ECN
MSc Computational Mechanics
Sem 3

Model Reduction

Lab 4a

Due 05/11/2018

Prasad ADHAV

Atul AGRAWAL

Prasad ADHAV Sample Report

Contents

1 Problem Definition **1**

2 PGD Code Completion And Validation **1**

3 Error Evolution: Approximation Error **2**

4 Non-Constant Source Term **3**

A `Main.m` **4**

List of Figures

1 (a) X-direction first 5 modes, (b) Y-direction first 5 modes 2
2 (a) PGD solution field for the poissons equation, (b) Exact Solution field 2
3 Pointwise error between PGD and exact solution for 10 enrichment terms 2
4 Error evolution with respect to different number of enrichment terms M and different node numbers N 2
5 Solution field with Non Constant source term 3

List of Tables

Prasad ADHAV Sample Report

1 Problem Definition

Consider the 2D Poisson equation shown in equation 1.1, with $L = 2$ and $H = 1$ and $f(x, y) = 1$

$$\bigtriangledown^2 u = -f(x,y) \quad \text{in} \quad]0, L[\times]0, H] \tag{1.1}$$

$$u = 0 \quad \text{on boundaries}$$

The separation of variables was performed as follows:
Supposing that the solution can be written in one term, i.e. $u(x, y) = R(x)S(y)$, where $R(x)$ is the solution in x-direction and $S(y)$ is the solution in y-direction; a weak form of the problem in equation 1.1 was written with test function $v = R(x)S^*(y) + R^*(x)S(y)$, see equation 1.2

$$\int_\Omega \left(\frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} + f(x,y) \right) \mathrm{d}\Omega \cdot v = 0 \tag{1.2}$$

In equation 1.2, the solution form $u(x, y) = R(x)S(y)$ and test function were then substituted in the above equation and the integral was expanded. Depending on the perturbation direction, BVP in x and y were obtained (in simplified integral representation) as $\alpha_1 (S, S) - \beta_1 (S'', S) - \gamma_1(f_2, S) = 0$ and $\alpha_2 (R, R) - \beta_2 (R'', R) - \gamma_2(f_1, R) = 0$, respectively. The coefficients expressions are shown below, and full formulation shown in equations 1.3 and 1.4

$$\beta_1 = \int_0^1 S(y)S dy \qquad \beta_2 = \int_0^2 R(x)R dx$$

$$\alpha_1 = \int_0^1 \frac{\partial^2 S(y)}{\partial y^2} S dy \qquad \alpha_2 = \int_0^2 \frac{\partial^2 R(x)}{\partial x^2} R dx$$

$$\gamma_i = \int_0^1 S(y) f_2 dy \qquad \gamma_2 = \int_0^2 R(x) f_1 dx$$

$$\alpha_1^i = \int_0^1 \frac{\partial^2 F_2^i}{\partial y^2} S dy \qquad \beta_2^i = \int_0^2 F_1^i R dx$$

$$\beta_1^i = \int_0^1 F_2^i S dy \qquad \alpha_2^i = \int_0^2 \frac{\partial^2 F_1^i}{\partial x^2} R dx$$

BVP - x direction

$$(\alpha_1(\mathbf{S})\mathbf{I}_x + \beta_1(\mathbf{S})\mathbf{K}_x)\,\mathbf{R} = \gamma_1(\mathbf{S}, \mathbf{f}_2) - \left(\sum_{i=1}^{M} \left[\alpha_1^i(\mathbf{S}, \mathbf{F}_2^i)\mathbf{I}_x\mathbf{F}_1^i + \beta_1^i(\mathbf{S}, \mathbf{F}_2^i)\mathbf{K}_x\mathbf{F}_1^i \right] \right) \tag{1.3}$$

BVP - y direction

$$(\alpha_2(\mathbf{R})\mathbf{I}_y + \beta_2(\mathbf{R})\mathbf{K}_y)\,\mathbf{S} = \gamma_2(\mathbf{R}, \mathbf{f}_1) - \left(\sum_{i=1}^{M} \left[\alpha_2^i(\mathbf{R}, \mathbf{F}_1^i)\mathbf{I}_y\mathbf{F}_2^i + \beta_2^i(\mathbf{R}, \mathbf{F}_1^i)\mathbf{K}_y\mathbf{F}_2^i \right] \right) \tag{1.4}$$

2 PGD Code Completion And Validation

As per the formulation, the α_1,α_2,β_1,β_2,γ_1 and γ_2 are implemented in the MATLAB code. Trapezoid rule is used for integration approximation. For validation purposes, the normalized modes are plotted for X-direction and y-direction as shown in Figure 1a and Figure 1b. As clear from the graph, the first modes in both the cases carry maximum amount of information with is coharent with the theory.

The PGD result obtained from the completed code is compared with exact solution, Figure 2a and Figure 2b resp. We see the PGD solution field is in close agreement with the Exact solution.

The point wise error is also analyzed for 101 nodes in both direction and 10 enrichment terms (Figure 5).

Prasad ADHAV Sample Report

Figures/X_Directn_Modes.jpg

(a)

Figures/Y_Directn_Modes.jpg

(b)

Figure 1: (a) X-direction first 5 modes, (b) Y-direction first 5 modes

Figures/PGDSOln.jpg

(a)

Figures/Exactsol.jpg

(b)

Figure 2: (a) PGD solution field for the poissons equation, (b) Exact Solution field

Figures/Pointwise.jpg

Figure 3: Pointwise error between PGD and exact solution for 10 enrichment terms

3 Error Evolution: Approximation Error

PGD is an approximation method. Its error depends on the number of the enrichment terms used and on the discretization of the equation by FE,FD or any other method. Here the number of the nodes are denoted by N and number of enrichment terms by M. The PGD solution is compared with the exact solution given by u_N^{ex}. FIg shows the exact solution and FIg shows the PGD solution for $M = 10$ and $N = 51$. Fig shows the node wise difference between the above two solutions.

For the study of evolution of error, we have taken the following values $N = \begin{bmatrix} 11 & 21 & 41 & 61 \end{bmatrix}$ and $M = \begin{bmatrix} 1 & 2 & 3 & 4 & 5 & 6 & 7 & 8 & 9 & 10 \end{bmatrix}$. The fixed point tolerance ϵ for each term and $\tilde{\epsilon}$ for global convergance was set at 10^{-8} for the problem setup. The max terms for non linear iteration set at 10. The error is computed by taking infinity norm given by equation 3.1.

$$e_N^M = \frac{||u_N^{ex} - u_N^M||_\infty}{||u_N^{ex}||_\infty} \tag{3.1}$$

Figures/ApproximationError.jpg

Figure 4: Error evolution with respect to different number of enrichment terms M and different node numbers N

The error formulation is coded in Matlab environment to obtain the results. The graph shown in Figure 4 represents the evolution of error with varying number of enrichment terms and mesh sizes. It can be inferred from the graph that as the number of enrichment terms increases, the error decreases as more enrichment terms captures finer details. Also finer mesh gives less error as evident from the graph. It is interesting to note that rate of change of error is same for all mesh sizes till 2 enrichment terms and after that the error remains almost constant for coarse meshes. This can be explained by stating that initial enrichment modes are sufficient to model all the mesh sizes to a certain extent, the moment further modes are introduced, they try to capture the finer details and reduces the error further for finer meshes. Further it can also be noted that for a given mesh size, after the error has attained a constant value, adding more enrichment term wont make a difference in the solution accuracy. For example, for 21 nodes, 3 enrichment terms are sufficient to model the solution. Adding more terms will just just increase the cost of computation with no increase in accuracy.

Prasad ADHAV Sample Report

4 Non-Constant Source Term

Here, instead of a constant source term, source term is a function of x and y given by the equation 4.1:

$$\begin{cases} \nabla^2 u = -(x^2 - y^2) \quad \text{in} \quad]-1,1[^2 \\ u = 0 \quad \text{on boundaries} \end{cases} \tag{4.1}$$

Separated represtation of f can be written in the form $f(x,y) = f_1(x).f_2(y)$ where $f_1(x)$ and $f_2(y)$ are

$$f_1(x) = \begin{bmatrix} \vdots & \vdots \\ -x^2 & 1 \\ \vdots & \vdots \end{bmatrix}, \quad f_2(y) = \begin{bmatrix} \cdots & 1 & \cdots \\ \cdots & y^2 & \cdots \end{bmatrix} \tag{4.2}$$

The separated source term is implemented in the Matlab code and RHS of the equation modified accordingly. In brief the modification are as follows

- f_1, f_2 inputs were prescribed as `f1 = ones(length(x),2)` and `f2 = ones(length(y),2)`; and the set `f1(:,1) = -x.^2` and `f1(:,2) = y.^2`
- With the change in size of f_1, f_2, the structure of γ_1, γ_2 changed as well, γ_1 being `trapz(y,bsxfun(@times,S,f2))` and similarly for γ_2 as f is multicolumn now. This change had to be accounted for in the function `HT2D_PGD_XY` as follows `bx = f1*gamma_1'` and `by = f2*gamma_2'`.

The number of enrichment term was set at 10 and no of iterations at 20, and 101 nodes in both the directions. ϵ and $\tilde{\epsilon}$ were set at 10^{-8}.

Figures/NonConstantSourceTerm.jpg

Figure 5: Solution field with Non Constant source term

The graph in the Figure 5 shows the solution of the above problem setup for the non constant source term.

Prasad ADHAV Sample Report

A Main.m

```
clear
clc
%%%%%%%%%%%%%%%%%% ECN %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%%%%%%%%%%%%%%%% Model Reduction %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Author: Atul Agrawal 1/11/2018 atulagrawal0711@gmail.com %%%%%%

%%Argument for the functions
% Geomtrical Dimesnions
X = 2; Y = 1;
N = 100;

% ELEMENT DIMENSION, TIME INCREMENT
dx = X/N; dy = Y/N;

x = linspace(0,X,X/dx+1);
y = linspace(0,Y,Y/dy+1);

Max_terms = 10;
Max_fp_iter = 20;
epsilon = 1.e-8;
epsilon_tilde = 1.e-8;
f1 = -1*ones(length(x),1);
f2 = ones(length(y),1);

%%Function call
[U_x,U_y]= HT2D_PGD_XY(x,y,Max_terms,Max_fp_iter,epsilon,epsilon_tilde,
    f1,f2);
U_ex= ExactSolution(x,y,N);
U_PGD=U_x*U_y';


%%Graphs and Results%%%%%%%%%%%%%%%%%%%%
%%
%%Normalized mode graphs
figure(1)
plot(x,U_x(:,1),'r',x,U_x(:,2),'g',x,U_x(:,3),'b',x,U_x(:,4),'y',x,U_x
    (:,5),'b','linewidth',1.5);

title('Normalized x-Direction Modes','fontsize',18)
xlabel('x','fontsize',14)
ylabel('R(x)','fontsize',14)

figure(2)

plot(y,U_y(:,1),'r',y,U_y(:,2),'g',y,U_y(:,3),'b',y,U_y(:,4),'y',y,U_y
    (:,5),'b','linewidth',1.5);
title('Normalized y-Direction Modes','fontsize',18)
xlabel('y','fontsize',14)
ylabel('S(y)','fontsize',14)
```

Prasad ADHAV Sample Report

```
% Solution Fields
figure(3)
surf(x,y,U_PGD);
shading interp

title('PGD SOlution','fontsize',18)
xlabel('x','fontsize',14)
ylabel('y','fontsize',14)

figure(4)
surf(x,y,U_ex);
shading interp

title('Exact Solution','fontsize',18)
xlabel('x','fontsize',14)
ylabel('y','fontsize',14)

%%
%Point Wise Error Calculation
figure(5)

Error = (U_PGD-U_ex);
surf(x,y,Error);
shading interp

title('Point Wise Error','fontsize',18)
xlabel('x','fontsize',14)
ylabel('y','fontsize',14)



%%
%Exercise 3: Approximation error
    %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% Geomtrical Dimesnions
X = 2; Y = 1;
N=100;
N_pt=[11 21 41 61];
Max_terms=1:1:10;
error=zeros(numel(Max_terms),numel(N_pt));

for i=1:4
    x = linspace(0,X,N_pt(i));
    y = linspace(0,Y,N_pt(i));
    for j=1:10
        Max_fp_iter = 20;
        epsilon = 1.e-8;
        epsilon_tilde = 1.e-8;
        f1 = -1*ones(length(x),1);
```

Prasad ADHAV Sample Report

```
        f2 = ones(length(y),1);
        F1_0 = zeros(N_pt(i),1);
        F2_0 = zeros(N_pt(i),1);
        %%Function call
        [U_x,U_y]= HT2D_PGD_XY(x,y,Max_terms(j),Max_fp_iter,epsilon,epsilon_tilde,f1,f2,F1_0,F2_0);
        U_ex= ExactSolution(x,y,N);
        U_PGD=U_x*U_y';
        error(j,i)= norm(U_ex'-U_PGD,'inf')/norm(U_ex','inf');

    end

end
figure(7)
semilogy(Max_terms,error(:,1),'b--*',Max_terms,error(:,2),'r-+',Max_terms,error(:,3),'m-x',Max_terms,error(:,4),'c-o')
legend('N = 11','N = 21','N = 41','N = 61')
xlabel('Enrichment Terms'),ylabel('e_{N}^{M}')


%%
%%Exercise 4: Varrying Source Term %%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%Source Term writtes as combination of [Nx,2].[2,Ny] where Nx has values
%-x^2 and Ny has values y^2.

% BAR LENGTH AND TIME INTERVAL
X = 2;
Y =2;
N = 100;

% ELEMENT DIMENSION, TIME INCREMENT
dx = X/N; dy = Y/N;

x = linspace(-1,1,X/dx+1)';
y = linspace(-1,1,Y/dy+1)';

Max_terms = 10;
Max_fp_iter = 20;
epsilon = 1.e-8;
epsilon_tilde = 1.e-8;

f1 = ones(length(x),2);
f2 = ones(length(y),2);
f1(:,1)= -x.^2;
f2(:,2)= y.^2;

%%Function call
[U_x,U_y]= HT2D_PGD_XY(x,y,Max_terms,Max_fp_iter,epsilon,epsilon_tilde,f1,f2);
```

Prasad ADHAV Sample Report

```
U_PGD=U_x*U_y';
figure(8)
surf(x,y,U_PGD);
shading interp; colorbar;

xlabel('x','fontsize',14)
ylabel('y','fontsize',14)
```

Prasad ADHAV Sample Report