MODELLING CELLULAR PROCESSES IN SPACE AND TIME
10-17 OCTOBER 2015
A PRIMER ON NUMERICAL METHODS FOR SOLVING DIFFERENTIAL EQUATIONS

KARIN SASAKI (EMBL CENTRE FOR BIOLOGICAL MODELLING)

Contents

1. Introduction to Euler's method

Ordinary differential equations, like the ones introduced in [3], are usually of the form $dy/dt = f(t, y)$. Stochastic differential equations (SDE), such as the Langevin equation used to model noisy biological systems (see [4] for an introduction) are of the form $dy/dt = f(t, y) + \xi(t)$, where $\xi(t)$ is *random noise*.

Euler's method is a method for numerical integration of equations, such as those above. The principle is as follows: To solve $dy/dt = f(t, y)$, with initial condition $y(t_0) = y_0$, Euler's method instructs us to pick a small increment h, the 'step', of time, and then say that $y_{n+1} = y_n + hf(t_n, y_n)$, using straight-line interpolation between the points $t_0, t_h, t_{2h}, ..., t_{nh}$. Similarly, for the stochastic case, y_{n+1} is approximated, using Euler's method, by $y_n + hf(t_n, y_n) + \xi_{n+1}$.

Under 'suitable' conditions on the function $f(x)$ (or $f(x) + \xi(t)$), as h shrinks, the function obtained converges to the correct solution. (Suitable conditions include making a good guess of the value $x(0)$.)

Let's look at a simple example.

1.1. Example 1: Deterministic exponential decay.

A quantity is subject to exponential decay if it decreases at a rate proportional to its current value. The differential equation for the decay $A \to 0$, with rate λ, is

$$\frac{dA}{dt} = -\lambda A$$

For simplicity, assume $\lambda = 1$. The analytic (exact) solution to this equation is

$$A(t) = A_0 e^{-t}$$

which is illustrated in figure 1(A).

The following MATLAB code shows how to numerically approximate the solution, using Euler's method. The result is illustrated in figure 1(B). (See [5] for an introduction to MATLAB.)

```
*In MATLAB, in a new script file*

% Euler method to solve an ODE of decay, numerically

tmax=5;                          % maximum time to go to
h=0.01;                          % time step
N=ceil(tmax/h);                  % number of steps to take

t=zeros(N,1);                    % prepare a place to store times
A=zeros(N,1);                    % prepare a place to store concentrations

t(1) = 0;                        % initial time
A(1) = 1;                        % initial guess (i.e. x(0) in explanation above)
for i=1:N                        % take N steps
  t(i+1) = t(i) + h;             % update time
  A(i+1) = A(i) + h*(-A(i));     % update A; the function f is f(A) = -A
end;

figure
plot(t,A);
axis([0 tmax (min(A)-2) (max(A)+2)]);   % set axis limits
title('Exponential decay');
grid on
axis([0 5 0 1.2])
```

The following MATLAB code shows how to numerically approximate the solution, using MATLAB's *ode45* function (use the MATLAB help to see how to use it). The result is illustrated in figure 1(C).

```
*In a MATLAB function file called decay_odes.m*

function dAdt = decay_odes(t, A)
% R1: A -> 0 with rate k=1

% mathematical model
dAdt = -1*A;
```

```
*In a new MATLAB script file*

% initial concentration
y0 = 1;

% time of simulation
tspan = [0 5];
```

```
% solve system of ODEs describing decay A -> 0
[Tode, Yode] = ode45(@decay_odes, tspan, y0);

% Plot the decay of A, with time
figure
plot (Tode, Yode);
legend('A');
xlabel('Time');
ylabel('Concentration');
```

1.2. **Example 2: Stochastic exponential decay.** The following MATLAB code shows how to numerically approximate the solution to a Langevin model of decay, using Euler's method. Figure 2 shows the results and compares between different step h sizes.

```
*In a new MATLAB script*

% Euler method to solve the Langevin equation for noisy decay, with noise xi(t) and N steps.
% This script also illustrates the effect of using smaller time steps.

% Note: change xi values 0, 0.2, 0.4 to see effect of noise

xi = 0.2;

figure

tmax = 5;                          % maximum time to go to
M = [5 10 100 1000 10000];

V = length(M);                     % number of plots

for v=1:V
  N = M(v);                        % number of steps
  h = tmax/N;                      % time step

  t = zeros(N,1);                 % prepare a place to store times
  A = zeros(N,1);                 % prepare a place to store number of molecules

  t(1) = 0;                        % initial time
  A(1) = 1;                        % initial location

  for i=1:N                        % take N steps
    t(i+1) = t(i) + h;
    A(i+1) = A(i) - A(i)*h + xi*randn*sqrt(h);  % the function here is f(A) = -A
  end;

  subplot(V,1,v)
        plot(t,A);
  hold on
  plot([0 tmax], [0 0],':');
  if N < 15,
```

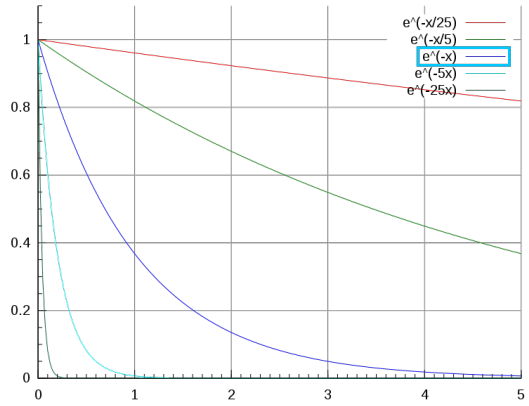
```
  plot(t,A,'.')
  end
  axis([0 tmax (min(A)-1) (max(A)+1)]);   % set axis limits

  title(['Exponential decay to 0 in ' int2str(N) ' steps, h = ' num2str(h) ',
  \xi = ' num2str(xi)]);
end
```

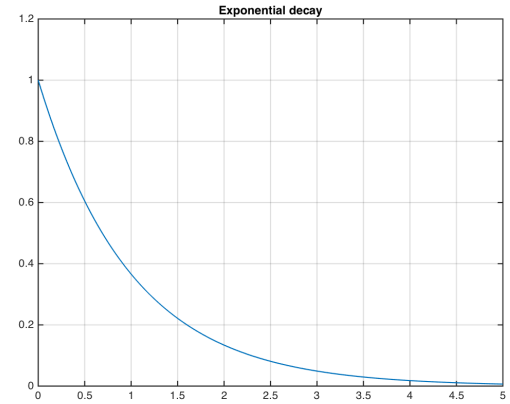
Appendix A - pseudo-code for the Euler method

define $f(y, t)$
input $t0$ and $y0$
input step size, h and the number of steps, n
for j from 1 to n do
  $y1 = y0 + h * m$
  $t1 = t0 + h$
  Print $t1$ and $y1$
  $t0 = t1$
  $y0 = y1$
end

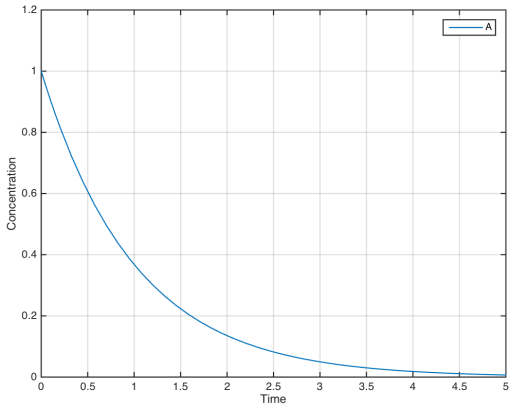
FIGURE 1. (A). Plot of analytic solution to the decay equation (from Wikipedia). (Concentration vs time.) (B). Plot of numerical approximation to the solution of the decay equation, using Euler's method. (Concentration vs time.) (C). Plot of numerical approximation to the solution of the decay equation, using MATLAB's *ode45* function. (Concentration vs time.)



(A)

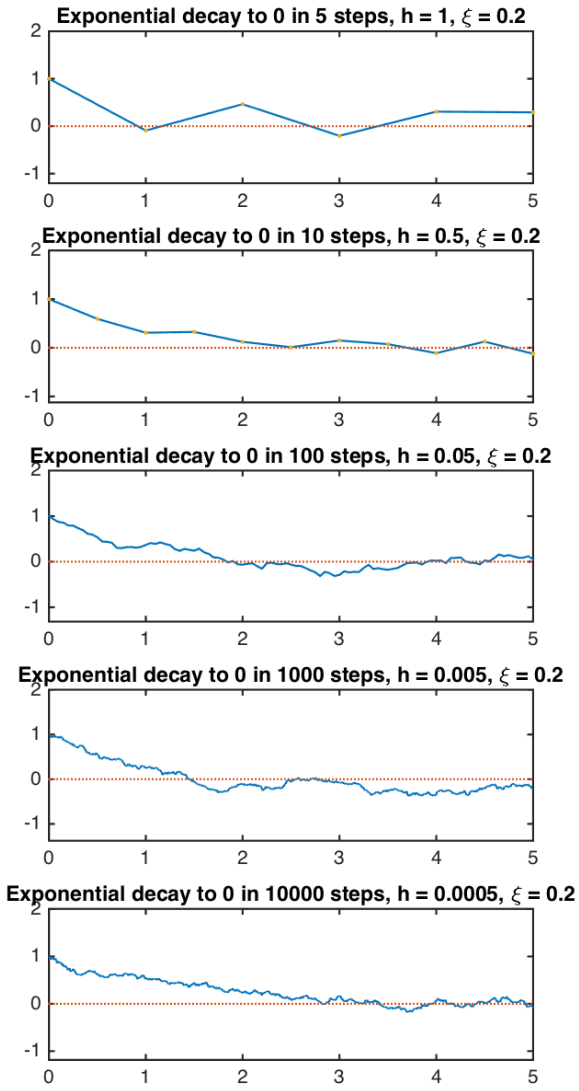


(B)



(C)

FIGURE 2. Plot of numerical approximation to the solution of the decay equation with noise, using Euler's method.



2. References

[1] Dieder Gonze, Stochastic simulations - Application to molecular networks, *Les Houches Spring School*, 2007.
[2] http://www-math.bgsu.edu/ zirbel/sde/matlab/
[3] Karin Sasaki, EMBL-CBM, A primer on deterministic models of biochemical reactions.
[4] Karin Sasaki, EMBL-CBM, A primer on stochastic models of biochemical reactions.
[5] Karin Sasaki, EMBL-CBM, A primer on MATLAB.