Mini-course on Geometric Numerical Integration: Solutions to the 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 18, 2016

This document gives propositions for the solutions to the assignments.

1 Background: Ordinary differential equations and first numerical schemes

Task 1: The code for this task could read

```
%% Code for task 1
clear all
k=0.1+10^(-3)*77; % growth parameter
p0=10; % initial number of parameter
tExact=[0:0.05:20]; % time interval
pExact=p0.*exp(k.*tExact); % exact solution

% plot of the exact solution
figure(),plot(tExact,pExact) % plot solution wrt time
xlabel('Time','FontSize',15) % x-axis
legend('Exact solution') % legend
title('Parasites in my body') % title
%print -djpeg90 task1a.jpg % can be used to save of the plot

% Euler's method for h=0.5
h=0.5;
t0=0;tend=20;
N=tend/h;% compute the number of steps
tE=t0;
pE=p0;
for n=1:N
  % compute one step of the method
  tE=tE+h;
  pE=pE+h*k*pE;
  tEuler1(n)=tE;
  pEuler1(n)=pE; % approx at pExact(t_n) for step size h
end
```

```
% Euler's method for h=0.25
h=0.25;
t0=0;tend=20;
N=tend/h;% compute the number of steps
tE=t0;
pE=p0;
for n=1:N
  % compute one step of the method
  tE=tE+h;
  pE=pE+h*k*pE;
  tEuler2(n)=tE;
  pEuler2(n)=pE; % approx at pExact(t_n) for step size h
end

% Euler's method for h=0.1
h=0.1;
t0=0;tend=20;
N=tend/h;% compute the number of steps
tE=t0;
pE=p0;
for n=1:N
  % compute one step of the method
  tE=tE+h;
  pE=pE+h*k*pE;
  tEuler3(n)=tE;
  pEuler3(n)=pE; % approx at pExact(t_n) for step size h
end
tExact=[t0:h:tend]; % time interval
pExact=p0.*exp(k.*tExact); % exact solution
figure(),plot(tExact,pExact,tEuler1,pEuler1,'.-', ...
          tEuler2,pEuler2,'.-',tEuler3,pEuler3,'.-')
xlabel('Time','FontSize',15) % x-axi
legend('Exact solution','Euler with h=0.5','Euler with h=0.25', ...
    'Euler with h=0.1','Location','NorthWest') % legend
```

Task 2: Heun's method can be rewritten as

$$
\begin{aligned}
k_1 &= f(x_0, y_0) \\
k_2 &= f(x_0 + h, y_0 + hk_1) \\
y_1 &= y_0 + \frac{h}{2}(k_1 + k_2).
\end{aligned}
$$

This is a Runge-Kutta method with $s = 2$ and having the Butcher tableau

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

A Taylor expansion in x followed by one in y gives

$$\begin{aligned}
f(x_0+h, y_0+hf(x_0,y_0)) &= f(x_0, y_0+hf(x_0,y_0)) + \frac{\partial f}{\partial x}(x_0, y_0+hf(x_0,y_0))h + \mathcal{O}(h^2) \\
&= f(x_0,y_0) + h\frac{\partial f}{\partial y}(x_0,y_0)f(x_0,y_0) + h\frac{\partial f}{\partial x}(x_0,y_0) + \mathcal{O}(h^2) \\
&= y'(x_0) + h\Big(\frac{\partial f}{\partial x}(x_0,y_0) + \frac{\partial f}{\partial y}(x_0,y_0)f(x_0,y_0)\Big) + \mathcal{O}(h^2) \\
&= y'(x_0) + hy''(x_0) + \mathcal{O}(h^2).
\end{aligned}$$

Now it follows that

$$\begin{aligned}
y_1 - y(x_0+h) &= y_0 + \frac{h}{2}f(x_0,y_0) + \frac{h}{2}f(x_0+h, y_0+hf(x_0,y_0)) - y_0 - y'(x_0)h - \frac{h^2}{2}y''(x_0) + \mathcal{O}(h^3) \\
&= y_0 + \frac{h}{2}f(x_0,y_0) + \frac{h}{2}(y_0' + hy_0'' + \mathcal{O}(h^2)) - y_0 - y_0'h - y_0''\frac{h^2}{2} + \mathcal{O}(h^3) = \mathcal{O}(h^3).
\end{aligned}$$

This implies that $p=2$ and thus the order of convergence of Heun's method is 2.

2 Geometric Numerical Integration: A taste

Task 1: We have

$$\frac{\mathrm{d}u}{\mathrm{d}v} = \frac{u(v-2)}{v(1-u)}$$

which is equivalent to

$$\frac{1-u}{u}\mathrm{d}u = \frac{v-2}{v}\mathrm{d}v$$

and after integration to

$$\ln(u) - u = v - 2\ln(v) + \text{Const.}$$

or, after rearrangement of the terms,

$$I(u,v) := \ln(u) - u + 2\ln(v) - v = \text{Const.}$$

Task 2: The code could look like this

```
clear all;
% Initial values:
e=0.6;
q1(1)=1-e;
q2(1)=0;
p1(1)=0;
p2(1)=sqrt((1+e)/(1-e));
% Time steps
h=2*10^-3;
t=0:h:40*2*pi;
N=round(40*2*pi/h)-1;

% Explicit Euler
for n = 1:N
  p1(n+1)=p1(n)-h*q1(n)/(q1(n)^2+q2(n)^2)^(3/2);
  p2(n+1)=p2(n)-h*q2(n)/(q1(n)^2+q2(n)^2)^(3/2);
```

```
  q1(n+1)=q1(n)+h*p1(n);
  q2(n+1)=q2(n)+h*p2(n);
end
% Energy:
H=1/2*(p1.^2+p2.^2)-1./sqrt(q1.^2+q2.^2);
% Angular momentum:
L=q1.*p2-q2.*p1;

% Plots
subplot(3,3,1)
plot(q1,q2)
xlabel('q_1(t)');
ylabel('q_2(t)','rotation',0);
title('Explicit Euler');

subplot(3,3,2)
plot(t,H)
axis([0 20*2*pi -0.6 0.6]);
xlabel('t');
ylabel('H','rotation',0);
title('Explicit Euler');

subplot(3,3,3)
plot(t,L)
axis([0 20*2*pi 0.75 1]);
xlabel('t');
ylabel('L','rotation',0);
title('Explicit Euler');

% Midpoint rule
for n=1:N
  p1_it=p1(n);
  p2_it=p2(n);
  q1_it=q1(n);
  q2_it=q2(n);
  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);
  p2(n+1)=p2(n)-h*((q2(n)+q2_it)/2)/ ...
    (((q1(n)+q1_it)/2)^2+((q2(n)+q2_it)/2)^2)^(3/2);
  q1(n+1)=q1(n)+h*(p1(n)+p1_it)/2;
  q2(n+1)=q2(n)+h*(p2(n)+p2_it)/2;

  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);
    q1_it=q1(n+1);
    q2_it=q2(n+1);
    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);
    p2(n+1)=p2(n)-h*((q2(n)+q2_it)/2)/ ...
      (((q1(n)+q1_it)/2)^2+((q2(n)+q2_it)/2)^2)^(3/2);
    q1(n+1)=q1(n)+h*(p1(n)+p1_it)/2;
    q2(n+1)=q2(n)+h*(p2(n)+p2_it)/2;
```

```
    end
end
% Energy+momentum
H=1/2*(p1.^2+p2.^2)-1./sqrt(q1.^2+q2.^2);
L=q1.*p2-q2.*p1;
% Plots
subplot(3,3,4)
plot(q1,q2)
xlabel('q_1(t)');
ylabel('q_2(t)','rotation',0);
title('Midpoint');

subplot(3,3,5)
plot(t,H)
axis([0 20*2*pi -0.6 0.6]);
xlabel('t');
ylabel('H','rotation',0);
title('Midpoint');

subplot(3,3,6)
plot(t,L)
axis([0 20*2*pi 0.75 1]);
xlabel('t');
ylabel('L','rotation',0);
title('Midpoint');

% Symplectic Euler

for n=1:N
    p1(n+1)=p1(n)-h*q1(n)/(q1(n)^2+q2(n)^2)^(3/2);
    p2(n+1)=p2(n)-h*q2(n)/(q1(n)^2+q2(n)^2)^(3/2);
    q1(n+1)=q1(n)+h*p1(n+1);
    q2(n+1)=q2(n)+h*p2(n+1);
end
% Energy+momentum
H=1/2*(p1.^2+p2.^2)-1./sqrt(q1.^2+q2.^2);
L=q1.*p2-q2.*p1;
% Plots
subplot(3,3,7)
plot(q1,q2)
xlabel('q_1(t)');
ylabel('q_2(t)','rotation',0);
title('Symplectic Euler');

subplot(3,3,8)
plot(t,H)
axis([0 20*2*pi -0.6 0.6]);
xlabel('t');
ylabel('H','rotation',0);
title('Symplectic Euler');

subplot(3,3,9)
plot(t,L)
axis([0 20*2*pi 0.75 1]);
```

```
xlabel('t');
ylabel('L','rotation',0);
title('Symplectic Euler');
```

3 Numerical integration of Hamiltonian systems

Task 1: By the fundamental Theorem of calculus, an application of the chain rule, the definition of the AVF scheme, and the skew-symmetry of J^{-1}, we have

$$\begin{aligned} H(y_{n+1}) - H(y_n) &= \int_0^1 \frac{\mathrm{d}}{\mathrm{d}\theta}\Big(H(\theta y_{n+1} + (1-\theta)y_n)\Big)\,\mathrm{d}\theta \\ &= \int_0^1 \nabla H(\theta y_{n+1} + (1-\theta)y_n)^T (y_{n+1} - y_n)\,\mathrm{d}\theta \\ &= \Big(\int_0^1 \nabla H(\theta y_{n+1} + (1-\theta)y_n)\,\mathrm{d}\theta\Big)^T hJ^{-1} \\ &\quad \Big(\int_0^1 \nabla H(\theta y_{n+1} + (1-\theta)y_n)\,\mathrm{d}\theta\Big) = 0. \end{aligned}$$

This implies $H(y_{n+1}) = H(y_n)$ for all n.

Task 2: The code could look like

```
clear all

Qzero=1.;
Pzero=0.2;

t_0=0;
t_end=50;

h=2^-6;
Nt=floor((t_end-t_0)./h);

y0=[Pzero; Qzero];y1t=y0;

for t=1:Nt
  %% AVF
  err=1;
  while err > 10.^(-6)
    yint1=@(x)( -sin(y0(2)+x.*(y1t(2)-y0(2))) );
    yint2=@(x)( (y0(1)+x.*(y1t(1)-y0(1))) );
    y1=y0+[quadl(yint1,0,1)*h;quadl(yint2,0,1)*h];
    err=norm(y1-y1t,2);y1t=y1;
  end
  y0=y1;y1t=y0;
  Qep(t)=y1(2);Pep(t)=y1(1);
  t*h
end

% plot of energy and position Q VS time
kk=1;
ttime=[0:h:t_end];
```

```
time=ttime(1:kk:end);
EnergEP=Pep(1:kk:end).^2./2-cos(Qep(1:kk:end));
figure,
plot(time,[Pzero.^2./2-cos(Qzero) EnergEP],'k')
title('Energy','FontSize',20)
xlabel('Time')
axis([0 t_end -1 1])
figure,
kk=1;
ttime=[0:h:t_end];
time=ttime(1:kk:end);
set(0,'DefaultLineMarkerSize',6)
plot(time,[Qzero Qep(1:kk:end)],'k')
title('Positions','FontSize',20)
xlabel('Time')
axis([0 t_end -2 2])
```

4 Highly oscillatory problems

Task 1: This is done using the definition of symplecticity $\Phi_h'(Y)J\Phi_h'(Y) = J$ and the definition 4.1 of the trigonometric methods $Y_{n+1} = \Phi_h(Y_n)$, see lecture notes.

Task 2: (a) The solution of the linear system

$$\begin{pmatrix} x'(t) \\ x''(t) \end{pmatrix} = \underbrace{\begin{pmatrix} 0 & t \\ -t\omega^2 & 0 \end{pmatrix}}_{tA} \begin{pmatrix} x(t) \\ x'(t) \end{pmatrix}$$

is given by

$$\begin{pmatrix} x(t) \\ x'(t) \end{pmatrix} = \exp(tA) \begin{pmatrix} x_0 \\ x_0' \end{pmatrix}.$$

Since the exponential function is analytic, we compute the series expansion thereof. Computing

$$\begin{aligned}
A^2 &= \begin{pmatrix} -(t\omega)^2 & 0 \\ 0 & -(t\omega)^2 \end{pmatrix} \\
A^{2k} &= \begin{pmatrix} (-1)^k (t\omega)^{2k} & 0 \\ 0 & (-1)^k (t\omega)^{2k} \end{pmatrix} \\
A^{2k+1} &= \begin{pmatrix} \frac{1}{\omega}(-1)^k (t\omega)^{2k+1} & 0 \\ 0 & -\omega(-1)^k (t\omega)^{2k+1} \end{pmatrix}
\end{aligned}$$

one thus obtains

$$\begin{aligned}
\exp(tA) &= \sum_{k\geq 0} \frac{1}{k!} A^k \\
&= \sum_{k\geq 0} \frac{1}{(2k)!} \begin{pmatrix} (-1)^k (t\omega)^{2k} & 0 \\ 0 & (-1)^k (t\omega)^{2k} \end{pmatrix} + \sum_{k\geq 0} \frac{1}{(2k+1)!} \begin{pmatrix} \frac{1}{\omega}(-1)^k (t\omega)^{2k+1} & 0 \\ 0 & -\omega(-1)^k (t\omega)^{2k+1} \end{pmatrix} \\
&= \begin{pmatrix} \cos(t\omega) & t\,\mathrm{sinc}(t\omega) \\ -\omega\,\mathrm{sinc}(t\omega) & \cos(t\omega) \end{pmatrix}.
\end{aligned}$$

Since all the appearing trigonometric functions are also analytic, this matrix exists and this yields the desired result.

(b) Midpoint rule: Applying the midpoint rule to the above differential equation gives

$$\begin{pmatrix} x_{n+1} \\ x'_{n+1} \end{pmatrix} = \begin{pmatrix} x_n \\ x'_n \end{pmatrix} + hA \begin{pmatrix} \frac{1}{2}(x_{n+1}+x_n) \\ \frac{1}{2}(x'_{n+1}+x'_n) \end{pmatrix} \iff$$

$$(I-\frac{h}{2}A)\begin{pmatrix} x_{n+1} \\ x'_{n+1} \end{pmatrix} = (I+\frac{h}{2}A)\begin{pmatrix} x_n \\ x'_n \end{pmatrix} \iff$$

$$\begin{pmatrix} \omega x_{n+1} \\ x'_{n+1} \end{pmatrix} = \begin{pmatrix} \omega & 0 \\ 0 & 1 \end{pmatrix} (I-\frac{h}{2}A)^{-1}(I+\frac{h}{2}A) \begin{pmatrix} \frac{1}{\omega} & 0 \\ 0 & 1 \end{pmatrix} \begin{pmatrix} \omega x_n \\ x'_n \end{pmatrix}.$$

Now we compute the matrix $M(h\omega)$:

$$(I-\frac{h}{2}A)^{-1} = \begin{pmatrix} 1 & -\frac{h}{2} \\ \frac{h}{2}\omega^2 & 1 \end{pmatrix}^{-1} = \frac{1}{1+(\frac{h\omega}{2})^2} \begin{pmatrix} 1 & \frac{h}{2} \\ -\frac{h}{2}\omega^2 & 1 \end{pmatrix}$$

$$(I-\frac{h}{2}A)^{-1}(I+\frac{h}{2}A) = \frac{1}{1+(\frac{h\omega}{2})^2} \begin{pmatrix} 1-(\frac{h\omega}{2})^2 & h \\ -h\omega^2 & 1-(\frac{h\omega}{2})^2 \end{pmatrix}$$

$$M(h\omega) = \frac{1}{1+(\frac{h\omega}{2})^2} \begin{pmatrix} 1-(\frac{h\omega}{2})^2 & h\omega \\ -h\omega & 1-(\frac{h\omega}{2})^2 \end{pmatrix}.$$

Finally, we compute the eigenvalues of the matrix $M(h\omega)$. To do this, we set $z = h\omega$ and first consider the matrix without the prefactor.

$$\big(1-(\tfrac{z}{2})^2-\lambda\big)^2 + z^2 = 0 \implies$$

$$\big(1-(\tfrac{z}{2})^2\big)^2 - 2\big(1-(\tfrac{z}{2})^2\big)\lambda + \lambda^2 + z^2 = 0 \iff$$

$$\big(1+(\tfrac{z}{2})^2\big)^2 - 2\big(1-(\tfrac{z}{2})^2\big)\lambda + \lambda^2 = 0 \implies$$

$$\lambda_{12} = 1-(\tfrac{z}{2})^2 \pm \sqrt{\big(1-(\tfrac{z}{2})^2\big)^2 - \big(1+(\tfrac{z}{2})^2\big)^2}$$

$$= 1 \pm iz - (\tfrac{z}{2})^2 \implies$$

$$|\lambda_{12}|^2 = \big(1-(\tfrac{z}{2})^2\big)^2 + z^2 = \big(1+(\tfrac{z}{2})^2\big)^2.$$

Dividing by the prefactor we obtain the eigenvalues of the matrix $M(h\omega)$, which are of modulus 1 independently of $z = h\omega$. Therefore, the midpoint rule is stable for all time step sizes.

(c) Störmer-Verlet method: One first applies the numerical method to the above second-order differential equation. Using again the notation $z = h\omega$. one gets

$$
\begin{aligned}
x'_{n+1/2} &= x'_n - \frac{z}{2}\omega x_n \\
x_{n+1} &= x_n + h\big(x'_n - \frac{z}{2}\omega x_n\big) = \big(1-\frac{z^2}{2}\big)x_n + hx'_n \\
x'_{n+1} &= x'_n - \frac{z}{2}\omega x_n - \frac{z}{2}\omega\Big(\big(1-\frac{z^2}{2}\big)x_n + hx'_n\Big) = \omega\big(\frac{z^3}{4} - z\big)x_n + \big(1-\frac{z^2}{2}\big)x'_n && \Longrightarrow \\
\begin{pmatrix} x_{n+1} \\ x'_{n+1} \end{pmatrix} &= \begin{pmatrix} 1-\frac{z^2}{2} & h \\ \omega z\big(\frac{z^2}{4}-1\big) & 1-\frac{z^2}{2} \end{pmatrix} \begin{pmatrix} x_n \\ x'_n \end{pmatrix} && \Longrightarrow \\
\begin{pmatrix} \omega x_{n+1} \\ x'_{n+1} \end{pmatrix} &= \begin{pmatrix} 1-\frac{z^2}{2} & z \\ z\big(\frac{z^2}{4}-1\big) & 1-\frac{z^2}{2} \end{pmatrix} \begin{pmatrix} \omega x_n \\ x'_n \end{pmatrix}
\end{aligned}
$$

Next, we compute the eigenvalues of the matrix $M(h\omega)$:

$$
\begin{aligned}
\big(a - \frac{z^2}{2} - \lambda\big)^2 - z^2\big(\frac{z^2}{4} - 1\big) = 0 && \Longleftrightarrow \\
\lambda^2 - \lambda(2 - z^2) + 1 = 0
\end{aligned}
$$

We know that $\lambda_1 + \lambda_2 = 2 - z^2$ and $\lambda_1\lambda_2 = 1$. From the latter relation we can see, that either one of the eigenvalues is larger than 1 in modulus or both are equal to 1 in modulus. The method is stable only in the case, where both eigenvalues are less than or equal to 1 in modulus, therefore in this case, both eigenvalues are equal to 1 in modulus. Now we consider the first relation.

$$
\begin{aligned}
1 = |\lambda_1| = |2 - z^2 - \lambda_2| \geq |2 - z^2| - |\lambda_2| = |2 - z^2| - 1 && \Longleftrightarrow \\
2 \geq |2 - z^2| && \Longleftrightarrow \\
2 \geq 2 - z^2 \quad \text{and} \quad -2 \leq 2 - z^2 && \Longleftrightarrow \\
0 \geq -z^2 \quad \text{and} \quad 4 \geq z^2.
\end{aligned}
$$

The last relation yields $|h\omega| \leq 2$ in order for the Störmer-Verlet scheme to be stable.

5 GNI for SDEs

Task 1: Propositions for the codes are

(a)

```
clear all
randn('state',100)
% discretised BM for dt=2^(-4)
Tend=1;
dt=2^(-4);
N=1/dt;
W(1)=0;
for l=1:N
  dW=sqrt(dt)*randn(1,1);
  W(l+1)=W(l)+dW;
end
randn('state',101)
% discretised BM for dt=2^(-6)
```

```
Tend=1;
dt1=2^(-6);
N=1/dt1;
W1(1)=0;
for l=1:N
  dW=sqrt(dt1)*randn(1,1);
  W1(l+1)=W1(l)+dW;
end
randn('state',102)
% discretised BM for dt=2^(-8)
Tend=1;
dt2=2^(-8);
N=1/dt2;
W2(1)=0;
for l=1:N
  dW=sqrt(dt2)*randn(1,1);
  W2(l+1)=W2(l)+dW;
end
% plot
figure(),
plot([0:dt:Tend],W,'b','LineWidth',3)
hold on
plot([0:dt1:Tend],W1,'k','LineWidth',3)
hold on
plot([0:dt2:Tend],W2,'r','LineWidth',3)
hold off
xlabel('$t_\ell$','Interpreter','latex','FontSize',15)
ylabel('$W_\ell$','Interpreter','latex','FontSize',15,'Rotation',0)
legend('2^{-4}','2^{-6}','2^{-8}')
set(gca,'FontSize',15);
```

```
clear all
randn('state',100)
% discretised BM for dt=2^(-8)
% 200 samples
Tend=1;dt=2^(-8);N=1/dt;
M=200; % samples
dW=sqrt(dt)*randn(M,N+1);
W=zeros(M,N+1);
W(:,1)=0;
for l=1:N
  W(:,l+1)=W(:,l)+dW(:,l);
end
Wmean=mean(W);
% discretised BM for dt=2^(-8)
% 2000 samples
Tend=1;dt=2^(-8);N=1/dt;
M=2000; % samples
dW=sqrt(dt)*randn(M,N+1);
W=zeros(M,N+1);
W(:,1)=0;
for l=1:N
  W(:,l+1)=W(:,l)+dW(:,l);
end
```

```
Wmean1=mean(W);
% discretised BM for dt=2^(-8)
% 20000 samples
Tend=1;dt=2^(-8);N=1/dt;
M=20000; % samples
dW=sqrt(dt)*randn(M,N+1);
W=zeros(M,N+1);
W(:,1)=0;
for l=1:N
  W(:,l+1)=W(:,l)+dW(:,l);
end
Wmean2=mean(W);
% plot mean over M samples
figure(),
plot([0:dt:Tend],Wmean,'b','LineWidth',3)
hold on
plot([0:dt:Tend],Wmean1,'k','LineWidth',3)
hold on
plot([0:dt:Tend],Wmean2,'m','LineWidth',3)
hold off
xlabel('Time','FontSize',15)
ylabel('Means','FontSize',15,'Rotation',0)
legend('200','2000','20000')
set(gca,'FontSize',15);
```

(b)

```
clear all
randn('state',100)
% discretised BM for dt=2^(-8)
% 50000 samples
Tend=1;dt=2^(-8);N=1/dt;
M=50000; % samples
dW=sqrt(dt)*randn(M,N+1);
W=zeros(M,N+1);
W(:,1)=0;
for l=1:N
  W(:,l+1)=W(:,l)+dW(:,l);
end
Wmean=mean(W);
% plot mean and some samples
figure(),
plot([0:dt:Tend],Wmean,'b','LineWidth',3)
hold on
plot([0:dt:Tend],[W(1:5,:)],'k','LineWidth',3) % plot 5 samples
hold off
xlabel('Time','FontSize',15)
ylabel('Means','FontSize',15,'Rotation',0)
set(gca,'FontSize',15);
```

Task 2: A proposition for the solution to this task reads

```
clear all
randn('state',100)
% Initial parameters
omega=5;X0=0;Y0=1;alpha=1;
```

```
% number of paths sampled for
% the approximation of the expectation
M=10^3;
% initial energy
Haminit=Y0^2./2+omega^2*X0^2./2
Tend=5;
N=2^(5);
h=Tend/N;

Hamem=zeros(M,N);
Hamstm=zeros(M,N);

for s=1:M
  % Initial values
  Xtempem=X0;Ytempem=Y0;
  Xtempstm=X0;Ytempstm=Y0;
  for j=1:N
    Winc=sqrt(h)*randn(1,1);
    % EM
    X1em=Xtempem+h*Ytempem;
    Y1em=Ytempem-h*omega^2*Xtempem+alpha*Winc;
    Xtempem=X1em;Ytempem=Y1em;
    % energy for EM
    Hamem(s,j)=.5*(Y1em^2+omega^2*X1em^2);

    % STM
    X1stm=Xtempstm*cos(h*omega)+sin(h*omega)/omega*Ytempstm+ ...
         alpha*sin(h*omega)/omega*Winc;
    Y1stm=-omega*Xtempstm*sin(h*omega)+cos(h*omega)*Ytempstm+ ...
         alpha*cos(h*omega)*Winc;
    Xtempstm=X1stm;Ytempstm=Y1stm;
    % energy for STM
    Hamstm(s,j)=.5*(Y1stm^2+omega^2*X1stm^2);
   end
end

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

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);
```