Lucerne University of
Applied Sciences and Arts

HOCHSCHULE
LUZERN

Information Technology
FH Zentralschweiz

ODEs - Exercise I

Prof. Dr. Josef F. Bürgler

I.BA_IMATH, Semesterweek 12

The solution of the exercises should be presented in a clear and concise manner. Numerical results should be accurate to 4 digits. The exercises are accepted if You solve 75% of the exercises adequately. Please hand in the exercises no later than at the end of the last lecture in semesterweek 13.

1 Spread of rumors

In a population of size N a rumor is spread by word of mouth. A member of the population knows about the rumor, if another member tells him/her the rumor. Assume $I(t)$ is the number of "*informed members*", i.e. members who know the rumor, at time t. Assuming, that each member meets k members in a time interval and spreads the rumor, the fraction of members not knowing the rumor is

$$q = \frac{N-I}{N}$$

During the time interval dt each informed meetsmeets $k\,q\,\mathrm{d}t$ not informed users, which then are told the rumor. Hence in each time interval dt the number of informed members increases by

$$\mathrm{d}I = I\,k\,q\,\mathrm{d}t \;=\; I\,k\,\frac{N-I}{N}\,\mathrm{d}t.$$

We divide by dt and find the ODE which desribes the spread of a rumor in a population

$$\dot{I}(t) = I(t)\,k\,\frac{N-I(t)}{N} \quad \text{subject to the initial condition} \quad I(0) = I_0.$$

(a) Verify, that

$$I(t) = \frac{I_0 N}{I_0 + (N-I_0)e^{-kt}}$$

satisfies the ODE and its initial condition.

(b) Draw the solution for $k = 5/\text{day}$ and $N = 10^6$ in the range of zero to five days.

Solution: First we verify the initial condition. We have

$$I(0) = \frac{I_0 N}{I_0 + (N-I_0)e^0} \;=\; \frac{I_0 N}{I_0 + (N-I_0)} \;=\; \frac{I_0 N}{N} \;=\; I_0$$

Now let's compute the derivative of $I(t)$

$$\begin{aligned}\dot{I}(t) &= I_0 N\,(-1)\left(I_0 + (N-I_0)e^{-kt}\right)^{-2}(-k)\,(N-I_0)e^{-kt} \\ &= k\,\frac{I_0 N}{I_0+(N-I_0)e^{-kt}}\,\frac{(N-I_0)e^{-kt}}{I_0+(N-I_0)e^{-kt}} \\ &= kI(t)\,\frac{I_0+(N-I_0)e^{-kt}-I_0}{I_0+(N-I_0)e^{-kt}} \\ &= kI(t)\left(1-\frac{I(t)}{N}\right) \\ &= kI(t)\,\frac{N-I(t)}{N}\end{aligned}$$

And this is exactly the ODE.

2 Analytical solution of linear ODE with constant coefficients

Solve the linear differential equation with constant coefficients subject to the initial conditions $u(0) = 1$:

$$u'(x) + ku(x) = e^{-2x} \quad \text{where } k \text{ is a real constants.}$$

Solution: We compute the solution of this ODE in three steps:

1. First compute the general solution of the homogeneuous equation using the Ansatz $u(x) = e^{\lambda x}$. Using $u'(x) = \lambda e^{\lambda x}$ the (homogeneous) ODE

   $$u'(x) + ku(x) = 0$$

   reads

   $$(\lambda + k)e^{\lambda x} = 0 \quad\Longleftrightarrow\quad \lambda + k = 0 \quad\Longleftrightarrow\quad \lambda = -k.$$

   Hence the most general solution is $u_h(x) = c_1 e^{-kx}$ where c_1 is some real constant.

2. Find one particular solution of the inhomogeneous ODE using an Ansatz similar to the perturbation. We use the Ansatz $u_p(x) = \tilde{c}e^{-2x}$ (where $\tilde{c}$ is some real constant to determined later), insert this and its derivative $u_p'(x) = -2\tilde{c}e^{-2x}$ into the (inhomogeneous) ODE and find

   $$u_p'(x) + ku_p(x) \;=\; (-2\tilde{c} + k\tilde{c})e^{-2x} \;=\; e^{-2x} \quad\Longleftrightarrow\quad \tilde{c}\,(-2+k) \;=\; 1 \quad\Longleftrightarrow\quad \tilde{c} = \frac{1}{k-2}.$$

   Hence $u_p(x) = \dfrac{1}{k-2}e^{-2x}$.

3. The most general solution of the inhomogeneous ODE is the sum of the two, i.e.

   $$u(x) = u_h(x) + u_p(x) \;=\; c_1 e^{-kx} + \frac{1}{k-2}e^{-2x}.$$

   Note: the constant c_1 will be determined in the sequel using the initial condition $u(0) = 1$.

We would like to compute a particular solution for the initial condition $u(0) = 1$. In order to do so, we require

$$1 = u(0) \;=\; c_1 e^{-k0} - \frac{1}{k-2}e^{-2\cdot 0} \;=\; c_1 - \frac{1}{k-2} \quad\Longleftrightarrow\quad c_1 \;=\; 1 + \frac{1}{k-2} \;=\; \frac{k-3}{k-2}.$$

Hence the particular solution of the initial value problem is

$$u(x) = \frac{k-3}{k-2}e^{-kx} + \frac{1}{k-2}e^{-2x}. \;=\; \frac{1}{k-2}\left((k-3)e^{-kx} + e^{-2x}\right).$$

It is easy to check, that $u(0) = 1$, and with

$$u'(x) = \frac{1}{k-2}\left((k-3)ke^{-kx} - 2e^{-2x}\right).$$

we find

$$\begin{aligned} u'(x) + ku(x) &= \frac{1}{k-2}\left((k-3)(-k)e^{-kx} - 2e^{-2x}\right) + \frac{1}{k-2}\left((k-3)ke^{-kx} + ke^{-2x}\right) \\ &= \frac{1}{k-2}\,(k-2)\,e^{-2x} \;=\; e^{-2x} \end{aligned}$$

which shows, that $u(x)$ is indeed a solution of the ODE satisfying the initial conditions.

3 Numerical solution using the Euler methods

Adapt the above Matlab/Octave-files to compare the exact solution $u(t) = e^{-t}$ of the model problem $u' = -u$ with the numerical solution for step sizes $h = 0.1, 0.5, 1$ in the interval $[0, 2]$. The plot should contain the graph of the exact solution as well as the two (piecewise linear) numerical approximations from forward and backward Euler.

- As a first step compute the formula, that allows to compute u_{n+1} from u_n.
- Next, adapt the Matlab/Octave-procedure `ForwardEuler` appropriately!
- Finally do the computation, produce the plot and compare with the forward Euler method.

Solution: Backward Euler reads

$$u_{n+1} = u_n + hf(t_{n+1}, u_{n+1}).$$

For the model problem, $u'' = -u$, i.e. $f(t,u) = -u$ and therefore $f(t_{n+1}, u_{n+1}) = -u_{n+1}$, this simplifies to

$$u_{n+1} = u_n - hu_{n+1}$$

which can be solved for u_{n+1}:

$$u_{n+1} = \frac{1}{1+h}u_n.$$

Therefore the following M-file (`BackwardEulerModelProblem.m`) implements the backward Euler method

```
function [t,u] = BackwardEulerModelProblem(tRange, u0, N)
% Use the backward Euler method to solve the model
% problem u' = -u subject to initial conditions u(0)=u0
% tRange = [t0,t1], where the solution will be computed,
% therefore t0 <= t <= t1. Also
% u0 = column vector of initial values for u at t0
% N  = number of equally-sized steps from t0 to t1
% t  = row vector of values of t
% u  = matrix whose n-th row is the approx. solution at t(n).

t      = zeros(N+1,1);            % initialize t
t(1)   = tRange(1);
h      = (tRange(2)-tRange(1))/N;
u(:,1) = u0;                     % u0
for n = 1 : N
  t(n+1)   = t(n) + h;
  u(:,n+1) = 1/(1 + h) * u(:,n);
end
```

To invoke this M-file with $h = 0.1$ and compare the result with exact solution (see Figure 1) use the following MATLAB-commands:

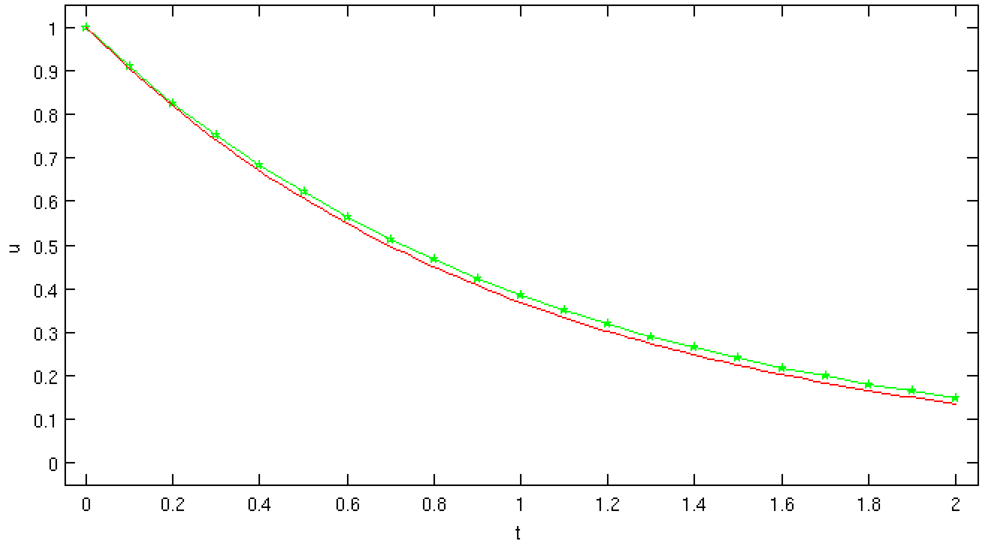

Figure 1: Model problem $u' = -u$ solved with backward Euler and $h = 0.1$ (green) compared with exact solution (red).

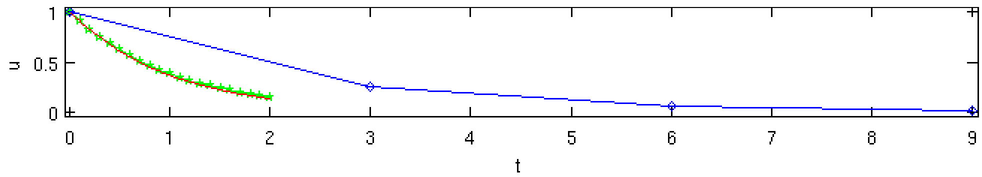

Figure 2: Model problem $u' = -u$ solved with backward Euler and $h = 3$ (blue) compared with part of the exact solution (red).

```
u0 = 1;
N  = 20;
[t, u] = BackwardEulerModelProblem([0, 2], u0, N);
plot(t, u, 'gp-');
hold on;
t = linspace(0,2,100);
plot(t,exp(-t),'r-')
axis equal
axis([-0.05 2.05 -0.05 1.05])
xlabel('t'), ylabel('u')
```

Even if we increase the step size above the limit, which must be imposed in the forward Euler method (where we must have $0 \leq h \leq 2$), the numerical solution is still stable (see Figure 2):

```
u0 = 1;
N  = 3;
[t, u] = BackwardEulerModelProblem([0, 9], u0, N);
plot(t, u, '-b');
```

4 Numerical solution of the rumor problem

Use any method (Euler forward or backward) to solve the rumor problem. Watch the error!

Solution: We try forward Euler

$$I_{n+1} = I_n + hf(t_n, I_n).$$

For our problem

$$\dot{I}(t) = I(t)\,k\,\frac{N - I(t)}{N} \quad \text{subject to the initial condition} \quad I(0) = I_0.$$

where

$$f(t,I) = I(t)\,k\,\frac{N - I(t)}{N}$$

Inserting into the forward Euler method yields

$$I_{n+1} = I_n + hI_n\,k\,\frac{N - I_n}{N} \;=\; I_n\left(1 + h\,k\,\frac{N - I_n}{N}\right).$$

We use the M-file `ForwardEuler.m` (as explained on the slides)

```
function [t,u] = ForwardEuler(f, tRange, u0, N)
% Use the forward Euler method to solve (a system of) 1-st
% order ODE(s) of the form u'=f(u,t), where f = name of an
% m-file which computes "du = f(u,t)" i.e. the RHS of the
% ODE as a row vector. tRange = [t0,t1], where the solution
% will be computed for t0 <= t <= t1. Also
% u0 = column vector of initial values for u at t0
%  N = number of equally-sized steps from t0 to t1
%  t = row vector of values of t
%  u = matrix whose n-th column is the approx. solution at t(n).

t      = zeros(N+1,1);            % initialize t
t(1)   = tRange(1);
h      = (tRange(2)-tRange(1))/N;
u(:,1) = u0;                      % u0
for n = 1 : N
  t(n+1)   = t(n) + h;
  u(:,n+1) = u(:,n) + h * feval(f, t(n), u(:,n));
end
```

To invoke this M-file with $h = 0.1$ and compare the result with exact solution (see Figure 3) use the following MATLAB-commands:

```
Npop   = 1E6;
k      = 5;
tbegin = 0;
tend   = 5;
Nint   = 20;
I0     = 10;
% inline function for f
fun01 = @(t,I) I*k*(Npop-I)/Npop;
% call the forward Euler method
[t, I] = ForwardEuler(fun01, [tbegin, tend], I0, N);
% plot the results and the analytical solution
plot(t, I, 'gp-');
hold on;
t = linspace(tbegin,tend,100);
plot(t,I0*Npop/(I0+(Npop-I0)*exp(-k*t)),'r-')
```

If we use $N = 100$ (i.e. `Nint = 100;`) we get the figure 4. If instead we use Euler Backward or Crank-Nicolson, we get the results shown in figure 5.

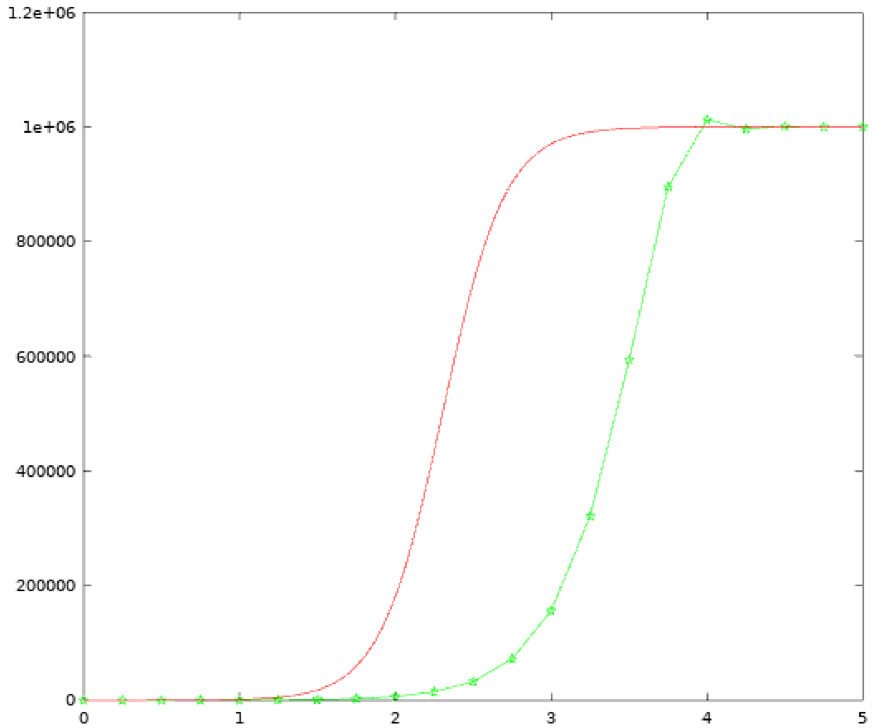

Figure 3: Rumor problem solved with forward Euler and $h = 0.25$ (green) compared with exact solution (red). Notice the huge difference not in the shape but in the time, when the rumor gets viral.

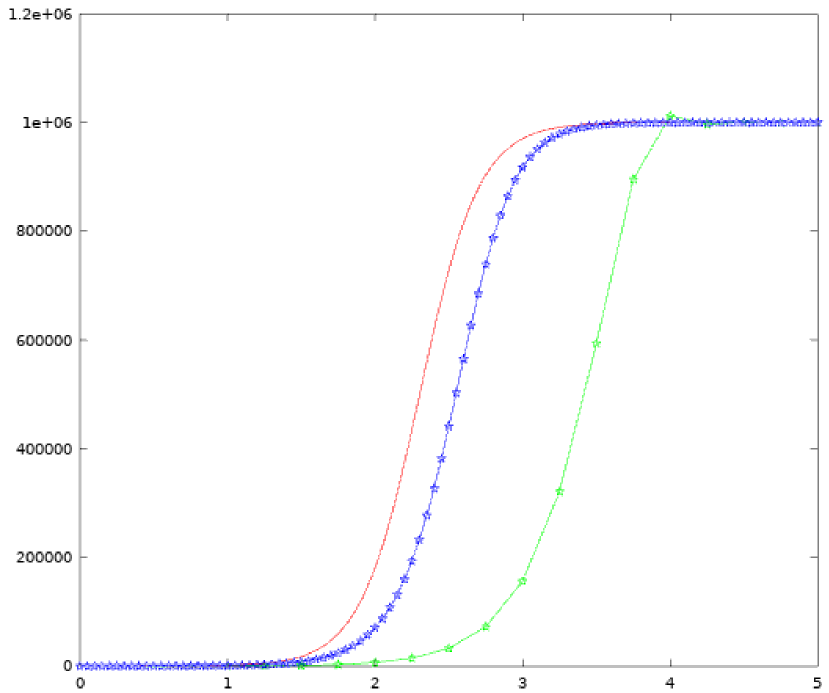

Figure 4: Rumor problem solved with forward Euler and $h = 0.05$ (blue), and $h = 0.25$ (green), compared with exact solution (red). Notice the huge difference not in the shape but in the time, when the rumor gets viral.

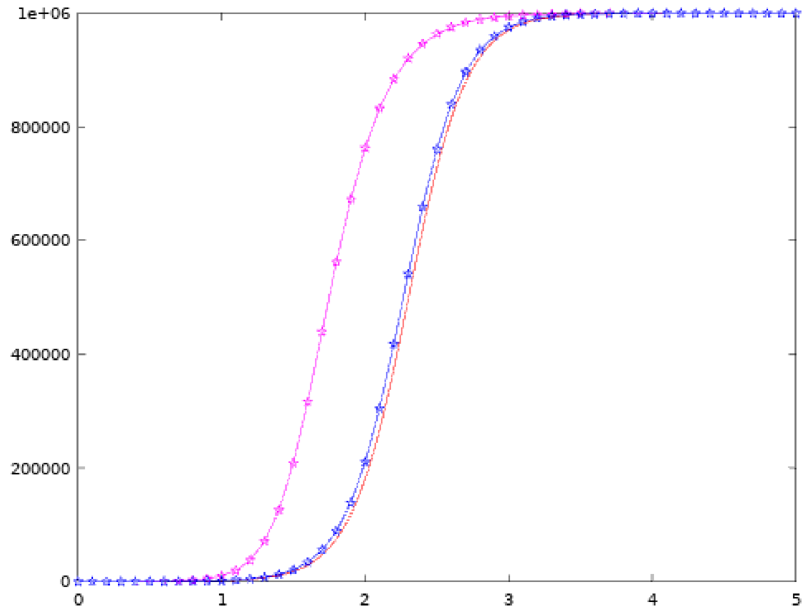

Figure 5: Rumor problem solved with $h = 0.1$ using backward Euler (magenta), and Crank-Nicolson (blue), compared with exact solution (red). Notice, Crank-Nicolson is a second-order method and thus much more accurate then the first-order Euler methods.