

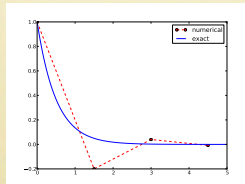
ON SCHEMES FOR EXPONENTIAL DECAY

Hans Petter Langtangen^{1,2}

Center for Biomedical Computing, Simula Research Laboratory¹

Department of Informatics, University of Oslo²

Apr 7, 2015



GOAL

The primary goal of this demo talk is to demonstrate how to write talks with `DocOnce` and get them rendered in numerous HTML formats.

LAYOUT

This version utilizes beamer slides with the theme `vintage`.

PROBLEM SETTING AND METHODS

RESULTS

PROBLEM SETTING AND METHODS



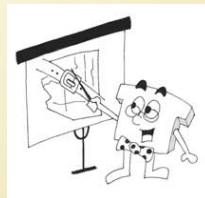
WE AIM TO SOLVE THE (ALMOST) SIMPLEST POSSIBLE DIFFERENTIAL EQUATION PROBLEM

$$u'(t) = -au(t) \quad (1)$$

$$u(0) = I \quad (2)$$

Here,

- $t \in (0, T]$
- a , I , and T are prescribed parameters
- $u(t)$ is the unknown function
- The ODE (1) has the initial condition (2)



THE ODE PROBLEM IS SOLVED BY A FINITE DIFFERENCE SCHEME

- Mesh in time: $0 = t_0 < t_1 \cdots < t_N = T$
- Assume constant $\Delta t = t_n - t_{n-1}$
- u^n : numerical approx to the exact solution at t_n

The θ rule,

$$u^{n+1} = \frac{1 - (1 - \theta)a\Delta t}{1 + \theta a\Delta t} u^n, \quad n = 0, 1, \dots, N - 1$$

contains the Forward Euler ($\theta = 0$), the Backward Euler ($\theta = 1$), and the Crank-Nicolson ($\theta = 0.5$) schemes.

THE ODE PROBLEM IS SOLVED BY A FINITE DIFFERENCE SCHEME

- Mesh in time: $0 = t_0 < t_1 \cdots < t_N = T$
- Assume constant $\Delta t = t_n - t_{n-1}$
- u^n : numerical approx to the exact solution at t_n

The θ rule,

$$u^{n+1} = \frac{1 - (1 - \theta)a\Delta t}{1 + \theta a\Delta t} u^n, \quad n = 0, 1, \dots, N - 1$$

contains the Forward Euler ($\theta = 0$), the Backward Euler ($\theta = 1$), and the Crank-Nicolson ($\theta = 0.5$) schemes.

THE ODE PROBLEM IS SOLVED BY A FINITE DIFFERENCE SCHEME

- Mesh in time: $0 = t_0 < t_1 \cdots < t_N = T$
- Assume constant $\Delta t = t_n - t_{n-1}$
- u^n : numerical approx to the exact solution at t_n

The θ rule,

$$u^{n+1} = \frac{1 - (1 - \theta)a\Delta t}{1 + \theta a\Delta t} u^n, \quad n = 0, 1, \dots, N - 1$$

contains the Forward Euler ($\theta = 0$), the Backward Euler ($\theta = 1$), and the Crank-Nicolson ($\theta = 0.5$) schemes.

THE ODE PROBLEM IS SOLVED BY A FINITE DIFFERENCE SCHEME

- Mesh in time: $0 = t_0 < t_1 \cdots < t_N = T$
- Assume constant $\Delta t = t_n - t_{n-1}$
- u^n : numerical approx to the exact solution at t_n

The θ rule,

$$u^{n+1} = \frac{1 - (1-\theta)a\Delta t}{1 + \theta a\Delta t} u^n, \quad n = 0, 1, \dots, N-1$$

contains the Forward Euler ($\theta = 0$), the Backward Euler ($\theta = 1$), and the Crank-Nicolson ($\theta = 0.5$) schemes.

THE ODE PROBLEM IS SOLVED BY A FINITE DIFFERENCE SCHEME

- Mesh in time: $0 = t_0 < t_1 < \dots < t_N = T$
- Assume constant $\Delta t = t_n - t_{n-1}$
- u^n : numerical approx to the exact solution at t_n

The θ rule,

$$u^{n+1} = \frac{1 - (1 - \theta)a\Delta t}{1 + \theta a\Delta t} u^n, \quad n = 0, 1, \dots, N - 1$$

contains the **Forward Euler** ($\theta = 0$), the **Backward Euler** ($\theta = 1$), and the **Crank-Nicolson** ($\theta = 0.5$) schemes.

THE FORWARD EULER SCHEME EXPLAINED

<http://youtube.com/PtJrPEIHNJw>

IMPLEMENTATION

IMPLEMENTATION IN A PYTHON FUNCTION:

```
def solver(I, a, T, dt, theta):  
    """Solve  $u' = -a*u$ ,  $u(0) = I$ , for  $t$  in  $(0, T]$ ; step:  $dt$ ."""  
    dt = float(dt) # avoid integer division  
    N = int(round(T/dt)) # no of time intervals  
    T = N*dt # adjust T to fit time step dt  
    u = zeros(N+1) # array of  $u[n]$  values  
    t = linspace(0, T, N+1) # time mesh  
  
    u[0] = I # assign initial condition  
    for n in range(0, N): #  $n=0, 1, \dots, N-1$   
        u[n+1] = (1 - (1-theta)*a*dt)/(1 + theta*dt*a)*u[n]  
    return u, t
```

HOW TO USE THE SOLVER FUNCTION

A COMPLETE MAIN PROGRAM

```
# Set problem parameters
I = 1.2
a = 0.2
T = 8
dt = 0.25
theta = 0.5
|\pause|

from solver import solver, exact_solution
u, t = solver(I, a, T, dt, theta)
|\pause|

import matplotlib.pyplot as plt
plt.plot(t, u, t, exact_solution)
plt.legend(['numerical', 'exact'])
plt.show()
```

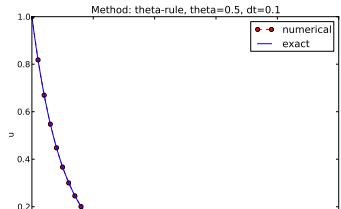
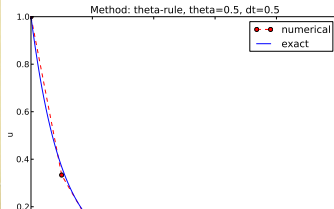
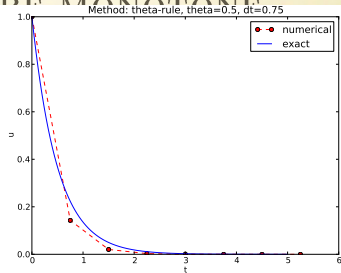
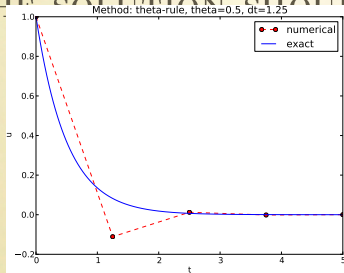
PROBLEM SETTING AND METHODS

RESULTS

RESULTS



THE CRANK-NICOLSON METHOD SHOWS
OSCILLATORY BEHAVIOR FOR NOT
SUFFICIENTLY SMALL TIME STEPS, WHILE
THE SOLUTION SHOULD BE MONOTONE



THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.

THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.

THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.

THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.

THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.

THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.

THE ARTIFACTS CAN BE EXPLAINED BY SOME THEORY

Exact solution of the scheme:

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

Key results:

- Stability: $|A| < 1$
- No oscillations: $A > 0$
- $\Delta t < 1/a$ for Forward Euler ($\theta = 0$)
- $\Delta t < 2/a$ for Crank-Nicolson ($\theta = 1/2$)

CONCLUDING REMARKS:

Only the Backward Euler scheme is guaranteed to always give qualitatively correct results.