Partial differential equations and finite difference methods.

Anne Kværnø

Dec 3, 2018

1 Introduction

In this note the finite difference method for solving partial differential equations (PDEs) will be presented.

Roughly speaking, a finite difference method is composed by the following steps:

1. Discretize the domain on which the equation is defined.

2. On each grid point, replace the derivatives with an approximation, using the values in neighbouring grid points.

3. Replace the exact solutions by its approximations.

4. Solve the resulting system of equations.

We will first see how to find approximations to the derivative of a function, and then how this can be used to solve boundary value problems like

$$u'' + p(x)u' + q(x)u = r(x), \qquad a \le x \le b, \qquad u(a) = u_a, \quad u(b) = u_b,$$

and time dependent partial differential equations with the heat equation

$$u_t = u_{xx}$$

as the example of choice. The technique described here is however applicable several other PDEs, so please concentrate on understanding the underlying idea.

2 Numerical differentiation.

This is the main tool for finite difference methods.

Given a sufficiently smooth function f. How can we find an approximation to $f'(x)$ or $f''(x)$ in some given point x, just by using evaluation of the function itself?

The derivative of of f is defined by

$$f'(x) = \lim_{h \to 0} \frac{f(x+h) - f(x)}{h}.$$

Given a sufficiently small value of h, the right hand side can be used an approximation to $f'(x)$. A small collection of the most used approximations to $f'(x)$ is:

$$f'(x) \approx \begin{cases} \dfrac{f(x+h) - f(x)}{h}, & \text{Forward difference,} \\ \dfrac{f(x) - f(x-h)}{h}, & \text{Bakward difference,} \\ \dfrac{f(x+h) - f(x-h)}{2h}, & \text{Central difference.} \end{cases}$$

The first one is taken directly from the definition, so is the second, and the third is just the mean of the first two. A common approximation to the second derivative is

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}.$$

Numerical example 1: Test the method on the function $f(x) = \sin(x)$ at the point $x = \frac{\pi}{4}$. Compare with the exact derivative. Try different step sizes, e.g. $h = 0.1, h = 0.01, h = 0.001$. Notice how the error in each case change with h.

2.1 Error analysis

In this case the error analysis is quite simple: Do a Taylor expansion of the error around x. The Taylor expansion becomes a power series in h.

The expansion for the error of the forward difference is:

$$e(x;h) = f'(x) - \frac{f(x+h) - f(x)}{h} = f'(x) - \frac{f(x) + f'(x)h + \frac{1}{2}f''(\xi)h^2 - f(x)}{h} = -\frac{1}{2}f''(\xi)h$$

where $\xi \in (x, x+h)$.

The expansion for the error of the central difference is slightly more complicated:

$$\begin{aligned} e(x;h) &= f'(x) - \frac{f(x+h) - f(x-h)}{2h} \\ &= f'(x) \\ &\quad - \frac{\left(f(x) + f'(x)h + \frac{1}{2}f''(x)h^2 + \frac{1}{6}f'''(\xi_1)h^2\right) - \left(f(x) - f'(x)h + \frac{1}{2}f''(x)h^2 - \frac{1}{6}f'''(\xi_2)h^2\right)}{2h} \\ &= -\frac{1}{12}\left(f'''(\xi_1) + f'''(\xi_2)\right)h^2 \\ &= -\frac{1}{6}f'''(\eta)h^2, \qquad \eta \in (x-h, x+h), \end{aligned}$$

where the two remainder terms have been combined by the intermediate value theorem (Result 2 at the end of *Preliminaries*). The error for the approximation of the second order derivative can be found similarly.

The order of an approximation is p if there exist a constant C independent on h such that

$$|e(h;x) \le Ch^p,$$

see *Preliminaries*.

In practice, it is sufficient to show that the power expansion of the error satisfies

$$e(x, h) = C_p h^p + C_{p+1}h^{p+1} + \cdots, \qquad C_p \neq 0$$

The forward and backward approximations are of order 1, the central differences of order 2.

We are going to use these formulas a lot in the sequel, so let us just summarize the results, including the error terms:

Difference formulas for derivatives:

$$f'(x) = \begin{cases} \dfrac{f(x+h) - f(x)}{h} - \dfrac{h}{2}f''(\xi), & \text{Forward difference} \\[3mm] \dfrac{f(x) - f(x-h)}{h} + \dfrac{h}{2}f''(\xi), & \text{Backward difference} \\[3mm] \dfrac{f(x+h) - f(x-h)}{2h} - \dfrac{h^2}{6}f'''(\xi). & \text{Central difference} \end{cases}$$

$$f''(x) = \frac{f(x+h) - 2f(x) + f(x-h)}{h^2} - \frac{h^2}{12}f^{(4)}(\xi), \qquad \text{Central difference}$$

3 Two point boundary problems (BVP)

Given a two point boundary value problem:

$$u'' + p(x)u' + q(x)u = r(x), \qquad a \le x \le b, \qquad u(a) = u_a, \quad u(b) = u_b,$$

where p, q are given functions of x and the boundary values u_a and u_a are given constants.

A finite difference method for this problem is constructed by the following steps:

Step 1: Given the interval $[a, b]$. Choose N, let $h = (b-a)/N$ and let $x_i = a + ih$, $i = 0, 1, \ldots, N$.

Step 2: For each inner grid point x_i, $i = 1, \ldots, N-1$, replace the derivatives by their approximations in the BVP. The result is:

$$\frac{u(x_i + h) - 2u(x_i) + u(x_i - h)}{h^2} + p(x_i)\frac{u(x_i + h) - u(x_i - h)}{2h} + q(x_i)u(x_i) + \mathcal{O}(h^2) = r(x_i)$$

for each $i = 1, 2 \ldots, N-1$, and the term $\mathcal{O}(h^2)$ represents the errors in the difference formulas.

Step 3: Ignore the error term, and replace the exact solution $u(x_i)$ by its numerical (and still unknown) approximation U_i:

$$\frac{U_{i+1} - 2U_i + U_{i-1}}{h^2} + p(x_i)\frac{U_{i+1} - U_{i-1}}{2h} + q(x_i)U_i = r(x_i), \qquad i = 1, \ldots, N-1.$$

This is the *discretization* of the BVP. If we know include the two boundary values as equations, the discretization is a linear system of equations

$$A\mathbf{U} = \mathbf{b},$$

where A er en $N + 1 \times N + 1$ matrix and $\mathbf{U} = [U_0, \ldots, U_N]^T$. Or more specific, by multiplying the equations by h^2 we end up with:

$$A = \begin{bmatrix} 1 & 0 & & & & & \\ v_1 & d_1 & w_1 & & & & \\ & v_2 & d_2 & w_2 & & & \\ & & v_3 & \ddots & \ddots & & \\ & & & \ddots & \ddots & w_{N-2} & \\ & & & & v_{N-1} & d_{N-1} & w_{N-1} \\ & & & & & 0 & 1 \end{bmatrix} \qquad \text{with} \qquad \begin{array}{l} v_i = 1 - \dfrac{h}{2}p(x_i) \\[2mm] d_i = -2 + h^2 q(x_i) \\[2mm] w_i = 1 + \dfrac{h}{2}p(x_i) \end{array} .$$

The right hand side $\mathbf{b}$ is given by

$$\mathbf{b} = [u_a, h^2 r(x_1), \ldots, h^2 r(x_{N-1}), u_b]^T.$$

Obviously, the first and last equations are trivial to solve, and is therefore often included in the right hand side.

Step 4: Solve $A\mathbf{U} = \mathbf{b}$ with respect to $\mathbf{U}$.

Example 1: Given the equation

$$u'' + 2u' - 3u = 9x, \qquad u(0) = u_a = 1, \quad u(1) = u_b = e^{-3} + 2e - 5 = 0.486351,$$

with exact solution $u(x) = e^{-3x} + 2e^x - 3x - 2$.

Choose N, let $h = 1/N$. Use the central differences for u' and u'', such that

$$\frac{u(x_i + h) - 2u(x_i) + u(x_i - h)}{h^2} + 2\frac{u(x_i + h) - u(x_i - h)}{2h} - 3u(x_i) + \mathcal{O}(h^2) = 9x_i, \qquad i = 1, \ldots, N$$

Let $U_i \approx u(x_i)$. Multiply by h^2 on both sides, include $U_0 = u_a$ og $U_N = u_b$ and clean the mess:

$$U_0 = 1$$
$$(1 - h)U_{i-1} + (-2 - 3h^2)U_i + (1 + h)U_{i+1} = 9x_i h^2, \qquad i = 1, \cdots N - 1,$$
$$U_N = 0.486351$$

To be even more concrete, for $N = 4$, we get $h = 0.25$. The linear system of equations becomes

$$\begin{pmatrix} 1 & 0 & 0 & 0 & 0 \\ 0.75 & -2.1875 & 1.25 & 0 & 0 \\ 0 & 0.75 & -2.1875 & 1.25 & 0 \\ 0 & 0 & 0.75 & -2.1875 & 1.25 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix} = \begin{pmatrix} 1. \\ 0.140625 \\ 0.28125 \\ 0.421875 \\ 0.48635073 \end{pmatrix}.$$

The first and the last equation is trivial to solve, so in practice you have a system of 3 equations in 3 unknowns,

$$\begin{pmatrix} -2.1875 & 1.25 & 0 \\ 0.75 & -2.1875 & 1.25 \\ 0 & 0.75 & -2.1875 \end{pmatrix} \begin{pmatrix} U_1 \\ U_2 \\ U_3 \end{pmatrix} = \begin{pmatrix} 0.140625 - 0.75 \cdot 1 \\ 0.28125 \\ 0.421875 - 1.25 \cdot 0.48635073 \end{pmatrix},$$

with the solution
$$U_1 = 0.293176, \qquad U_2 = 0.025557, \qquad 0.093820.$$

For comparison, the exact solution in these points are:

$$u(0.25) = 0.290417, \qquad u(0.5) = 0.020573, \qquad u(0.75) = 0.089400.$$

3.1 Implementation

For simplicity, the implementation below is only done for BVPs with constant coefficients, that is $p(x) = p$ and $q(x) = q$. This makes the diagonal, sub- and super-diagonals constant, except at the first and the last row. An extra function is included to construct matrices of the form $A = \text{tridiag}\{v, d, w\}$.

The implementation consist of

1. Choose N, let $h = (b - a)/N$ and $x_i = a + ih$, $i = 0, \ldots, N$.

2. Construct the matrix $A \in \mathbb{R}^{N+1 \times N+1}$ and the vector $b \in \mathbb{R}^{N+1}$. The matrix A is tridiagonal, and except from the first and last row, has the elements $v = 1 - \frac{h}{2}p$ below the diagonal, $d = -2 + h^2 q$ as diagonal elements and $w = 1 + \frac{h}{2}p$ above the diagonal.

3. Construct the vector $\mathbf{b} = [b_0, \ldots, b_N]^T$ with elements $b_i = h^2 r(x_i)$ for $i = 1, \ldots, N-1$.

4. Modify the first and the last row of the matrix A and the first and last element of the vector $\mathbf{b}$, depending on the boundary conditions.

5. Solve the system $A\mathbf{U} = \mathbf{b}$.

3.2 Boundary conditions

To get a unique solution of a BVP (or a PDE), some information about the solution, usually given on the boundaries has to be known. The most common boundary conditions are:

1. Dirichlet condition: The solution is known at the boundary.

2. Neumann condition: The derivative is known at the boundary.

3. Robin (or mixed) condition: A combination of those.

In the example above, Dirichlet conditions were used. We will now see how to handle Neumann conditions. Robin conditions can be treated similarly.

Given the BVP with a Neumann condition at the left boundary:

$$u'' + p(x)u' + q(x)u = r(x), \qquad a \leq x \leq b, \qquad u'(a) = u'_a, \quad u(b) = u_b.$$

Here, u'_a is some given value. In this case, the solution $u(a)$ and its corresponding approximation U_0 are unknown, and we need some difference formula also for the point $a = x_0$. The simplest option is to use a forward difference

$$u'_a = \frac{u(x_1) - u(x_0)}{h} + \mathcal{O}(h) \qquad \Rightarrow \qquad \frac{U_1 - U_0}{h} = u'_a$$

but this is only a first order approximation, and thus lower accuracy is to be expected. We could also use a second order approximation using the values in the grid points x_0, x_1 and x_2, but this will ruin the nice tridiagonal structure of the coefficient matrix. Instead, use the idea of a *false boundary*:

Assume that the solution can be stretched outside the boundary $x = a$, all the way to a fictitious grid point $x_{-1} = a - h$, where we also assume there is an approximate and equally fictitious approximation U_{-1} to $u(x_{-1})$. Then we have two difference formulas in the point a, one for the BVP itself and a central difference for the boundary conditions:

$$\frac{U_1 - 2U_0 + U_{-1}}{h^2} + p(x_0)\frac{U_1 - U_{-1}}{2h} + q(x_0)U_0 = r(x_0)$$
$$\frac{U_1 - U_{-1}}{2h} = u_a'$$

Solve the second equation with respect to U_{-1}, insert this into the first equation which then becomes:

$$\frac{2U_1 - 2U_0 - 2hu_a'}{h^2} + p(x_0)u_a' + q(x_0)U_0 = r(x_0).$$

So the only thing that has changed is the first equation. And since central differences have been used both for the BVP and the boundary condition, the overall order of this approximation can be proved to be 2.

Example 2: Given the same example as before, but now with a Neumann condition at the left boundary:

$$u'' + 2u' - 3u = 9x, \qquad u'(0) = u_a' = -4, \quad u(1) = u_b = -2e^{-3} + e - 5 = 0.48635073,$$

with exact solution $u(x) = e^{-3x} - 2e^x - 3x - 2$.

The modified difference equation at the boundary $x_0 = 0$ is:

$$\frac{2U_1 - 2U_0 - 2u_a'h}{h^2} + 2u_a' - 3U_0 = 0.$$

Multiply this equation by h^2, and include the equation as the discretization for the grid point x_0.

$$(-2 - 3h^2)U_0 - 2U_1 = (2h - 2h^2)u_a'$$
$$(1-h)U_{i-1} + (-2-3h^2)U_i + (1+h)U_{i+1} = 9h^2 x_i, \qquad\qquad i = 1, \cdots N-1.$$
$$U_N = u_b,$$

which, for $N = 4$ og $h = 0.25$ becomes:

$$\begin{pmatrix} -2.1875 & 2 & 0 & 0 & 0 \\ 0.75 & -2.1875 & 1.25 & 0 & 0 \\ 0 & 0.75 & -2.1875 & 1.25 & 0 \\ 0 & 0 & 0.75 & -2.1875 & 1.25 \\ 0 & 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} U_0 \\ U_1 \\ U_2 \\ U_3 \\ U_4 \end{pmatrix} = \begin{pmatrix} -1.5 \\ 0.140625 \\ 0.28125 \\ 0.421875 \\ 0.48635073 \end{pmatrix}.$$

The solution of this is

$$U_0 = 0.92103219, \quad U_1 = 0.25737896, \quad U_2 = 0.01029386, \qquad U_3 = 0.08858688.$$

Numerical exercises:

1. Modify the code above to solve this problem. Use $N = 4$ to check your solution, but try also $N = 10$ and $N = 20$.

2. Modify the code above to solve the same BVP, but now with the left boundary condition $u'(a) + u(a)/4 = 0$.

4 The heat equation

In this section we will see how to solve the heat equation by finite difference methods. It should however be emphasized that the strategies applies to a lot of time-dependent PDEs, the heat equation is just an example.

Given the equation, well known from the first part of this course:

$$u_t = u_{xx}, \qquad\qquad 0 \le x \le 1$$
$$u(0,t) = g_0(t), \quad u(1,t) = g_1(t), \qquad \text{Boundary conditions}$$
$$u(x,0) = f(x) \qquad\qquad \text{Initial conditions}$$

The equation is solved from $t = 0$ to $t = t_{end}$.

4.1 Semi-discretization

This is a technique which combines the discretization of boundary problems explained above with the techniques for solving ordinary differential equations.

The idea is as follows:

Step 1: Make a discretization of the interval in the x-direction. Choose some M, let $\Delta x = 1/M$ (since the interval is $[0,1]$) and the grid points are $x_i = i\Delta x$, $i = 0, 1, \ldots, M$.

Note that for each grid point x_i the solution $u(x_i, t)$ is a function of t alone.

Step 2: Fix some arbitrary point time t, and make a discretization of the right hand side of the PDE. Using central differences to approximate u_{xx}, this will give

$$\frac{\partial u}{\partial t}(x_i, t) = \frac{u(x_{i+1}, t) - 2u(x_i, t) + u(x_{i-1}, t)}{\Delta x^2} + \mathcal{O}(\Delta x^2).$$

Step 3: Ignore the error term $\mathcal{O}(\Delta x^2)$ and replace $u(x_i, t)$ with the approximation $U_i(t)$ in the formula above. The result is

$$U_i'(t) = \frac{U_{i+1}(t) - 2U_i(t) + U_{i-1}(t)}{\Delta x^2}, \qquad i = 1, 2, \ldots, M-1,$$

where $U_i'(t) = dU_i(t)/dt$. And this, together with the boundary conditions $U_0(t) = g_0(t)$, $U_M(t) = g_1(t)$ and the initial condition $U_i(0) = f(x_i)$, $i = 0, 1, \ldots, M$ forms a well defined system of ordinary differential equations.

This system is usually called a *semi-discretization* of the PDE.

Step 4: Solve the system of ODEs by the method of your preference.

The explicit Euler method with step size Δt applied to these ODEs is:

$$U_i^{n+1} = U_i^n + r\left(U_{i+1}^n - 2U_i^n + U_{i-1}^n \right), \quad i = 1, 2, \ldots, M-1, \qquad \text{where } r = \frac{\Delta t}{\Delta x^2}.$$

So $U_i^n \approx u(x_i, t_n)$ where $t_n = n\Delta t$, and to distinguish between the indices representing space and time, we have used the time indices n as superscripts.

Let us test this algorithm on two examples.

Numerical examples 1:   Solve the heat equation $u_t = u_{xx}$ on the interval $[0, 1]$ with

$$u(x, 0) = \sin(\pi x), \qquad\qquad \text{Initial value}$$
$$g_0(t) = g_1(t) = 0. \qquad\qquad \text{Boundary values}$$

The exact solution of this problem is given by

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

Numerical example 2:   Repeat example 1, but now with the initial values

$$u(x, 0) = \begin{cases} 2x & 0 \le x \le 0.5, \\ 2(1 - x) & 0.5 < x \le 2 - 2x, \end{cases}$$

In this case, the exact solution is not given.

Run the codes below with

1. $M = 4$, $N = 20$.

2. $M = 8$, $N = 40$.

3. $M = 16$, $N = 80$.

Both initial values are already implemented.

4.2   Implementation

The solution is stable for $M = 4$, $N = 20$, and apparently unstable for $M = 16$, $N = 80$. Why?

4.3   Stability analysis

The semi-discretized system

$$\dot{U}_i(t) = \frac{U_{i+1}(t) - 2U_i(t) + U_{i-1}(t)}{\Delta x^2}, \qquad i = 1, 2, \ldots, M - 1, \qquad U_0(t) = g_0(t), \qquad U_M(t) = g_1(t),$$

is a linear ordinary differential equation:

$$\dot{\mathbf{U}} = \frac{1}{\Delta x^2}\big(A\mathbf{U} + \mathbf{g}(t)\big),$$

where

$$\mathbf{U} = \begin{pmatrix} U_1 \\ U_2 \\ \vdots \\ U_{M-1} \end{pmatrix}, \qquad A = \begin{pmatrix} -2 & 1 & & \\ 1 & \ddots & \ddots & \\ & \ddots & \ddots & 1 \\ & & 1 & -2 \end{pmatrix} \qquad \text{and} \qquad \mathbf{g}(t) = \begin{pmatrix} g_0(t) \\ 0 \\ \vdots \\ 0 \\ g_1(t) \end{pmatrix}$$

Stability requirements for such problems were discussed in the note on stiff ordinary differential equation. We proved there that the stability depends on the eigenvalues λ_k of the matrix $\frac{1}{\Delta x^2} A$. For the forward

Euler method, it was proved that the step size has to chosen such that $-2 \leq \Delta t \lambda_k \leq 0$ for all λ_k. Otherwise, the numerical solution will be unstable.

It is possible to prove that the eigenvalues of the matrix A is given by

$$\lambda_k = -4 \sin^2 \left(\frac{k\pi}{M}\right), \qquad k = 1, \cdots, M-1.$$

So all the eigenvalues λ_k of $\frac{1}{\Delta x^2} A$ satisfy

$$-\frac{4}{\Delta x^2} < \lambda_k < 0.$$

The numerical solution is stable if $\Delta t < -2/\lambda_k$ for all k, that means whenever

$$r = \frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}.$$

Exercise: Repeat the two experiments above (for the two different initial values) to justify the bound above. Use $M = 16$, and in each case find the corresponding r and observe from the experiments whether the solution is stable or not.

1. Let $N = 256$.

2. Let $N = 128$.

3. Let $N = 250$.

In the last case, it seems like the metod is unstable for the first initial value, and unstable for the second. Do you have any idea why? (Both solutions will be unstable if integrated over a longer time periode).

Hint: Relate to the Fourier expansion solution of the heat equation from the first part of the course.

4.4 Implicit methods

The semi-discretized system is an example of a stiff ODE, which can only be handled reasonable efficiently by $A(0)$-stable methods, like implicit Euler or the trapezoidal rule, see the note on stiff ODEs.

Implicit Euler. The implicit Euler method for the discritized system $\dot{\mathbf{U}} = \frac{1}{\Delta x^2} \left(A\mathbf{U} + \mathbf{g}(t)\right)$ is given by

$$\mathbf{U}^{n+1} = \mathbf{U}^n + r\,A\mathbf{U}^{n+1} + r\,\mathbf{g}(t_{n+1}), \qquad \text{med} \qquad r = \frac{\Delta t}{\Delta x^2}.$$

where $\mathbf{U}^n = [U_1^n, U_2^n, \ldots, U_{M-1}^n$ and $U_i^n \approx u(x_i, t_n)$.

For each time step, the following system of linear equations has to be solved:

$$(I_{M-1} - r\,A)\mathbf{U}^{n+1} = \mathbf{U}^n + r\,\mathbf{g}(t_{n+1}),$$

where I_{M-1} is the identity matrix of dimension $(M-1) \times (M-1)$.

The error in the gridpoints can be shown to be $\mathcal{O}(\Delta t + \Delta x^2)$.

Crank-Nicolson (trapezoidal rule). The trapezoidal rule applied to the semi-discretized system is often referred to as *Crank-Nicolson's method*. The method is $A(0)$-stable and of order 2 in time, so we can expect better accuracy. The method is written as:

$$\mathbf{U}^{n+1} = \mathbf{U}^n + \frac{\Delta t}{2\Delta x^2} A(\mathbf{U}^{n+1} + \mathbf{U}^n) + \frac{\Delta t}{2\Delta x^2}(\mathbf{g}(t_n) + \mathbf{g}(t_{n+1})).$$

So for each timestep the following system of equations has to be solved with respect to $\mathbf{U}^n$:

$$(I_{M-1} - \frac{r}{2}A)\mathbf{U}^{n+1} = (I_{M-1} + \frac{r}{2}A)\mathbf{U}^n + \frac{r}{2}(\mathbf{g}(t_n) + \mathbf{g}(t_{n+1})), \qquad r = \frac{\Delta t}{\Delta x^2}$$

The error in the gridpoints can be shown to be $\mathcal{O}(\Delta t^2 + \Delta x^2)$.

Implementation. It is possible to solve the system of ODEs directly by the methods developed in the note on stiff ODEs, or by using some other existing ODE solver. For nonlinear problems, this is often advisable (but not always). Mostly for the purpose of demonstration, the implicit Euler method as well as the Crank-Nicolson scheme is implemented directly in the following code.

For each time step, a system of linear equation has to be solved:

$$K\mathbf{U}_{n+1} = \mathbf{b}$$

where

Implicit Euler:

$$K = I_{M-1} - rA, \qquad \mathbf{b} = \mathbf{U}_n + r[g_0(t_{n+1}), 0, \ldots, 0, g_1(t_{n+1})]^T$$

Crank-Nicolson:

$$K = I_{M-1} - \frac{r}{2}A, \qquad \mathbf{b} = (I_{M-1} + \frac{r}{2}A)\mathbf{U}_n + r\left[\frac{1}{2}(g_0(t_n) + g_0(t_{n+1})), 0, \ldots, 0, \frac{1}{2}(g_1(t_n) + g_1(t_{n+1}))\right]^T.$$

The methods can of course be applied to the problems from Numerical examples 1 and 2. But for the fun of it, we now include a problem with nontrivial boundary conditions.

Numerical example 3: Solve the equation

$$u_t = u_{xx}, \qquad u(0,t) = e^{-\pi^2 t}, \quad u(1,t) = -e^{-\pi^2 t}, \quad u(x,0) = \cos(\pi x).$$

up to $t_{end} = 0.2$ by implicit Euler and Crank-Nicolson. Plot the solution and the error. The exact solution is $u(x,t) = e^{-\pi^2 t}\cos(\pi x)$.

Use $N = M$, and $M = 10$ and $M = 100$ (for example). Notice that there are no stability issues, even for r large. Also notice the difference in accuracy for the two methods.