;e difference methods for diffusion proce;

Hans Petter Langtangen[1,2]

[1]Center for Biomedical Computing, Simula Research Laboratory
[2]Department of Informatics, University of Oslo

Jul 14, 2014

VERY PRELIMINARY VERSION

nts

ist of Exercises and Projects

ıe 1D diffusion equation

ısion equation, 1D indexheat equation, 1D
ımous *diffusion equation*, also known as the *heat equation*, reads

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

:, t) is the unknown function to be solved for, x is a coordinate in space, and t
cient α is the *diffusion coefficient* and determines how fast u changes in time.
ı for the diffusion equation is $u_t = \alpha u_{xx}$.
ared to the wave equation, $u_{tt} = c^2 u_{xx}$, which looks very similar, but the c
features solutions that are very different from those of the wave equation. A
equation makes quite different demands to the numerical methods.
ıl diffusion problems may experience rapid change in the very beginning, but t
of u becomes slower and slower. The solution is usually very smooth, and aft
cannot recognize the initial shape of u. This is in sharp contrast to solution
ation where the initial shape is preserved - the solution is basically a movin
. The standard wave equation $u_{tt} = c^2 u_{xx}$ has solutions that propagates with
ithout changing shape, while the diffusion equation converges to a *stationary*
$\to \infty$. In this limit, $u_t = 0$, and $\bar{u}$ is governed by $\bar{u}''(x) = 0$. This stationary lim
equation is called the *Laplace* equation and arises in a very wide range of appl
ıt the sciences.
ossible to solve for $u(x,t)$ using a explicit scheme, but the time step restrictio
ıuch less favorable than for an explicit scheme for the wave equation. And
:e, since the solution u of the diffusion equation is very smooth and changes slow
s are not convenient and not required by accuracy as the diffusion process conv
ıry state.

he initial-boundary value problem for 1D diffusion

a unique solution of the diffusion equation, or equivalently, to apply numerical r
ıitial and boundary conditions. The diffusion equation goes with one initial c
$I(x)$, where I is a prescribed function. One boundary condition is required
:he boundary, which in 1D means that u must be known, u_x must be known,
on of them.
all start with the simplest boundary condition: $u = 0$. The complete initial-b
ısion problem in one space dimension can then be specified as

$$\begin{aligned}
\frac{\partial u}{\partial t} &= \alpha \frac{\partial^2 u}{\partial x^2}, \quad x \in (0,L),\ t \in (0,T] \\
u(x,0) &= I(x), \qquad x \in [0,L] \\
u(0,t) &= 0, \qquad t > 0, \\
u(L,t) &= 0, \qquad t > 0\,.
\end{aligned}$$

(1) is known as a one-dimensional *diffusion equation*, also often referred to a
With only a first-order derivative in time, only one *initial condition* is needed, w
der derivative in space leads to a demand for two *boundary conditions*. The pa
e given and is referred to as the *diffusion coefficient*.

Diffusion equations like (1) have a wide range of applications throughout phy
ıd financial sciences. One of the most common applications is propagation of he
epresents the temperature of some substance at point x and time t. Section **??**
idely occurring applications.

.2 Forward Euler scheme

he first step in the discretization procedure is to replace the domain $[0,L] \times$
ıesh points. Here we apply equally spaced mesh points

$$x_i = i\Delta x, \quad i = 0, \ldots, N_x,$$

ıd

$$t_n = n\Delta t, \quad n = 0, \ldots, N_t\,.$$

Ioreover, u_i^n denotes the mesh function that approximates $u(x_i, t_n)$ for i =
$= 0, \ldots, N_t$. Requiring the PDE (1) to be fulfilled at a mesh point (x_i, t_n) leads

$$\frac{\partial}{\partial t} u(x_i, t_n) = \alpha \frac{\partial^2}{\partial x^2} u(x_i, t_n),$$

he next step is to replace the derivatives by finite difference approximations. The
mplest method arises from using a forward difference in time and a central dif

$$[D_t^+ u = \alpha D_x D_x u]_i^n\,.$$

Vritten out,

$$\frac{u_i^{n+1} - u_i^n}{\Delta t} = \alpha \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}\,.$$

Ve have turned the PDE into algebraic equations, also often called discrete equ
roperty of the equations is that they are algebraic, which makes them easy to
e anticipate that u_i^n is already computed such that u_i^{n+1} is the only unknow
ith respect to this unknown is easy:

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

The computational algorithm then becomes

1. compute $u_i^0 = I(x_i)$for $i = 0, \ldots, N_x$
2. for $n = 0, 1, \ldots, N_t$:
   (a) apply (8) for all the internal spatial points $i = 1, \ldots, N_x - 1$
   (b) set the boundary values $u_i^{n+1} = 0$ for $i = 0$ and $i = N_x$

he algorithm is compactly fully specified in Python:

```
pace(0, L, Nx+1)    # mesh points in space
] - x[0]
pace(0, T, Nt+1)    # mesh points in time
] - t[0]
t/dx**2
ros(Nx+1)
ros(Nx+1)

itial condition u(x,0) = I(x)
 range(0, Nx+1):
i] = I(x[i])

 range(0, Nt):
mpute u at inner mesh points
i in range(1, Nx):
u[i] = u_1[i] + Fo*(u_1[i-1] - 2*u_1[i] + u_1[i+1])

sert boundary conditions
 = 0;  u[Nx] = 0

date u_1 before next step
:]= u
```

ackward Euler scheme

pply a backward difference in time in (5), but the same central difference in sp

$$[D_t^- u = D_x D_x u]_i^n,$$

tten out reads

$$\frac{u_i^n - u_i^{n-1}}{\Delta t} = \alpha \frac{u_{i+1}^n - 2u_i^n + u_{i-1}^n}{\Delta x^2}.$$

ssume u_i^{n-1} is computed, but all quantities at the "new" time level n are unknow not possible to solve with respect to u_i^n because this value couples to its ne u_{i-1}^n and u_{i+1}^n, which are also unknown. Let us examine this fact for the ca Equation (10) written for $i = 1, \ldots, Nx - 1 = 1, 2$ becomes

$$\frac{u_1^n - u_1^{n-1}}{\Delta t} = \alpha \frac{u_2^n - 2u_1^n + u_0^n}{\Delta x^2}$$
$$\frac{u_2^n - u_2^{n-1}}{\Delta t} = \alpha \frac{u_3^n - 2u_2^n + u_1^n}{\Delta x^2}$$

dary values u_0^n and u_3^n are known as zero. Collecting the unknown new values left-hand side gives

$$\left(1 + 2\alpha \frac{\Delta t}{\Delta x^2}\right) u_1^n - \alpha \frac{\Delta t}{\Delta x^2} u_2^n = u_1^{n-1},$$
$$-\alpha \frac{\Delta t}{\Delta x^2} u_1^n + \left(1 + 2\alpha \frac{\Delta t}{\Delta x^2}\right) u_2^n = u_2^{n-1}.$$

coupled 2×2 system of algebraic equations for the unknowns u_1^n and u_2^n. Discre that lead to a coupled system of equations for the unknown function at a n

vel are said to be *implicit methods*. The counterpart, *explicit methods*, refers ethods where there is a simple explicit formula for the values of the unknown fu ne spatial mesh points at the new time level. From an implementational point ethods are more comprehensive to code since they require the solution of co e., a matrix system, at each time level.

In the general case, (10) gives rise to a coupled $(Nx - 1) \times (Nx - 1)$ syst quations for all the unknown u_i^n at the interior spatial points $i = 1, \ldots, Nx - 1$ nknowns on the left-hand side, and introducing the *numerical Fourier number*

$$F_o = \alpha \frac{\Delta t}{\Delta x^2},$$

.0) can be written

$$-F_o u_{i-1}^n + (1 + 2F_o)\, u_i^n - F_o u_{i+1}^n = u_{i-1}^{n-1},$$

or $i = 1, \ldots, Nx - 1$. One can either view these equations as a system for where ne internal grid points, $i = 1, \ldots, N_x - 1$, are unknown, or we may append the $_0^n$ and $u_{N_x}^n$ to the system. In the latter case, all u_i^n for $i = 0, \ldots, N_x$ are unkno dd the boundary equations to the $N_x - 1$ equations in (16):

$$u_0^n = 0,$$
$$u_{N_x}^n = 0.$$

A coupled system of algebraic equations can be written on matrix form, and t we want to call up ready-made software for solving the system. The equations (1 orrespond to the matrix equation

$$AU = b$$

here $U = (u_0^n, \ldots, u_{N_x}^n)$, and the matrix A has the following structure:

$$A = \begin{pmatrix} A_{0,0} & A_{0,1} & 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 \\ A_{1,0} & A_{1,1} & 0 & \ddots & & & & & \vdots \\ 0 & A_{2,1} & A_{2,2} & A_{2,3} & \ddots & & & & \vdots \\ \vdots & \ddots & & \ddots & \ddots & 0 & & & \vdots \\ \vdots & & \ddots & \ddots & \ddots & \ddots & \ddots & & \vdots \\ \vdots & & & 0 & A_{i,i-1} & A_{i,i} & A_{i,i+1} & \ddots & \vdots \\ \vdots & & & & \ddots & \ddots & \ddots & \ddots & 0 \\ \vdots & & & & & \ddots & \ddots & \ddots & A_{N_x-} \\ 0 & \cdots & \cdots & \cdots & \cdots & \cdots & 0 & A_{N_x,N_x-1} & A_{N_x,} \end{pmatrix}$$

he nonzero elements are given by

$$A_{i,i-1} = -F_o$$
$$A_{i,i} = 1 + 2F_o$$
$$A_{i,i+1} = -F_o$$

uations for internal points, $i = 1, \ldots, N_x - 1$. The equations for the boundary
d to

$$
\begin{aligned}
A_{0,0} &= 1, \\
A_{0,1} &= 0, \\
A_{N_x,N_x-1} &= 0, \\
A_{N_x,N_x} &= 1\,.
\end{aligned}
$$

-hand side b is written as

$$
b = \begin{pmatrix} b_0 \\ b_1 \\ \vdots \\ b_i \\ \vdots \\ b_{N_x} \end{pmatrix}
$$

$$
\begin{aligned}
b_0 &= 0, \\
b_i &= u_i^{n-1}, \quad i = 1, \ldots, N_x - 1, \\
b_{N_x} &= 0\,.
\end{aligned}
$$

serve that the matrix A contains quantities that do not change in time. Ther
rmed once and for all before we enter the recursive formulas for the time ev
-hand side b, however, must be updated at each time step. This leads to the fo
ional algorithm, here sketched with Python code:

```
pace(0, L, Nx+1)   # mesh points in space
] - x[0]
pace(0, T, N+1)    # mesh points in time
ros(Nx+1)
ros(Nx+1)

tructures for the linear system
s((Nx+1, Nx+1))
s(Nx+1)

 range(1, Nx):
i-1] = -Fo
i+1] = -Fo
i] = 1 + 2*Fo
 A[Nx,Nx] = 1

itial condition u(x,0) = I(x)
 range(0, Nx+1):
i] = I(x[i])

cipy.linalg

 range(0, Nt):
mpute b and solve linear system
```



```
    for i in range(1, Nx):
        b[i] = -u_1[i]
    b[0] = b[Nx] = 0
    u[:] = scipy.linalg.solve(A, b)

    # Update u_1 before next step
    u_1[:] = u
```

.4 Sparse matrix implementation

Ve have seen from (19) that the matrix A is tridiagonal. The code segment a
ense matrix representation of A, which stores a lot of values we know are zero
orse, the solution algorithm computes with all these zeros. With $N_x + 1$ unk
y the solution algorithm is $\frac{1}{3}(N_x + 1)^3$ and the storage requirements $(N_x + 1$
ue fact that A is tridiagonal and employing corresponding software tools, the w
emands can be proportional to N_x only.

The key idea is to apply a data structure for a tridiagonal or sparse matrix. Th
ackage has relevant utilities. For example, we can store the nonzero diagonals
ackage also has linear system solvers that operate on sparse matrix data struc
elow illustrates how we can store only the main diagonal and the upper and lo

```
# Representation of sparse matrix and right-hand side
main  = zeros(Nx+1)
lower = zeros(Nx-1)
upper = zeros(Nx-1)
b     = zeros(Nx+1)

# Precompute sparse matrix
main[:] = 1 + 2*Fo
lower[:] = -Fo  #1
upper[:] = -Fo  #1
# Insert boundary conditions
main[0] = 1
main[Nx] = 1

A = scipy.sparse.diags(
    diagonals=[main, lower, upper],
    offsets=[0, -1, 1], shape=(Nx+1, Nx+1),
    format='csr')
print A.todense()

# Set initial condition
for i in range(0,Nx+1):
    u_1[i] = I(x[i])

for n in range(0, Nt):
    b = u_1
    b[0] = b[-1] = 0.0  # boundary conditions
    u[:] = scipy.sparse.linalg.spsolve(A, b)
    u_1[:] = u
```

he `scipy.sparse.linalg.spsolve` function utilizes the sparse storage stru
erforms in this case a very efficient Gaussian elimination solve.

.5 The θ rule

he θ rule provides a family of finite difference approximations in time:

0 gives the Forward Euler scheme in time

1 gives the Backward Euler scheme in time

$\frac{1}{2}$ gives the Crank-Nicolson scheme in time

o the 1D diffusion problem we have

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

me also leads to a matrix system with entries $1+2F_o\theta$ on the main diagona
nd $-F_o\theta$ on the super- and sub-diagonal. The right-hand side entry b_i is

$$b_i = u_i^n + F_o(1-\theta)\frac{u_{i+1}^n-2u_i^n+u_{i-1}^n}{\Delta x^2}.$$

he Laplace and Poisson equation

ace equation, $\nabla^2 u = 0$, or the Poisson equation, $-\nabla^2 u = f$, occur in m
ns throughout science and engineering. We can solve 1D variants of the
with the listed software, because we can interpret $u_{xx} = 0$ as the limiting
u_{xx} when u reach a steady state limit where $u_t \to 0$. Similarly, Poisson's e
arises from solving $u_t = u_{xx} + f$ and letting $t \to$ so $u_t \to 0$.
ically in a program, we can simulate $t \to \infty$ by just taking one large time
tly, set α to a large value. All we need is to have F_o large. As $F_o \to \infty$, we c
es see that the limiting discrete equation becomes

$$\frac{u_{i+1}^{n+1}-2u_i^{n+1}+u_{i-1}^{n+1}}{\Delta x^2} = 0,$$

othing but the discretization $[D_xD_xu]_i^{n+1} = 0$ of $u_{xx} = 0$.
ackward Euler scheme can solve the limit equation directly and hence produce a
Laplace equation. With the Forward Euler scheme we must do the time steppi
is illegal and leads to instability. We may interpret this time stepping as solv
system from u_{xx} by iterating on a time pseudo time variable.

xtensions

ensions are performed exactly as for a wave equation as they only affect the
s (which are the same as in the wave equation).

iable coefficients

mann and Robin conditions

and 3D

rsions of this document will for completeness and independence of the wave e
feature info on the three points. The Robin condition is new, but straightfor

$$-\alpha\frac{\partial u}{\partial n} = h_T(u-U_s), \quad [-\alpha D_x u = h_T(u-U_s)]_i^n$$

Analysis of schemes for the diffusion equation

.1 Properties of the solution

particular characteristic of diffusive processes, governed by an equation like

$$u_t = \alpha u_{xx},$$

that the initial shape $u(x,0) = I(x)$ spreads out in space with time, along
mplitude. Three different examples will illustrate the spreading of u in space a
me.

imilarity solution. The diffusion equation (31) admits solutions that depe
$)/\sqrt{4\alpha t}$ for a given value of c. One particular solution is

$$u(x,t) = a\,\mathrm{erf}(\eta) + b,$$

here

$$\mathrm{erf}(\eta) = \frac{2}{\sqrt{\pi}}\int_0^\eta e^{-\zeta^2}d\zeta,$$

the *error function*, and a and b are arbitrary constants. The error function l
dd around $\eta = 0$, and goes relatively quickly to ± 1:

$$\begin{aligned}
\lim_{\eta\to-\infty}\mathrm{erf}(\eta) &= -1,\\
\lim_{\eta\to\infty}\mathrm{erf}(\eta) &= 1,\\
\mathrm{erf}(\eta) &= -\mathrm{erf}(-\eta),\\
\mathrm{erf}(0) &= 0,\\
\mathrm{erf}(2) &= 0.99532227,\\
\mathrm{erf}(3) &= 0.99997791\,.
\end{aligned}$$

As $t \to 0$, the error function approaches a step function centered at $x = c$.
roblem posed on the unit interval $[0,1]$, we may choose the step at $x = 1/2$ (m
$= -1/2$, $b = 1/2$. Then

$$u(x,t) = \frac{1}{2}\left(1-\mathrm{erf}\left(\frac{x-\frac{1}{2}}{\sqrt{4\alpha t}}\right)\right) = \frac{1}{2}\mathrm{erfc}\left(\frac{x-\frac{1}{2}}{\sqrt{4\alpha t}}\right),$$

here we have introduced the *complementary error function* $\mathrm{erfc}(\eta) = 1-\mathrm{erf}(\eta)$
34) implies the boundary conditions

$$\begin{aligned}
u(0,t) &= \frac{1}{2}\left(1-\mathrm{erf}\left(\frac{-1/2}{\sqrt{4\alpha t}}\right)\right),\\
u(1,t) &= \frac{1}{2}\left(1-\mathrm{erf}\left(\frac{1/2}{\sqrt{4\alpha t}}\right)\right).
\end{aligned}$$

or small enough t, $u(0,t) \approx 1$ and $u(1,t) \approx 1$, but as $t \to \infty$, $u(x,t) \to 1/2$ on

for a Gaussian pulse. The standard diffusion equation $u_t = \alpha u_{xx}$ admits a G
as solution:

$$u(x,t) = \frac{1}{\sqrt{4\pi\alpha t}} \exp\left(-\frac{(x-c)^2}{4\alpha t}\right).$$

his is a Dirac delta function, so for computational purposes one must start to v
t some time $t = t_\epsilon > 0$. Replacing t by $t_\epsilon + t$ in (37) makes it easy to operate
at starts at $t = 0$ with an initial condition with a finite width. The important fe
t the standard deviation σ of a sharp initial Gaussian pulse increases in time ac
$\sqrt{2\alpha t}$, making the pulse diffuse and flatten out.

for a sine component. For example, (31) admits a solution of the form

$$u(x,t) = Qe^{-at}\sin(kx)\ .$$

neters Q and k can be freely chosen, while inserting (38) in (31) gives the con

$$a = -\alpha k^2\,.$$

y important feature is that the initial shape $I(x) = Q\sin kx$ undergoes a d
^2t), meaning that rapid oscillations in space, corresponding to large k, are ver
pened than slow oscillations in space, corresponding to small k. This feature le
g of the initial condition with time.
llowing examples illustrates the damping properties of (38). We consider the

$$\begin{aligned} u_t &= u_{xx}, \quad x \in (0,1),\ t \in (0,T], \\ u(0,t) &= u(1,t) = 0, \quad t \in (0,T], \\ u(x,0) &= \sin(\pi x) + 0.1\sin(100\pi x)\,. \end{aligned}$$

l condition has been chosen such that adding two solutions like (38) constr
solution to the problem:

$$u(x,t) = e^{-\pi^2 t}\sin(\pi x) + 0.1e^{-\pi^2 10^4 t}\sin(100\pi x)\,.$$

llustrates the rapid damping of rapid oscillations $\sin(100\pi x)$ and the very muc
of the slowly varying $\sin(\pi x)$ term. After about $t = 0.5\cdot 10^{-4}$ the rapid oscilla
a visible amplitude, while we have to wait until $t \sim 0.5$ before the amplitude of
$\pi x)$ becomes very small.

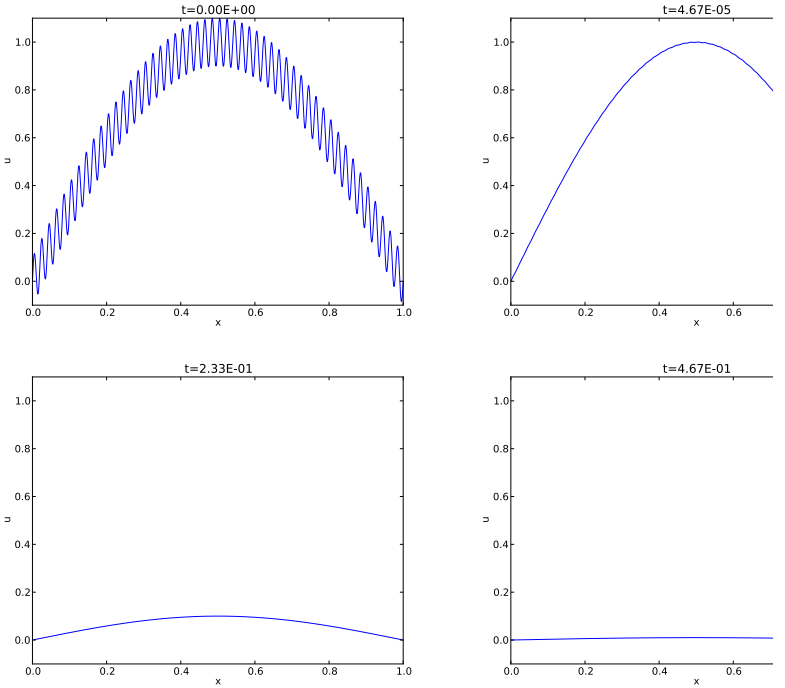
nalysis of discrete equations

part to (38) is the complex representation of the same function:

$$u(x,t) = Qe^{-at}e^{ikx},$$

$= \sqrt{-1}$ is the imaginary unit. We can add such functions, often referred to
its, to make a Fourier representation of a general solution of the diffusion equa

$$u(x,t) \approx \sum_{k\in K} b_k e^{-\alpha k^2 t}e^{ikx},$$



igure 1: Evolution of the solution of a diffusion problem: initial condition (u
eduction of the small waves (upper right), 1/10 reduction of the long wave (
/100 reduction of the long wave (lower right).

here K is a set of an infinite number of k values needed to construct the solut
owever, the series is truncated and K is a finite set of k values need build a go
olution. Note that (39) is a special case of (40) where $K = \{\pi, 100\pi\}$, $b_\pi = 1$,

The amplitudes b_k of the individual Fourier waves must be determined
ondition. At $t = 0$ we have $u \approx \sum_k b_k \exp(ikx)$ and find K and b_k such that

$$I(x) \approx \sum_{k\in K} b_k e^{ikx}\,.$$

The relevant formulas for b_k come from Fourier analysis, or equivalently, a least
or approximating $I(x)$ in a function space with basis $\exp(ikx)$.)

Much insight about the behavior of numerical methods can be obtained by in
wave component $\exp(-\alpha k^2 t)\exp(ikx)$ is treated by the numerical scheme.
ch wave components are also solutions of the schemes, but the damping fac
aries among the schemes. To ease the forthcoming algebra, we write the damp
he exact amplification factor corresponding to A is $A_e = \exp(-\alpha k^2\Delta t)$.

nalysis of the finite difference schemes

een that a general solution of the diffusion equation can be built as a linear coml
omponents

$$e^{-\alpha k^2 t} e^{ikx}\,.$$

ıental question is whether such components are also solutions of the finite di
This is indeed the case, but the amplitude $\exp(-\alpha k^2 t)$ might be modified (wl
vhen solving the ODE counterpart $u' = -\alpha u$). We therefore look for numerical s
m

$$u_q^n = A^n e^{ikq\Delta x} = A^n e^{ikx},$$

amplification factor A must be determined by inserting the component into a

. The exact amplification factor is $A_\mathrm{e} = \exp(-\alpha^2 k^2 \Delta t)$. We should therefore
› have a decaying numerical solution as well. If $-1 \leq A < 0$, A^n will change si
. to time level, and we get stable, non-physical oscillations in the numerical s
ıot present in the exact solution.

y. To determine how accurately a finite difference scheme treats one wave cor
e that the basic deviation from the exact solution is reflected in how well A^n appro
w well A approximates A_e.

nalysis of the Forward Euler scheme

ard Euler finite difference scheme for $u_t = \alpha u_{xx}$ can be written as

$$[D_t^+ u = \alpha D_x D_x u]_q^n\,.$$

a wave component (42) in the scheme demands calculating the terms

$$e^{ikq\Delta x}[D_t^+ A]^n = e^{ikq\Delta x} A^n \frac{A-1}{\Delta t},$$

$$A^n D_x D_x [e^{ikx}]_q = A^n \left(-e^{ikq\Delta x}\frac{4}{\Delta x^2}\sin^2\left(\frac{k\Delta x}{2}\right)\right).$$

these terms in the discrete equation and dividing by $A^n e^{ikq\Delta x}$ leads to

$$\frac{A-1}{\Delta t} = -\alpha\frac{4}{\Delta x^2}\sin^2\left(\frac{k\Delta x}{2}\right),$$

quently

$$A = 1 - 4F_o \sin^2\left(\frac{k\Delta x}{2}\right),$$

$$F_o = \frac{\alpha\Delta t}{\Delta x^2}$$

the *numerical Fourier number*. The complete numerical solution is then

$$u_q^n = \left(1 - 4F_o \sin^2\left(\frac{k\Delta x}{2}\right)\right)^n e^{ikq\Delta x}\,.$$

tability. We easily see that $A \leq 1$. However, the A can be less than -1, w
rowth of a numerical wave component. The criterion $A \geq -1$ implies

$$4F_o \sin^2(p/2) \leq 2\,.$$

he worst case is when $\sin^2(p/2) = 1$, so a sufficient criterion for stability is

$$F_o \leq \frac{1}{2},$$

r expressed as a condition on Δt:

$$\Delta t \leq \frac{\Delta x^2}{2\alpha}\,.$$

ote that halving the spatial mesh size, $\Delta x \to \frac{1}{2}\Delta x$, requires Δt to be reduce
/4. The method hence becomes very expensive for fine spatial meshes.

ccuracy. Since A is expressed in terms of F_o and the parameter we now cal
lso express A_e by F_o and p:

$$A_\mathrm{e} = \exp(-\alpha k^2 \Delta t) = \exp(-4F_o p^2)\,.$$

omputing the Taylor series expansion of A/A_e in terms of F_o can easily be
ympy:

```
def A_exact(Fo, p):
    return exp(-4*Fo*p**2)

def A_FE(Fo, p):
    return 1 - 4*Fo*sin(p)**2

from sympy import *
Fo, p = symbols('Fo p')
A_err_FE = A_FE(Fo, p)/A_exact(Fo, p)
print A_err_FE.series(Fo, 0, 6)
```

he result is

$$\frac{A}{A_\mathrm{e}} = 1 - 4F_o\sin^2 p + 2F_o p^2 - 16F_o^2 p^2 \sin^2 p + 8F_o^2 p^4 + \cdots$$

ecalling that $F_o = \alpha\Delta t/\Delta x$, $p = k\Delta x/2$, and that $\sin^2 p \leq 1$, we realize that
rror terms are at most

$$1 - 4\alpha\frac{\Delta t}{\Delta x^2} + \alpha\Delta t - 4\alpha^2\Delta t^2 + \alpha^2\Delta t^2\Delta x^2 + \cdots\,.$$

nalysis of the Backward Euler scheme

ng $u_t = \alpha u_{xx}$ by a Backward Euler scheme,

$$[D_t^- u = \alpha D_x D_x u]_q^n,$$

ting a wave component (42), leads to calculations similar to those arising fi
Euler scheme, but since

$$e^{ikq\Delta x}[D_t^- A]^n = A^n e^{ikq\Delta x}\frac{1 - A^{-1}}{\Delta t},$$

$$\frac{1 - A^{-1}}{\Delta t} = -\alpha\frac{4}{\Delta x^2}\sin^2\left(\frac{k\Delta x}{2}\right),$$

$$A = \left(1 + 4F_o \sin^2 p\right)^{-1}.$$

plete numerical solution can be written

$$u_q^n = \left(1 + 4F_o \sin^2 p\right)^{-n} e^{ikq\Delta x}.$$

. We see from (48) that $0 < A < 1$, which means that all numerical wave com
and non-oscillatory for any $\Delta t > 0$.

nalysis of the Crank-Nicolson scheme

k-Nicolson scheme can be written as

$$[D_t u = \alpha D_x D_x \overline{u}^x]_q^{n+\frac{1}{2}},$$

$$[D_t u]_q^{n+\frac{1}{2}} = \frac{1}{2}\alpha\left([D_x D_x u]_q^n + [D_x D_x u]_q^{n+1}\right).$$

(42) in the time derivative approximation leads to

$$[D_t A^n e^{ikq\Delta x}]^{n+\frac{1}{2}} = A^{n+\frac{1}{2}} e^{ikq\Delta x}\frac{A^{\frac{1}{2}} - A^{-\frac{1}{2}}}{\Delta t} = A^n e^{ikq\Delta x}\frac{A - 1}{\Delta t}.$$

(42) in the other terms and dividing by $A^n e^{ikq\Delta x}$ gives the relation

$$\frac{A - 1}{\Delta t} = -\frac{1}{2}\alpha\frac{4}{\Delta x^2}\sin^2\left(\frac{k\Delta x}{2}\right)(1 + A),$$

some more algebra,

$$A = \frac{1 - 2F_o \sin^2 p}{1 + 2F_o \sin^2 p}.$$

numerical solution is hence

$$u_q^n = \left(\frac{1 - 2F_o \sin^2 p}{1 + 2F_o \sin^2 p}\right)^n e^{ikp\Delta x}.$$

tability. The criteria $A > -1$ and $A < 1$ are fulfilled for any $\Delta t > 0$.

.7 Summary of accuracy of amplification factors

Ve can plot the various amplification factors against $p = k\Delta x/2$ for different c
arameter. Figures 2, 3, and 4 show how long and small waves are damped by the
ompared to the exact damping. As long as all schemes are stable, the amplif
ositive, except for Crank-Nicolson when $F_o > 0.5$.

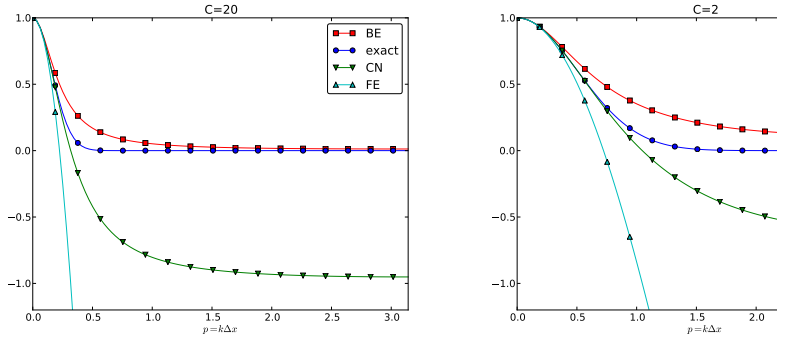


Figure 2: Amplification factors for large time steps.

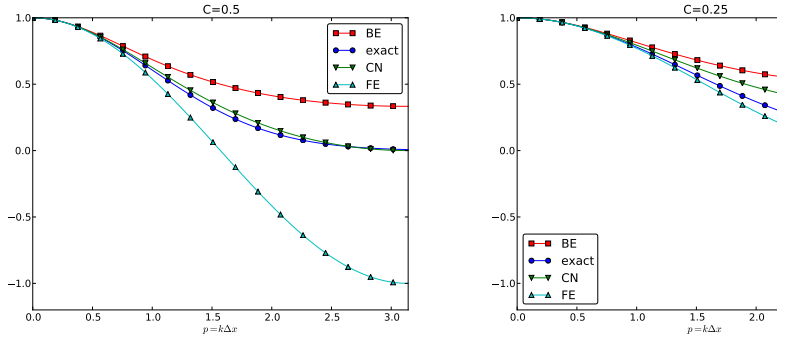


Figure 3: Amplification factors for time steps around the Forward Euler sta

The effect of negative amplification factors is that A^n changes sign from o
ne next, thereby giving rise to oscillations in time in an animation of the soluti
igure 2 that for $F_o = 20$, waves with $p \geq \pi/2$ undergo a damping close to $-$
nat the amplitude does not decay and that the wave component jumps up an
or $F_o = 2$ we have a damping of a factor of 0.5 from one time level to the nex
nuch smaller than the exact damping. Short waves will therefore fail to be effect
hese waves will manifest themselves as high frequency oscillatory noise in the

A value $p = \pi/4$ corresponds to four mesh points per wave length of e^{ikx}
nplies only two points per wave length, which is the smallest number of points
epresent the wave on the mesh.

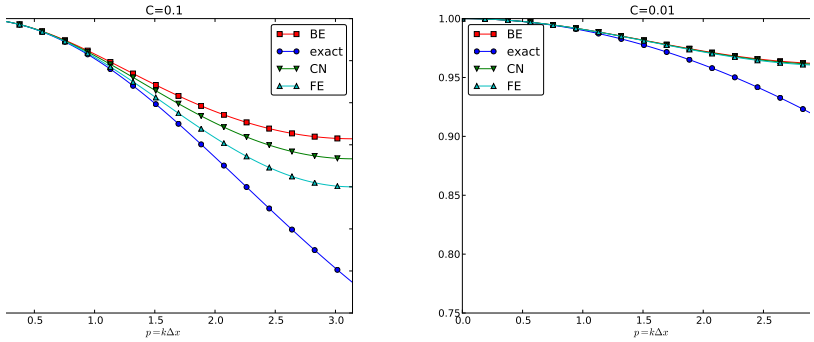


Figure 4: Amplification factors for small time steps.

nonstrate the oscillatory behavior of the Crank-Nicolson scheme, we choose a
that leads to short waves with significant amplitude. A discontinuous $I(x)$
serve this purpose.
$_o$ = ...

e 1: Use an analytical solution to formulate a 1D test

cise explores the exact solution (37). We shall formulate a diffusion problem
nain for half of the Gaussian pulse. Then we shall investigate the impact of u
boundary condition, which we in general cases often are forced due if the solutic
rough finite boundaries undisturbed.

olution (37) is seen to be symmetric at $x = c$, because $\partial u/\partial x = 0$ always vani
e this property to formulate a complete initial boundary value problem in 1D i
ion equation $u_t = \alpha u_{xx}$ on $[0, L]$ with $u_x(0, t) = 0$ and $u(L, t)$ known.

ıe exact solution to set up a convergence rate test for an implementation of the
e if a one-sided difference for $u_x(0, t)$, say $u_0 = u_1$, destroys the second-order a

ıe that we want to solve the problem numerically on $[0, L]$, but we do not k
ıtion and cannot of that reason assign a correct Dirichlet condition at $x =$
simply set $u(L, t) = 0$ since this will be an accurate approximation before the
hes $x = L$ and even thereafter it might be a satisfactory condition. Let u_e be t
nd let u be the solution of $u_t = \alpha u_{xx}$ with an initial Gaussian pulse and the b
s $u_x(0, t) = u(L, t) = 0$. Derive a diffusion problem for the error $e = u_e - u$. S
ıumerically using an exact Dirichlet condition at $x = L$. Animate the evolutio
make a curve plot of the error measure

$$E(t) = \sqrt{\frac{\int_0^L e^2 dx}{\int_0^L u dx}}.$$

ıuitable error measure for the present problem?

) Instead of using $u(L, t) = 0$ as approximate boundary condition for lett
Gaussian pulse out of our finite domain, one may try $u_x(L, t) = 0$ since the sol
quite flat. Argue that this condition gives a completely wrong asymptotic s
o do this, integrate the diffusion equation from 0 to L, integrate u_{xx} by part
ivergence theorem in 1D) to arrive at the important property

$$\frac{d}{dt}\int_0^L u(x, t)dx = 0,$$

nplying that $\int_0^L u dx$ must be constant in time, and therefore

$$\int_0^L u(x, t)dx = \int_0^L I(x)dx\,.$$

he integral of the initial pulse is 1.

) Another idea for an artificial boundary condition at $x = L$ is to use a coolin

$$-\alpha u_x = q(u - u_S),$$

here q is an unknown heat transfer coefficient and u_S is the surrounding ten
ıedium outside of $[0, L]$. (Note that arguing that u_S is approximately $u(L, t)$
ondition from the previous subexercise that is qualitatively wrong for large t.) De
roblem for the error in the solution using (52) as boundary condition. Assu
$_S = 0$ "outside the domain" as $u \to 0$ for $x \to \infty$. Find a function $q = q(t$
xact solution obeys the condition (52). Test some constant values of q and a
orresponding error function behaves. Also compute $E(t)$ curves as suggested i
ilename: `diffu_symmetric_gaussian.py`.

Exercise 2: Use an analytical solution to formulate a 2D te

eneralize (37) to multi dimensions by assuming that one-dimensional solutions c
o solve $u_t = \alpha\nabla^2 u$. Use this solution to formulate a 2D test case where the peak
at the origin and where the domain is a rectangle in the first quadrant.
oundary conditions $\partial u/\partial n = 0$ whereever possible, and use exact Dirichlet c
emaining boundaries. Filename: `diffu_symmetric_gaussian_2D.pdf`.

Exercise 3: Examine stability of a diffusion model with a s

onsider a diffusion equation with a linear u term:

$$u_t = \alpha u_{xx} + \beta u\,.$$

) Derive in detail a Forward Euler scheme, a Backward Euler scheme, and a Cr
ıis type of diffusion model. Thereafter, formulate a θ-rule to summarize the t

) Assume a solution like (38) and find the relation between a, k, α, and β.

) Calculate the stability of the Forward Euler scheme. Design numerical experi
ıe results.

) Repeat c) for the Backward Euler scheme.

) Repeat c) for the Crank-Nicolson scheme.

oes the extra term bu impact the accuracy of the three schemes?

ompare the numerical and exact amplification factors, either in graphs or by
ansion (or both).
`diffu_stab_uterm.pdf`.

ercises

e 4: Stabilizing the Crank-Nicolson method by Rannacher
g

known that the Crank-Nicolson method may give rise to non-physical oscillation
f diffusion equations if the initial data exhibit jumps (see Section 2.6). Ranna
a stabilizing technique consisting of using the Backward Euler scheme for the f
with step length $\frac{1}{2}\Delta t$. One can generalize this idea to taking $2m$ time steps of s
Backward Euler method and then continuing with the Crank-Nicolson method,
-order in time. The idea is that the high frequencies of the initial solution are
ut, and the Backward Euler scheme treats these high frequencies correctly. Th
requency content of the solution is gone and the Crank-Nicolson method will
his idea for $m = 1, 2, 3$ on a diffusion problem with a discontinuous initial co
he convergence rate using the solution (34) with the boundary conditions (35)-(
h that the conditions are in the vicinity of ± 1. For example, $t < 5a1.6 \cdot 10^{-2}$ m
liffusion from a step to almost a straight line. The program `diffu_erf_sol.p`
mpute the analytical solution.

5: Energy estimates for diffusion problems

ect concerns so-called *energy estimates* for diffusion problems that can be u
e analytical insight and for verification of implementations.

art with a 1D homogeneous diffusion equation with zero Dirichlet conditions:

$$\begin{aligned} u_t &= \alpha u_x x, && x \in \Omega = (0, L),\ t \in (0, T], \\ u(0, t) = u(L, t) &= 0, && t \in (0, T], \\ u(x, 0) &= I(x), && x \in [0, L]\,. \end{aligned}$$

y estimate for this problem reads

$$||u||_{L^2} \leq ||I||_{L^2},$$

$||\cdot||_{L^2}$ norm is defined by

$$||g||_{L^2} = \sqrt{\int_0^L g^2 dx}\,.$$

tify $||u||_{L^2}$ or $\frac{1}{2}||u||_{L^2}$ is known as the *energy* of the solution, although it is
nergy of the system. A mathematical tradition has introduced the notion *energy*

The estimate (56) says that the "size of u" never exceeds that of the initial co
quivalently, that the area under the u curve decreases with time.

To show (56), multiply the PDE by u and integrate from 0 to L. Use that uu_t
s the time derivative of u^2 and that $u_x xu$ can integrated by parts to form an int
hat the time derivative of $||u||_{L^2}^2$ must be less than or equal to zero. Integrate
nd derive (56).

) Now we address a slightly different problem,

$$\begin{aligned} u_t &= \alpha u_x x + f(x, t), && x \in \Omega = (0, L),\ t \in (0 \\ u(0, t) = u(L, t) &= 0, && t \in (0 \\ u(x, 0) &= 0, && x \in [0 \end{aligned}$$

he associated energy estimate is

$$||u||_{L^2} \leq ||f||_{L^2}\,.$$

This result is more difficult to derive.)

Now consider the compound problem with an initial condition $I(x)$ and a
(x, t):

$$\begin{aligned} u_t &= \alpha u_x x + f(x, t), && x \in \Omega = (0, L),\ t \in (0 \\ u(0, t) = u(L, t) &= 0, && t \in (0 \\ u(x, 0) &= I(x), && x \in [0 \end{aligned}$$

how that if w_1 fulfills (53)-(55) and w_2 fulfills (58)-(60), then $u = w_1 + w_2$ is
2)-(64). Using the triangle inequality for norms,

$$||a + b|| \leq ||a|| + ||b||,$$

ow that the energy estimate for (62)-(64) becomes

$$||u||_{L^2} \leq ||I||_{L^2} + ||f||_{L^2}\,.$$

) One application of (65) is to prove uniqueness of the solution. Suppose u_1 ar
2)-(64). Show that $u = u_1 - u_2$ then fulfills (62)-(64) with $f = 0$ and $I = 0$. Us
hat the energy must be zero for all times and therefore that $u_1 = u_2$, which
olution is unique.

) Generalize (65) to a 2D/3D diffusion equation $u_t = \nabla \cdot (\alpha \nabla u)$ for $x \in \Omega$.

Hint. Use integration by parts in multi dimensions:

$$\int_\Omega u \nabla \cdot (\alpha \nabla u)\,\mathrm{d}x = -\int_\Omega \alpha \nabla u \cdot \nabla u\,\mathrm{d}x + \int_{\partial\Omega} u\alpha \frac{\partial u}{\partial n},$$

here $\frac{\partial u}{\partial n} = \boldsymbol{n} \cdot \nabla u$, $\boldsymbol{n}$ being the outward unit normal to the boundary $\partial\Omega$ of th

re also consider the multi-dimensional PDE $u_t = \nabla \cdot (\alpha \nabla u)$. Integrate both side
auss' divergence theorem, $\int_\Omega \nabla \cdot \boldsymbol{q}\,\mathrm{d}x = \int_{\partial\Omega} \boldsymbol{q}\cdot\boldsymbol{n}\,\mathrm{d}s$ for a vector field $\boldsymbol{q}$. Show th
ogeneous Neumann conditions on the boundary, $\partial u/\partial n = 0$, area under the u
onstant in time and

$$\int_\Omega u\,\mathrm{d}x = \int_\Omega I\,\mathrm{d}x\,.$$

ish a code in 1D, 2D, or 3D that can solve a diffusion equation with a source
ıdition I, and zero Dirichlet or Neumann conditions on the whole boundary.
n use (65) and (66) as a partial verification of the code. Choose some functions
: that (65) is obeyed at any time when zero Dirichlet conditions are used. Iter
I functions and check that (66) is fulfilled when using zero Neumann conditio

a list of some possible bugs in the code, such as indexing errors in arrays, failu
ct boundary conditions, evaluation of a term at a wrong time level, and simi
ıe bugs, see if the verification tests from the previous subexercise pass or fa
ion shows how strong the energy estimates and the estimate (66) are for poin
he implementation.
`diffu_energy.pdf`.

ences

nnacher. Finite element solution of diffusion problems with irregular data. *Nur
matik*, 43:309–327, 1984.

ndex