Homework - Introduction to Frontiers of Computational Science

Ridlo W. Wibowo || 1215011069

October 18, 2013

Problem 1.
Solve this simple oscillator equation.

$$m\ddot{x} = -kx$$

with initial condition $x(0) = 1$ and $\dot{x}(0) = 0$, theoretically and numerically.

Theoretically
general solution for a function $x = e^{rt}$

$$\begin{aligned}
\frac{d^2x}{dt^2} + \frac{k}{m}x &= 0 \\
r^2e^{rt} + \frac{k}{m}e^{rt} &= 0 \\
e^{rt}(r^2 + \frac{k}{m}) &= 0 \\
r^2 + \frac{k}{m} &= 0 \\
r &= \pm\sqrt{-\frac{k}{m}} = \pm i\sqrt{\frac{k}{m}} \\
r &= \pm i\omega
\end{aligned}$$

so the general solution by linear combination is

$$x(t) = c_1e^{+i\omega t} + c_2e^{-\omega t}$$

rewrite above equation using euler formula

$$\begin{aligned}
x(t) &= (c_1 + c_2)\cos(\omega t) + i(c_1 - c_2)\sin(\omega t) \\
x(t) &= A\cos(\omega t) + B\sin(\omega t)
\end{aligned}$$

or we can use simpler form:

$$\begin{aligned} x(t) &= C\cos(\omega t+\varphi) \\ \dot{x}(t) &= -C\omega\sin(\omega t+\varphi) \end{aligned}$$

using the initial condition we will get the solution:

$$x(t) = \cos(\omega t)$$

Numerically
we can use Taylor expansions:

$$\begin{aligned} x(t+\Delta t) &= x(t) + v(t)\Delta t + \frac{a(t)\Delta t^2}{2} + \frac{b(t)\Delta t^3}{6} + \mathcal{O}(\Delta t^4) \\ x(t-\Delta t) &= x(t) - v(t)\Delta t + \frac{a(t)\Delta t^2}{2} - \frac{b(t)\Delta t^3}{6} + \mathcal{O}(\Delta t^4) \end{aligned}$$

where x is the position, $v = \dot{x}$ the velocity, $a = \ddot{x}$ the acceleration and b the jerk (third derivative of the position with respect to the time) t.
Adding these two expansions gives

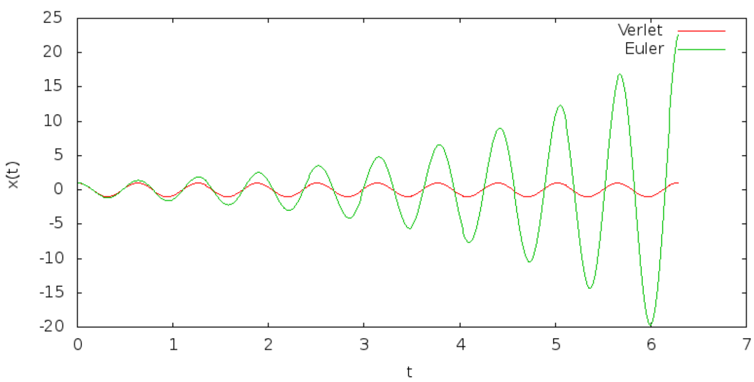
$$x(t+\Delta t) = 2x(t) - x(t-\Delta t) + a(t)\Delta t^2 + \mathcal{O}(\Delta t^4).$$

this method which is so called Verlet algorithm is an order more accurate than integration by simple Taylor expansion alone because we can see that the first and third-order terms from the Taylor expansion cancel out.
for this problem $a(t) = -\omega^2 x(t)$, and using this method we need two position for one iteration, so for the first time step we can use Taylor expansion to get the position

$$x_1 = x_0 + v_0\Delta t + \frac{1}{2}a_0\Delta t^2(\mathcal{O}(\Delta t^3))$$

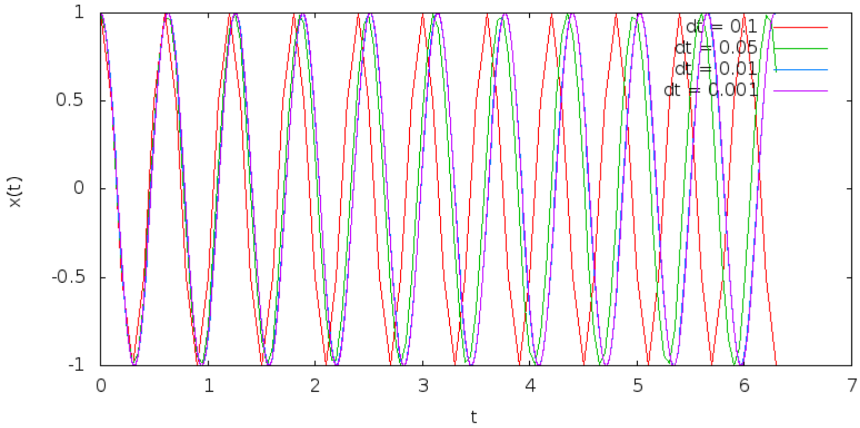
The error is not considered a problem because on a simulation of over a large amount of timesteps, the error on the first timestep is only a negligibly small amount of the total error.

Calculation using initial condition above with $\omega = 10$ and $\Delta t = 0.01$ (10 periods):



Comparison between Verlet and Euler method for the same problem.

Calculation using Verlet algorithm with different Δt:



Comparison of the result from Verlet algorthm using different Δt.

Program

```
#include <iostream>
#include <stdlib.h>
#include <math.h>
#include <fstream>
using namespace std;

void plot();
int main(){
    double omega = 10.;
    double x0 = 1.;
    double v0 = 0.;
    double ti = 0., tf = 2.*M_PI;
    double dt = 0.01, x=x0, v=v0, t=ti, prev, next, a;

    double N = (tf - ti)/dt;

    // Verlet method
    ofstream out("output.txt");
    out << t << " " << x << endl;
    a = -1.*omega*omega*x;
    prev = x;
    next = x + v*dt + 0.5*a*dt*dt;
    t = t+dt;
    out << t << " " << next << endl;
    for (int i=1; i<=N; i++){
        a = -1*omega*omega*next;
        x = 2*next - prev + a*dt*dt;
        prev = next;
        next = x;
        t = t+dt;
        out << t << " " << x << endl;
```

```
    }
    out.close();

    // Euler method
    ofstream out2("output2.txt");
    x=x0; v=v0; t=ti;
    out2 << t << " " << x << endl;
    for (int i=0;i<=N;i++){
        a = -1.*omega*omega*x;
        x = x + v*dt;
        v = v + a*dt;
        t = t + dt;
        out2 << t << " " << x << endl;
    }
    out2.close();

    plot();
    return 0;
}

void plot(){
    ofstream ploter("inp.plt");
    ploter << "#gnuplot input file\n";
    ploter << "set term png size 600,400\n";
    ploter << "set output \"verlet.png\"\n";
    ploter << "set xlabel \"t\"\n";
    ploter << "set ylabel \"x(t)\"\n";
    ploter << "plot \"output.txt\" u 1:2 w l t \"Verlet\", \"output2.txt\"
        u 1:2 w l t \"Euler\"\n";
    ploter.close();

    system("gnuplot inp.plt");
}
```