Study guide: Analysis of exponential decay models

Hans Petter Langtangen[1,2]

Center for Biomedical Computing, Simula Research Laboratory[1]

Department of Informatics, University of Oslo[2]

Aug 25, 2023

Slides selected/modified by Mikael Mortensen

Recap - Finite differencing of exponential decay

The ordinary differential equation

$$u'(t) = -au(t), \quad u(0) = I, \quad y \in (0, T],$$

where $a > 0$ is a constant.

Solve the ODE by finite difference methods:

- Discretize in time:

$$0 = t_0 < t_1 < t_2 < \cdots < t_{N_t - 1} < t_{N_t} = T$$

- Satisfy the ODE at N_t discrete time steps $n \in [1, 2, \ldots, N_t]$:

$$u'(t_n) = -au(t_n) \quad \text{or} \quad u'(t_{n-\frac{1}{2}}) = -au(t_{n-\frac{1}{2}})$$

Finite difference algorithms

- Discretization by a generic θ-rule

$$\frac{u^n - u^{n-1}}{\triangle t} = -(1-\theta)au^{n-1} - \theta u^n$$

$$\begin{cases} \theta = 0 & \text{Forward Euler} \\ \theta = 1 & \text{Backward Euler} \\ \theta = 1/2 & \text{Crank-Nicolson} \end{cases}$$

Note $u^n = u(t_n)$

- Solve recursively: Set $u^0 = I$ and then

$$u^n = \frac{1 - (1-\theta)a\triangle t}{1 + \theta a\triangle t} u^{n-1} \quad \text{for } n > 0$$

Implicit vs explicit algorithms

- Discretization by a generic θ-rule

$$\frac{u^n - u^{n-1}}{\triangle t} = -(1-\theta)au^{n-1} - \theta u^n$$

$$\begin{cases} \theta = 0 & \text{Forward Euler} \\ \theta = 1 & \text{Backward Euler} \\ \theta = 1/2 & \text{Crank-Nicolson} \end{cases}$$

- Implicit Backward Euler: $-au$ computed from **unknown** u^n
- Explicit Forward Euler: $-au$ computed from **known** u^{n-1}
- Crank-Nicolson is semi-implicit

Analysis of finite difference equations

Model:

$$u'(t) = -au(t), \quad u(0) = I \tag{1}$$

Method:

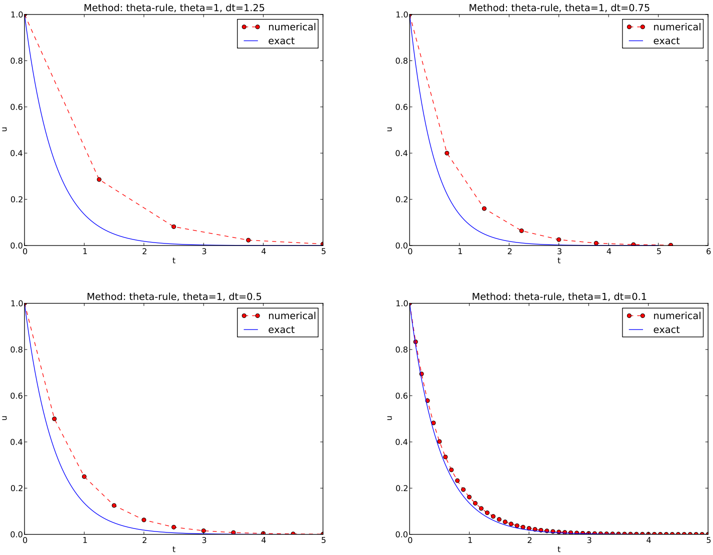
$$u^{n+1} = \frac{1 - (1-\theta)a\Delta t}{1 + \theta a\Delta t} u^n \tag{2}$$

Problem setting

How good is this method? Is it safe to use it?

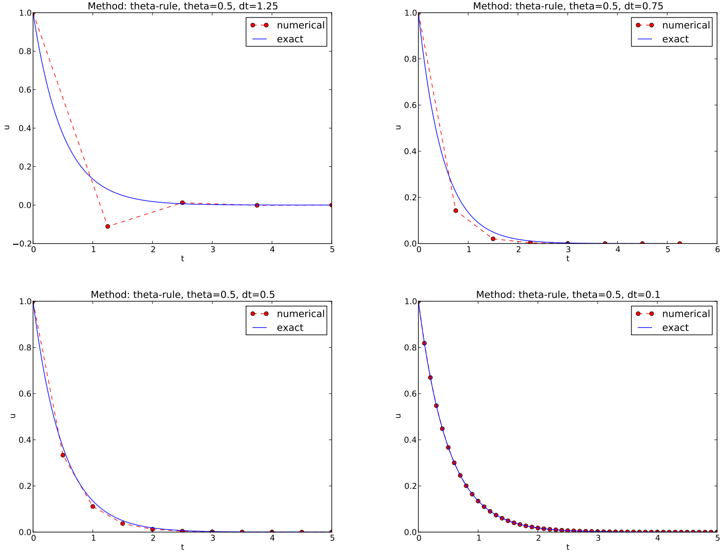
Encouraging numerical solutions; Backward Euler

$I = 1,\ a = 2,\ \theta = 1,\ \Delta t = 1.25, 0.75, 0.5, 0.1.$



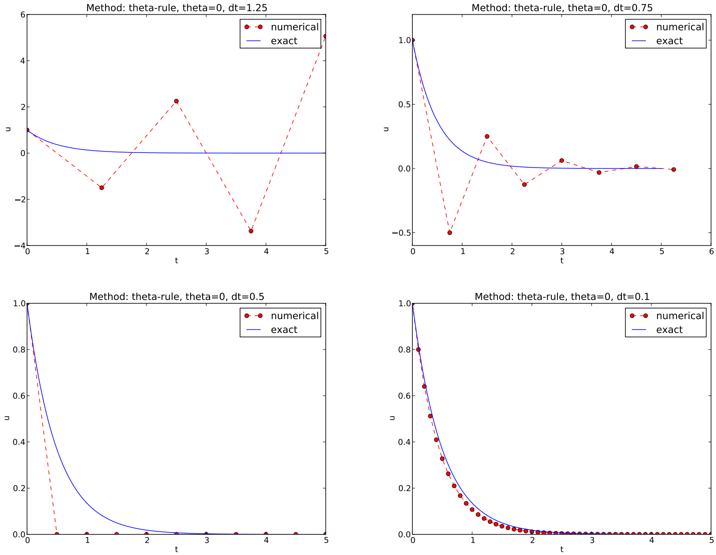
Discouraging numerical solutions; Crank-Nicolson

$I = 1,\ a = 2,\ \theta = 1/2,\ \Delta t = 1.25, 0.75, 0.5, 0.1.$



Discouraging numerical solutions; Forward Euler

$I = 1$, $a = 2$, $\theta = 0$, $\Delta t = 1.25, 0.75, 0.5, 0.1$.



Summary of observations

The characteristics of the displayed curves can be summarized as follows:

- The Backward Euler scheme *always* gives a monotone solution, lying above the exact solution.
- The Crank-Nicolson scheme gives the most accurate results, but for $\Delta t = 1.25$ the solution oscillates.
- The Forward Euler scheme gives a growing, oscillating solution for $\Delta t = 1.25$; a decaying, oscillating solution for $\Delta t = 0.75$; a strange solution $u^n = 0$ for $n \geq 1$ when $\Delta t = 0.5$; and a solution seemingly as accurate as the one by the Backward Euler scheme for $\Delta t = 0.1$, but the curve lies *below* the exact solution.

Problem setting

Goal

We ask the question

- Under what circumstances, i.e., values of the input data I, a, and Δt will the Forward Euler and Crank-Nicolson schemes result in undesired oscillatory solutions?

Techniques of investigation:

- Numerical experiments
- Mathematical analysis

Another question to be raised is

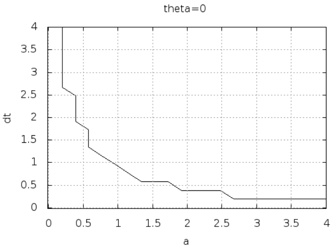
- How does Δt impact the error in the numerical solution?

Experimental investigation of oscillatory solutions

The solution is oscillatory if for some n

$$u^n > u^{n-1}$$

("Safe choices" of Δt lie under following curve as a function of a.)



Seems that $a\Delta t < 1$ for FE and 2 for CN.

Exact numerical solution

Starting with $u^0 = I$, the simple recursion (2) can be applied repeatedly n times, with the result that

$$u^n = IA^n, \quad A = \frac{1-(1-\theta)a\Delta t}{1+\theta a\Delta t} \tag{3}$$

Such a formula for the exact discrete solution is unusual to obtain in practice, but very handy for our analysis here.

Note: An exact dicrete solution fulfills a discrete equation (without round-off errors), whereas an exact solution fulfills the original mathematical equation.

Stability

Since $u^n = IA^n$,

- $A < 0$ gives a factor $(-1)^n$ and oscillatory solutions
- $|A| > 1$ gives growing solutions
- Recall: the exact solution is *monotone* and *decaying*
- If these qualitative properties are not met, we say that the numerical solution is *unstable*

Computation of stability in this problem

$A < 0$ if

$$\frac{1 - (1-\theta)a\Delta t}{1 + \theta a \Delta t} < 0$$

To avoid oscillatory solutions we must have $A > 0$ and

$$\Delta t < \frac{1}{(1-\theta)a} \tag{4}$$

- Always fulfilled for Backward Euler
- $\Delta t \leq 1/a$ for Forward Euler
- $\Delta t \leq 2/a$ for Crank-Nicolson

Computation of stability in this problem

$|A| \leq 1$ means $-1 \leq A \leq 1$

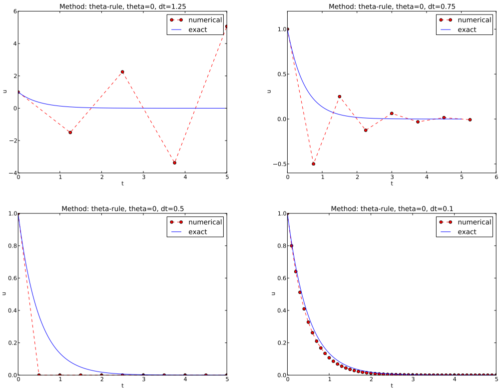
$$-1 \leq \frac{1-(1-\theta)a\Delta t}{1+\theta a\Delta t} \leq 1 \tag{5}$$

-1 is the critical limit (because $A \leq 1$ is always satisfied):

$$\Delta t \leq \frac{2}{(1-2\theta)a}, \quad \text{when } \theta < \frac{1}{2}$$

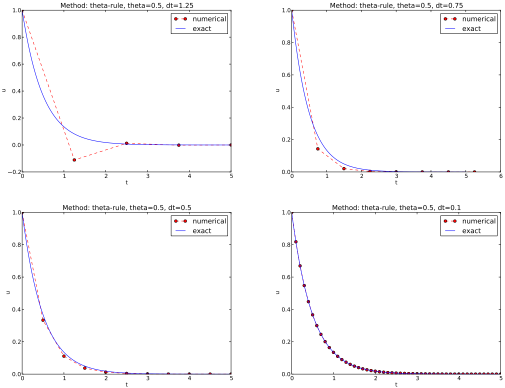
- Always fulfilled for Backward Euler and Crank-Nicolson
- $\Delta t \leq 2/a$ for Forward Euler

Explanation of problems with Forward Euler



- $a\Delta t = 2 \cdot 1.25 = 2.5$ and $A = -1.5$: oscillations and growth
- $a\Delta t = 2 \cdot 0.75 = 1.5$ and $A = -0.5$: oscillations and decay
- $\Delta t = 0.5$ and $A = 0$: $u^n = 0$ for $n > 0$
- Smaller Δt: qualitatively correct solution

Explanation of problems with Crank-Nicolson



- $\Delta t = 1.25$ and $A = -0.25$: oscillatory solution
- Never any growing solution

Summary of stability

1. Forward Euler is *conditionally stable*
   - $\Delta t < 2/a$ for avoiding growth
   - $\Delta t \leq 1/a$ for avoiding oscillations
2. The Crank-Nicolson is *unconditionally stable* wrt growth and conditionally stable wrt oscillations
   - $\Delta t < 2/a$ for avoiding oscillations
3. Backward Euler is unconditionally stable

Comparing amplification factors

u^{n+1} is an amplification A of u^n:

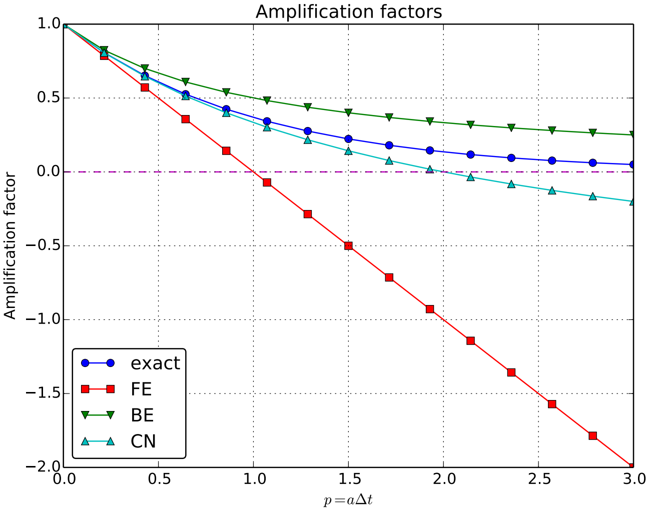
$$u^{n+1} = Au^n, \quad A = \frac{1 - (1-\theta)a\Delta t}{1 + \theta a\Delta t}$$

The exact solution is also an amplification:

$$u(t_{n+1}) = A_{\rm e} u(t_n), \quad A_{\rm e} = e^{-a\Delta t}$$

A possible measure of accuracy: $A_{\rm e} - A$

Plot of amplification factors



$p = a\Delta t$ is the important parameter for numerical performance

- $p = a\Delta t$ is a dimensionless parameter
- all expressions for stability and accuracy involve p
- Note that Δt alone is not so important, it is the combination with a through $p = a\Delta t$ that matters

Another "proof" why $p = a\Delta t$ is key

If we scale the model by $\bar{t} = at$, $\bar{u} = u/I$, we get $d\bar{u}/d\bar{t} = -\bar{u}$, $\bar{u}(0) = 1$ (no physical parameters!). The analysis show that $\Delta\bar{t}$ is key, corresponding to $a\Delta t$ in the unscaled model.

Series expansion of amplification factors

To investigate $A_e - A$ mathematically, we can Taylor expand the expression, using $p = a\Delta t$ as variable.

```
>>> from sympy import *
>>> # Create p as a mathematical symbol with name 'p'
>>> p = Symbol('p')
>>> # Create a mathematical expression with p
>>> A_e = exp(-p)
>>>
>>> # Find the first 6 terms of the Taylor series of A_e
>>> A_e.series(p, 0, 6)
1 + (1/2)*p**2 - p - 1/6*p**3 - 1/120*p**5 + (1/24)*p**4 + O(p**6)

>>> theta = Symbol('theta')
>>> A = (1-(1-theta)*p)/(1+theta*p)
>>> FE = A_e.series(p, 0, 4) - A.subs(theta, 0).series(p, 0, 4)
>>> BE = A_e.series(p, 0, 4) - A.subs(theta, 1).series(p, 0, 4)
>>> half = Rational(1,2)  # exact fraction 1/2
>>> CN = A_e.series(p, 0, 4) - A.subs(theta, half).series(p, 0, 4)
>>> FE
(1/2)*p**2 - 1/6*p**3 + O(p**4)
>>> BE
-1/2*p**2 + (5/6)*p**3 + O(p**4)
>>> CN
(1/12)*p**3 + O(p**4)
```

Error in amplification factors

Focus: the error measure $A - A_e$ as function of Δt (recall that $p = a\Delta t$):

$$A - A_e = \begin{cases} \mathcal{O}(\Delta t^2), & \text{Forward and Backward Euler},\\ \mathcal{O}(\Delta t^3), & \text{Crank-Nicolson} \end{cases} \tag{6}$$

The fraction of numerical and exact amplification factors

Focus: the error measure $1 - A/A_e$ as function of $p = a\Delta t$:

```
>>> FE = 1 - (A.subs(theta, 0)/A_e).series(p, 0, 4)
>>> BE = 1 - (A.subs(theta, 1)/A_e).series(p, 0, 4)
>>> CN = 1 - (A.subs(theta, half)/A_e).series(p, 0, 4)
>>> FE
(1/2)*p**2 + (1/3)*p**3 + O(p**4)
>>> BE
-1/2*p**2 + (1/3)*p**3 + O(p**4)
>>> CN
(1/12)*p**3 + O(p**4)
```

Same leading-order terms as for the error measure $A - A_e$.

The true/global error at a point

- The error in A reflects the *local (amplification) error* when going from one time step to the next
- What is the *global (true) error* at t_n?
  $e^n = u_{\rm e}(t_n) - u^n = Ie^{-at_n} - IA^n$
- Taylor series expansions of e^n simplify the expression

Computing the global error at a point

```
>>> n = Symbol('n')
>>> u_e = exp(-p*n)    # I=1
>>> u_n = A**n         # I=1
>>> FE = u_e.series(p, 0, 4) - u_n.subs(theta, 0).series(p, 0, 4)
>>> BE = u_e.series(p, 0, 4) - u_n.subs(theta, 1).series(p, 0, 4)
>>> CN = u_e.series(p, 0, 4) - u_n.subs(theta, half).series(p, 0, 4)
>>> FE
(1/2)*n*p**2 - 1/2*n**2*p**3 + (1/3)*n*p**3 + O(p**4)
>>> BE
(1/2)*n**2*p**3 - 1/2*n*p**2 + (1/3)*n*p**3 + O(p**4)
>>> CN
(1/12)*n*p**3 + O(p**4)
```

Substitute n by $t/\Delta t$:

- Forward and Backward Euler: leading order term $\frac{1}{2}ta^2\Delta t$
- Crank-Nicolson: leading order term $\frac{1}{12}ta^3\Delta t^2$

Convergence

The numerical scheme is convergent if the global error $e^n \to 0$ as $\Delta t \to 0$. If the error has a leading order term Δt^r, the convergence rate is of order r.

Integrated errors

Focus: norm of the numerical error

$$||e^n||_{\ell^2} = \sqrt{\Delta t \sum_{n=0}^{N_t} (u_e(t_n) - u^n)^2}$$

Forward and Backward Euler:

$$||e^n||_{\ell^2} = \frac{1}{4}\sqrt{\frac{T^3}{3}} a^2 \Delta t$$

Crank-Nicolson:

$$||e^n||_{\ell^2} = \frac{1}{12}\sqrt{\frac{T^3}{3}} a^3 \Delta t^2$$

Summary of errors

Analysis of both the pointwise and the time-integrated true errors:

- 1st order for Forward and Backward Euler
- 2nd order for Crank-Nicolson

Truncation error

- How good is the discrete equation?
- Possible answer: see how well $u_{\rm e}$ fits the discrete equation

$$[D_t^+ u = -au]^n$$

i.e.,

$$\frac{u^{n+1}-u^n}{\Delta t} = -au^n$$

Insert $u_{\rm e}$ (which does not in general fulfill this discrete equation):

$$\frac{u_{\rm e}(t_{n+1})-u_{\rm e}(t_n)}{\Delta t} + au_{\rm e}(t_n) = R^n \neq 0 \qquad (7)$$

Computation of the truncation error

- The residual R^n is the *truncation error*.
- How does R^n vary with Δt?

Tool: Taylor expand u_{e} around the point where the ODE is sampled (here t_n)

$$u_{\mathrm{e}}(t_{n+1}) = u_{\mathrm{e}}(t_n) + u_{\mathrm{e}}'(t_n)\Delta t + \frac{1}{2}u_{\mathrm{e}}''(t_n)\Delta t^2 + \cdots$$

Inserting this Taylor series in (7) gives

$$R^n = u_{\mathrm{e}}'(t_n) + \frac{1}{2}u_{\mathrm{e}}''(t_n)\Delta t + \ldots + au_{\mathrm{e}}(t_n)$$

Now, u_{e} solves the ODE $u_{\mathrm{e}}' = -au_{\mathrm{e}}$, and then

$$R^n \approx \frac{1}{2}u_{\mathrm{e}}''(t_n)\Delta t$$

This is a mathematical expression for the truncation error.

The truncation error for other schemes

Backward Euler:

$$R^n \approx -\frac{1}{2}u_{\mathrm{e}}''(t_n)\Delta t$$

Crank-Nicolson:

$$R^{n+\frac{1}{2}} \approx \frac{1}{24}u_{\mathrm{e}}'''(t_{n+\frac{1}{2}})\Delta t^2$$

Consistency, stability, and convergence

- Truncation error measures the residual in the difference equations. The scheme is *consistent* if the truncation error goes to 0 as $\Delta t \rightarrow 0$. Importance: the difference equations approaches the differential equation as $\Delta t \rightarrow 0$.
- *Stability* means that the numerical solution exhibits the same qualitative properties as the exact solution. Here: monotone, decaying function.
- *Convergence* implies that the true (global) error $e^n = u_e(t_n) - u^n \rightarrow 0$ as $\Delta t \rightarrow 0$. This is really what we want!

The Lax equivalence theorem for *linear* differential equations: consistency + stability is equivalent with convergence.

(Consistency and stability is in most problems much easier to establish than convergence.)