Mini-course on Geometric Numerical Integration: Assignments

David Cohen
Umeå University and University of Innsbruck
✉ david.cohen@umu.se

This mini-course is supported by an initiation grant from STINT
The Swedish Foundation for International Cooperation
in Research and Higher Education

February 14, 2016

1 Background: Ordinary differential equations and first numerical schemes

Task 1: You don't feel well after eating in a churrascaria in downtown Rio de Janeiro ... it seems that you have picked up a parasite that grows exponentially fast until treated. Returning home there are 10 of them in you. To model this problem, you should use the **population dynamic model** from the lecture $y' = ay$. The growth parameter for the parasites in your body is $a = 0.1 + 10^{-3}A$, where A is the number corresponding to the last two digits of your year of birth.

a) Open an M-file called *churrasc.m* and write a code in order to plot the exact solution of the problem for the time interval $[0,20]$. The following may be useful

```
%% Code for task churrascaria
clear all
...
p0= ...; % initial number of parasites
tExact=[0:0.05:20]; % time interval
pExact= ...; % exact solution of the ODE
figure(),plot(tExact,pExact, ... ) % plot solution wrt time
xlabel('Time','FontSize',15) % x-axis with nice fonts
legend(...) % legend
title(...) % title
print -depsc2 -r0 taskChurrasc.eps % save the plot in .eps
```

b) Complete your M-file with an implementation of Euler's method. You should plot the numerical approximations given by the method with the step sizes $h = 0.5, 0.25$ and 0.1 on the interval $[0,20]$. Compare with the exact solution. The following may be useful

```
%% Code for task Churrasc
h=0.5;
t0=0;tend=20;
N=...; % compute the number of steps used by Euler's method
...; % initial time and initial value
for n=1:N
  ...; % compute one step of the method
  tEuler1(n)= ...; % store the time for the plot, see below
```

```
  pEuler1(n)= ...; % approx of pExact(t_n) for step size h
  ...; % update the method
end
...
h=0.25; % for the next step
...
% plot of the exact and numerical solutions
tExact=[0:.1:20]; % time interval
pExact= ...; % exact solution
figure(),plot(tExact,pExact,tEuler1,pEuler1, ... )
legend(...) % legend, etc
...
```

Task 2: Consider **Heun's method** (1900)

$$\begin{aligned} Y &:= y_0 + hf(x_0, y_0) \\ y_1 &:= y_0 + \frac{h}{2}\big(f(x_0, y_0) + f(x_0 + h, Y)\big) \end{aligned}$$

or

$$\boxed{y_{k+1} = y_k + \underbrace{\frac{h}{2}\big(f(x_k, y_k) + f(x_k + h, y_k + hf(x_k, y_k))\big)}_{=:h\phi(x_k, y_k, h)},\ k = 0, 1, 2, \ldots.}$$

Show that this numerical method has the following Butcher tableau

$$\begin{array}{c|cc} 0 & & \\ 1 & 1 & \\ \hline & 1/2 & 1/2 \end{array}$$

Show that the order of convergence of this method is $p = 2$.

Hint: Use the fact that $f(x_0 + h, y_0 + hf(x_0, y_0)) = y'(x_0) + hy''(x_0) + \mathcal{O}(h^2), \quad h \to 0.$

2 Geometric Numerical Integration: A taste

Task 1: Show that

$$I(u, v) := \ln(u) - u + 2\ln(v) - v$$

is an **invariant** for the Lotka-Volterra (or predator-prey) problem

$$\begin{aligned} \dot{u} &:= \frac{\mathrm{d}u}{\mathrm{d}t} = u \cdot (v - 2) \\ \dot{v} &:= \frac{\mathrm{d}v}{\mathrm{d}t} = v \cdot (1 - u). \end{aligned} \tag{1}$$

This means that $I(u(t), v(t)) = I(u_0, v_0)$ for all time t along the exact solution of our problem. Here, (u_0, v_0) are the initial values of the above problem.

Hint: Divide the two equations in (1) *formally and use the method of separation of variables to find the invariant* I.

Task 2: The motion of two bodies which attract each other by gravity is described by **Kepler's problem**

$$\begin{aligned} \ddot{q}_1 &= -\frac{q_1}{(q_1^2+q_2^2)^{3/2}} \\ \ddot{q}_2 &= -\frac{q_2}{(q_1^2+q_2^2)^{3/2}}, \end{aligned} \tag{2}$$

where $q(t) := (q_1(t), q_2(t)) \in \mathbb{R}^2$ denotes the position of the second body relatively to the (fixed) first body at time t. This is a Hamiltonian problem with

$$H(p,q) = \frac{1}{2}(p_1^2+p_2^2) - \frac{1}{\sqrt{q_1^2+q_2^2}}$$

and $p_i := \dot{q}_i$ for $i = 1,2$. We consider the initial values

$$q_1(0) = 1-e,\ q_2(0) = 0,\ p_1(0) := \dot{q}_1(0) = 0,\ p_2(0) := \dot{q}_2(0) = \sqrt{\frac{1+e}{1-e}},$$

where $e := 0.6$, then the trajectory is elliptic with eccentricity e and the motion is periodic with period 2π (no need to prove this).

Implement the following numerical schemes (with step size $h = 2\cdot 10^{-3}$) for the Kepler problem. Plot the numerical trajectories in the plane (q_1, q_2) and monitor the evolution of the Hamiltonian $H(p,q)$ and of the angular momentum $L(p,q) := q_1p_2 - q_2p_1$ on the time interval $[0, 40\pi]$ (both quantities are invariant along the exact solution of (2)). Use the following numerical methods:

a) Euler's method

$$\begin{aligned} q_{n+1} &= q_n + h\frac{\partial H}{\partial p}(p_n, q_n) \\ p_{n+1} &= p_n - h\frac{\partial H}{\partial q}(p_n, q_n). \end{aligned}$$

b) The symplectic Euler method

$$\begin{aligned} q_{n+1} &= q_n + h\frac{\partial H}{\partial p}(p_{n+1}, q_n) \\ p_{n+1} &= p_n - h\frac{\partial H}{\partial q}(p_{n+1}, q_n). \end{aligned}$$

c) The implicit midpoint rule.

Open an M-file called *taskKepler.m* in order to discretise numerically Kepler's problem. The following skeleton may be useful.

```
clear all
% Initial values
e=0.6;
q1(1)=1-e; % first component of q
q2(1)=0;   % second component of q
p1(1)=0;
p2(1)=sqrt((1+e)/(1-e));

h=2*10^-3; % stepsize
```

```
t=0:h:40*2*pi; % time interval
N=round(40*2*pi/h)-1;

% Explicit Euler method
for n = 1:N
  % compute p_{n+1},q_{n+1} using Euler's method
  p1(n+1)= ...;
  p2(n+1)= ...;
  q1(n+1)= ...;
  q2(n+1)= ...;
end

% Energy (vector containing all num. energies)
H=1/2*(p1.^2+p2.^2)- ... ;

% Angular momentum
L= ...

% Various plots for Euler's methods
...
% Do the same with symp. Euler and midpoint
...
```

Since the midpoint rule is implicit for the Kepler problem, one has to solve a system of implicit equations for the determination of q_{n+1} and p_{n+1}. This system is of the form $Y = G(Y)$, with $Y := (q_{n+1}, p_{n+1})$, and one can use successive iterations in order to find Y. Basically, starting with an initial guess for Y, denoted $Y^{(0)}$, one iterates $Y^{(n+1)} = G(Y^{(n)})$ until the norm of two successive approximations is below a certain tolerance, say 10^{-6}. The following may be useful

```
% midpoint scheme
for n = 1:N
  % initial guess for p,q
  p1_it=p1(n);p2_it=p2(n);
  ...
  % first iteration
  p1(n+1)=p1(n)-h*((q1(n)+q1_it)/2)/ ...
                (((q1(n)+q1_it)/2)^2+((q2(n)+q2_it)/2)^2)^(3/2);
  ...
  % successive iterations
  while norm([p1(n+1);p2(n+1);q1(n+1);q2(n+1)]- ...
             [p1_it;p2_it;q1_it;q2_it])>10^-6
    p1_it=p1(n+1);p2_it=p2(n+1);
    ...
    % compute new values for p,q using p_it, q_it
    p1(n+1)= ... ;
    ...
  end
end
% Compute H and L and do the plots for implicit midpoint as above
```

3 Numerical integration of Hamiltonian systems

Task 1: Consider Hamiltonian systems of the form

$$\dot{y} = f(y), \quad \text{where} \quad f(y) = J^{-1}\nabla H(y), \tag{3}$$

with $y \in \mathbb{R}^{2d}$ and the skew-symmetric matrix $J = \begin{pmatrix} 0 & I \\ -I & 0 \end{pmatrix}$, where $I \in \mathbb{R}^{d\times d}$ is the identity matrix. Consider the **average vector field method** (AVF)

$$y_{n+1} = y_n + h\int_0^1 f(\theta y_{n+1} + (1-\theta)y_n)\,\mathrm{d}\theta.$$

Using the fact that

$$H(y_{n+1}) - H(y_n) = \int_0^1 \frac{\mathrm{d}}{\mathrm{d}\theta}\big(H(\theta y_{n+1} + (1-\theta)y_n)\big)\,\mathrm{d}\theta,$$

show that the AVF method exactly conserves the energy, i. e. $H(y_{n+1}) = H(y_n)$.

Task 2: Consider the Hamiltonian function $H(y) = \frac{1}{2}p^2 - \cos(q)$, with $y = (p,q) \in \mathbb{R}^2$, in (3) and implement the AVF method using the step size $h = 2^{-6}$ and initial values $q_0 = 1$ and $p_0 = 0.2$. Plot the position of the pendulum q versus the time and monitor the evolution of the total energy on the time interval $[0,50]$. Since the AVF method is an implicit method, you may use the same techniques as described in Kepler's problem. Furthermore, you may use matlab command *quadl* in order to numerically evaluate the integrals present in the AVF method. The following may be useful

```
clear all
% initial values, time, stepsize
Qzero=1.; Pzero=0.2;
t_0=0; t_end=50;
h=2^-6; N=floor((t_end-t_0)./h);
y0=[Pzero;Qzero];

% first initial guess for y
y1t=y0;

for t=1:N
  err=1;
  while err > 10.^(-6)
    % define the functions to integrate
    yint1=@(x)( -sin(y0(2)+x.*(y1t(2)-y0(2))) );
    yint2= ...
    % definition of the AVF method
    y1=y0+[quadl( ... )*h; ... ];
    % error between two consecutive iterations
    err=norm(y1-y1t,2);
    % update of the iterate
    y1t=y1;
  end
  y0=y1;y1t=y0;
  % save the AVF approximations for the plots
  Qavf(t)=y1(2);Pavf(t)=y1(1);
end
% plots
...
```

4 Highly oscillatory problems

Task 1: Show that a trigonometric method (see Definition 4.1 in the lecture notes) satisfying

$$\psi(\zeta) = \sin(\zeta)\psi_1(\zeta)$$
$$\psi_0(\zeta) = \cos(\zeta)\psi_1(\zeta)$$

is symplectic if and only if

$$\psi(\zeta) = \operatorname{sinc}(\zeta)\phi(\zeta) \quad \text{for} \quad \zeta = h\omega.$$

Task 2: Consider the differential equation

$$x''(t) = -\omega^2 x(t)\,, \quad \omega \gg 1 \tag{4}$$

with initial conditions $x(0) = x_0$ and $x'(0) = x_0'$.

(a) Show, that the exact solution of the above problem is given by

$$\begin{pmatrix} \omega x(t) \\ x'(t) \end{pmatrix} = \begin{pmatrix} \cos(\omega t) & \sin(\omega t) \\ -\sin(\omega t) & \cos(\omega t) \end{pmatrix} \begin{pmatrix} \omega x_0 \\ x_0' \end{pmatrix}.$$

(b) Consider the midpoint rule

$$y_{n+1} = y_n + hf\left(\tfrac{y_n + y_{n+1}}{2}\right)$$

for differential equations of the form $y' = f(y)$ as well as the Störmer-Verlet method

$$x'_{n+1/2} = x'_n + \frac{h}{2} g(x_n)$$
$$x_{n+1} = x_n + h x'_{n+1/2}$$
$$x'_{n+1} = x'_{n+1/2} + \frac{h}{2} g(x_{n+1})$$

for differential equations of the form $x'' = g(x)$. Compute one step of each of the above numerical methods applied to the problem (4). Note that for the midpoint rule, the problem needs to be rewritten as a first order system. Analyse the stability of the numerical methods with respect to the time step size h. That is, write the numerical solution as

$$\begin{pmatrix} \omega x_{n+1} \\ x'_{n+1} \end{pmatrix} = M(h\omega) \begin{pmatrix} \omega x_n \\ x'_n \end{pmatrix}$$

and compute the eigenvalues of the matrix $M(h\omega)$. If these eigenvalues have modulus ≤ 1 for all time step sizes h, then the numerical method is stable.

5 GNI for SDEs

Task 1: **Simulation of a Brownian motion**.

(a) Write a Matlab code to simulate one realisation of a discretised Brownian motion $W(t)$ on $[0,1]$. For this, one uses the fact that

$$W(t_\ell) = W(t_{\ell-1}) + \mathrm{d}W_\ell,$$

where each $\mathrm{d}W_\ell$ is an independent random variable of the form $\sqrt{h}N(0,1)$. Consider grid points given by $t_\ell = \ell h$, where $h = 2^{-4}, 2^{-6}$, and 2^{-8}. Plot your numerical results $W(t_\ell)$ for these three different values of h.
You may use the following

```
clear all
randn('state',100) % set the state of randn
% discretised Brownian motion for h=2^(-4)
Tend= ... ;h=2^(-4) ;N= ...;
W(1)= ...; % BM starts at 0 a.s.
for l=...
  dW=sqrt(h)*randn(1,1); % increment/normal rand. var.
  W(l+1)= ... ; % iter. procedure using def. of BM
end
% plot W against time
...
% discretised Brownian motion for h=2^(-6)
...
```

(b) With the help of the above part, compute the mean of $W(t)$ over 200, 2000, 20000 trajectories of $W(t)$ on $[0,1]$ with $h = 2^{-8}$. Plot your results.

```
...
dW=sqrt(h)*randn(M,N+1); % generate M samples of Wiener increments
...
Wmean=mean(...);
...
```

(c) Finally, in the same figure, display 5 sample paths of $W(t)$ and the mean over 50000 trajectories of $W(t)$ on $[0,1]$ for $h = 2^{-8}$.

Task 2: **Scalar stochastic oscillators**.

Consider the system of SDEs on the time interval $[0, T]$:

$$\begin{aligned} \mathrm{d}X(t) &= Y(t)\,\mathrm{d}t \\ \mathrm{d}Y(t) &= -\omega^2 X(t)\,\mathrm{d}t + \alpha\,\mathrm{d}W(t), \end{aligned}$$

where $W(t)$ is a scalar Wiener process, the parameters $\omega \gg 1$ and $\alpha \in \mathbb{R}$, and we denote by X_0 and Y_0 the initial values of the problem. You may choose: $\omega = 5, \alpha = 1, T = 1$ for the above parameters and $X_0 = 0, Y_0 = 1$ for the initial values.

In the lecture, we have seen that the exact solution verifies the following trace formula (i. e. linear drift in the expected energy of the harmonic oscillator)

$$\mathbb{E}\Big[\frac{1}{2}Y(t)^2 + \frac{\omega^2}{2}X(t)^2\Big] = \frac{1}{2}Y_0^2 + \frac{\omega^2}{2}X_0^2 + \frac{\alpha^2}{2}t \quad \text{for all time} \quad t.$$

Compute and plot the linear drift in the expected energy along the numerical solutions given by the Euler-Maruyama scheme

$$\begin{aligned} X_{n+1} &= X_n + hY_n \\ Y_{n+1} &= Y_n - h\omega^2 X_n + \alpha\Delta W_n, \end{aligned}$$

where $\Delta W_n \sim \sqrt{h}N(0,1)$, and the stochastic trigonometric method

$$\begin{aligned} X_{n+1} &= X_n\cos(\omega h) + Y_n\omega^{-1}\sin(\omega h) + \alpha\omega^{-1}\sin(\omega h)\Delta W_n \\ Y_{n+1} &= -X_n\omega\sin(\omega h) + Y_n\cos(\omega h) + \alpha\cos(\omega h)\Delta W_n. \end{aligned}$$

The following code may be helpful:

```
clear all;randn('state',100)

% number of paths samples for
% the approximation of the expectation
M=10^3;

% Initial parameters
omega=5;X0=0;Y0=1;alpha=1;

% initial energy
Haminit= ...

Tend=5;N=2^(5);h=Tend/N;

% initialisation of matrix containing
% numerical energies
Hamem=zeros(M,N);
Hamstm=zeros(M,N);

for s=1:M % loop on the samples
  % Initial values
  Xtempem= ... ;Ytempem= ... ;
  Xtempstm= ...;Ytempstm= ...;
  for j=1:N
    Winc=sqrt(h)*randn(1,N); % Wiener increment

    % Euler-Maruyama
    X1em=Xtempem+h*Ytempem;
    Y1em= ... ;
    Xtempem=X1em;Ytempem= ... ;
    % energy for EM
    Hamem(s,j)= ... ;

    % Stoch. trigon. method
    X1stm=Xtempstm*cos(h*omega)+sin(h*omega)/omega*Ytempstm+ ...
         alpha*sin(h*omega)/omega*Winc;
    Y1stm= ... ;
    ...
   end
end

set(0,'DefaultTextFontSize',12)
set(0,'DefaultAxesFontSize',12)
set(0,'DefaultLineLineWidth',2)
set(0,'DefaultLineMarkerSize',10)

% plots of the expected energy for EM, STM, exact
Dtvals=[h:h:Tend];
figure(),
plot(Dtvals,mean(Hamem),'ks-', ...
Dtvals,mean(Hamstm),'k+-', ...
Dtvals,Haminit+alpha^2.*Dtvals./2,'r'),
axis([Dtvals(1) Dtvals(end) 0 30])
```

```
hold off
xlabel('Time','FontSize',14)
ylabel('Energy','Rotation',0,'FontSize',14)
legend('EM','STM','Exact')
set(gca,'FontSize',15);
```

In the above code, the expectation of a random variable $\mathbb{E}[Z]$ is approximated by the mean $\mathbb{E}[Z] \approx \frac{1}{M}\sum_{s=1}^{M} Z^s$, where Z^s are independent realisations of Z.