Michael Cowan

Computational Physics

Problem Set 3

Chapter 2

15. (see attached page) In the problem we showed that using the Euler method to calculate the motion of a pendulum led to a monotonically increasing value for the energy. This was done by showing that Ei+1=Ei + x, where x is an expression that is always ≥0.

17. In this problem we compared three different methods for calculating the period of a pendulum. The first approximation was the standard small-angle approximation, . The second approximation came from manipulating the conservation of energy equation to produce an elliptic integral of the first kind, . The third approximation used the Verlet method to produce even more accurate results for the period. For all three methods, the system is normalized so that L/g = 1.

With regards to the Verlet method, in order to find ri+1, it is necessary to know both ri and ri-1. r1 is obviously the initial position, but to find r2, we must have a value for r0. This is typically done by doing a backwards Euler step from r1. In this problem, though, v1 = 0 so the calculation for the backwards Euler step gives r0 = r1.

As a final step, we found the initial angle for which each of the first two approximations exceeds 10% error when compared to the Verlet method.

Code:

1. tau=input('Enter the time step: ');
2. maxtime=60/tau;
3. for i=1:179
4. thetam(i)=i\*pi/180; %set the value of the initial angle, in radians
5. approx1(i)=2\*pi; %small angle approximation
6. approx2(i)=2\*pi\*(1+(thetam(i)).^2/16); %second approximation method
7. %Verlet method for calculating r
8. for n=1:maxtime
9. if (n==1 || n==2)
10. r(n)=thetam(i); %set first and second steps
11. else
12. r(n)=2\*r(n-1)-r(n-2)+tau^2\*(-sin(r(n-1))); %calculate subsequent steps
13. if (r(n)<0) %stop calculating after 1/4 cycle
14. break;
15. end
16. end
17. end
18. per(i)=(n\*tau)\*4; %calculate the period of the Verlet method
19. end
20. for x=1:179
21. dif\_1(x)=per(x)-approx1(x); %find the difference between the approximation and the Verlet method
22. if (dif\_1(x)>0.1\*per(x)) %stop when the difference exceeds 10%
23. break;
24. end
25. end
26. for y=1:179
27. dif\_2(y)=per(y)-approx2(y); %find the difference between the approximation and the Verlet method
28. if (dif\_2(y)>0.1\*per(y)) %stop when the difference exceeds 10%
29. break;
30. end
31. end
32. thetam=thetam.\*(180/pi); %convert thetam to degrees
33. fprintf('The first approximation reaches 10 percent error at %g degrees\n',x);
34. fprintf('The second approximation reaches 10 percent error at %g degrees\n',y);
35. plot(thetam,approx1,'-',thetam,approx2,'--',thetam,per,'o');
36. xlabel('Initial angle (degrees)');
37. ylabel('Period (s)');
38. title('Graph of period vs initial angle');
39. legend('1st approximation','2nd approximation','Verlet method','location','northwest');

Results:

Enter the time step: 0.01

The first approximation reaches 10 percent error at 69 degrees

The second approximation reaches 10 percent error at 128 degrees

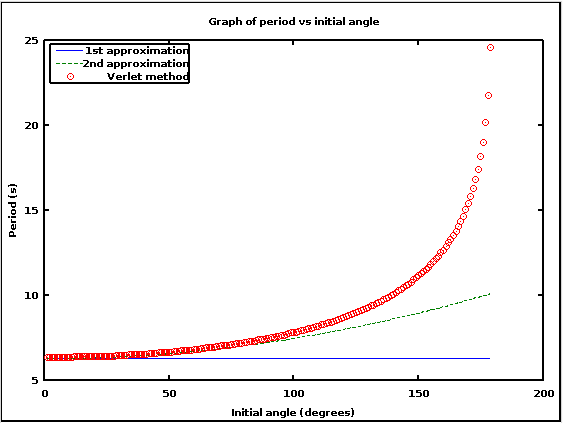


Figure 1: Comparison of period vs initial angle for three different methods of calculation

21. The purpose of this problem is to simulate a pendulum with a harmonically driven pivot. The question asks us to show that when the amplitude of the driving force is large compared to g that it is possible for the pendulum to be stable in the inverted position.

Code (portions copied from the **pendul.m** downloaded file are in italics):

1. clear all; % Clear the memory
2. *%\* Set initial position and velocity of pendulum*
3. *theta0 = input('Enter initial angle (in degrees): ');*
4. *theta = theta0\*pi/180; % Convert angle to radians*
5. *omega = 0; % Set the initial velocity*
6. *%\* Set the physical constants and other variables*
7. *g\_over\_L=1; % The constant g*
8. *time = 0; % Initial time*
9. A0=input('Input the amplitude of the driving force as a multiple of g: ');
10. Td=input('Input the period of the driving force (s): ');
11. *tau=input('Input the time step (s): ');*
12. *%\* Take one backward step to start Verlet*
13. *accel = -(g\_over\_L)\*sin(theta); % Gravitational acceleration*
14. *theta\_old = theta - omega\*tau + 0.5\*tau^2\*accel;*
15. *%\* Loop over desired number of steps with given time step*
16. *% and numerical method*
17. maxtime=100\*Td/tau;
18. for istep=1:maxtime
19. *%\* Record angle and time for plotting*
20. *t\_plot(istep) = time;*
21. *th\_plot(istep) = theta\*180/pi; % Convert angle to degrees*
22. *time = time + tau;*
23. ad=A0\*sin(2\*pi\*time/Td); %acceleration from the driving force
24. accel = -((g\_over\_L+ad))\*sin(theta); % total acceleration
25. *theta\_new = 2\*theta - theta\_old + tau^2\*accel;*
26. *theta\_old = theta; % Verlet method*
27. *theta = theta\_new;*
28. *end*
29. *plot(t\_plot,th\_plot,'-');*
30. *xlabel('Time'); ylabel('\theta (degrees)');*

Result:

Enter initial angle (in degrees): 170

Input the amplitude of the driving force as a multiple of g: 100

Input the period of the driving force (s): .2

Input the time step (s): .004

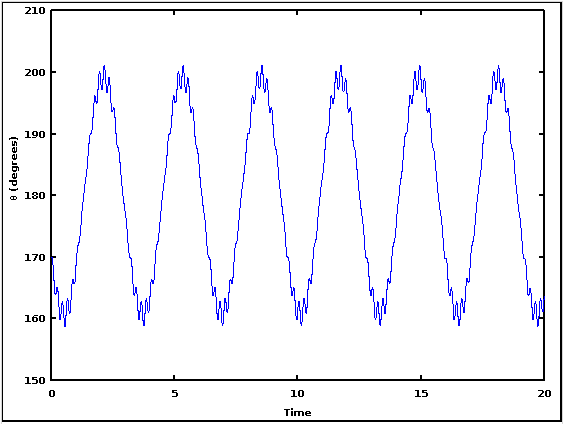


Figure 2: Angle of the driven pendulum as a function of time

The graph shows that the driven pendulum can be stable in the in the inverted position for a large enough driving force. I also found, through trying other values, that the pendulum needs to start at a fairly large angle, otherwise it will just oscillate around the non-inverted position.