# A SIMPLE GUIDE TO NAVIGATE THE MATLAB ALGORITHMS SHARED AT

[https://github.com/gurkanceren/Stable-XHDG-Project](https://github.com/gurkanceren/Stable-XHDG-Project)

- Go to the folder that is corresponding to the problem you want to solve. Folders are named systematically "methodusedfordiscretization_partialdifferentialequationsolved_geometry(voidorbimaterial interface)_timediscretizationmethod(if applies)"
- Within the folder go to the file that has a name "xxx_main.m"
- As a representative say you choose to model Laplace Equation using XHDG discretization on a voided geometry. So go to the folder "XHDG_Laplace_Void_OK" and there go to "Laplace_main.m"
- First 37 lines of code set the path to the folders that has the codes you want to reach to and set the constants of the problem.

```
2
3   % LaCaN 2012 (www-lacan.upc.edu) -- edited to handle XHDG from HDG code by
4   % Ceren Gurkan
5   %
6   % Academic 2D XHDG code for solving the Laplace equation with Dirichlet
7   % boundary conditions.
8   %
9   % Main data variables:
10  %  X: nodal coordinates
11  %  T: mesh connectivitity matrix
12  %  F: faces (here sides) for each element
13  %    (Faces are numbered so that interior faces are first)
14  %  elemInfo: element type
15  %  infoFaces.intFaces: [elem1 face1 elem2 face2 rotation] for each face
16  %    (Each face is assumed to have the orientation given by the
17  %    first element, it is flipped when seen from the second element)
18  %  infoFaces.extFaces: [elem1 face1] for each exterior face
19  %  referenceElement: integration points and shape functions (volume and
20  %    boundary sides)
21  %
22
23
24  clc, clear all, close all, %home
25  restoredefaultpath, setpath
26
27  % initime = cputime;
28  % profile on
29  % plot(magic(35))
30  % profile viewer
31  % p = profile('info');
32  % profsave(p,'profile_results')
33  % Stabilization parameter
34  tau = 1;
35  %Viscosity
36  mu=1;
37  errors1=[];   errorspost1=[];
38  %Changing mesh name (for cluster)
```

- Between lines 40 and 58 choose the mesh and approximation degree you want to work with. m represents mesh number (increasing m means finer mesh) and p represents the polynomial degree of approximation.

```
40  for p=1 %degree
41      errors = []; errorsPost = []; hs=[];
42      for m=1:6 %mesh number
43
44          filename = ['mesh' num2str(m) '_P' num2str(p) ];
45          disp('Solving...')
46          fname1=[filename '.dcm'];
47          disp(fname1)
48          hs=[hs,0.5^(m-1)];
49
50          %meshName = 'Mesh2_P2.dcm';
51          % Load data
52           meshName = fname1;
53          if all(meshName(end-2:end)=='dcm')
54              GenerateMatFileFromEZ4U(['Meshes/' meshName]);
55          end
56          load(['Meshes/' meshName(1:end-3) 'mat']);
57          X = 2*X - 1; % modify the mesh to have it defined in [-1,1]
```

- At line 59 reference element is defined
- At line 61 void geometry is defined. Use 1 for circular void. Use other values (see comments) for peanut shaped voids or linear interfaces
- At line 65 elements are categorized as standard (inside or outside the void) and cut
- Between lines 66-71 plotting of the mesh and level set boundary of the domain is done.
- Between lines 75-82, faces are numbered, they are put in matrix to tell between which two elements the face is shared.

```
74
75          %% HDG preprocess
76          disp('HDG preprocess...')
77          [F, infoFaces] = hdg_preprocess(T);
78          nOfElements = size(T,1);
79          nOfElementNodes = size(T,2);
80          nOfFaceNodes = size(referenceElement.NodesCoord1d,1);
81          nOfFaces = max(max(F));
82          nOfExteriorFaces = size(infoFaces.extFaces,1);
83
84
85
```

- At line 90, all elemental computations are done, local solver is created.
- At line 92, nodes that are on Dirichlet boundary is eliminated from the system matrix
- System is solved for “lambda” the hybrid unknown defined at inner element faces at line 95
- “uhat” is constructed between lines 97-99 that is, putting together the “lambda” and Dirichlet nodes together, following the correct numbering of the nodes.
- In line 102 elemental unknowns u and q are recovered from the hybrid unknown uhat using the local solver for every element
- Between lines 105-112 the super convergent post processing is done for every element
- Between lines 116-175 for every element (separately for standard and cut elements) the error is calculated
- Between lines 210-220 convergence graphs are produced.

```
209 %% Other plots
210 figure(20), clf
211 plot(log10(hs),log10(errors),'-o',log10(hs),log10(errorsPost),'o-');
212 legend('u','u*')
213
214 slopes = (log10(errors(2:end))-log10(errors(1:end-1)))./(log10(hs(2:end))-log10(hs(1:end-1)))
215 slopesPost = (log10(errorsPost(2:end))-log10(errorsPost(1:end-1)))./(log10(hs(2:end))-log10(hs(1:end-1)))
216
217
218 errors1=[errors1 ; errors ];
219 errorspost1=[errorspost1 ; errorsPost];
220 lhs = log10(hs);
221
222 end
```