WIKIPEDIA

Finite difference method

In numerical analysis, **finite-difference methods** (**FDM**) are discretizations used for solving differential equations by approximating them with difference equations that finite differences approximate the derivatives.

FDMs convert a linear ordinary differential equations (ODE) or non-linear partial differential equations (PDE) into a system of equations that can be solved by matrix algebra techniques. The reduction of the differential equation to a system of algebraic equations makes the problem of finding the solution to a given ODE/PDE ideally suited to modern computers, hence the widespread use of FDMs in modern numerical analysis[1]. Today, FDMs are the dominant approach to numerical solutions of PDEs.[1]

Contents

Derivation from Taylor's polynomial

First, assuming the function whose derivatives are to be approximated is properly-behaved, by Taylor's theorem, we can create a Taylor series expansion

$$f(x_0+h)=f(x_0)+\frac{f'(x_0)}{1!}h+\frac{f^{(2)}(x_0)}{2!}h^2+\cdots+\frac{f^{(n)}(x_0)}{n!}h^n+R_n(x),$$

where $n!$ denotes the factorial of n, and $R_n(x)$ is a remainder term, denoting the difference between the Taylor polynomial of degree n and the original function. We will derive an approximation for the first derivative of the function "f" by first truncating the Taylor polynomial:

$$f(x_0 + h) = f(x_0) + f'(x_0)h + R_1(x),$$

Setting, x_0=a we have,

$$f(a + h) = f(a) + f'(a)h + R_1(x),$$

Dividing across by h gives:

$$\frac{f(a+h)}{h} = \frac{f(a)}{h} + f'(a) + \frac{R_1(x)}{h}$$

Solving for f'(a):

$$f'(a) = \frac{f(a+h) - f(a)}{h} - \frac{R_1(x)}{h}$$

Assuming that $R_1(x)$ is sufficiently small, the approximation of the first derivative of "f" is:

$$f'(a) \approx \frac{f(a+h) - f(a)}{h}.$$

Accuracy and order

The error in a method's solution is defined as the difference between the approximation and the exact analytical solution. The two sources of error in finite difference methods are round-off error, the loss of precision due to computer rounding of decimal quantities, and truncation error or discretization error, the difference between the exact solution of the original differential equation and the exact quantity assuming perfect arithmetic (that is, assuming no round-off).

To use a finite difference method to approximate the solution to a problem, one must first discretize the problem's domain. This is usually done by dividing the domain into a uniform grid (see image to the right). This means that finite-difference methods produce sets of discrete numerical approximations to the derivative, often in a "time-stepping" manner.

An expression of general interest is the local truncation error of a method. Typically expressed using Big-O notation, local truncation error refers to the error from a single application of a method. That is, it is the quantity $f'(x_i) - f'_i$ if $f'(x_i)$ refers to the exact value and f'_i to the numerical approximation. The remainder term of a Taylor polynomial is convenient for analyzing the local truncation error. Using the Lagrange form of the remainder from the Taylor polynomial for $f(x_0 + h)$, which is

$$R_n(x_0 + h) = \frac{f^{(n+1)}(\xi)}{(n+1)!}(h)^{n+1}, \text{where } x_0 < \xi < x_0 + h,$$

the dominant term of the local truncation error can be discovered. For example, again using the forward-difference formula for the first derivative, knowing that $f(x_i) = f(x_0 + ih)$,

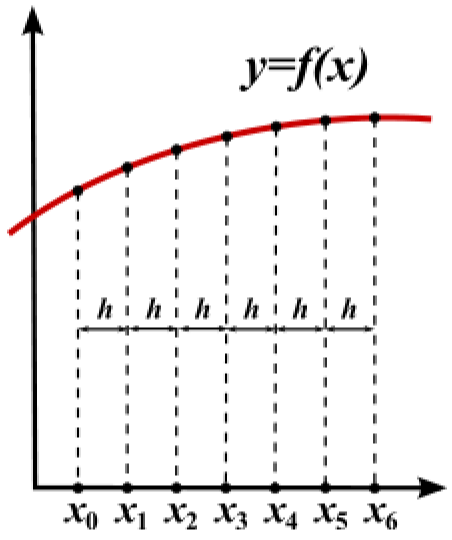
$$f(x_0+ih)=f(x_0)+f'(x_0)ih+\frac{f''(\xi)}{2!}(ih)^2,$$

and with some algebraic manipulation, this leads to

$$\frac{f(x_0+ih)-f(x_0)}{ih}=f'(x_0)+\frac{f''(\xi)}{2!}ih,$$

and further noting that the quantity on the left is the approximation from the finite difference method and that the quantity on the right is the exact quantity of interest plus a remainder, clearly that remainder is the local truncation error. A final expression of this example and its order is:

$$\frac{f(x_0+ih)-f(x_0)}{ih}=f'(x_0)+O(h).$$



The finite difference method relies on discretizing a function on a grid.

This means that, in this case, the local truncation error is proportional to the step sizes. The quality and duration of simulated FDM solution depends on the discretization equation selection and the step sizes (time and space steps). The data quality and simulation duration increase significantly with smaller step size.[2] Therefore, a reasonable balance between data quality and simulation duration is necessary for practical usage. Large time steps are useful for increasing simulation speed in practice. However, time steps which are too large may create instabilities and affect the data quality.[3][4]

The von Neumann and Courant-Friedrichs-Lewy criteria are often evaluated to determine the numerical model stability.[3][4][5][6]

Example: ordinary differential equation

For example, consider the ordinary differential equation

$$u'(x)=3u(x)+2.$$

The Euler method for solving this equation uses the finite difference quotient

$$\frac{u(x+h)-u(x)}{h}\approx u'(x)$$

to approximate the differential equation by first substituting it for u'(x) then applying a little algebra (multiplying both sides by h, and then adding u(x) to both sides) to get

$$u(x+h)=u(x)+h(3u(x)+2).$$

The last equation is a finite-difference equation, and solving this equation gives an approximate solution to the differential equation.

Example: The heat equation

Consider the normalized heat equation in one dimension, with homogeneous Dirichlet boundary conditions

$$U_t = U_{xx}$$
$$U(0,t) = U(1,t) = 0 \text{ (boundary condition)}$$
$$U(x,0) = U_0(x) \text{ (initial condition)}$$

One way to numerically solve this equation is to approximate all the derivatives by finite differences. We partition the domain in space using a mesh $x_0, \dots, x_J$ and in time using a mesh $t_0, \dots, t_N$. We assume a uniform partition both in space and in time, so the difference between two consecutive space points will be h and between two consecutive time points will be k. The points

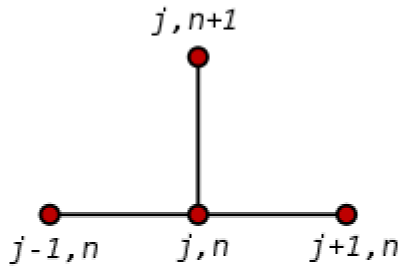
$$u(x_j, t_n) = u_j^n$$

will represent the numerical approximation of $u(x_j, t_n)$.

Explicit method

Using a forward difference **at time** t_n and a second-order central difference for the space derivative at position x_j (FTCS) we get the recurrence equation:

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{h^2}.$$



The stencil for the most common explicit method for the heat equation.

This is an explicit method for solving the one-dimensional heat equation.

We can obtain u_j^{n+1} from the other values this way:

$$u_j^{n+1} = (1-2r)u_j^n + ru_{j-1}^n + ru_{j+1}^n$$

where $r = k/h^2$.

So, with this recurrence relation, and knowing the values at time n, one can obtain the corresponding values at time n+1. u_0^n and u_J^n must be replaced by the boundary conditions, in this example they are both 0.

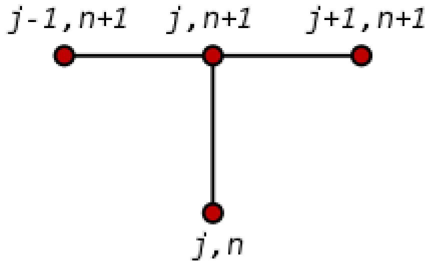
This explicit method is known to be numerically stable and convergent whenever $r \le 1/2$.[7] The numerical errors are proportional to the time step and the square of the space step:

$$\Delta u = O(k) + O(h^2)$$

Implicit method

If we use the backward difference **at time** t_{n+1} and a second-order central difference for the space derivative at position x_j (The Backward Time, Centered Space Method "BTCS") we get the recurrence equation:

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2}.$$



The implicit method stencil.

This is an implicit method for solving the one-dimensional heat equation.

We can obtain u_j^{n+1} from solving a system of linear equations:

$$(1+2r)u_j^{n+1} - ru_{j-1}^{n+1} - ru_{j+1}^{n+1} = u_j^n$$

The scheme is always numerically stable and convergent but usually more numerically intensive than the explicit method as it requires solving a system of numerical equations on each time step. The errors are linear over the time step and quadratic over the space step:

$$\Delta u = O(k) + O(h^2).$$

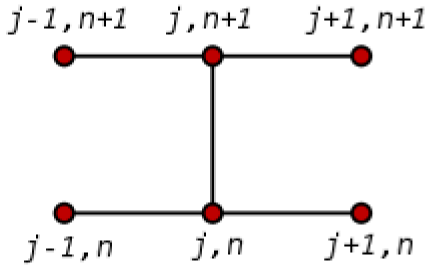
Crank–Nicolson method

Finally if we use the central difference at time $t_{n+1/2}$ and a second-order central difference for the space derivative at position x_j ("CTCS") we get the recurrence equation:

$$\frac{u_j^{n+1} - u_j^n}{k} = \frac{1}{2}\left(\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{h^2} + \frac{u_{j+1}^{n} - 2u_j^{n} + u_{j-1}^{n}}{h^2}\right).$$

This formula is known as the Crank–Nicolson method.

We can obtain u_j^{n+1} from solving a system of linear equations:



The Crank–Nicolson stencil.

$$(2+2r)u_j^{n+1} - ru_{j-1}^{n+1} - ru_{j+1}^{n+1} = (2-2r)u_j^n + ru_{j-1}^n + ru_{j+1}^n$$

The scheme is always numerically stable and convergent but usually more numerically intensive as it requires solving a system of numerical equations on each time step. The errors are quadratic over both the time step and the space step:

$$\Delta u = O(k^2) + O(h^2).$$

Usually the Crank–Nicolson scheme is the most accurate scheme for small time steps. The explicit scheme is the least accurate and can be unstable, but is also the easiest to implement and the least numerically intensive. The implicit scheme works the best for large time steps.

Comparison

The figures below present the solutions given by the above methods to approximate the heat equation

$$U_t = \alpha U_{xx}, \quad \alpha = \frac{1}{\pi^2},$$

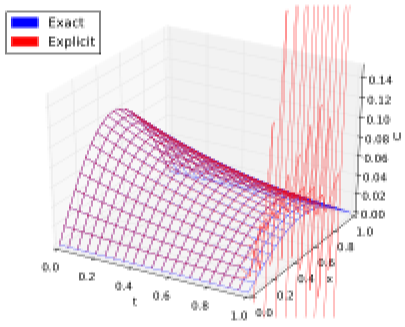
with the boundary condition

$$U(0,t) = U(1,t) = 0.$$

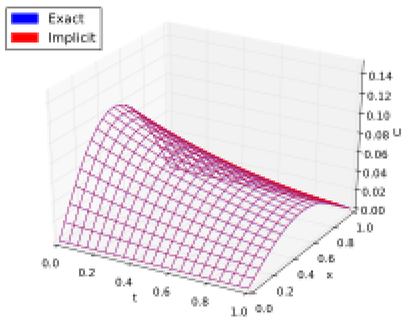
The exact solution is

$$U(x,t) = \frac{1}{\pi^2} e^{-t} \sin(\pi x).$$

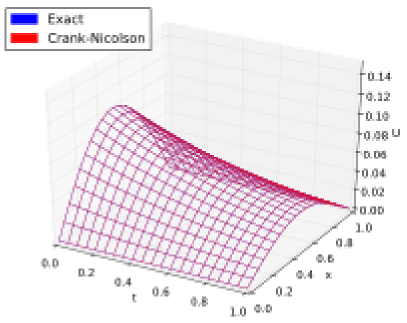
Comparison of Finite Difference Methods



Explicit method (*not* stable)



Implicit method (stable)



Crank-Nicolson method (stable)

Example: The Laplace operator

The (continuous) Laplace operator in n-dimensions is given by $\Delta u(x)=\sum_{i=1}^{n}\partial_i^2 u(x)$. The discrete Laplace operator $\Delta_h u$ depends on the dimension n.

In 1D the Laplace operator is approximated as

$$\Delta u(x)=u''(x)\approx\frac{u(x-h)-2u(x)+u(x+h)}{h^2}=:\Delta_h u(x)\,.$$

This approximation is usually expressed via the following stencil

$$\frac{1}{h^2}\begin{bmatrix}1 & -2 & 1\end{bmatrix}$$

and which represents a symmetric, tridiagonal matrix. For an equidistant grid one gets a Toeplitz matrix.

The 2D case shows all the characteristics of the more general nD case. Each second partial derivative needs to be approximated similar to the 1D case

$$\begin{aligned}\Delta u(x,y)&=u_{xx}(x,y)+u_{yy}(x,y)\\&\approx\frac{u(x-h,y)-2u(x,y)+u(x+h,y)}{h^2}+\frac{u(x,y-h)-2u(x,y)+u(x,y+h)}{h^2}\\&=\frac{u(x-h,y)+u(x+h,y)-4u(x,y)+u(x,y-h)+u(x,y+h)}{h^2}\\&=:\Delta_h u(x,y)\,,\end{aligned}$$

which is usually given by the following stencil

$$\frac{1}{h^2}\begin{bmatrix} & 1 & \\ 1 & -4 & 1\\ & 1 & \end{bmatrix}.$$

Consistency

Consistency of the above-mentioned approximation can be shown for highly regular functions, such as $u\in C^4(\Omega)$. The statement is

$$\Delta u-\Delta_h u=\mathcal{O}(h^2)\,.$$

To proof this one needs to substitute Taylor Series expansions up to order 3 into the discrete Laplace operator.

Properties

Subharmonic

Similar to continuous subharmonic functions one can define *subharmonic functions* for finite-difference approximations u_h

$$-\Delta_h u_h \leq 0\,.$$

Mean value

One can define a general stencil of *positive type* via

$$\begin{bmatrix} & \alpha_N & \\ \alpha_W & -\alpha_C & \alpha_E \\ & \alpha_S & \end{bmatrix}\,, \quad \alpha_i > 0\,, \quad \alpha_C = \sum_{i \in \{N,E,S,W\}} \alpha_i\,.$$

If u_h is (discrete) subharmonic then the following *mean value property* holds

$$u_h(x_C) \leq \frac{\sum_{i \in \{N,E,S,W\}} \alpha_i u_h(x_i)}{\sum_{i \in \{N,E,S,W\}} \alpha_i}\,,$$

where the approximation is evaluated on points of the grid, and the stencil is assumed to be of positive type.

A similar mean value property also holds for the continuous case.

Maximum principle

For a (discrete) subharmonic function u_h the following holds

$$\max_{\Omega_h} u_h \leq \max_{\partial\Omega_h} u_h\,,$$

where $\Omega_h, \partial\Omega_h$ are discretizations of the continuous domain Ω, respectively the boundary $\partial\Omega$.

A similar maximum principle also holds for the continuous case.

The SBP-SAT method

The SBP-SAT method is a stable and accurate technique for discretizing and imposing boundary conditions of a well-posed partial differential equation using high order finite differences.[8][9]The method is based on finite differences where the differentiation operators exhibit summation-by-parts properties. Typically, these operators consist of differentiation matrices with central difference stencils in the interior with carefully chosen one-sided boundary stencils designed to mimic integration-by-parts in the discrete setting. Using the SAT technique, the boundary conditions of the PDE are imposed weakly, where the boundary values are "pulled" towards the desired conditions rather than exactly fulfilled. If the tuning parameters (inherent to the SAT technique) are chosen properly, the resulting system of ODE's will exhibit similar energy behavior as the continuous PDE, i.e. the system has no non-physical energy growth. This guarantees stability if an integration scheme with a stability region that includes parts of the imaginary axis, such as the fourth order Runge-Kutta method, is used. This makes the SAT technique an attractive method of imposing boundary conditions for higher order finite difference methods, in contrast to for example the injection method, which typically will not be stable if high order differentiation operators are used.

See also

- Finite element method
- Finite difference
- Finite difference time domain
- Infinite difference method
- Stencil (numerical analysis)
- Finite difference coefficients
- Five-point stencil
- Lax–Richtmyer theorem
- Finite difference methods for option pricing
- Upwind differencing scheme for convection
- Central differencing scheme
- Discrete Poisson equation
- Discrete Laplace operator

References

1. Christian Grossmann; Hans-G. Roos; Martin Stynes (2007). *Numerical Treatment of Partial Differential Equations*. Springer Science & Business Media. p. 23. ISBN 978-3-540-71584-9.
2. Arieh Iserles (2008). *A first course in the numerical analysis of differential equations*. Cambridge University Press. p. 23. ISBN 9780521734905.
3. Hoffman JD; Frankel S (2001). *Numerical methods for engineers and scientists*. CRC Press, Boca Raton.
4. Jaluria Y; Atluri S (1994). "Computational heat transfer". *Computational Mechanics*. **14**: 385–386. doi:10.1007/BF00377593 (https://doi.org/10.1007%2FBF00377593).
5. Majumdar P (2005). *Computational methods for heat and mass transfer* (1st ed.). Taylor and Francis, New York.
6. Smith GD (1985). *Numerical solution of partial differential equations: finite difference methods* (3rd ed.). Oxford University Press.
7. Crank, J. *The Mathematics of Diffusion*. 2nd Edition, Oxford, 1975, p. 143.
8. Bo Strand (1994). *Summation by Parts for Finite Difference Approximations for d/dx*. *Journal of Computational Physics*. doi:10.1006/jcph.1994.1005 (https://doi.org/10.1006%2Fjcph.1994.1005).
9. Mark H. Carpenter; David I. Gottlieb; Saul S. Abarbanel (1994). *Time-stable boundary conditions for finite-difference schemes solving hyperbolic systems: Methodology and application to high-order compact schemes*. *Journal of Computational Physics*. doi:10.1006/jcph.1994.1057 (https://doi.org/10.1006%2Fjcph.1994.1057).

Further reading

- K.W. Morton and D.F. Mayers, *Numerical Solution of Partial Differential Equations, An Introduction*. Cambridge University Press, 2005.
- Autar Kaw and E. Eric Kalu, *Numerical Methods with Applications*, (2008) [1] (http://www.autarkaw.com/books/numericalmethods/index.html). Contains a brief, engineering-oriented introduction to FDM (for ODEs) in Chapter 08.07 (http://numericalmethods.eng.usf.edu/topics/finite_difference_method.html).
- John Strikwerda (2004). *Finite Difference Schemes and Partial Differential Equations* (2nd ed.). SIAM. ISBN 978-0-89871-639-9.
- Smith, G. D. (1985), *Numerical Solution of Partial Differential Equations: Finite Difference Methods, 3rd ed.*, Oxford University Press

- Peter Olver (2013). *Introduction to Partial Differential Equations* (http://www.math.umn.edu/~olver/pde.html). Springer. Chapter 5: Finite differences. ISBN 978-3-319-02099-0..
- Randall J. LeVeque, *Finite Difference Methods for Ordinary and Partial Differential Equations (http://faculty.washington.edu/rjl/fdmbook/)*, SIAM, 2007.

Various lectures and lecture notes

- Finite-Difference Method in Electromagnetics (see and listen to lecture 9) (https://web.archive.org/web/20140302111344/http://emlab.utep.edu/ee5390cem.htm)
- Lecture Notes (https://web.archive.org/web/20090206071754/http://ltl.iams.sinica.edu.tw/document/training_lectures/2006/SH_Chen/Finite_Difference_Methods.pdf) Shih-Hung Chen, National Central University
- Numerical Methods for time-dependent Partial Differential Equations (http://jwhaverkort.net23.net/documents/NumMethPDEs.pdf)