




WIKIPEDIA
The Free Encyclopedia

Main page
Contents
Featured content
Current events
Random article
Donate to Wikipedia
Wikipedia store

Interaction
Help
About Wikipedia
Community portal
Recent changes
Contact page

Tools
What links here
Related changes
Upload file
Special pages
Permanent link
Page information
Wikidata item
Cite this page

Print/export
Create a book
Download as PDF
Printable version

Languages 
Deutsch
Español
فارسی
Français
Polski
Português
Українська
中文

 Edit links

[Create account](#) [Log in](#)

Article [Talk](#)

[Read](#) [Edit](#) [More](#)

Crank–Nicolson method

From Wikipedia, the free encyclopedia
(Redirected from [Crank-Nicolson method](#))

In **numerical analysis**, the **Crank–Nicolson method** is a [finite difference method](#) used for numerically solving the [heat equation](#) and similar [partial differential equations](#).^[1] It is a [second-order](#) method in time, it is [implicit](#) in time and can be written as an [implicit Runge–Kutta method](#), and it is [numerically stable](#). The method was developed by [John Crank](#) and [Phyllis Nicolson](#) in the mid 20th century.^[2]

For diffusion equations (and many other equations), it can be shown the Crank–Nicolson method is unconditionally [stable](#).^[3] However, the approximate solutions can still contain (decaying) spurious oscillations if the ratio of time step Δt times the [thermal diffusivity](#) to the square of space step, Δx^2 , is large (typically larger than 1/2 per [Von Neumann stability analysis](#)). For this reason, whenever large time steps or high spatial resolution is necessary, the less accurate [backward Euler method](#) is often used, which is both stable and immune to oscillations.

Contents

[\[hide\]](#)

- The method
- Example: 1D diffusion
- Example: 1D diffusion with advection for steady flow, with multiple channel connections
- Example: 2D diffusion
- Application in financial mathematics
- See also
- References
- External links

The method [\[edit\]](#)

The Crank–Nicolson method is based on the [trapezoidal rule](#), giving second-order convergence in time. For example, in one dimension, if the [partial differential equation](#) is

$$\frac{\partial u}{\partial t} = F\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right)$$

then, letting $u(i\Delta x, n\Delta t) = u_i^n$, the equation for Crank–Nicolson method is a combination of the [forward Euler method](#) at n and the [backward Euler method](#) at $n + 1$ (note, however, that the method itself is *not* simply the average of those two methods, as the equation has an implicit dependence on the solution):

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = F_i^n\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right) \quad (\text{forward Euler})$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = F_i^{n+1}\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right) \quad (\text{backward Euler})$$

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{1}{2} \left[F_i^{n+1}\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right) + F_i^n\left(u, x, t, \frac{\partial u}{\partial x}, \frac{\partial^2 u}{\partial x^2}\right) \right] \quad (\text{Crank–Nicolson}).$$

Note that this is an *implicit method*: to get the "next" value of u in time, a system of algebraic equations must be solved. If the partial differential equation is nonlinear, the [discretization](#) will also be nonlinear so that advancing in time will involve the solution of a system of nonlinear algebraic equations, though linearizations are possible. In many problems, especially linear diffusion, the algebraic problem is [tridiagonal](#) and may be efficiently solved with the [tridiagonal matrix algorithm](#), which gives a fast $\mathcal{O}(n)$ direct solution as opposed to the usual $\mathcal{O}(n^3)$ for a full matrix.

Example: 1D diffusion [\[edit\]](#)

The Crank–Nicolson method is often applied to [diffusion problems](#). As an example, for linear diffusion,

$$\frac{\partial u}{\partial t} = a \frac{\partial^2 u}{\partial x^2}$$

applying a [finite difference](#) spatial discretization for the right hand side, the Crank–Nicolson discretization is then :

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \frac{a}{2(\Delta x)^2} \left((u_{i+1}^{n+1} - 2u_i^{n+1} + u_{i-1}^{n+1}) + (u_{i+1}^n - 2u_i^n + u_{i-1}^n) \right)$$

or, letting $r = \frac{a\Delta t}{(\Delta x)^2}$:

$$-ru_{i+1}^{n+1} + 2(1+r)u_i^{n+1} - ru_{i-1}^{n+1} = ru_{i+1}^n + 2(1-r)u_i^n + ru_{i-1}^n$$

which is a [tridiagonal](#) problem, so that u_i^{n+1} may be efficiently solved by using the [tridiagonal matrix algorithm](#) in favor of a much more costly [matrix inversion](#).

A quasilinear equation, such as (this is a minimalistic example and not general)

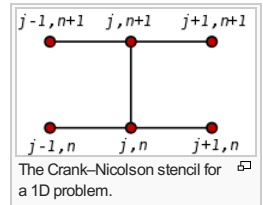
$$\frac{\partial u}{\partial t} = a(u) \frac{\partial^2 u}{\partial x^2}$$

would lead to a nonlinear system of algebraic equations which could not be easily solved as above; however, it is possible in some cases to linearize the problem by using the old value for a , that is $a_i^n(u)$ instead of $a_i^{n+1}(u)$. Other times, it may be possible to estimate $a_i^{n+1}(u)$ using an explicit method and maintain stability.

Example: 1D diffusion with advection for steady flow, with multiple channel connections [\[edit\]](#)

This is a solution usually employed for many purposes when there is a contamination problem in streams or rivers under steady flow conditions but information is given in one dimension only. Often the problem can be simplified into a 1-dimensional problem and still yield useful information.

Here we model the concentration of a solute contaminant in water. This problem is composed of three parts: the known diffusion equation (D_x chosen as constant), an advective component (which means the system is evolving in space due to a velocity field), which we choose to be a constant U_x , and a lateral interaction between longitudinal channels (k).



$$\frac{\partial C}{\partial t} = D_x \frac{\partial^2 C}{\partial x^2} - U_x \frac{\partial C}{\partial x} - k(C - C_N) - k(C - C_M) \quad (1)$$

where C is the concentration of the contaminant and subscripts N and M correspond to *previous* and *next* channel.

The Crank–Nicolson method (where i represents position and j time) transforms each component of the PDE into the following:

$$\frac{\partial C}{\partial t} \Rightarrow \frac{C_i^{j+1} - C_i^j}{\Delta t} \quad (2)$$

$$\frac{\partial^2 C}{\partial x^2} \Rightarrow \frac{1}{2(\Delta x)^2} \left((C_{i+1}^{j+1} - 2C_i^{j+1} + C_{i-1}^{j+1}) + (C_{i+1}^j - 2C_i^j + C_{i-1}^j) \right) \quad (3)$$

$$\frac{\partial C}{\partial x} \Rightarrow \frac{1}{2} \left(\frac{(C_{i+1}^{j+1} - C_{i-1}^{j+1})}{2(\Delta x)} + \frac{(C_{i+1}^j - C_{i-1}^j)}{2(\Delta x)} \right) \quad (4)$$

$$C \Rightarrow \frac{1}{2}(C_i^{j+1} + C_i^j) \quad (5)$$

$$C_N \Rightarrow \frac{1}{2}(C_{Ni}^{j+1} + C_{Ni}^j) \quad (6)$$

$$C_M \Rightarrow \frac{1}{2}(C_{Mi}^{j+1} + C_{Mi}^j). \quad (7)$$

Now we create the following constants to simplify the algebra:

$$\lambda = \frac{D_x \Delta t}{2\Delta x^2}$$

$$\alpha = \frac{U_x \Delta t}{4\Delta x}$$

$$\beta = \frac{k\Delta t}{2}$$

and substitute (2), (3), (4), (5), (6), (7), α , β and λ into (1). We then put the *new time* terms on the left ($j+1$) and the *present time* terms on the right (j) to get:

$$-\beta C_{Ni}^{j+1} - (\lambda + \alpha) C_{i-1}^{j+1} + (1 + 2\lambda + 2\beta) C_i^{j+1} - (\lambda - \alpha) C_{i+1}^{j+1} - \beta C_{Mi}^{j+1} = \beta C_{Ni}^j + (\lambda + \alpha) C_{i-1}^j + (1 - 2\lambda - 2\beta) C_i^j + (\lambda - \alpha) C_{i+1}^j + \beta C_{Mi}^j.$$

To model the *first* channel, we realize that it can only be in contact with the following channel (M), so the expression is simplified to:

$$-(\lambda + \alpha) C_{i-1}^{j+1} + (1 + 2\lambda + \beta) C_i^{j+1} - (\lambda - \alpha) C_{i+1}^{j+1} - \beta C_{Mi}^{j+1} = +(\lambda + \alpha) C_{i-1}^j + (1 - 2\lambda - \beta) C_i^j + (\lambda - \alpha) C_{i+1}^j + \beta C_{Mi}^j.$$

In the same way, to model the *last* channel, we realize that it can only be in contact with the previous channel (N), so the expression is simplified to:

$$-\beta C_{Ni}^{j+1} - (\lambda + \alpha) C_{i-1}^{j+1} + (1 + 2\lambda + \beta) C_i^{j+1} - (\lambda - \alpha) C_{i+1}^{j+1} = \beta C_{Ni}^j + (\lambda + \alpha) C_{i-1}^j + (1 - 2\lambda - \beta) C_i^j + (\lambda - \alpha) C_{i+1}^j.$$

To solve this linear system of equations we must now see that boundary conditions must be given first to the beginning of the channels:

C_0^j : initial condition for the channel at present time step

C_0^{j+1} : initial condition for the channel at next time step

C_{N0}^j : initial condition for the previous channel to the one analyzed at present time step

C_{M0}^j : initial condition for the next channel to the one analyzed at present time step.

For the last cell of the channels (z) the most convenient condition becomes an adiabatic one, so

$$\frac{\partial C}{\partial x} \Big|_{x=z} = \frac{(C_{i+1} - C_{i-1})}{2\Delta x} = 0.$$

This condition is satisfied if and only if (regardless of a null value)

$$C_{i+1}^{j+1} = C_{i-1}^{j+1}.$$

Let us solve this problem (in a matrix form) for the case of 3 channels and 5 nodes (including the initial boundary condition). We express this as a linear system problem:

$$[AA] [C^{j+1}] = [BB][C^j] + [d]$$

where

$$\mathbf{C}^{j+1} = \begin{bmatrix} C_{11}^{j+1} \\ C_{12}^{j+1} \\ C_{13}^{j+1} \\ C_{14}^{j+1} \\ C_{21}^{j+1} \\ C_{22}^{j+1} \\ C_{23}^{j+1} \\ C_{24}^{j+1} \\ C_{31}^{j+1} \\ C_{32}^{j+1} \\ C_{33}^{j+1} \\ C_{34}^{j+1} \end{bmatrix} \quad \text{and} \quad \mathbf{C}^j = \begin{bmatrix} C_{11}^j \\ C_{12}^j \\ C_{13}^j \\ C_{14}^j \\ C_{21}^j \\ C_{22}^j \\ C_{23}^j \\ C_{24}^j \\ C_{31}^j \\ C_{32}^j \\ C_{33}^j \\ C_{34}^j \end{bmatrix}.$$

Now we must realize that AA and BB should be arrays made of four different subarrays (remember that only three channels are considered for this example but it covers the main part discussed above).

$$\mathbf{AA} = \begin{bmatrix} AA1 & AA3 & 0 \\ AA3 & AA2 & AA3 \\ 0 & AA3 & AA1 \end{bmatrix} \quad \text{and}$$

$$\mathbf{BB} = \begin{bmatrix} BB1 & -AA3 & 0 \\ -AA3 & BB2 & -AA3 \\ 0 & -AA3 & BB1 \end{bmatrix}$$

where the elements mentioned above correspond to the next arrays and an additional 4x4 full of zeros. Please note that the sizes of AA and BB are 12x12:

$$\begin{aligned}
\mathbf{AA1} &= \begin{bmatrix} (1+2\lambda+\beta) & -(\lambda-\alpha) & 0 & 0 \\ -(\lambda+\alpha) & (1+2\lambda+\beta) & -(\lambda-\alpha) & 0 \\ 0 & -(\lambda+\alpha) & (1+2\lambda+\beta) & -(\lambda-\alpha) \\ 0 & 0 & -2\lambda & (1+2\lambda+\beta) \end{bmatrix}, \\
\mathbf{AA2} &= \begin{bmatrix} (1+2\lambda+2\beta) & -(\lambda-\alpha) & 0 & 0 \\ -(\lambda+\alpha) & (1+2\lambda+2\beta) & -(\lambda-\alpha) & 0 \\ 0 & -(\lambda+\alpha) & (1+2\lambda+2\beta) & -(\lambda-\alpha) \\ 0 & 0 & -2\lambda & (1+2\lambda+2\beta) \end{bmatrix}, \\
\mathbf{AA3} &= \begin{bmatrix} -\beta & 0 & 0 & 0 \\ 0 & -\beta & 0 & 0 \\ 0 & 0 & -\beta & 0 \\ 0 & 0 & 0 & -\beta \end{bmatrix}, \\
\mathbf{BB1} &= \begin{bmatrix} (1-2\lambda-\beta) & (\lambda-\alpha) & 0 & 0 \\ (\lambda+\alpha) & (1-2\lambda-\beta) & (\lambda-\alpha) & 0 \\ 0 & (\lambda+\alpha) & (1-2\lambda-\beta) & (\lambda-\alpha) \\ 0 & 0 & 2\lambda & (1-2\lambda-\beta) \end{bmatrix} \& \\
\mathbf{BB2} &= \begin{bmatrix} (1-2\lambda-2\beta) & (\lambda-\alpha) & 0 & 0 \\ (\lambda+\alpha) & (1-2\lambda-2\beta) & (\lambda-\alpha) & 0 \\ 0 & (\lambda+\alpha) & (1-2\lambda-2\beta) & (\lambda-\alpha) \\ 0 & 0 & 2\lambda & (1-2\lambda-2\beta) \end{bmatrix}.
\end{aligned}$$

The \mathbf{d} vector here is used to hold the boundary conditions. In this example it is a 12x1 vector:

$$\mathbf{d} = \begin{bmatrix} (\lambda+\alpha)(C_{10}^{j+1} + C_{10}^j) \\ 0 \\ 0 \\ 0 \\ (\lambda+\alpha)(C_{20}^{j+1} + C_{20}^j) \\ 0 \\ 0 \\ 0 \\ (\lambda+\alpha)(C_{30}^{j+1} + C_{30}^j) \\ 0 \\ 0 \\ 0 \end{bmatrix}.$$

To find the concentration at any time, one must iterate the following equation:

$$[C^{j+1}] = [AA^{-1}][[BB][C^j] + [\mathbf{d}]].$$

Example: 2D diffusion [\[edit\]](#)

When extending into two dimensions on a uniform [Cartesian grid](#), the derivation is similar and the results may lead to a system of [band-diagonal](#) equations rather than [tridiagonal](#) ones. The two-dimensional heat equation

$$\frac{\partial u}{\partial t} = a \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} \right)$$

can be solved with the Crank–Nicolson discretization of

$$\begin{aligned}
u_{i,j}^{n+1} = u_{i,j}^n + \frac{1}{2} \frac{a\Delta t}{(\Delta x)^2} & [(u_{i+1,j}^{n+1} + u_{i-1,j}^{n+1} + u_{i,j+1}^{n+1} + u_{i,j-1}^{n+1} - 4u_{i,j}^{n+1}) \\
& + (u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n - 4u_{i,j}^n)]
\end{aligned}$$

assuming that a square grid is used so that $\Delta x = \Delta y$. This equation can be simplified somewhat by rearranging terms and using the [CFL number](#)

$$\mu = \frac{a\Delta t}{(\Delta x)^2}.$$

For the Crank–Nicolson numerical scheme, a low [CFL number](#) is not required for stability, however it is required for numerical accuracy. We can now write the scheme as:

$$\begin{aligned}
(1+2\mu)u_{i,j}^{n+1} - \frac{\mu}{2} (u_{i+1,j}^{n+1} + u_{i-1,j}^{n+1} + u_{i,j+1}^{n+1} + u_{i,j-1}^{n+1}) \\
= (1-2\mu)u_{i,j}^n + \frac{\mu}{2} (u_{i+1,j}^n + u_{i-1,j}^n + u_{i,j+1}^n + u_{i,j-1}^n).
\end{aligned}$$

Application in financial mathematics [\[edit\]](#)

Further information: [Finite difference methods for option pricing](#)

Because a number of other phenomena can be [modeled](#) with the [heat equation](#) (often called the diffusion equation in [financial mathematics](#)), the Crank–Nicolson method has been applied to those areas as well.^[4] Particularly, the [Black–Scholes](#) option pricing model's [differential equation](#) can be transformed into the heat equation, and thus [numerical solutions](#) for [option pricing](#) can be obtained with the Crank–Nicolson method.

The importance of this for finance, is that option pricing problems, when extended beyond the standard assumptions (e.g. incorporating changing dividends), cannot be solved in closed form, but can be solved using this method. Note however, that for non-smooth final conditions (which happen for most financial instruments), the Crank–Nicolson method is not satisfactory as numerical oscillations are not damped. For [vanilla options](#), this results in oscillation in the [gamma value](#) around the [strike price](#). Therefore, special damping initialization steps are necessary (e.g., fully implicit finite difference method).

See also [\[edit\]](#)

- [Financial mathematics](#)
- [Trapezoidal rule \(differential equations\)](#)

[edit]

- ET.

[edit]

-

[show][show]

F

43.

Text is available under the [Creative Commons Attribution-ShareAlike 4.0 International license](#), a trademark of the [Wikimedia Foundation, Inc.](#), a non-profit organization.

D

