# Lecture 4b  Approximations of diffusions

Mathématiques appliquées (MATH0504-1)

B. Dewals, Ch. Geuzaine

# Learning objectives

Become aware of the existence of stability conditions for the design of numerical schemes

Derive formally the stability condition for the 1D diffusion equation

Understand the general *von Neumann* stability condition, involving the concepts of

- amplification factor
- stable / unstable modes

Discover that numerical schemes may be unstable, conditionally stable or unconditionally stable

# Outline

1. Hands on activity: Matlab computation

2. Derivation of a stability criterion for an explicit discretization of the 1D diffusion wave equation

3. General von Neumann stability condition, and application in practice

4. Crank-Nicolson scheme

LIÈGE université
Sciences Appliquées

# 1 – Hands on activity: Matlab computation

In this section, we use finite differences to compute numerical approximations of the solution of the 1D diffusion equation. We highlight the influence of the choice of the time step (with respect to the grid spacing) on the stability of the computation (Section 8.2 in Strauss, 2008).

# A first attempt to compute a solution of the 1D diffusion equation was a total failure!

According to the maximum principle, the true solution of the diffusion equation remains <u>bounded</u> by the minimum and maximum values in the initial condition (0 and 1 in the example) ...

In contrast, the computational "approximation"

- lead to negative values
- as well as <u>growing</u> values far above the maximum in the initial condition

| | | | | | | | | | | |
|---|---|---|---|---|---|---|---|---|---|---|
| $n=4$ | 1 | −4 | 10 | −16 | 19 | −16 | 10 | −4 | 1 | |
| $n=3$ | 0 | 1 | −3 | 6 | −7 | 6 | −3 | 1 | 0 | |
| $n=2$ | 0 | 0 | 1 | −2 | 3 | −2 | 1 | 0 | 0 | |
| $n=1$ | 0 | 0 | 0 | 1 | −1 | 1 | 0 | 0 | 0 | |
| $n=0$ | 0 | 0 | 0 | 0 | 1 | 0 | 0 | 0 | 0 | $\to x$ |

Hence, the computational result was nowhere near the true solution!

# Let us have another try!

Let's solve again the diffusion problem

$$u_t = u_{xx}, \qquad u(x,0) = \phi(x)$$

using finite differences.

We use a forward difference for $u_t$ and a centered difference for $u_{xx}$.

The difference equation is

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}$$

It has a local truncation error of $O(\Delta t)$ and $O(\Delta x)^2$.

# Let us have another try!

In contrast with the first attempt, we do not specify *now* the choice of the mesh $\Delta t$ and the mesh $\Delta x$.

We introduce the following notation:

$$s = \frac{\Delta t}{(\Delta x)^2}$$

The difference equation

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2}$$

becomes $u_j^{n+1} = s\left(u_{j+1}^n + u_{j-1}^n\right) + (1 - 2s)u_j^n$

# Explicit *vs.* implicit numerical schemes

The scheme

$$u_j^{n+1} = s(u_{j+1}^n + u_{j-1}^n) + (1 - 2s)u_j^n$$

is said to be explicit because the values at time step $n + 1$ are given explicitly in terms of the values at the earlier times.

In contrast, one example of an implicit scheme would write:

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = \frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2}$$

# Explicit *vs.* implicit numerical schemes

A scheme may also be semi-implicit, such as:

$$\frac{u_j^{n+1}-u_j^n}{\Delta t}=(1-\theta)\frac{u_{j+1}^n-2u_j^n+u_{j-1}^n}{(\Delta x)^2}+\theta\frac{u_{j+1}^{n+1}-2u_j^{n+1}+u_{j-1}^{n+1}}{(\Delta x)^2}$$

where $\theta$ is a parameter usually set between $0$ an $1$.

Opting for a (semi-)implicit scheme generally improves the stability of the scheme; but it requires the resolution of (large) systems of algebraic equations.

The value of $\theta$ may be chosen to enhance the scheme accuracy. We will come back to this later.

# Now, use the explicit scheme to solve with Matlab a standard diffusion problem

The explicit scheme writes:

$$u_j^{n+1} = s\left(u_{j+1}^n + u_{j-1}^n\right) + (1 - 2s)u_j^n$$

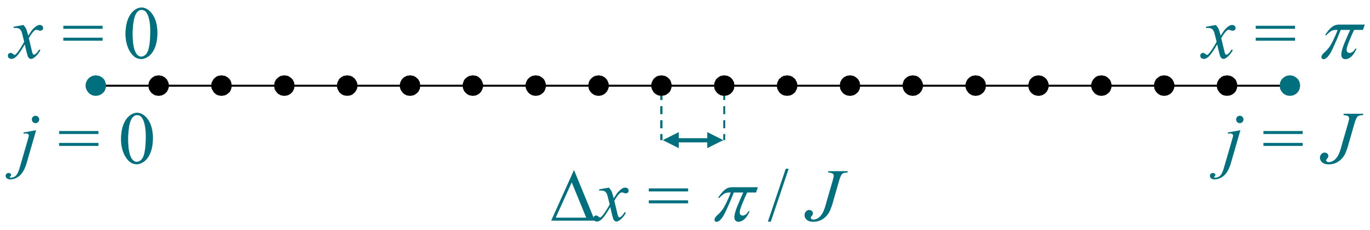
Consider the following standard problem:

$$\begin{aligned} u_t &= u_{xx} && \text{for } 0 < x < \pi, t > 0 \\ u &= 0 && \text{at } x = 0, \pi \end{aligned}$$

$$u(x,0) = \phi(x) = \begin{cases} x & \text{in } \left(0, \frac{\pi}{2}\right) \\ \pi - x & \text{in } \left(\frac{\pi}{2}, \pi\right). \end{cases}$$

# Now, use the explicit scheme to solve with Matlab a standard diffusion problem

First, discretize in space the 1D domain $(0, \pi)$ ,
with a grid of $J + 1$ nodes and a spacing $\Delta x = \pi / J$.




The discrete boundary and initial conditions are

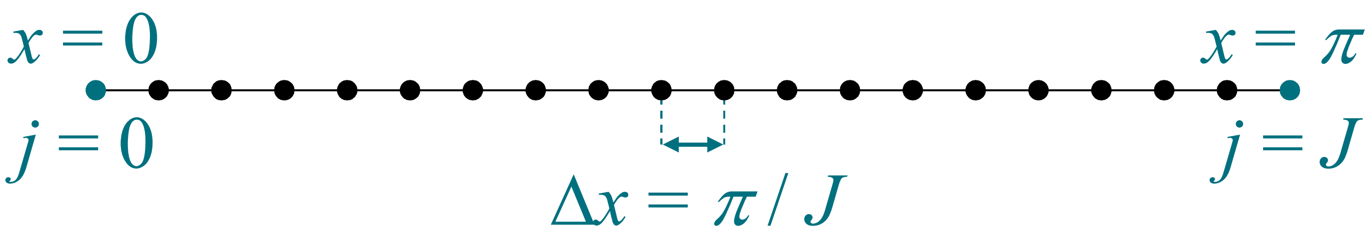
$$u_0^n = u_J^n = 0 \qquad \text{and} \qquad u_j^0 = \phi(j\Delta x)$$

$$\text{with } \phi(x) = \begin{cases} x & \text{in } \left(0, \frac{\pi}{2}\right) \\ \pi - x & \text{in } \left(\frac{\pi}{2}, \pi\right). \end{cases}$$

# It is your turn! Hands on! Launch Matlab and look at how the computation behaves …

① Create the domain

Consider $J = 20$




```
x_start = 0; x_end = pi; J = 20;
dx = ( x_end - x_start ) / J;
X = [x_start:dx:x_end];
```

# It is your turn! Hands on! Launch Matlab and look at how the computation behaves …

② You will compute the solution for $t = \pi^2 / 25$ by considering $s = \Delta t / (\Delta x)^2 = 0.4$.

```
t_end = pi^2 / 25;
s = 0.4;
dt = s * dx^2;
```

③ Initialize a matrix to store the solution:

```
N = uint8(t_end / dt);
U = zeros(N+1,J+1);
```

## It is your turn! Hands on! Launch Matlab and look at how the computation behaves …

④ Prescribe the initial condition …

```
II = find(X<pi/2);
U(1,II) = X(II);
II = find(X>=pi/2);
U(1,II) = pi - X(II);
```

⑤ … and the boundary conditions

```
U(:,1) = 0;
U(:,end) = 0;
```

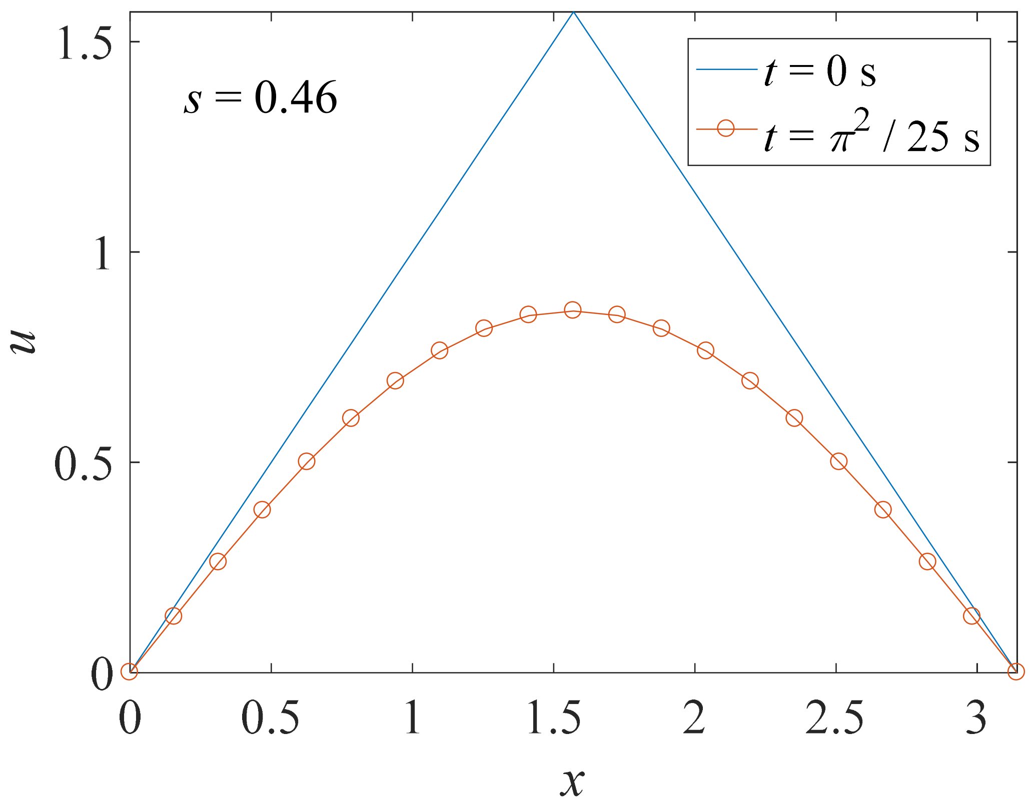
# It is your turn! Hands on! Launch Matlab and look at how the computation behaves …

⑥ Now, you are ready for your first numerical resolution of a PDE:

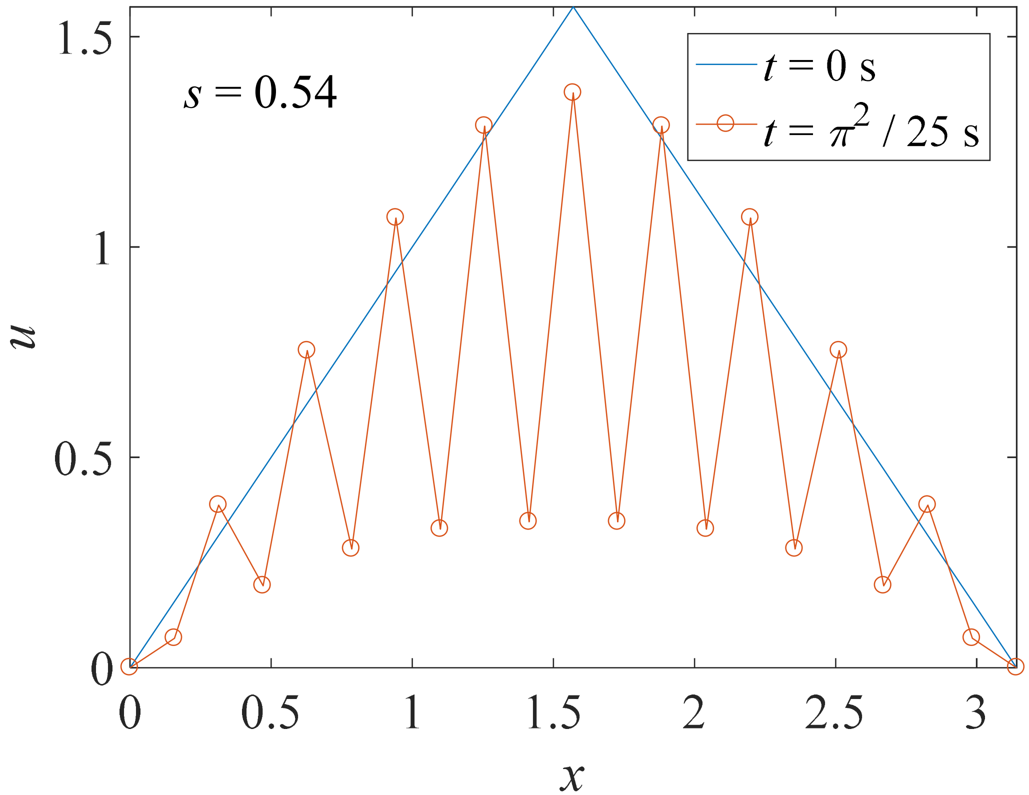
```
for n = 1:N
   U(n+1,2:end-1) …
   = s*(U(n,3:end)+U(n,1:end-2))…
     + (1-2*s) * U(n,2:end-1);
end
```

How does the result look like?

# It is your turn! Hands on! Launch Matlab and look at how the computation behaves …



# It is your turn! Hands on! Launch Matlab and look at how the computation behaves …



# 2 – Derivation of a stability criterion

In this section, we derive a theoretical stability criterion for an explicit discretization of the 1D diffusion wave equation. The theoretical criterion agrees amazingly well with the observations made in the numerical experiments described in the previous section.

# A stability criterion can be formally derived

Heuristically, we find that the computation remains stable provided that $s < 1/2$.

A hint on this can be found in the discretized equation

$$u_j^{n+1} = s\left(u_{j+1}^n + u_{j-1}^n\right) + (1 - 2s)u_j^n$$

where the coefficient $1 - 2\,s$ becomes negative for $s > 1/2$.

Let us demonstrate now the stability condition

$$s < 1/2.$$

We proceed in six simple steps.

## ① Separate the variables in the difference equation

We look for solutions of the difference equation

$$u_j^{n+1} = s(u_{j+1}^n + u_{j-1}^n) + (1 - 2s)u_j^n$$

of the form

$$u_j^n = X_j \, T_n,$$

with $X_j$ a function of space only ($x = j\,\Delta x$), and

$T_n$ a function of time only ($t = n\,\Delta t$).

Substituting in the difference equation, we get

$$X_j \, T_{n+1} = s\left(X_{j+1} \, T_n + X_{j-1} \, T_n\right) + (1-2s) X_j \, T_n$$

# ① Separate the variables in the difference equation

The difference equation may be rewritten as

$$\frac{T_{n+1}}{T_n} = 1 - 2s + s\,\frac{X_{j+1} + X_{j-1}}{X_j} = \xi\left(\cancel{n}, \cancel{j}\right)$$

The left-hand side (LHS) and the right-hand side (RHS) of this equation are functions of independent variables (respectively $n$ and $j$).

Therefore, the equality may hold only if each side is a constant <u>independent </u>of $n$ and $j$.

We note this constant $\xi$.

# ② Solve the time equation

From

$$\frac{T_{n+1}}{T_n} = \xi$$

we get

$$T_n = \xi^n . \boldsymbol{T_0}$$

The factor $\xi$ plays a major part in the assessment of the stability of numerical schemes.

It is called *amplification factor*.

# ③ Solve the spatial equation

The difference equation

$$1-2s+s\frac{X_{j+1}+X_{j-1}}{X_j}=\xi$$

may be rewritten

$$s\left(X_{j+1}-2X_j+X_{j-1}\right)+\left(1-\xi\right)X_j=0$$

This is a discretized form of a second-order ODE, which has sine and cosine solutions.

Therefore, we guess solutions of the form

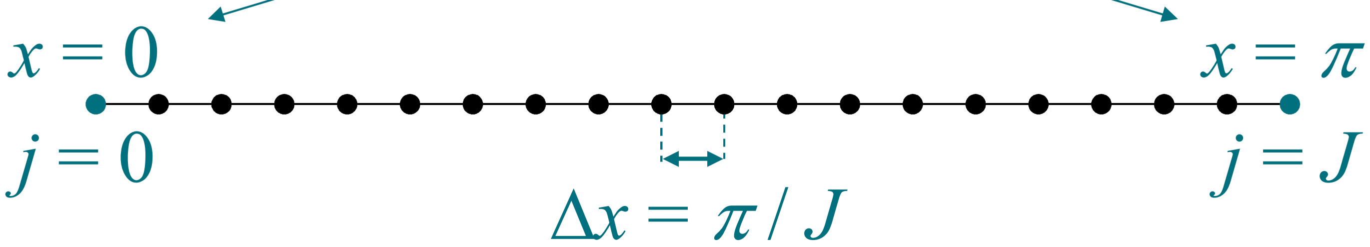
$$X_j=A\cos\theta j+B\sin\theta j$$

with $A$, $B$ arbitrary constants and $\theta$ to be determined.

# ④ Prescribe the boundary conditions

The boundary conditions of the problem were formulated as

$$u_0^n = u_J^n = 0$$




Setting $X_0 = 0$ at $j = 0$ implies that $A = 0$.

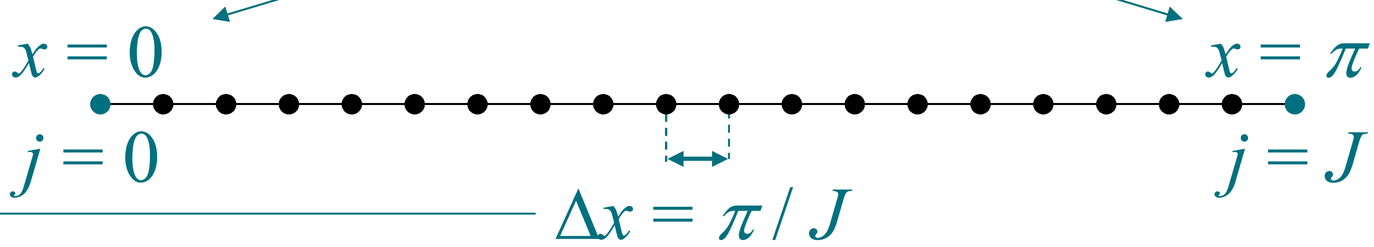
We can freely set $B = 1$, so that $X_j$ becomes

$$X_j = A\cos\theta j + B\sin\theta j \quad \Rightarrow \quad X_j = \sin\theta j$$

# ④ Prescribe the boundary conditions

The boundary conditions of the problem were formulated as

$$u_0^n = u_J^n = 0$$




Setting $X_J = 0$ at $j = J$ implies that $\sin J\theta = 0$.

Thus, $J\theta = k\,\pi$, with $k$ an integer (*wave number*).

Combining with $J = \pi / \Delta x$ leads to $\theta = k\,\pi$, and

$$X_j = \sin\left( jk\Delta x \right)$$

# ⑤ Determine $\xi$ from the spatial equation

Let us substitute $X_j = \sin\,(\,j\;k\;\Delta x\;)$ into the spatial equation

$$1 - 2s + s\,\frac{X_{j+1} + X_{j-1}}{X_j} = \xi$$

It leads to

$$1 - 2s + s\,\frac{\sin\left[\left(j+1\right)k\Delta x\right] + \sin\left[\left(j-1\right)k\Delta x\right]}{\sin\left(jk\Delta x\right)} = \xi$$

or

$$1 - 2s + s\,\frac{2\sin\left(jk\Delta x\right)\cos\left(k\Delta x\right)}{\sin\left(jk\Delta x\right)} = \xi$$

Hence, $\xi = \xi\left(k\right) = 1 - 2s\left[1 - \cos\left(k\Delta x\right)\right]$

# ⑥ Discuss the value of $\xi$ as a function of $s$

We know from the solution of the time equation

$$T_n = \xi^n T_0$$

that $|\xi(k)|$ must remain below $1$; otherwise

- the numerical solution would amplify with time, and we have no chance to get a stable solution
- we have no chance that the numerical solution converges towards the true solution

$$u(x,t) \to 0 \qquad \text{for} \qquad t = n\,\Delta t \to \infty.$$

Therefore let us look at the condition(s) to be prescribed on $s$ so that $|\xi|$ remains below $1$ for all $k$.

## ⑥ Discuss the value of $\xi$ as a function of $s$

Since the factor $1 - \cos(k\Delta x)$ in

$$\xi = \xi(k) = 1 - 2s\left[1 - \cos(k\Delta x)\right]$$

varies between $0$ and $2$, we have

$$1 - 4\,s \leq \xi\,(k) \leq 1.$$

So, stability requires that $1 - 4\,s \geq -\,1$, hence

$$\boxed{s = \frac{\Delta t}{\Delta x^2} \leq \frac{1}{2}}$$

This is the condition required for stability of the computation!

# Which is the most unstable mode?

The analysis above shows that the “most dangerous” mode is the mode for which $\xi(k) = -1$.

From

$$\xi = \xi(k) = 1 - 2s\left[1 - \cos(k\Delta x)\right]$$

we can infer that this “most dangerous” mode corresponds to $\cos(k\,\Delta x) = -1$.

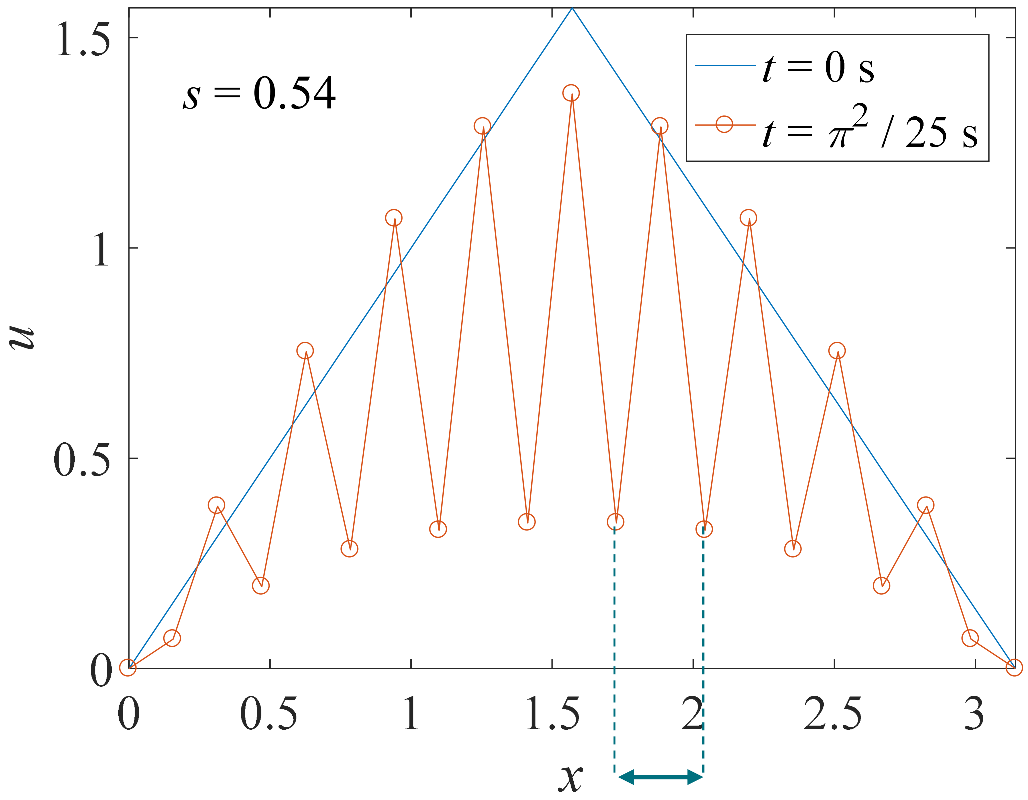
The corresponding wave number $k$ is

$$k = \frac{\pi}{\Delta x}$$

which is a fairly high wave number.

# Which is the most unstable mode?

This theoretical result is, again, fully consistent with the observations in the “failed” computation.




Period of the “most dangerous” mode according to theory

$$= 2\pi / k$$
$$= 2\pi / (\pi / \Delta x)$$
$$= 2\,\Delta x$$

LIÈGE université
Sciences Appliquées

# 3 – Von Neumann stability condition

In this section, we formulate a more general stability condition and describe how to apply it in practice.

# Outline

1. Hands on activity: Matlab computation
2. Derivation of a stability criterion for an explicit discretization of the 1D diffusion wave equation
3. General *von Neumann* stability condition, and application in practice
4. Crank-Nicolson scheme

# General procedure for assessing the stability in a diffusion or a wave problem

The example discussed above suggests the following general procedure:

- separate the variables
- evaluate the amplification factor $\xi(k)$ as a function of the wave number and
  - $\Delta t \,/\, \Delta x^2$ (diffusion)
  - $\Delta t \,/\, \Delta x$ (wave)
- use the so-called *von Neuman stability condition*:

$$|\xi\,(k)| \leq 1 \qquad \text{for all } k.$$

# Practical trick, to be used in the exercises

The result

$$\xi = \xi(k) = 1 - 2s\left[1 - \cos(k\Delta x)\right]$$

could have been obtained very quickly from the difference equation

$$1 - 2s + s\frac{X_{j+1} + X_{j-1}}{X_j} = \xi$$

by simply substituting into the difference equation an exponential mode of the form:

$$X_j = \left(e^{ik\Delta x}\right)^j$$

# 4 – Crank-Nicholson scheme

In this section, we present a different numerical scheme than used so far, and we show that this scheme may be unconditionally stable.

# Implication for engineering

The stability criterion

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

means that in practice the time steps must be taken very short.

Particularly, for the numerical scheme considered so far, $\Delta t$ scales with the square of $\Delta x$!

Let us investigate whether a slightly different scheme may lead to a less restrictive stability condition … or even no stability condition at all.

# Crank-Nicolson scheme

Let us come back to the semi-implicit scheme introduced earlier

$$\frac{u_j^{n+1}-u_j^n}{\Delta t}=(1-\theta)\frac{u_{j+1}^n-2u_j^n+u_{j-1}^n}{(\Delta x)^2}+\theta\frac{u_{j+1}^{n+1}-2u_j^{n+1}+u_{j-1}^{n+1}}{(\Delta x)^2}$$

with $\theta$ a number between $0$ and $1$.

If $\theta = 0$, it reduces to the previous explicit scheme.

Otherwise, it is implicit since $u^{n+1}$ appears on both sides of the equation. This means that, at each time step, a system of linear algebraic equations must be solved.

# Let us analyze the stability of the scheme

As before, we plug the separated solution

$$u_j^n = \left(e^{ik\Delta x}\right)^j \left[\xi(k)\right]^n$$

into the difference equation

$$\frac{u_j^{n+1} - u_j^n}{\Delta t} = (1-\theta)\frac{u_{j+1}^n - 2u_j^n + u_{j-1}^n}{(\Delta x)^2} + \theta\frac{u_{j+1}^{n+1} - 2u_j^{n+1} + u_{j-1}^{n+1}}{(\Delta x)^2}$$

It leads to

$$\frac{\xi - 1}{\Delta t} = (1-\theta)\frac{e^{ik\Delta x} - 2 + e^{-ik\Delta x}}{(\Delta x)^2} + \theta\xi\frac{e^{ik\Delta x} - 2 + e^{-ik\Delta x}}{(\Delta x)^2}$$

$$\xi - 1 = 2(1-\theta)s\left[\cos(k\Delta x) - 1\right] + \xi 2\theta s\left[\cos(k\Delta x) - 1\right]$$

# Let us analyze the stability of the scheme

From

$$\xi - 1 = 2(1-\theta)s\left[\cos(k\Delta x) - 1\right] + \xi 2\theta s\left[\cos(k\Delta x) - 1\right]$$

we get the following expression for $\xi(k)$:

$$\xi = \frac{1 - 2(1-\theta)s\left[1 - \cos(k\Delta x)\right]}{1 + 2\theta s\left[1 - \cos(k\Delta x)\right]}$$

By examining this result, we find out that

- $\xi(k) \le 1$ is always true;
- while $\xi(k) \ge -1$ requires that:

$$s(1-2\theta)\left[1 - \cos(k\Delta x)\right] < 1$$

# Let us analyze the stability of the scheme

If $1 - 2\theta \leq 0$, the condition

$$s(1-2\theta)\left[1-\cos(k\Delta x)\right] < 1$$

is fulfilled whatever the value of $s$

This means that

for $\theta \geq 1/2$, the scheme is **unconditionally stable**.

The particular case of $\theta = 1/2$ is called the Crank-Nicholson scheme. It is second-order accurate in $\Delta t$.

For $\theta < 1/2$, the stability condition writes:

$$s < \frac{1}{2(1-2\theta)}$$

# Take-home messages

We empirically observed a stability criterion, which we also demonstrated theoretically.

In an explicit discretization of the diffusion equation, $\Delta t$ scales with $\Delta x^2$, which may be very restrictive.

Implicit schemes can be unconditionally stable.

The stability of a numerical scheme may be assessed by plugging an exponential mode $u_j^n = \left(e^{ik\Delta x}\right)^j \left[\xi(k)\right]^n$

into the difference equation and checking that the absolute value of the amplification factor $\xi$ remains below 1 for all wave numbers $k$.