Kharagpur Lectures

Numerical Accuracy, Error Control and Applications

Carol G. Hoover & William G. Hoover
Ruby Valley Nevada

December 2016

Outline

1. Integration Methods and Accuracy
   - Accuracy checks with the harmonic oscillator or a benchmark problems
   - Symplectic Integrators for Hamiltonian Systems
   - Runge-Kutta Integrators for Non-Hamiltonian Systems

2. Cell Model evaluation of integration techniques

3. Predictor-Corrector Methods
   - Milne method
   - Gear predictor-Corrector

4. Stiff Differential Equations :
   - The Nosé Oscillator
   - Adaptive Integration

5. Addendum
   - Periodic orbits for the Nosé and Nosé-Hoover oscillator
   - Regular and Chaotic Orbits for the springy pendulum
1. Integration Methods and Accuracy

Numerical Approximations with Taylor’s Series at $t_n$

$$x(t_{n+1}) = x(t_n) + \Delta t \frac{dx}{dt} + \frac{(\Delta t^2) d^2x}{2!} \frac{dt^2}{dt^2} + \frac{(\Delta t^3) d^3x}{3!} \frac{dt^3}{dt^3} + O(\Delta t^4)$$

$$x(t_{n-1}) = x(t_n) - \Delta t \frac{dx}{dt} + \frac{(\Delta t^2) d^2x}{2!} \frac{dt^2}{dt^2} - \frac{(\Delta t^3) d^3x}{3!} \frac{dt^3}{dt^3} + O(\Delta t^4)$$

$$x(t_{n+1}) - 2x(t_n) + x(t_{n-1}) = \Delta t^2 F(x_n) + O(\Delta t^4)$$

Second-Order Symplectic Integration Algorithms

Harmonic oscillator: $\dot{q} = p; \dot{p} = -q; (q,p)_0 = (1,0)$

Solve one second order equation or two first order equations:

$$\frac{d^2 x}{dt^2} = F(x) \quad \frac{dx}{dt} = v \quad \frac{dv}{dt} = F(x).$$

Störmer-Verlet.

$$x_{n+1} = 2x_n - x_{n-1} + \Delta t^2 F(x_n)$$

Leapfrog; 3 stages.

$$x_n = x_{n-1} + v_{n-(1/2)} \Delta t; \quad a_n = F(x_n); \quad v_{n+(1/2)} = v_{n-(1/2)} + a_n \Delta t.$$ 

The Stormer-Verlet difference equation can be solved analytically:

$$q_0 = e^{io_d 0}; q_{+1} = e^{io_d \Delta t}; q_{-1} = e^{io_d (-\Delta t)}; \quad \omega = (1/\Delta t) \cos^{-1}(1 - \Delta t^2/2)$$

Then

$$q_{error} = \max \left( \sqrt{ (\cos(time) - \cos(\omega dt))^2 } \right)$$

**Stormer-Verlet Coordinate and Energy Errors**

The momentum must be approximated from the values of the coordinates.

The two curves shown for the energy error are for second- and fourth-order centered approximations for the momentum:

\[
P_2 \quad p_i = \frac{(q_{i+1} - q_{i-1})}{2dt} - \frac{(dt^2)}{6} q'''
\]

\[
P_4 \quad p_i = \frac{2(q_{i+1} + q_{i-1})}{3dt} - \frac{(q_{i+2} + q_{i-2})}{12dt} + \frac{(dt^4)}{30} q''''
\]

Notice that the energy remains second order. However, the calculated data shows that the energy error is about five times smaller for the higher order momentum.

---

**Fourth-Order Symplectic Algorithm (Candy and Rozmus)**

The fourth-order symplectic algorithms solve first order differential equations. The number of stages in a timestep is four.

\[
\dot{q} = p ; \quad \dot{p} = F
\]

Specify \( q_0 \) and \( p_0 \) :

\[
q_i = q_{i-1} + a_i p_{i-1} dt ; \quad p_i = p_{i-1} + b_i F_i dt ; \quad \text{for } i = 1, 4
\]

\[
a_1 = a_4 = \frac{(2 + 2^{1/3} + 2^{-1/3})}{6} ; \quad b_1 = b_3 = \frac{(2 - 2^{1/3})^{-1}}{2}
\]

\[
a_2 = a_3 = \frac{(1 - 2^{1/3} - 2^{-1/3})}{6} ; \quad b_2 = \frac{(1 - 2^{2/3})^{-1}}{2}
\]

Programming steps:

\[
Q = Q + a_1 p dt \\
P = P + b_1 F dt \\
Q = Q + a_2 p dt \\
P = P + b_2 F dt \\
Q = Q + a_3 p dt \\
P = P + b_3 F dt \\
Q = Q + a_4 p dt
\]

---

Fourth-Order Symplectic Monte-Carlo Algorithm

An algorithm with five force evaluations per timestep was developed by Monte-Carlo sampling adjusting the coefficients subject to the constraints of time reversibility and normalization so that the Monte-Carlo trajectory optimization occurs in a four-dimensional space. This method was successful in modeling many-body dynamics. There are 5 force evaluations per step. The programming steps for the oscillator are given below.

\[
\begin{align*}
q &= q + 0.005904d00*p*dt \\
p &= p + 0.171669d00*(-q)*dt \\
q &= q + 0.515669d00*p*dt \\
p &= p - 0.516595d00*(-q)*dt \\
q &= q - 0.021573d00*p*dt \\
p &= p + 1.689852d00*(-q)*dt \\
q &= q - 0.021573d00*p*dt \\
p &= p + 0.171669d00*(-q)*dt \\
q &= q + 0.005904d00*p*dt
\end{align*}
\]

Notice that for each variable the coefficients add up to 1.0 .


Runge-Kutta Methods

Fourth-Order Runge-Kutta Method

1. Four steps calculate approximate derivatives :
   - Calculate new values of v and F using updated values of x and p (right-hand-side evaluation)
   - Update x and v for dt/2 or dt .

2. Calculate updated values of x and v over the full time step using a weighted average of the derivatives computed in the previous four-step calculation .

Fifth-Order Runge-Kutta Method

1. Six update steps to calculate approximate derivatives . Some update steps use weighted derivatives .

2. Final step uses a weighted average of six approximate derivatives .
Derivative points for Runge-Kutta integrators

Harmonic Oscillator

\[
\dot{q} = +p \\
\dot{p} = -q
\]

5th order Runge-Kutta

dt = \frac{2\pi}{5}

4th order Runge-Kutta

dt = \frac{2\pi}{10}

Storage Layout for the Runge-Kutta Algorithms

5 particles

<table>
<thead>
<tr>
<th>YY Array</th>
<th>YYP Array</th>
</tr>
</thead>
<tbody>
<tr>
<td>Position and Momenta</td>
<td>Derivatives</td>
</tr>
<tr>
<td>x1 = yy(1)</td>
<td>yyp(1) = p1</td>
</tr>
<tr>
<td>x2 = yy(2)</td>
<td>yyp(2) = p2</td>
</tr>
<tr>
<td>x3 = yy(3)</td>
<td>yyp(3) = p3</td>
</tr>
<tr>
<td>x4 = yy(4)</td>
<td>yyp(4) = p4</td>
</tr>
<tr>
<td>x5 = yy(5)</td>
<td>yyp(5) = p5</td>
</tr>
<tr>
<td>p1 = yy(6)</td>
<td>yyp(6) = f1</td>
</tr>
<tr>
<td>p2 = yy(7)</td>
<td>yyp(7) = f2</td>
</tr>
<tr>
<td>p3 = yy(8)</td>
<td>yyp(8) = f3</td>
</tr>
<tr>
<td>p4 = yy(9)</td>
<td>yyp(9) = f4</td>
</tr>
<tr>
<td>p5 = yy(10)</td>
<td>yyp(10) = f5</td>
</tr>
</tbody>
</table>
A Comparison of RK4 and RK5 for the Oscillator with a Large Timestep

Results for a large timestep:

The Fourth-Order Runge-Kutta solution decays.

The Fifth-Order Runge-Kutta solutions grows.

Similarly the energy drifts upward for the RK5 and decays for RK4.

Energy Errors for RK4 and RK5 for the Harmonic Oscillator

Global Energy Error ~ dt^5 for Both RK4 & RK5

Total time = 2.0

\[
\text{Slope } = 5
\]

\[
\text{Slope } \approx 5
\]
2. Cell Model evaluation of integration techniques

Integration Accuracy for a Chaotic System

• A Cell Model
  A Chaotic Trajectory generated with a very smooth, soft potential

• 6 Integration Methods
  Symplectic:
  2nd Order LeapFrog
  4th Order Candy-Rozmus
  4th Order Method with Coefficients determined by Monte-Carlo
  6th Order Yoshida Method
  Runge-Kutta Methods:
  4th Order Runge-Kutta
  5th Order Runge-Kutta

• Results compared with an accurate 8th order Telroy-Schlier-Sieter symplectic method.

A Very Smooth Cell-Model Trajectory

\[ \Phi = \sum [1 - (r - r_i)^2]^4 \quad \text{for} \quad |r - r_i| < 1 . \]

Periodic boundary conditions; RK5 quadruple precision; chaotic Hamiltonian.

Evaluation Criteria for the Cell Model Calculations

1. Trajectory Accuracy
We find the time for which the coordinates are accurate to 0.01 when compared with the Schlier-Seiter-Telroy 8th order integration.

2. Reversibility
The time for which a trajectory can be reversed to within 0.01 of the origin.

3. Energy Accuracy
Energy conservation is a diagnostic. The difference between the initial energy and the energy at the final time is the energy error.

4. What else do we need to consider?

<table>
<thead>
<tr>
<th>Method</th>
<th>Order</th>
<th>Accurate Trajectory Time</th>
<th>Energy Accuracy</th>
<th>Reversal Time</th>
<th>Force Evaluations</th>
</tr>
</thead>
<tbody>
<tr>
<td>Leapfrog</td>
<td>2nd</td>
<td>18</td>
<td>$10^{-7}$</td>
<td>47</td>
<td>1</td>
</tr>
<tr>
<td>Candy Rozmus</td>
<td>4th</td>
<td>34</td>
<td>$10^{-15}$</td>
<td>42</td>
<td>3</td>
</tr>
<tr>
<td>Monte Carlo</td>
<td>4th</td>
<td>31</td>
<td>$10^{-13}$</td>
<td>43</td>
<td>5</td>
</tr>
<tr>
<td>Yoshida</td>
<td>6th</td>
<td>36</td>
<td>$10^{-15}$</td>
<td>42</td>
<td>7</td>
</tr>
<tr>
<td>Runge-Kutta</td>
<td>4th</td>
<td>35</td>
<td>$10^{-13}$</td>
<td>42</td>
<td>4</td>
</tr>
<tr>
<td>Runge-Kutta</td>
<td>5th</td>
<td>34</td>
<td>$10^{-15}$</td>
<td>42</td>
<td>6</td>
</tr>
</tbody>
</table>

Accurate trajectory time occurs for coordinate errors less than 0.01 compared to 8th order Telroy-Schlier-Sieter symplectic method. Coefficients are extended precision. Here $dt = 0.001$, a typical MD timestep.
Conclusions
Comparison of 6 Integration Methods

- Energy conservation and reversibility do not guarantee an accurate trajectory. Leapfrog integration is an example.

- Leapfrog integration has a longer reversibility time than any of the other more accurate integration methods. For chaotic systems it is nearly as useful as the higher order, more accurate integrators because of the Lyapunov instability for the cell model.

- The Lyapunov instability is a physical instability rather than a numerical inaccuracy that occurs in chaotic systems. The Lyapunov exponent measures the exponential separation rate between two trajectories initially separated by a “small” perturbation (∼ \(10^{-6}\)). The largest Lyapunov exponent for the cell model is 0.7. Consider an initial error of \(10^{-16}\). For a run to a time of 50 the separation will increase to \(10^{15}\):

\[
e^{0.7 \times 50} = e^{35} \approx 10^{15}.
\]

- Trajectory accuracy is the most important criterion to use when calculating detailed atomistic mechanisms such as energy barrier crossings. Leapfrog integration (or the Monte-Carlo method) is a good choice for equilibrium problems at constant energy.

3. Predictor – Corrector Methods

- Used for stiff equations
- Often stiff equations arise for problems with two time scales in the solution
- Two steps:
  - The Predictor step is explicit
  - The Corrector step is implicit
- Predictor-Corrector schemes are not self-starting
  - Fourth and fifth order Runge-Kutta algorithms are self-starting!
  - Combining the two Runge-Kutta algorithms results in an excellent adaptive integrator for the Nosé oscillator
- Second Order Störmer-Verlet is self starting
- Stability is important **
- Stable time steps are usually larger for implicit than for explicit algorithms
- Molecular dynamics: Nosé’s thermostated oscillator
- Continuum mechanics: Viscoelasticity and other nonlinear effects
- Other examples include chemical reactions with two time scales, and circuit analysis
- Milne’s two methods can be analyzed with the Harmonic Oscillator. The first uses the two first order equations of motion and the second uses the second order equation of motion

** For more details on implicit methods see the following:

http://qucs.sourceforge.net/tech/node25.html
http://qucs.sourceforge.net/tech/node24.html
**Milne’s Predictor-Corrector Methods for the Oscillator**

**Method 1**: Two first order differential equations – Error term is 5th order.

- **Predictor Step**: Explicit
  \[ q_{n+1} = q_{n-3} + (4dt/3)(2\ddot{q}_n - \ddot{q}_{n-1} + 2\ddot{q}_{n-2}) + (28/90)dt^5 \dddot{q} \]
  \[ p_{n+1} = p_{n-3} + (4dt/3)(2\dot{p}_n - \dot{p}_{n-1} + 2\dot{p}_{n-2}) + (28/90)dt^5 \dddot{p} ; \ (\dot{p} = -q) \]

- **Corrector Step**: Implicit
  \[ q_{n+1} = q_{n-1} + (h/3)(\dddot{q}_{n+1} + 4\dddot{q}_n + \dddot{q}_{n-1}) - (1/90)dt^5 \dddot{q} \]
  \[ p_{n+1} = p_{n-1} + (h/3)(\dddot{p}_{n+1} + 4\dddot{p}_n + \dddot{p}_{n-1}) - (1/90)dt^5 \dddot{p} ; \ (\dddot{p} = -q) \]

**Method 2**: One second order differential equation – Error term is 6th order.

- **Predictor Step**: Explicit
  \[ q_{n+1} = q_n + q_{n-2} - q_{n-3} + (dt^2/4)(5\dddot{q}_n + 2\dddot{q}_{n-1} + 5\dddot{q}_{n-2}) + (17/240)dt^6 \dddot{q} \]

- **Corrector Step**: Implicit
  \[ q_{n+1} = 2q_n - q_{n-1} + (dt^2/12)(\dddot{q}_{n+1} + 10\dddot{q}_n + \dddot{q}_{n-1}) - (1/240)dt^6 \dddot{q} \]

**Predictor-Corrector Algorithms are not Self-Starting**

If the analytical form of the solution is known, it can be used to evaluate nearby points to start up the algorithm.

Taylor’s series can always be used to generate the extra points needed to start the algorithm. This can also be done numerically.

Evaluate the function and its derivative each at two nearby points. This is the finite difference technique.

Accuracy can be evaluated using analytical solutions for the numerical approximation and comparing the error when the time step is reduced in a systematic way.

**Student problem**

Any of the algorithms can be evaluated with the oscillator.

Is the stability behavior different when using previous points versus future points in Taylor series?
Numerically Evaluate the Order of the Global Error for Milne’s Methods Using the Harmonic Oscillator

Coordinate Error \( q(2\pi) = \text{True Solution} - \text{Numerical Solution} \)

Plot: \( \ln(\text{Error}) \) as a function of \( \ln(\text{dt}) \) for several values of \( \text{dt} \).

The slope of a straight line gives the power of \( \text{dt} \) in the error term.

Initial condition: \( q = \cos(t); \ p = -\sin(t); \ F = -q \).

\( \text{dt} = \frac{2\pi}{50}, \frac{2\pi}{100}, \ldots, \frac{2\pi}{1600} \)

Starting values calculated from the analytical solution.

**Method 1: First Order-Equations**

\[
\begin{align*}
\dot{q} &= p; \quad \dot{p} = -q \\
\end{align*}
\]

**Method 2: Second-Order Equation**

\[
\ddot{q} = -q
\]

Gear Predictor-Corrector Method

\[
\begin{align*}
\mathbf{r}_{t+dt}^P &= \mathbf{r}_t + \text{dt}(\mathbf{v}_t) + \frac{\text{dt}^2}{2} \mathbf{a}_t + \frac{\text{dt}^3}{6} \mathbf{b}_t + \cdots \\
\mathbf{v}_{t+dt}^P &= \mathbf{v}_t + \text{dt}(\mathbf{a}_t) + \frac{\text{dt}^2}{2} \mathbf{b}_t + \cdots \\
\mathbf{a}_{t+dt}^P &= \mathbf{a}_t + \text{dt}(\mathbf{b}_t) + \cdots \\
\mathbf{b}_{t+dt}^P &= \mathbf{b}_t
\end{align*}
\]

**Predictor Step**

\[
\begin{align*}
\begin{pmatrix} r \\ v \\ a \\ b \end{pmatrix}_{t+dt}^P &= \begin{pmatrix} 1 & \text{dt} & \frac{\text{dt}^2}{2} & \frac{\text{dt}^3}{6} \\ 0 & 1 & \text{dt} & \frac{\text{dt}^2}{2} \\ 0 & 0 & 1 & \text{dt} \\ 0 & 0 & 0 & 1 \end{pmatrix} \begin{pmatrix} r \\ v \\ a \\ b \end{pmatrix}_t \\
\end{align*}
\]

**Corrector Step**

\[
\begin{align*}
\begin{pmatrix} r \\ v \\ a \\ b \end{pmatrix}_{t+dt} &= \begin{pmatrix} r \\ v \\ a \\ b \end{pmatrix}_{t+dt}^P + (c_0 + c_1 + c_2 + c_3) \Delta a_0 \\
\end{align*}
\]

\( \Delta a = (\mathbf{a}_{t+dt}^P - \mathbf{a}_{t+dt}) / \text{dt}^2 / 2 \)

Student Problem

Consult Gear’s book for a fourth order gear algorithm with an iteration scheme for the corrector step.

Gear Algorithm with a single iteration of the Corrector Step

```
c Gear algorithm for the harmonic oscillator
ro  =  1.0d00  ! initial condition
vo  =  0.0d00
ao  = -1.0d00
bo  =  0.0d00
dt  =  (1/2)**n  ! Pick dt for n = 1,9
rp = 0.0d00  ! Predictor values
vp = 0.0d00
ap = 0.0d00
bp = 0.0d00
rc = 0.0d00  ! Corrector values
vc = 0.0d00
ac = 0.0d00
bc = 0.0d00
time = 0.0d00
Ham = 0.5d00*(ro**2 + vo**2)
Ham0 = Ham
write(6,*)"energy ",time,Ham

c0 = 1.0d00/6.0d00
c1 = 5.0d00/6.0d00
c2 = 1.0d00

do it = 1,itmax

c predictor
rp =  ro + dt*vo + 0.5d00*(dt**2)*ao + 
   (1.0d00/6.0d00)*(dt**3)*bo
vp =  vo + dt*ao + 0.5d00*(dt**2)*bo
ap =  ao + dt*bo
bp =  bo
ac = -rp
da = (ac - ap)
Da = 0.5d00*da*dt*dt

c corrector
rc = rp + c0*Da
vo = vp + c1*Da/dt
ac = ap + 2.0d00*c2*Da/(dt**2)
bc = bp + 6.0d00*c3*Da/(dt**3)
time = it*dt
Ham = 0.5d00*(rc**2 + vc**2)
write(6,*)time,rc,vc,ac,bc,Ham
ro = rc  ! reset the variables
vo = vc
ao = ac
bo = bc
end do
```
Gear integration for the Harmonic Oscillator
Energy, Coordinate, and Momentum Error in one period

Initial condition \( (q, p, a, b, H) = (1, 0, -1, 0, 0.5) \)
\[ dt = \pi / 2^n \text{ for } n = 4, \ldots, 11 \]

\[ \log\left( \frac{dq}{dt} \right) \text{ slope } = 5.0 \]
\[ \ln\left( \frac{dp}{dt} \right) \text{ slope } = 3.0 \]

http://www.chemcomp.com/Gear.pdf ;
https://people.maths.ox.ac.uk/suli/nsodes.pdf


If the numerical problem has some components that change quickly, but you have to integrate for a long time, the problem is called stiff (see p. 360 and Perry and Green, p. 3-50, 2008). Implicit methods are then needed. The best ones are based upon the work by Gear (1971). For the problem
\[ \frac{dy}{dt} = f(t,y), \quad y(0) = y_0, \]
where \( y \) and \( f \) can be vectors, the methods of different order are:

1. \( y^{n+1} = y^n + \Delta t f(y^n) \)
2. \( y^{n+1} = \frac{4}{3} y^n - \frac{1}{3} y^{n-1} + \frac{2}{3} \Delta t f(y^n) \)
3. \( y^{n+1} = \frac{18}{11} y^n - \frac{9}{11} y^{n-1} + \frac{2}{11} y^{n-2} + \frac{6}{11} \Delta t f(y^n) \)
4. \( y^{n+1} = \frac{48}{25} y^n - \frac{36}{25} y^{n-1} + \frac{16}{25} y^{n-2} - \frac{3}{25} y^{n-3} + \frac{12}{25} \Delta t f(y^n) \)
5. \( y^{n+1} = \frac{300}{137} y^n - \frac{300}{137} y^{n-1} + \frac{200}{137} y^{n-2} - \frac{75}{137} y^{n-3} + \frac{12}{137} y^{n-4} + \frac{60}{137} \Delta t f(y^n) \)

In Gear's method, the nonlinear equations are solved. If they cannot be solved, then the step size is reduced and you try again. Keep reducing the step size until the implicit equations can be solved. If the step size is extremely small, the computer program will stop. The order can be changed, too, and it is changed in the Gear algorithm to minimize the computational cost for a specified accuracy.

Reference


4. Stiff Differential Equations

Purpose: To understand adaptive integration and to study techniques for analyzing results.

1. Hamiltonian mechanics with temperature

Nosé (1984) modified Hamiltonian mechanics to include temperature. We compare two versions of the modified mechanics:
Nose mechanics and Nosé-Hoover mechanics for the harmonic oscillator.

2. We discuss numerical stiffness, errors, and Lyapunov instability.

3. We discuss fixed time step calculations (Fourth-Order Runge-Kutta) for the Nosé and Nosé-Hoover oscillators; we find numerical instability.

4. We present an adaptive integration technique based on fourth-order Runge-Kutta.

5. We compare the adaptive method for the Nosé and Nosé-Hoover oscillators.

6. Summary

The Thermostated Harmonic Oscillator

The Nosé oscillator is the motion of a harmonic oscillator with a specified average temperature! Nosé’s idea (1984) was to replace isoenergetic mechanics with a temperature-based mechanics. To do this he developed a temperature dependent Hamiltonian consistent with Gibbs’ canonical distribution. When this Hamiltonian is applied to the harmonic oscillator the resulting equations of motion are very stiff!

\[ 2\mathcal{H} = q^2 + p^2/s^2 + T\ln(s^2) + \zeta^2 \equiv 0 \rightarrow \]

Nose Oscillator Equations

\[ q = (p/s^2) ; p = -q ; s = \zeta ; \dot{\zeta} = (p^2/s^3) - 1/s \]

For \( \kappa, T, k, m = 1 \).

Dettmann and Morris later modified the original Nosé Hamiltonian and showed that the following Nosé-Hoover equations (Hoover 1985) can be derived from it.

Nosé-Hoover Oscillator Equations

\[ \dot{q} = p/s ; \dot{p} = -s q ; \dot{s} = s \zeta ; \dot{\zeta} = p^2/s^2 - 1 \quad \# 1 \]

\[ p/s \rightarrow p ; \dot{q} = p ; \dot{p} = -q - \xi p ; \dot{\xi} = p^2 - 1. \quad \# 2 \]
Adaptive Integration for stiff differential equations: Compare the Nosé and Nosé-Hoover Oscillators

- We will compare the smooth Nosé-Hoover equations to the stiff Nosé oscillator equations with the same phase-space trajectories.

- Nosé-Hoover oscillator trajectories are both (Lyapunov stable) and chaotic (Lyapunov unstable).

- A trajectory is considered “Lyapunov” unstable if the distance between the trajectory and nearby a neighbors’ trajectory grows exponentially:

\[ \delta \propto e^{+\lambda t} \text{ Lyapunov unstable} \quad \text{for } \lambda > 0 \]

\[ \text{Lyapunov stable} \quad \text{for } \lambda \leq 0 \]

- The trajectory considered in this Lecture is Lyapunov unstable (chaotic). However, Lyapunov instability occurs perpendicular to the trajectory whereas numerical errors give rise to phase-error fluctuations parallel to the trajectory.

- The unusual aspect of these two oscillators is that the two trajectories are exactly the same but the dynamical rates of progress along the common trajectory are different. This is referred to as “time scaling” in Nosé’s original paper. We will show that this is the cause of the numerical stiffness!

Numerical Stiffness

- In mathematics, a stiff equation is a differential equation for which the usual numerical methods for solving the equation are numerically unstable, unless the step size is taken to be extremely small.

- Adaptive methods can be used in some cases to overcome numerical instability. We will illustrate this with the Nosé oscillator example.

- Adaptive methods cannot be used for singular integrands such as the “event-driven” hard sphere collision models.

- Other examples of numerical stiffness arise in problems with two time scales as in circuit analysis. In these cases the stiffness is treated with predictor-corrector methods including implicit methods. Matrix solutions are required for the implicit methods.

- Most molecular dynamics motion equations are solved with explicit techniques!
Fourth-Order Runge-Kutta Integration for the Nosé and Nosé-Hoover Oscillators

If the Hamiltonian is set to 0 and $p = \zeta = 0$ initially then $0 < s \leq 1$, the Nosé and Nosé-Hoover oscillator phase-space trajectories are identical. We can use this fact to compare the integration results for the two oscillators.

Initial condition: $(q,p,s, \zeta) = (2.4, 0, e^{-2.88}, 0)$; $\mathcal{H} = 0$.

Nosé
\[ dt = 0.001 \]
\[ time = 5. \]
Nosé-Hoover
\[ dt = 0.01 \]
\[ time = 25. \]
Nosé-Hoover Oscillator
\[ dt = 0.01 \]
\[ time = 1000. \]

Identical phase-space trajectories. RK4 algorithm is unstable for Nosé oscillator using $dt = 0.001$, time $\sim 5.0$. Nosé-Hoover oscillator is stable for $dt = 0.01$.

What is a stable time step for the Nosé Oscillator?

The Hamiltonian will be zero if $0 < s < 1$. For the Nosé Oscillator the variables scale as $1/(s^2)$. For small $s$ the rates become large. What is the minimum value of $s$ for which the algorithm will remain stable and at what time does this minimum occur?

Initial condition: $(q,p,s, \zeta) = (2.4, 0, e^{-2.88}, 0)$; $\mathcal{H} = 0$.

\[ \dot{q} = (p/s^2); \quad \dot{p} = -q; \quad \dot{s} = \zeta; \quad \dot{\zeta} = (p^2/s^3) - 1/s \] Nosé
\[ \dot{q} = p/s; \quad \dot{p} = -sq; \quad \dot{s} = s\zeta; \quad \dot{\zeta} = p^2/s^2 - 1 \] Nosé-Hoover

$s$ is a time scaling variable

\[ dq/dt \sim p / e^{-14} \]
**Nosé Oscillator Trajectories for dt = 10^{-6}**

\[ \frac{dq}{dt} \approx \frac{p}{e^{14}} \]

\[ H = 0 \]

**How well does the algorithm perform for dt = 0.0001?**

- **Double Precision**
- **Quadruple Precision**

**Conclusion:** This algorithm is not able to integrate successfully the Nosé oscillator.

Use a logarithmic scale to find the minimum values of s.

What is another check can we make on this calculation?
We gain physical insights by checking all of the results !!

This trajectory in the time interval shown is reasonably smooth in spite of the small values of s. But notice the straight line approximations to the small radius-of-curvature loops.

The Hamiltonian is not zero!
In fact the Hamiltonian deviates from zero starting at a time of 5.0!
Smaller time steps have not helped the numerical stability.

Conclusions

1. Always test your numerical results for errors and consistency with the physical model.
   - Check the dependence of the error on the time step
   - Check the dependence of the results on precision (double versus quadruple)
   - Check the physical diagnostics (energy in this case)
   - Compare the result to a known result (Nosé-Hoover)
2. The fourth-order Runge-Kutta algorithm is not useful for integrating the Nosé oscillator
3. Develop an adaptive integration routine for the Nosé oscillator and study the dependence of the stiffness on the time-scaling variable.

Student problems
1. Run Nosé-Hoover #1 trajectories with the RK4 algorithm for the chaotic initial condition \((q, p, s, \zeta) = (2.4, 0, e^{-2.88}, 0)\). Test all of the checks shown above. Use a simulation time longer than 264.
2. Run Nosé-Hoover #2 trajectories for the stable-torus initial condition \((q, p, \zeta) = (0.0, 1.55, 0.0)\). Test all of the checks shown above. How do these calculations differ from the chaotic solution?
An Adaptive Integration Algorithm
Based on the Fourth-order Runge-Kutta Method

- Select an error band
  - For double precision ~ $10^{-12}$ to $10^{-10}$
  - For quadruple precision ~ $10^{-24}$ to $10^{-20}$
- Calculate the root-mean-square error using the difference between two $dt/2$ steps and the single $dt$ step
- Change the time step if outside the error band
  - Half the time step for errors greater than the upper limit
  - Double the time step for errors less than the lower limit
- Reset the array with the $dt$ step to the more nearly accurate solution at the end of the time step.

```fortran
  call rk4(x,xp,dt/2.0d00)
call rk4(x,xp,dt/2.0d00)
call rk4(y,yp,dt/1.0d00)
error2 = (x(1)-y(1))**2 + (x(2)-y(2))**2 + (x(3)-y(3))**2 +
        (x(4)-y(4))**2
error = dsqrt(error2)
if(error.gt.10.0d00**(-10)) dt = 0.5d00*dt
if(error.lt.10.0d00**(-12)) dt = 2.0d00*dt
```

Adaptive Integration
Nosé’s Oscillator is much stiffer than the Nosé-Hoover Oscillator

100,000 Adaptive time steps

We can show that the rates for the Nosé oscillator are about 3 times faster but progress along the trajectory is considerably reduced for the same number of adaptive time steps. This is evidence for the numerical stiffness for the Nosé oscillator. The trajectories are the same but progress at different rates.
Adaptive Time Step Comparisons
Nosé-Hoover Oscillator Versus Nosé Oscillator

\[
\ln(2) \left( \frac{dt}{\ln(2)} \right)
\]

\[
0 < \text{time} < 200
\]

\[
750 < \text{time} < 790
\]

\[
\dot{q} = \left( \frac{p}{s^2} \right) ; \quad \dot{p} = -q ; \quad \dot{s} = \frac{\xi}{s} ; \quad \dot{\xi} = \left( \frac{p^2}{s^3} \right) - \frac{1}{s}
\]

Nosé

\[
\dot{q} = \frac{p}{s} ; \quad \dot{p} = -sq ; \quad \dot{s} = s\xi ; \quad \dot{\xi} = \frac{p^2}{s^2} - 1
\]

Nosé-Hoover

Adaptive Time Step Comparisons
Double Precision Versus Quadruple Precision

Nosé Oscillator
Initial condition: \((q, p, s, z) = (2.4, 0, e^{-2.88}, 0)\); \(H = 0\).

Time = 28.1097, 22 crossings of \(p = 0\) section.

Double precision

\[
\ln(2) \left( \frac{dt}{\ln(2)} \right)
\]

12, 201 steps

Quadruple precision

\[
\ln(2) \left( \frac{dt}{\ln(2)} \right)
\]

2,000,000 steps

\[
\dot{q} = \left( \frac{p}{s^2} \right) ; \quad \dot{p} = -q ; \quad \dot{s} = \frac{\xi}{s} ; \quad \dot{\xi} = \left( \frac{p^2}{s^3} \right) - \frac{1}{s}
\]

for \(\kappa, T, k, m = 1\).
The Distance Between two $p = 0$ sections Measures a Length Along a Trajectory

A section is a two-dimensional set of points in the four-dimensional phase space. A section is defined by setting both of the other two variables set to 0.0. In this special case, we look at two $p = 0.0$ sections and let $s$ take on whatever values that occur along the trajectory between the two sections. This allows us to measure the rate at which the Nosé or Nosé-Hoover oscillator travels between two known points on the trajectory.

Numerical results confirm $p = 0.0 = p/s$ sections are identical!

The Distance Between two $p = 0$ sections Measures a Length Along a Trajectory

A section is a two-dimensional set of points in the four-dimensional phase space. A section is defined by setting both of the other two variables set to 0.0. In this special case, we look at two $p = 0.0$ sections and let $s$ take on whatever values that occur along the trajectory between the two sections. This allows us to measure the rate at which the Nosé or Nosé-Hoover oscillator travels between two known points on the trajectory.

Numerical results confirm $p = 0.0 = p/s$ sections are identical!

Compute the trajectory length between two crossings in a $p = 0$ plane

Trajectory length in a time step is $dL_t = \sqrt{\delta q^2 + \delta p^2 + \delta s^2 + \delta \zeta^2}$. For $N$ time steps the trajectory length between two crossings of a $p = 0$ plane is $L_{\text{cross}} = \sum dL_t$.

For $H = 0$ the trajectories for the two oscillators are identical.

Nosé-Hoover & Nosé crossings at $p = 0$

For $H = 0$ the trajectories for the two oscillators are identical.

Nosé-Hoover & Nosé crossings at $p = 0$
The Variable $s$ in the Equations of Motion Scales all of the Rates

Trajectory Length:
\[ dL_i = \text{length between two crossings of the } p = 0 \text{ plane} \]
\[ L = \sum dL_i \text{ for 178 crossings} \]

Initial condition
\[ (q, p, s, \zeta) = (2.4, 0, e^{-2.88}, 0) \]
Chosen to insure that \[ H = 0 \]

Notice that these equations differ by $1/s$ where \( s \) is the time-scaling factor.

\[ \dot{q} = \frac{p}{s^2} ; \quad \dot{p} = -q ; \quad \dot{s} = \zeta ; \quad \dot{\zeta} = \left(\frac{p^2}{s^3}\right) - \frac{1}{s} \quad \text{Nosé} \]

\[ \dot{q} = \frac{p}{s} ; \quad \dot{p} = -sq ; \quad \dot{s} = s\zeta ; \quad \dot{\zeta} = \frac{p^2}{s^2} - 1 \quad \text{Nosé-Hoover} \]

Nosé Oscillator with $H = 0$;
What is the Best Initial Value of $s$?

Vary $s$ to maximize the number of crossings in 100,000,000 steps

Initial condition:
\[ (q, p, \zeta) = (2.4, 0, 0) \]
\[ s = 2^n, \quad -11 < n < 11 \]

\[ s = 2^{-4} = 0.06250 \]
\[ s = e^{-2.88} = 0.056135 \]
\[ H = 0 \]
\[ s = 2^{-5} = 0.03125 \]

One more test:
Hamiltonian Systems Are Conservative!
Energy must be conserved in a Hamiltonian System!

Error in the Hamiltonian

<table>
<thead>
<tr>
<th>s</th>
<th>Double precision</th>
<th>Quadruple precision</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.062500</td>
<td>-3.2702049634902863E-05</td>
<td>-4.997996331072645417188952190057969E-13</td>
</tr>
<tr>
<td>0.056135*</td>
<td>-3.3960281428768724E-05</td>
<td>9.510041726002260340923701275951037E-13</td>
</tr>
<tr>
<td>0.031250</td>
<td>-3.3617859108492354E-05</td>
<td>-3.2076744813156868420757831612092E-11</td>
</tr>
</tbody>
</table>

* This is the value that corresponds to a vanishing Hamiltonian.

Lyapunov Exponents and Chaos

The instantaneous Lyapunov exponent measures the exponential growth of the separation of two nearby neighboring trajectories, \( \delta = \lambda \delta \). We choose a neighboring satellite trajectory \( x_s \) constrained a distance \( \delta = 0.000001 \) from a reference trajectory \( x_r \). After a time step \( dt \), the distance is rescaled to \( \delta \) providing a measure of the growth of separation during that interval.

\[
x_r = x_r + g(x_s - x_r) \quad \text{where} \quad g = \frac{\delta}{\sqrt{(x_s - x_r)^2}} \approx e^{-\lambda dt}
\]

\[
\lambda(t) = -\frac{\ln(g)}{dt}
\]

This instantaneous or local Lyapunov exponent averaged over a very long trajectory gives the Lyapunov exponent for the phase-space trajectory.

\[
\lambda = \frac{\sum (\lambda(t)dt)}{\sum (dt)}
\]

This Lyapunov is the maximum of the four Lyapunov exponent measuring the growth in phase space for the oscillator. Bill will present algorithms for calculating the full spectrum in a later lecture. The solutions of the thermostated oscillator are characterized by the value of the maximum exponent:

\[
\lambda > 0 \text{ chaotic solution}; \quad \lambda = 0 \text{ conservative tori}.
\]
Calculated Results for the Lyapunov Exponents using the time-scaling factors

\[
\begin{align*}
\lambda_N &= 0.0468 ; \quad d\mathcal{H} = 0.000197 \\
\lambda_{NH} &= 0.0140 ; \quad d\mathcal{H} = 0.000006 \\
<1/s>_{Nosé} \lambda_{NH} &= 0.0140 \times (3.28) = 0.0460 \\
<s>_{NH} \lambda_{Nosé} &= 0.305 \times (0.0468) = 0.0143 \\
\lambda_{Nosé} &= 3.28 \lambda_{NH} ; \quad t_{NH} = 3.28 t_{Nosé}
\end{align*}
\]

Student Problem

\[
\begin{align*}
<s> <1/s> &= (0.661) \times 3.28 = 2.17 \\
\text{which is true for all Nosé s-values calculated.}
\end{align*}
\]

\[
\begin{align*}
<s> <1/s> &= (0.305) \times 964.0 = 294.0 \\
\text{Is this true for any Nosé-Hoover s-values?}
\end{align*}
\]

The Nosé Oscillator Local Lyapunov Exponent Is Another Measure of Stiffness

Adaptive Integration

Quadruple precision with error band between \(10^{-28}\) and \(10^{-24}\)

The corresponding time 16.42 for the Nosé-Hoover #2 dynamics and corresponds to a broad minimum in \(s\). The amplitudes and the required number of steps differ by about six orders of magnitude. Fixed \(dt = 0.001\)!

Nosé Oscillator \(\dot{q} = \left(\frac{p}{s^2}\right) ; \quad \dot{p} = -q ; \quad \dot{s} = \xi ; \quad \dot{\xi} = \left(\frac{p^2}{s^3}\right) - 1/s\)

\((q, p, s, \xi)_0 = (2.4, 0, e^{-2.88}, 0) ; \quad \delta = 10^{-6}\)
The Nosé-Hoover #1 Oscillator Local Lyapunov Exponent Is Orders of Magnitude Smaller

Adaptive Integration
Double precision with error band between $10^{-16}$ and $10^{-14}$

Typical Local Lyapunov Peaks for the Nosé-Hoover Oscillator

Resolution of Lyapunov Peak for the Nosé-Hoover Oscillator

\[
\begin{align*}
\dot{q} &= p/s ; \quad \dot{p} = -sq ; \quad \dot{s} = s\zeta ; \quad \dot{\zeta} = p^2/s^2 - 1 \quad \# 1 \\
p/s &\rightarrow p ; \quad \dot{q} = p ; \quad \dot{p} = -q - \zeta p ; \quad \dot{\zeta} = p^2 - 1 . \quad \# 2
\end{align*}
\]

Correlation of $|p|$, $\lambda$, and $s$ with $dt$

Linear correlation on the log-log plot for all 3 variables and both oscillators!
Analyzing the Results With Histograms

We can understand better the motion of the oscillators by looking at the probability density for the logarithm of the time scaling variable and the time steps.

- With a variable time step care must be taken in computing the probability density for the time scaling variable $s$. A trajectory value, $\ln(s_{\text{traj}})$, is associated with the $i$th bin if

$$\ln(s)_i \leq \ln(s_{\text{traj}}) < \ln(s)_{i+1}.$$

However, the probability that the trajectory values will fall into the bin is equal to the fraction of the time that the trajectory spends within the bin. Thus, the probability density in the $i$th bin is given by

$$p(\ln(s)_i \leq \ln(s_{\text{traj}}) < \ln(s)_{i+1}) = \sum_{i=1}^{N} \Delta t_i / \sum \Delta t_j$$

for 4,000,000,000 time steps.

The time-step histograms show that there were three values of $dt$ used for the smooth Nosé-Hoover equations and more than 20 values of $dt$ (a factor of a million) for the Nosé oscillator. All these time steps are adaptive.

Probability Distribution for the Time-Scaling Variable

$p_{NH} \propto s$ ; $p_{\text{Nosé}} \propto s^2$
The time-step histograms show that there were three different time intervals used for the smooth Nosé-Hoover equations and there are about twenty different time intervals used for the Nosé oscillator. Keep in mind that in each interval the timestep is halved and doubled in the adaptive method.

Results and Conclusions: Adaptive Integration for Nosé and Nosé-Hoover Oscillators

- Nosé Oscillator can be treated with the Adaptive 4th-Order Runge-Kutta Algorithm
- The trajectories for the Nosé and Nosé-Hoover oscillators are the same but progress at different rates
- Trajectory intervals measured between two \( p = 0 \) sections show that the sections are identical for the two oscillators
- The time scaling variable provides the connection between the Lyapunov exponents for the two oscillators
- The local Lyapunov exponent is an important measure of stiffness
- We find linear correlations on log-log plots of the the momentum and time scaling versus time.
The Three Sections of the Nosé & NH Generate Conservative Tori and Points in the Chaotic Sea

Initial periodic orbits \( \{q, p, \zeta\} = (\pm 1.2144, 0, 0) \) incremented by \( \pm 0.1 \). The 12th increment \((2.4144, 0, 0)\) lies in the chaotic sea.

Time Scaling in Nosé and Nosé-Hoover Mechanics

Initial condition for a periodic orbit
\[ \{q, p, s, \zeta\}_0 = \left( 1.2145, 0, e^{-q^2/2}, 0 \right) \]

At time \( t_{NH2} = 4.018 \)
\[ \{q, p, s, \zeta\} = \{-0.2538, 1.5000, 0.3066, -0.2237\} \]
\[ p_{NH2} = \{p_{Nose} / s\}; p_{Nose} = p_{NH2}s = 0.4599 \]
The time scaling variable $s$ oscillates twice during a single period of the oscillator.

\[ \tau_N = 1.0757 \times 2 \quad \tau_{NH} = 2.8038 \times 2 \]

Lyapunov Exponents for Nosé & Nosé-Hoover Mechanics

$< \lambda >_{\text{Nosé}}$ differs from $< \lambda >_{\text{NH}}$ by a factor of $< s >_{\text{NH}}$.

\[ < \lambda > = .0475 \quad \text{Nosé} \]
\[ < \lambda > = .01414 \quad \text{NH1} \]
\[ < \lambda > = .01392 \quad \text{NH2} \]
"Springy Pendulum" with a Regular Orbit

\[
\{ x, y, p_x, p_y \}_0 = \{ 1, 0, 0, 0 \}
\]

\[
\{ r, \theta, p_r, p_\theta \}_0 = \{ 1, \pi/2, 0, 0 \}
\]

"Springy Pendulum" with a Chaotic Orbit

\[
\{ x, y, p_x, p_y \}_0 = \{ 2^{-1/2}, 2^{-1/2}, 0, 0 \}
\]

\[
\{ r, \theta, p_r, p_\theta \}_0 = \{ 1, 3\pi/4, 0, 0 \}
\]
Non-Ergodicity of the Springy Pendulum

\[ \theta \]

\[ -\pi/2 \quad +\pi/2 \]

\[ -\pi/4 \quad +\pi/4 \]

\[ X \]

\[ Y \]

\[ P_x \]

\[ P_y \]

\[ r \]

\[ r \]

\[ -1.5 \quad -1.0 \quad -0.5 \quad 0.0 \quad 0.5 \quad 1.0 \quad 1.5 \]

\[ 0.5 \quad 1.0 \quad 1.5 \]

\[ -2.0 \quad -1.5 \quad -1.0 \quad -0.5 \quad 0.0 \quad 0.5 \quad 1.0 \quad 1.5 \quad 2.0 \]

\[ P_r \]

\[ P_\theta \]