MTH 659 Computational Wave Propagation Assignment 3

Sandeep Koranne

June 3, 2017

Introduction

Consider the wave equation given by:

$$u_{tt} - \Delta u = 0, u \in [0, T] \times \Omega, \Omega = (0, 1)$$

and its variational formulation as discussed in class. For $r = 1, 2$, compute the mass and stiffness matrix using exact integrals involved in the semi-discrete formulation.

1 P1 and P2 FEM Mass Matrix and Stiffness Matrix

Consider the P1 FEM Lagrange polynomial as defined by the requirement on the reference element that $\psi_i(x_j) = \delta_{ij}$. For $\Omega = [0, 1]$, the support for the P1 FEM is restricted to exactly two nodes, so we can therefore say:

$$M_{ij} = \int_{\Omega} \psi_i(x)\psi_j(x)dx = \int_0^1 \psi_i(x)\psi_j(x)dx = \int_{x_{p-1}}^{x_{p+1}} \psi_i(x)\psi_j(x)dx$$

The equation for $\psi(x)$ can be constructed from Lagrange formula. We want $\psi_1(0) = 1, \psi_1(1) = 0$, similarly $\psi_2(0) = 0, \psi_2(1) = 1$. Using Lagrange formula we get

$$\begin{aligned} \psi_1(x) &= \frac{x_2 - x}{x_2 - x_1} = \frac{1 - x}{1 - 0} = 1 - x \\ \psi_2(x) &= \frac{x - x_1}{x_2 - x_1} = \frac{x - 0}{1 - 0} = x \end{aligned} \tag{1}$$

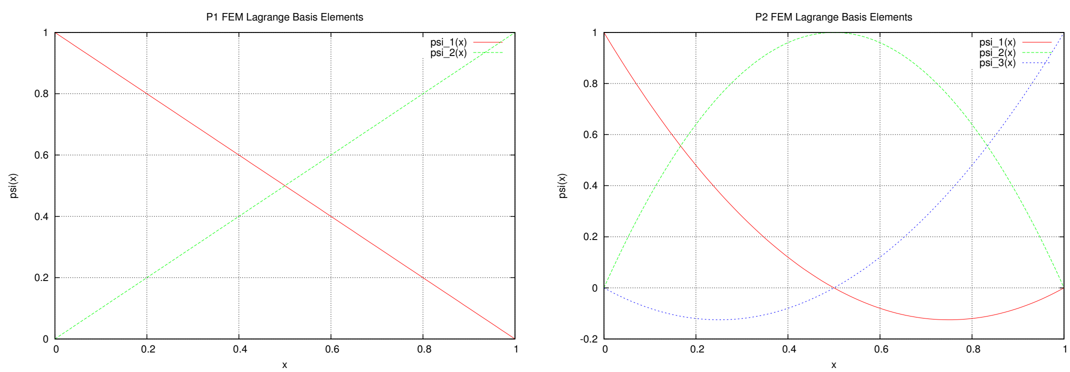
Now we can calculate the mass-matrix and stiffness matrix using integration, and we get

$$M = \frac{h}{6}\begin{bmatrix} 2 & 1 \\ 1 & 2 \end{bmatrix} K = \frac{1}{h}\begin{bmatrix} 1 & -1 \\ -1 & 1 \end{bmatrix}$$

Similarly for P2 FEM, we first construct the Lagrange polynomials. The points are 0, 0.5 and 1, so we want $\psi_1(0) = 1, \psi_1(0.5) = \psi_1(1) = 0$. This gives

$$
\begin{aligned}
\psi_1(x) &= \frac{(x-0.5)(x-1)}{(0-0.5)(0-1)} = (2x-1)(x-1) \\
\psi_2(x) &= \frac{(x-0)(x-1)}{(0.5-0)(0.5-1)} = (4x)(1-x) \\
\psi_3(x) &= \frac{(x-0)(x-0.5)}{(1-0)(1-0.5)} = (x)(2x-1)
\end{aligned}
$$

These Lagrange basis elements are shown in Figure 1. Using the Maxima computer algebra system we



(a) Lagrange P1 elements

(b) Lagrange P2 elements

Figure 1: Lagrange basis elements for P1 and P2 FEM.

computed the mass-matrix integrals and stiffness matrix integrals as shown below.

```
(%i19) h1: (2*x-1)*(x-1);
(%o19)                          (x - 1) (2 x - 1)
(%i20) h2 : 4*x*(1-x);
(%o20)                             4 (1 - x) x
(%i21) h3: x*(2*x-1);
(%o21)                             x (2 x - 1)
(%i22) integrate(h1*h1,x,0,1);
                                       2
(%o22)                                 --
                                       15
(%i30) integrate(diff(h3,x)*diff(h1,x),x,0,1);
                                        1
(%o30)                                  -
                                        3
(%i31) integrate(diff(h1,x)*diff(h1,x),x,0,1);
                                        7
(%o31)                                  -
```

```
                                                3
(%i32) integrate(diff(h1,x)*diff(h2,x),x,0,1);
                                                8
(%o32)                                        - -
                                                3
(%i33) integrate(diff(h2,x)*diff(h2,x),x,0,1);
                                              16
(%o33)                                        --
                                              3
```

We get:

$$M = \frac{h}{30}\begin{bmatrix} 4 & 2 & -1 \\ 2 & 16 & 2 \\ -1 & 2 & 4 \end{bmatrix} K = \frac{1}{3h}\begin{bmatrix} 7 & -8 & 1 \\ -8 & 16 & -8 \\ 1 & -8 & 7 \end{bmatrix}$$

2 Mass Lumping Using the Quadrature Rule

For P1, the Trapezoidal integration provides the quadrature as the points are the end points of the element. We get

$$M = \frac{h}{3}\begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}$$

For P2 FEM, we use Simpson integration, with the mid point of the reference element, to get

$$M = \frac{h}{30}\begin{bmatrix} 1 & 0 & 0 \\ 0 & 20 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

3 Discrete Equation

Simpson's rule for integration is:

$$\int_a^b f \approx \frac{b-a}{6}[f(a) + 4f(\frac{a+b}{2}) + f(b)]$$

For the middle coordinate $x_{p,1}$, the equation can be derived from:

$$u_{tt} = -M^{-1}KU$$

Using the Simpson's quadrature on the integral for K, we get

$$\frac{d^2 u_{p+1/2}}{dt^2}(t) = -M^{-1}KU$$

Since M has factor h, and K has factor $1/h$ we can remove the $\frac{1}{h^2}$ factor outside and get

$$\frac{d^2 u_{p+1/2}}{dt^2}(t) = \frac{30}{18h^2}[8u_p(t) - 16u_{p+1/2}(t) + 8u_{p+1}(t)]$$

Rearranging we get:

$$\frac{d^2u_{p+1/2}}{dt^2}(t) = \frac{4}{h^2}[u_p(t) - 2u_{p+1/2}(t) + u_{p+1}(t)]$$

as required.
For the elements nodes we again have:

$$\frac{d^2u_p}{dt^2}(t) = -M^{-1}KU$$

But this time we have to use $u_{p-1/2}$, $u_{p+1/2}$, u_{p-1} and u_{p+1} as the adjacent nodes. Given that K is:

$$K = \frac{1}{3h}\begin{bmatrix} 7 & -8 & 1 \\ -8 & 16 & -8 \\ 1 & -8 & 7 \end{bmatrix}$$

Using the first row of K, we get:

$$\frac{d^2u_p}{dt^2}(t) = \frac{-1}{h^2}[(7u_p - 8u_{p-1/2} + u_{p-1}) + (7u_p - 8u_{p+1/2} + u_{p+1})]$$

Rearranging this we get the required expression.

4 FEM Implementation

The implementation was completed in the Julia language and the code is shown in Listing 1.

```
##############################################################################
# File        : fem.jl
# Author      : Sandeep Koranne (C) 2017. All rights reserved
# Purpose     : Implementation of 1D P1 and P2 FEM for OSU Computational
#               Wave Propagation course by Dr. Bokil
#
# Algorithm : Galerkin FEM with Legendre polynomial and Gauss-Lobatto
#               quadrature. Maxima used for symbolic computation.
#               L2 error and H1 error is calculated
##############################################################################
function exact_soln(x,t,OMEGA)
    return cos( OMEGA*pi*t )*sin( OMEGA*pi*x );
end

function FEM_P1( HJ, OMEGA )
    N  = 2^HJ;
    h  = 1/N;
    K1 = eye(2) + [0 -1; -1 0];
    M  = N*10; # time steps
    END_TIME = 1.0;
    dt = END_TIME / M; # same as h
    dt = dt;
    M1 = h/6*[2 1; 1 2];
    A1 = zeros( N,N );
```

```
    B1 = zeros( N,N );
    for k in collect( 1:N-1 )
        for i in collect( 1:2 )
            for j in collect( 1:2 )
                ig = k + i - 1;
                jg = k + j - 1;
                A1[ig,jg] = A1[ig,jg] + K1[i,j];
                B1[ig,jg] = B1[ig,jg] + M1[i,j];
            end
        end
    end
    writedlm("B.txt", B1, " " );
    U  = zeros(N);
    UP = zeros(N);
    UPP= zeros(N);
    for i in collect(1:N)
        UP[i] =  exact_soln( i*h, dt, OMEGA );
        UPP[i] = exact_soln( i*h, 0, OMEGA );
    end
    U[1] = U[N] = UP[1] = UP[N] = UPP[1] = UPP[N] = 0; # Boundary condition
    writedlm("UO.txt", UP, " ");
    NDR1 = inv(B1)*1/h*A1;
    ERROR_L2 = 0;
    ERROR_H1 = 0;
    for time_step in collect(3:M)
        U = 2*UP - UPP - (dt*dt)*NDR1*UP;
        UPP = UP;
        UP  = U;
        U[1] = U[N] = UP[1] = UP[N] = UPP[1] = UPP[N] = 0; # Boundary condition
        ERROR = 0;
        UE = zeros(N);
        UE = -U;
        EXACT = zeros( N );
        for i in collect(1:N)
            EXACT[i] = exact_soln( i*h, time_step, OMEGA );
        end
        EXACT[1] = EXACT[N] = 0;
        # L2 norm calculation
        for i in collect(1:N)
            value = ( EXACT[i] - UE[i] );
            ERROR += value^2;
        end
        ERROR = sqrt.(h*ERROR);
        ERROR_L2 = max( ERROR, ERROR );
        # H1 error calculation
        H1_ERROR = 0;
        for i in collect(2:N)
            l2value = ( EXACT[i] - UE[i] );
            value = ( (EXACT[i] - EXACT[i-1])/h - (UE[i]-UE[i-1])/h );
            H1_ERROR += l2value^2+value^2; # this is the formula for H1
```

```
            end
            H1_ERROR = sqrt.(h*H1_ERROR);
            ERROR_H1 = max( H1_ERROR, H1_ERROR );
        end
        print("ERROR_L2 = ");
        println(ERROR_L2);
        print("ERROR_H1 = ");
        println(ERROR_H1);

        for i in collect(1:N)
            U[i] = -U[i];
        end
        EXACT = zeros( N );
        for i in collect(1:N)
            EXACT[i] = exact_soln( i*h, END_TIME, OMEGA );
        end
        EXACT[1] = EXACT[N] = 0;
        writedlm("E.txt", EXACT, " ");
        writedlm("U.txt", U, " ");
        return [ERROR_L2, ERROR_H1]';
end
OMEGA=1;
EXP_RES = zeros( 7, 3 );
for h in collect( 4:10 )
    A = FEM_P1( h, OMEGA );
    EXP_RES[h-3,1] = h;
    EXP_RES[h-3,2] = A[1];
    EXP_RES[h-3,3] = A[2];
end
writedlm("A.txt", EXP_RES, " " );
```

Listing 1: Implementation of P1 FEM for 1D wave propagation.

The plot for the wave propagation is shown in Figure 4, and the error plot for L2 and H1 error for $\omega = 1, 5, 10$ are shown in Figure 3, Figure 4 and Figure 5, respectively.

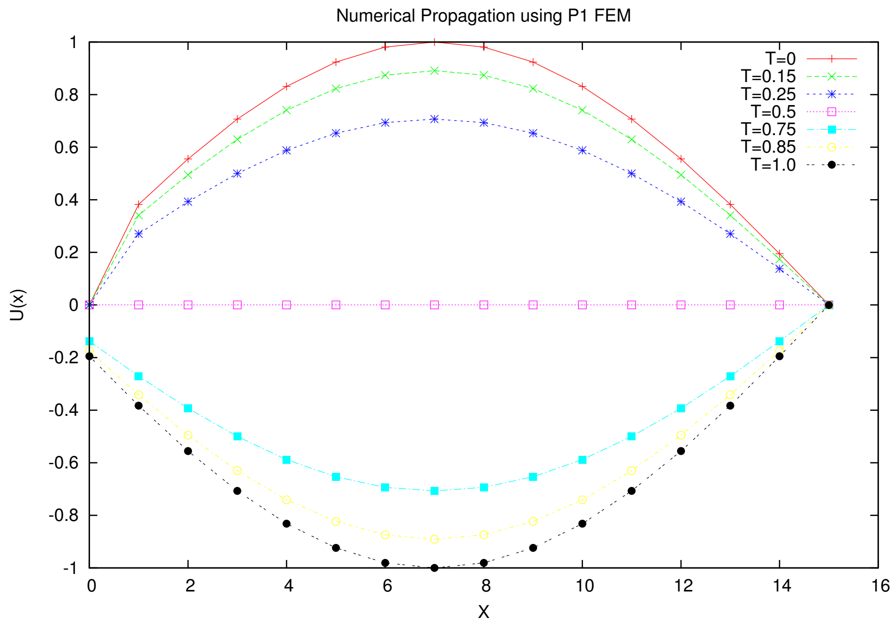
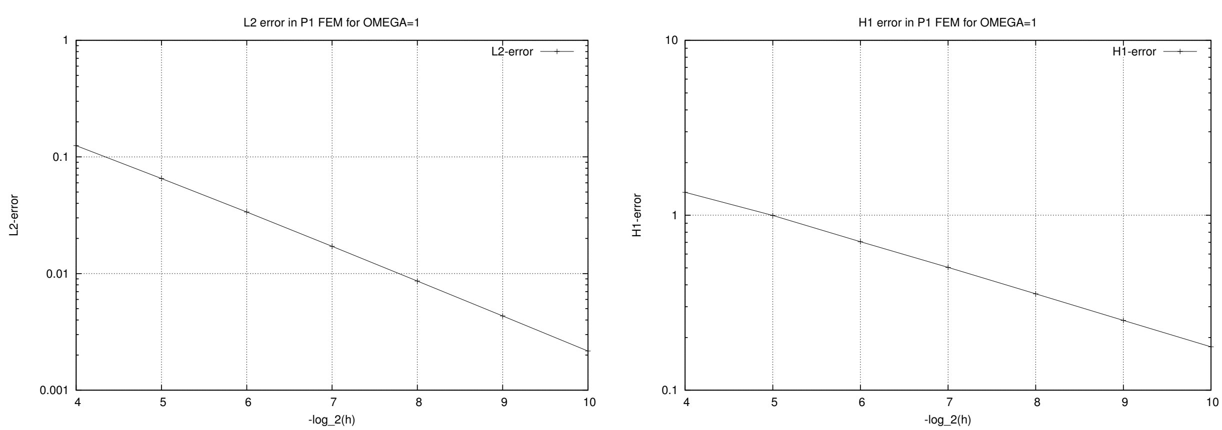


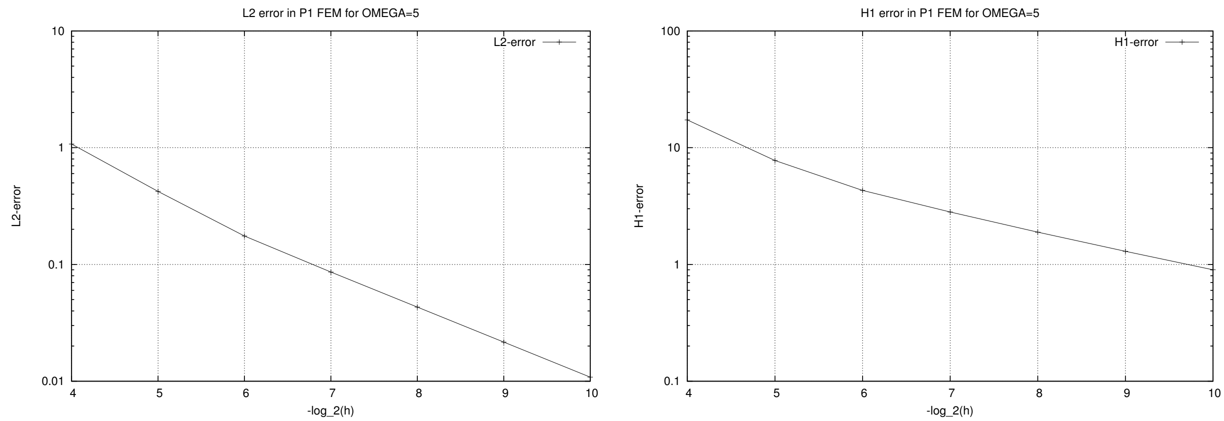
Figure 2: FEM Numerical solution for 1D Wave propagation



(a) L2 error for $\omega = 1$

(b) H1 error for $\omega = 1$

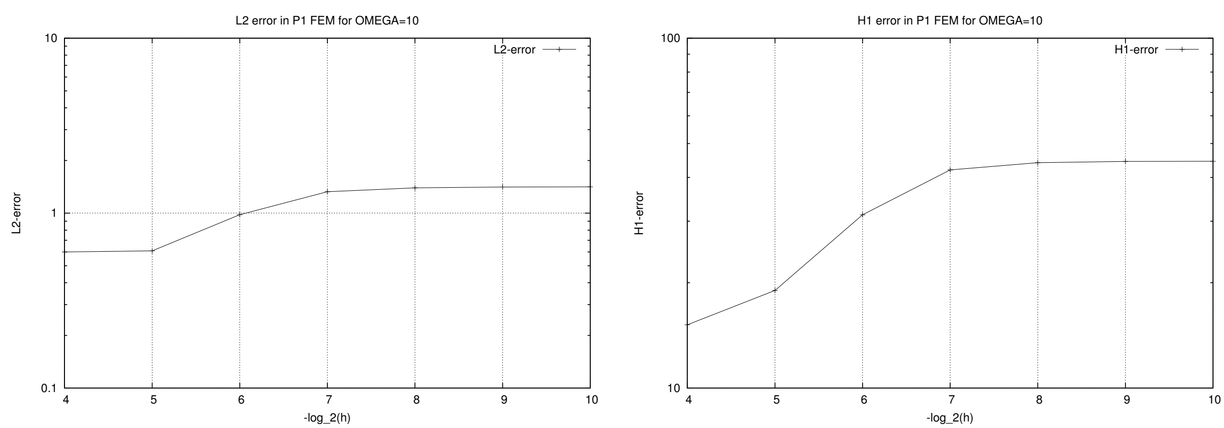
Figure 3: Numerical results for $\omega = 1$.



(a) L2 error for $\omega = 5$

(b) H1 error for $\omega = 5$

Figure 4: Numerical results for $\omega = 5$.



(a) L2 error for $\omega = 10$

(b) H1 error for $\omega = 10$

Figure 5: Numerical results for $\omega = 10$.