bayam.org

Implicit Euler with Newton-Raphson

Course Name

– • – • –

Last Revision: September 29, 2015

Table of Contents

Abstract

These notes are intended as a resource for myself and anyone interested in the material. If you spot any errors or would like to contribute, please contact me directly.

1 Mass-Spring-Damper

A system is defined with:

$$M\ddot{x} + B\dot{x} + Kx = F$$

where: M, B, K are mass, damping, and stiffness respectively.

2 Newton-Raphson

Newton-Raphson is used to solve $g(x) = 0$ by iterating:

$$x_{k+1} = x_k - \frac{g(x_k)}{g'(x(k))}$$

3 Implicit Euler

Solution to: $\dot{y} = f(t, y)$ can be found using:

$$y_{k+1} = y_k + hf(t_{k+1}, y_{k+1})$$

where h is the time step.
Implicit Euler can not be solved immediately since there is y_{k+1} in both left and right side of the equation. The equation need to be changed to:

$$g(y_{k+1}) = y_{k+1} - y_k - hf(t_{k+1}, y_{k+1}) = 0$$

Now, we can use Newton-Raphson to solve $g(x)$.

4 Implementation

$$\begin{aligned} M\ddot{x} + B\dot{x} + Kx &= F \\ \ddot{x} &= \frac{F - B\dot{x} + Kx}{M} \end{aligned}$$

Solve $\ddot{x}$ to get $\dot{x}$ using the implicit Euler:

$$\begin{aligned} \dot{x}_{k+1} - \dot{x}_k - h\ddot{x}_{k+1} &= 0 \\ \dot{x}_{k+1} - \dot{x}_k - h\left(\frac{F - B\dot{x}_{k+1} + Kx_{k+1}}{M}\right) &= 0 \\ g(\dot{x}_{k+1}) = \frac{M}{h}\dot{x}_{k+1} - \frac{M}{h}\dot{x}_k - F + B\dot{x}_{k+1} - Kx_{k+1} &= 0 \\ g(\dot{x}_{k+1}) = \frac{M}{h}\dot{x}_{k+1} - \frac{M}{h}\dot{x}_k - F + B\dot{x}_{k+1} - K(x_k + h\dot{x}_{k+1}) &= 0 \end{aligned}$$

and:

$$g'(\dot{x}_{k+1}) = \frac{M}{h} + B + Kh$$

We can then proceed with Newton-Raphson.

5 Some Insights

Implicit Euler has damping, even when you tried to simulate system with no damping. This is a well known characteristic of backward-Euler / implicit methods. To understand this you can write the update equations in phase-space matrix form and calculate the eigenvalues of the update matrix. You will see that it is always < 1 for a simple spring. Miles Macklin explained this to me.

6 MATLAB

```
% ==========================================================================
% Solve M x_ddot + B x_dot + K x = F
% Implicit Euler with Newton Raphson
% ==========================================================================

function implicit_euler_newton_raphson()
    clear all;
    close all;
    clc;

    figure;
    hold;

    K = 100;           % spring stiffness. N/m
    M = 5;             % mass, kg
    B = 2*sqrt(K*M);  %critical damping
    %B = 0;

    % initial condition:
    x0 = 0.1;
    v0 = 0;
    F0 = 0;

    h = 0.01;
    t_start = 0;
    t_end = 10;
    t=t_start:h:t_end;

    % Use ode45, 1kHz as ground truth:
    [tode45,xode45]=ode45(@msd, [t_start:0.001:t_end], [x0 v0], [], ...
                              M, B, K);
    plot(tode45,xode45(:,1), '--r')

    for k = 1 : length(t)
        % Newton-Raphosn relies on good intial value
        % As an initial guess, a 1-step forward Euler is used
```

```
        %v1_h = v1_hat(M, B, K, 0, x0, v0, h);
        %v1 = v1_h

        % Initializing v1 = 0 also works. It just needs more iterations
        v1 = 0;
        v1_ = inf;

        % Newton-Rapshon
        % x(k+1) = x(k) - g(x(k))/g'(x(k))
        % Since the system is linear, g'(x(k)) = constant
        g_d =  M/h + B + K * h;
        while(1) % Newton-Raphson iteration
            v1 = v1 - (g(M, B, K, F0, x0, v0, v1, h) / g_d);
            v1_ = v1;
            if abs(v1 - v1_) <0.0001
                break;
            end
        end

        x1 = x0 + v0 * h;
        x0 = x1;
        v0 = v1;

        x(k, :) = [x1 v1];
    end
    plot(t, x(:,1), 'b')

    legend('ode45', 'Implicit Euler');
    xlabel('Time (s)')
end

function output = v1_hat(M, B, K, F, x0, v0, h)
    v0_dot = (F - B * v0 - K * x0) / M;
    output = v0 + h * v0_dot;
end

function output = g(M, B, K, F, x0, v0, v1, h)
    output = M*(v1-v0)/h-F+B*v1+K*(x0+h*v0);
end

function xdot=msd(t,x, M, B, K)
    xdot_1 = x(2);
    xdot_2 = -(B/M)*x(2) - (K/M)*x(1);

    xdot = [xdot_1 ; xdot_2 ];
end
}
```