolson method converges at the rates of *Ο(δt2)* and *Ο(δS2)*. This is a faster rate of convergence than either the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html), or the [implicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-implicit.html).

A disadvantage of the Crank-Nicolson method (over the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html)) is that it requires the inverse of a matrix (i.e. *C -1*) to be calculated, and the inverse of a matrix is (computationally) an expense operation to perform. Fortunately, for tri-diagonal matrices such as *C* (i.e. matrices that only have non-zero elements down their diagonal and the terms directly above and below their diagonal) fast inversion algorithms are available.

Another issue with the Crank-Nicolson method is that it is known to be sensitive to non-smooth boundary conditions. This is particularly the case when calculating *the Greeks*. A modified approach, using the [explicit method](http://www.goddardconsulting.ca/option-pricing-finite-diff-explicit.html) for the first few time steps, then reverting to the Crank-Nicolson method for the remaining steps often overcomes this problem.

**Pricing American Style Options**

When pricing options that include the possibility of early exercise special care must be taken when solving [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CNMatrixFormulation) for *Fi*. Taking the inverse of *C* to calculate a value for *Fi*then comparing the calculated values to the intrinsic value of the option and taking the larger value (on an element by element basis) results in incorrect option values. This is because the modified *Fi* will no longer satisfy [Equation 3](http://www.goddardconsulting.ca/option-pricing-finite-diff-crank-nicolson.html#CNMatrixFormulation) as correct values must do.

However an iterative approach to calculating the inverse (such as the Guass-Seidel matrix inversion method) may be used successfully.