Homework #1: ODE’ s and Particle Dynamics

Huai-Ping Lee

September 20, 2007

Implementation

For both problems, we implemented two kinds of integrators: Euler method and 4th order Runge-Kutta method. The code can be found at ../code/PBM.zip. (This is a newer version, fixed some bugs in the demo version.)

Physical and Numerical Stability

Problem A. For a particle system that is only affected by gravity and drag force, we have the equations

$$\begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} = J \begin{bmatrix} x \\ \dot{x} \end{bmatrix} + \begin{bmatrix} v_o \\ g \end{bmatrix},$$

where *J* is the Jacobian matrix. The physical stability of the linear system depends on the eigenvalues of *J*: every solution is stable if all the eigenvalues have negative real part, or all the eigenvalues are less than or equal to zero if they are all real numbers [1]. In our case,

$$J = \begin{bmatrix} 0 & 1 \\ 0 & -k_d/m \end{bmatrix},$$

and the eigenvalues are $\lambda = 0, -k_d/m$. Therefore this linear system is physically stable as long as $k_d \geq 0$. In this case there is no equilibrium state, so we will not discuss its numerical stability.

Problem B. For spring-mass systems, the equations become

$$\begin{aligned} \begin{bmatrix} \dot{x} \\ \ddot{x} \end{bmatrix} &= J \begin{bmatrix} x \\ \dot{x} \end{bmatrix} \\ \ddot{x} &= -\frac{k_s}{m} x. \end{aligned}$$

We can then analyze the numerical stability of different methods using this equation. The first method we used is Euler step:

$$\begin{aligned} X_{i+1} &= X_i + h\dot{X}_i \\ &= X_i + h(JX_i) \\ &= (I + hJ)X_i, \end{aligned}$$

where $X = \begin{bmatrix} x \\ \dot{x} \end{bmatrix}$. The result is an exponential function that converges only if the magnitude of eigenvalue of $(I + hJ)$ is less than or equal to 1, i.e., $|1 + h\lambda| \leq 1$, where λ is the eigenvalue of *J*.

Another numerical method implemented is fourth-order Runge-Kutta. In this case,

$$\begin{aligned}
k_1 &= h\lambda X_i \\
k_2 &= h\lambda(1 + h\lambda/2)X_i \\
k_3 &= h\lambda(1 + (h\lambda/2)(1 + h\lambda/2))X_i \\
k_4 &= h\lambda(1 + h\lambda(1 + (h\lambda/2)(1 + h\lambda/2))X_i \\
X_{i+1} &= \left(1 + h\lambda + \frac{1}{2}(h\lambda)^2 + \frac{1}{6}(h\lambda)^3 + \frac{1}{24}(h\lambda)^4\right)X_i = q(h\lambda)X_i.
\end{aligned}$$

(Here we write λ in place of J for simplicity.) So we need $|q(h\lambda)| \leq 1$ to assure stability.

In our case,

$$J = \left[\begin{array}{cc} 0 & 1 \\ -\frac{k_s}{m} & 0 \end{array}\right],$$

and λ = �$\sqrt{-k_s/m}$ = �$i\sqrt{k_s/m}$, where $i = \sqrt{-1}$. Therefore $|1 + h\lambda| = \sqrt{1 + h^2(k_s/m)}$ is always greater than 1, and explicit Euler method is always unstable *if there is no damping* (!!) For RK4 method, it is hard to derive an inequality, but we can verify our example situation. Suppose we have $k_s = 30$, $m = 3$, and $h = 0.1$, then we can calculate $|q(h\lambda)| \approx 0.999993142337597 < 1$. So the RK4 integrator is still numerically stable under this setting.

In general, if the equation of $\ddot{x}$ have some damping term dependent on $\dot{x}$, we will then have the equation $\ddot{x} + \frac{k_d}{m}\dot{x} + \frac{k_s}{m}x = 0$, and the Jacobian matrix becomes

$$J = \left[\begin{array}{cc} 0 & 1 \\ -\frac{k_s}{m} & -\frac{k_d}{m} \end{array}\right],$$

so the eigenvalues are the roots of $\lambda^2 + \frac{k_d}{m}\lambda + \frac{k_s}{m} = 0$, namely

$$\lambda = \frac{-\frac{k_d}{m} \pm \sqrt{\frac{k_d}{m}^2 - 4\frac{k_s}{m}}}{2}.$$

Then there is a chance that $|1 + h\lambda| \leq 1$ (λ can be a complex number) so that the stability of explicit Euler method is assured.

Results

Here are some plots from our experiments. This is a case when Euler method blows up while RK4 is still stable.

Figure 1: Euler method with $k_s = 30$, $m = 3$, and $h = 0.1$

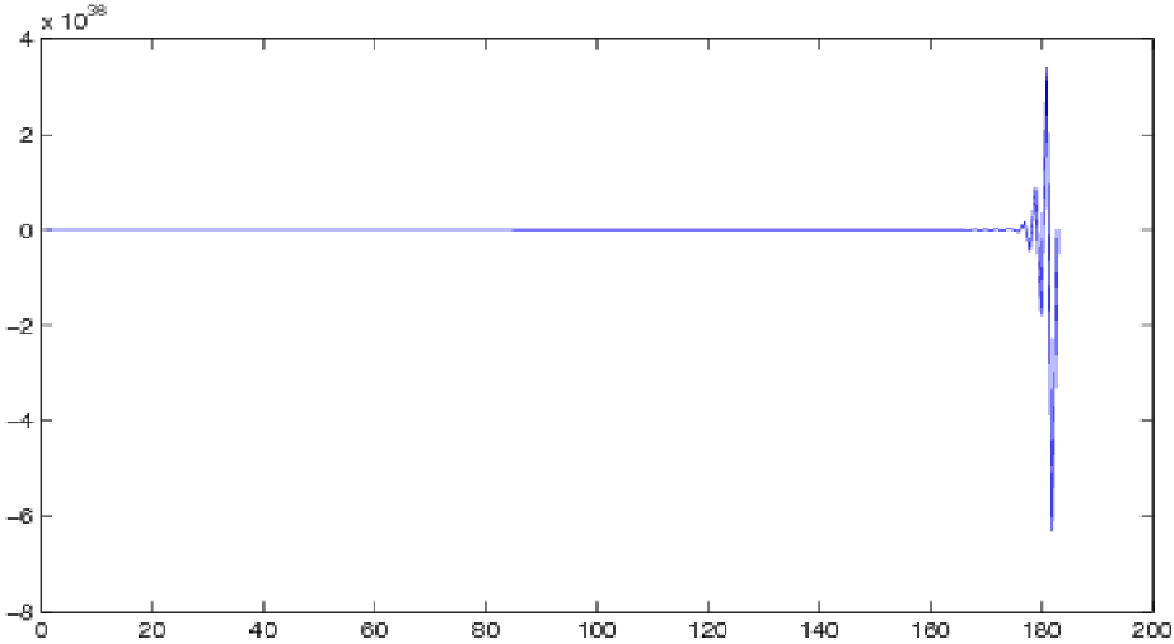
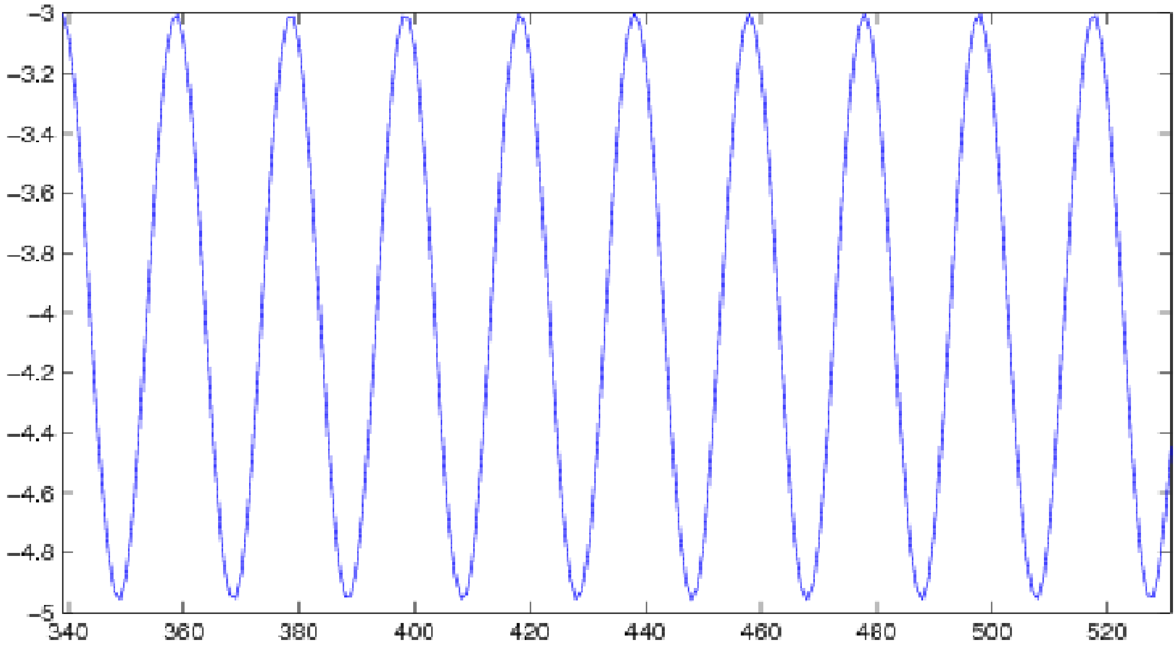


Figure 2: Fourth-order Runge-Kutta method with k_s = 30, m = 3, and h = 0.1



Summary

RK4 is superior to Euler method in terms of accuracy and stability. However, it is not free. Since the RK4 method evaluates the integral four times to compute the result, it takes at least four times as much work as an Euler integrator.

References

[1] David Eberly. Stability analysis for systems of differential equations, February 2003.