GRASIELE ROMANZINI BEZERRA

CDT IN FUTURE FLUIDS DYNAMICS

University of Leeds

Consider the Poisson system

$$-\nabla^2 u = f \quad \text{on } (x,y) \in [0,1]^2 \tag{1a}$$
$$f(x,y) = 2\pi^2 \sin(\pi x)\cos(\pi y) \tag{1b}$$
$$u(0,y) = u(1,y) = 0 \tag{1c}$$
$$\partial_y u(x,y)\big|_{y=0} = \partial_y u(x,y)\big|_{y=1} = 0 \tag{1d}$$

with variable or unknown $u(x,y)$, given function $f(x,y)$, and with Dirichlet and Neumann boundary conditions. The exact solution is $u_e(x,y) = \sin(\pi x)\cos(\pi y)$, please check.

Solution: (Checking exact solution.)

Given the Equation

$$u_e(x,y) = \sin(\pi x)\cos(\pi y), \tag{1}$$

we have that

$$\partial_x u_e = \pi \cos(\pi x)\cos(\pi y), \tag{2}$$
$$\partial_x^2 u_e = -\pi^2 \sin(\pi x)\cos(\pi y), \tag{3}$$
$$\partial_y u_e = -\pi \sin(\pi x)\sin(\pi y), \tag{4}$$
$$\partial_y^2 u_e = -\pi^2 \sin(\pi x)\cos(\pi y). \tag{5}$$

With this we can write

$$\nabla^2 u_e = -2\pi^2 \sin(\pi x)\cos(\pi y) = -f, \tag{6}$$

proving that u_e is the exact solution of Equation (1a) in the question.

QUESTION 1: *STEP 1*

Write down the Ritz-Galerkin principle for the above Poisson system and show that variation thereof yields the system. What are the conditions on the variation $\delta u(x,y)$? Derive the weak formulation for the above system. Show that the test function $w(x,y)$, say, used is the same as $w(x,y) = \delta u(x,y)$.

Solution:

Considering Equation (1a) from the question, we can multiply for a arbitrary weight function w and integrating over the domain $[0,1]^2$

$$\int_0^1 \int_0^1 \left\{-\nabla^2 u - f\right\} w dx dy = 0. \tag{7}$$

We can solve this equation by doing

$$\int_0^1 \int_0^1 \nabla w \cdot \nabla u dx dy - \oint \delta u \hat{n} \cdot \nabla u d\Gamma - \int_0^1 \int_0^1 w f dx dy = 0, \tag{8}$$

where Γ is the surface around the domain. This is a similar problem as the one on section 6.3.1 of [1]. However, for this problem the domain surface can be represented as in Figure 1, where Γ_1 are both horizontal limits, that are equivalents and because of that are represented with the same symbol. And Γ_2 are the vertical limits, also mutually equivalent.

With this we can re-write the second integral in Equation (8) as

$$-\oint w\hat{n} \cdot \nabla u d\Gamma = -2\int_{\Gamma_1} w\hat{n} \cdot \nabla u d\Gamma - 2\int_{\Gamma_2} w\hat{n} \cdot \nabla u d\Gamma. \tag{9}$$

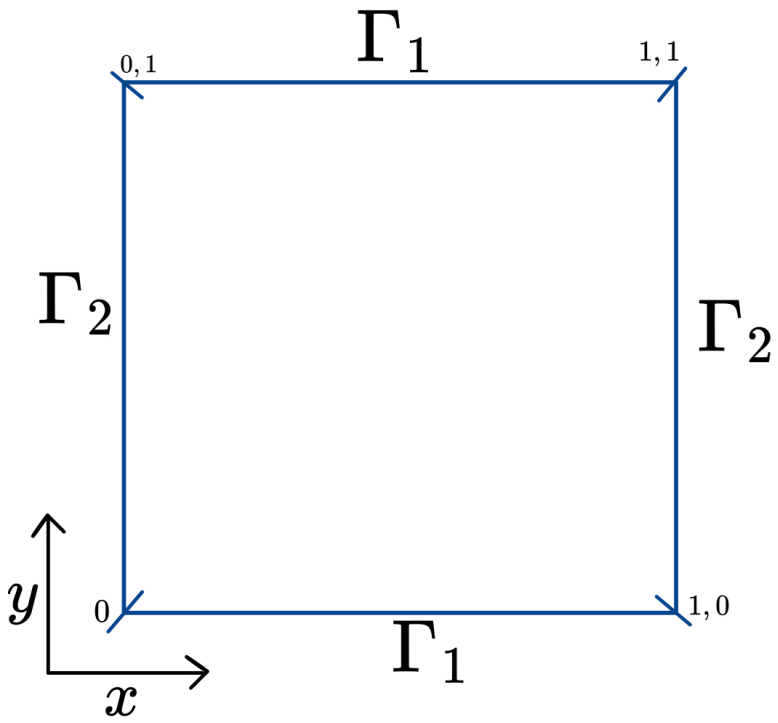

Figure 1: Domain surface for the Poisson problem in 1(a-d).

For the horizontal boundaries, Γ_1, at $y = 0, 1$, we have that

$$w\hat{n} \cdot \nabla u = w \begin{pmatrix} 0 \\ 1 \end{pmatrix} \cdot \begin{pmatrix} \partial_x u \\ \partial_y u \end{pmatrix} = w\partial_y u \tag{10}$$

but we have from (1d) that $\partial_y u = 0$ at $y = 0, 1$, and therefore

$$-2 \int_{\Gamma_1} w\hat{n} \cdot \nabla u d\Gamma = 0. \tag{11}$$

For the vertical boundaries, Γ_2, at $x = 0, 1$, we have that

$$w\hat{n} \cdot \nabla u = w \begin{pmatrix} 1 \\ 0 \end{pmatrix} \cdot \begin{pmatrix} \partial_x u \\ \partial_y u \end{pmatrix} = w\partial_x u, \tag{12}$$

but we have from (1c) that $u = 0$ at $x = 0, 1$, and we need that the test function w respect the same boundary conditions as u, this lead to $w(0, y) = w(1, y) = 0$ and

$$-2 \int_{\Gamma_2} w\hat{n} \cdot \nabla u d\Gamma = 0. \tag{13}$$

Now we can re-write Equation (8) as

$$\int_0^1 \int_0^1 \{\nabla w \cdot \nabla u - wf\} dxdy = 0. \tag{14}$$

or

$$\int_0^1 \int_0^1 \nabla w \cdot \nabla u dxdy = \int_0^1 \int_0^1 wf dxdy. \tag{15}$$

In this form, this equation can be implemented on Firedrake.

now, let's consider the test function $w(x, y) = \delta u(x, y)$. It still obey $w(0, y) = w(1, y) = 0$, as if $u = 0$ in these boundaries, then $\delta u = 0$ as well. Then we have that

$$\int_0^1 \int_0^1 \nabla(\delta u) \cdot \nabla u dxdy = \int_0^1 \int_0^1 \delta u f dxdy \tag{16}$$

$$\int_0^1 \int_0^1 \{\nabla(\delta u) \cdot \nabla u - \delta u f\} dxdy = 0 \tag{17}$$

$$\int_0^1 \int_0^1 \left\{ \delta \left(\frac{1}{2} |\nabla(u)|^2 \right) - \delta u f \right\} dxdy = 0 \tag{18}$$

And finally

$$\delta \int_0^1 \int_0^1 \left\{ \frac{1}{2} |\nabla(u)|^2 - uf \right\} dxdy = 0. \tag{19}$$

And this became a problem of minimizing $\int_0^1 \int_0^1 \left\{ \frac{1}{2} |\nabla(u)|^2 - uf \right\} dxdy$, as the δ operator represents a variation. Another way too see this minimization problem is to consider the function

$$J = \int_0^1 \int_0^1 \left\{ \frac{1}{2} |\nabla u|^2 - uf \right\} dxdy. \tag{20}$$

To find the minimum of this function one must do

$$\frac{dJ}{du} = 0, \tag{21}$$

this lead to

$$\frac{dJ}{du} = \int_0^1 \int_0^1 \{\nabla u \cdot \nabla \delta u - \delta u f\} \, dxdy = 0, \tag{22}$$

with $w = \delta u$ we have

$$\int_0^1 \int_0^1 \{\nabla u \cdot \nabla w - w f\} \, dxdy = 0. \tag{23}$$

Which recover Equation (14).

QUESTION 2: *STEP 2*

Write down the algebraic or discrete Ritz-Galerkin principle after introducing a FEM expansion $u_h(x, y)$ for $u(x, y) \approx u_h(x, y)$ in terms of global basis functions. Write down the algebraic or discrete weak formulation after introducing a FEM expansion in terms of global basis functions. Show that the variation of the former yields the latter.

Solution:

To discretize the equation of the Ritz minimization problem obtained in the previous question, I will first write it in the follow form

$$\int_0^1 \int_0^1 \{\delta(\nabla u) \cdot \nabla u - \delta u f\} \, dxdy = 0. \tag{24}$$

From Section 7.3.1 of [1], we can approximate u by

$$u_h = \sum_{j=1}^{n} \hat{u}_j \phi_j(x, y), \tag{25}$$

where $\hat{u}_j$ are unknowns coefficients and $\phi_j(x, y)$ are base functions that are linearly independent and sufficiently smooth in whiting the target space. We can then replace this on Equation (24) to obtain

$$\int_0^1 \int_0^1 \left\{ \delta \left(\nabla \sum_{j=1}^{n} \hat{u}_j \phi_j \right) \cdot \nabla \left(\sum_{i=1}^{n} \hat{u}_i \phi_i \right) - \delta \left(\sum_{j=1}^{n} \hat{u}_j \phi_j \right) f \right\} dxdy = 0, \tag{26}$$

$$\int_0^1 \int_0^1 \left\{ \left(\sum_{j=1}^{n} \delta \hat{u}_j \nabla \phi_j \right) \cdot \left(\sum_{i=1}^{n} \hat{u}_i \nabla \phi_i \right) - \sum_{j=1}^{n} \delta \hat{u}_j \phi_j f \right\} dxdy = 0, \tag{27}$$

$$\int_0^1 \int_0^1 \left\{ \sum_{j=1}^{n} \sum_{i=1}^{n} (\delta \hat{u}_j) \hat{u}_i \nabla \phi_j \cdot \nabla \phi_i - \sum_{j=1}^{n} \delta \hat{u}_j \phi_j f \right\} dxdy = 0, \tag{28}$$

$$\sum_{j=1}^{n} \sum_{i=1}^{n} (\delta \hat{u}_j) \hat{u}_i \int_0^1 \int_0^1 \nabla \phi_j \cdot \nabla \phi_i dxdy - \sum_{j=1}^{n} \delta \hat{u}_j \int_0^1 \int_0^1 \phi_j f dxdy = 0. \tag{29}$$

Now I can define a matrix A for which the components are given by

$$A_{ij} = \int_0^1 \int_0^1 \nabla\phi_j \cdot \nabla\phi_i dxdy, \tag{30}$$

And a vector $\mathbf{g}$, for which the components are given by

$$g_j = \int_0^1 \int_0^1 \phi_j f dxdy. \tag{31}$$

Then we can write Equation (29) as

$$\sum_{j=1}^{n}\sum_{i=1}^{n}(\delta\hat{u}_j)\hat{u}_i A_{ij} - \sum_{j=1}^{n}\delta\hat{u}_j g_j = 0, \tag{32}$$

$$\sum_{j=1}^{n}\delta\hat{u}_j\left(\sum_{i=1}^{n}A_{ij}\hat{u}_i - g_j\right) = 0. \tag{33}$$

Then we have that for each variation $\delta\hat{u}_j$, it is needed that

$$\sum_{i=1}^{n}A_{ij}\hat{u}_i - g_j = 0, \tag{34}$$

independently. And this reduces to the the system

$$\sum_{i=1}^{n}A_{ij}\hat{u}_i = g_j, \tag{35}$$

that is what needed to be solved to find the coefficients $\hat{u}_i$.

Now, if instead of start from the variation form we start from the weak form

$$\int_0^1 \int_0^1 \{\nabla w \cdot \nabla u - wf\}\, dxdy = 0. \tag{36}$$

We can consider the test function to be

$$w = \sum_i \phi_i, \tag{37}$$

by replacing this on Equation (36) we have

$$\int_0^1 \int_0^1 \left\{\nabla\left(\sum_j \phi_j\right) \cdot \nabla\left(\sum_i \hat{u}_i\phi_i\right) - \left(\sum_j \phi_j\right) f\right\} dxdy = 0, \tag{38}$$

$$\sum_j\sum_i \int_0^1 \int_0^1 \nabla\phi_j \cdot \hat{u}_i\nabla\phi_i dxdy - \sum_j \int_0^1 \int_0^1 \phi_j f dxdy = 0, \tag{39}$$

$$\sum_j \left\{\sum_i \hat{u}_i \int_0^1 \int_0^1 \nabla\phi_j \cdot \nabla\phi_i dxdy - \int_0^1 \int_0^1 \phi_j f dxdy\right\} = 0. \tag{40}$$

Finally using the definitions in (30) and (31) we have again that it is needed that each item in the j sum to be independently zero:

$$\sum_{i=1}^{n}A_{ij}\hat{u}_i - g_j = 0, \tag{41}$$

$$\sum_{i=1}^{n}A_{ij}\hat{u}_i = g_j. \tag{42}$$

QUESTION 4: *STEP 4*

Solve the system in Firedrake with the provided or other Firedrake codes. Plot the numerical results for $u_h(x, y)$ with Paraview as a contour plot (with clear labelling/indicating of the values used in that plot). Plot the difference $|u_h(x, y) - u_e(x, y)j$ (of numerical and exact solutions) as a contour plot in Firedrake for a few suitable resolutions (−). Mention the function spaces used and the order of accuracy. Explore different order (p-refinement) and mesh resolutions (h-refinements). Explain and show which h, p combinations are (roughly) equivalent and why? Provide clear figure captions with information on resolution, etc., such as h, p.

Solution:

The system is solved in firedrake using two methods: Method 1 ($u_{h,1}$) implements Equation (14) and Method 2 ($u_{h,2}$) implements Equation (19). The code that does it is in `code/Poissons_eq_v2_modified.py`. The result for using p-refinement $p = 1$, and a 16x16 mesh are displayed in Figure 2. In the upper row is plotted the contour of the approximations $u_{h,1}$, $u_{h,2}$ and the exact solution u_e. In the bottom row is plotted the error for $u_{h,1}$ calculated as $|u_{h,1} - u_e|$, the error in $u_{h,2}$ calculated as $|u_{h,2} - u_e|$ and the residual difference between both approximations. These contour plots where done embed in the aforementioned python code using `mathplotlib.pyplot.contourf`.

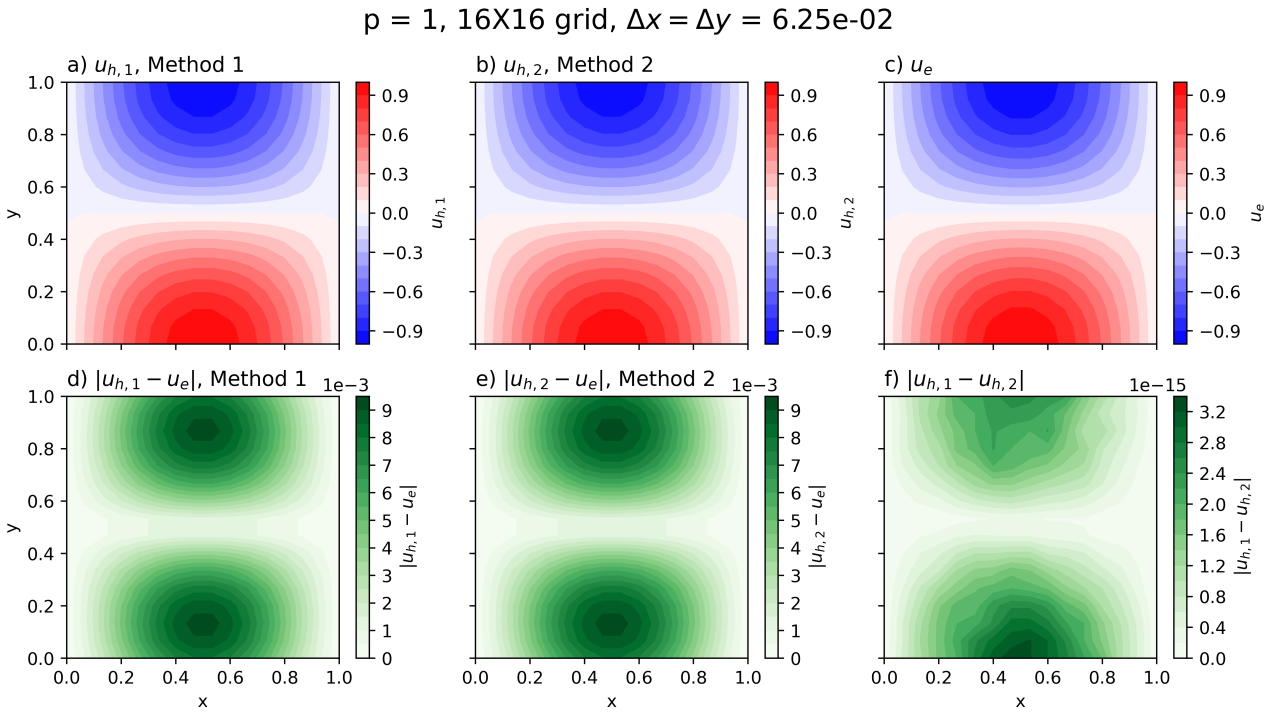

Figure 2: Solutions of the system in 1(a-d), using pronominal order $p = 1$, and mesh size 16x16, leading to h = 6.25e-2. In (a) is the contour plot of the numerical approximation $u_{h,1}$ obtained using equation (14), (b) is the contour plot of the numerical approximation $u_{h,2}$ obtained using equation (19), (c) is the contour plot of the exact solution u_e. In (d) is the contour plot of the error in $u_{h,1}$ calculated by doing $|u_{h,1} - u_e|$, (e) is the contour plot of the error in $u_{h,2}$ calculated by doing $|u_{h,2} - u_e|$ and (f) is the difference between the results of the two methods.

Next I refined the mesh by increasing the number of subdivisions to 32x32, results shown in Figure 3, and to 64x64, results shown in Figure 4. In a first analysis we can observe that the error drop to almost half when the mesh size double, it is also observable that the residue between both methods also decrease. However analysing it this way is somewhat qualitative.

To obtain a quantitative analysis of the error one might use the L^2-norm that is given by

$$|u|_{L^2} = \sqrt{\int_\Omega |u(x, y)|^2 dxdy}. \tag{43}$$

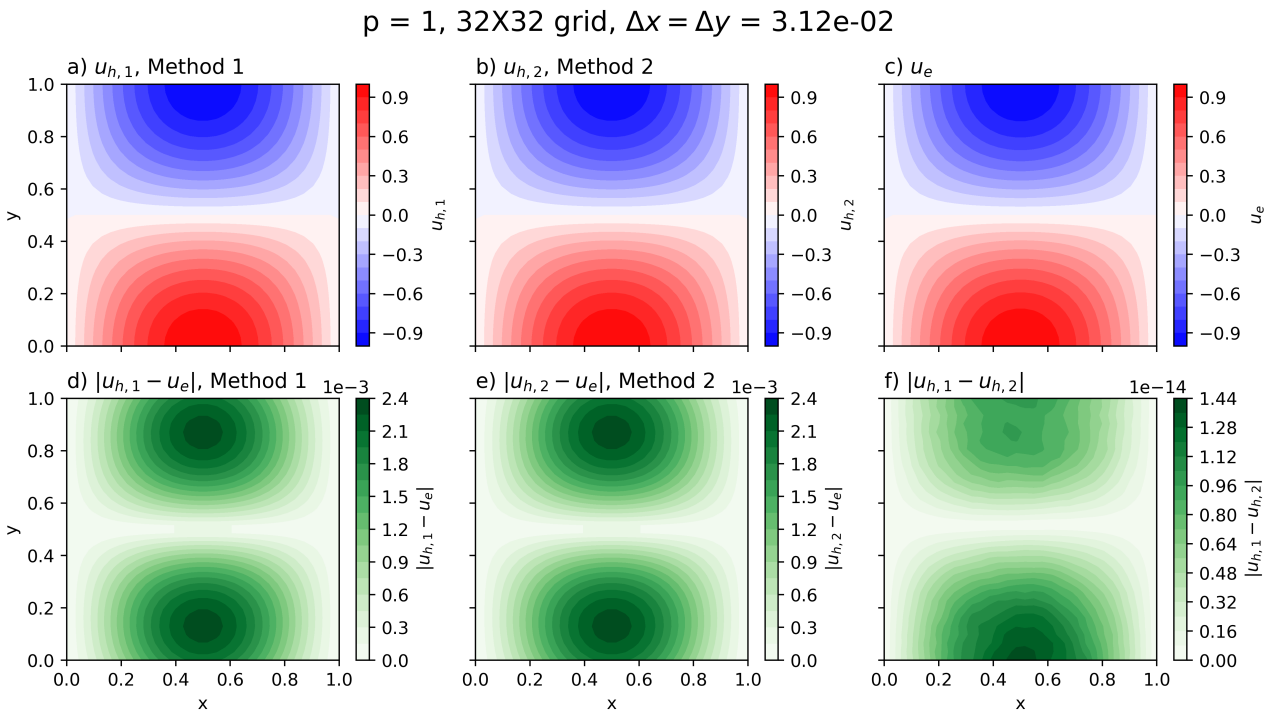

Figure 3: Solutions of the system in 1(a-d), using pronominal order $p = 1$, and mesh size 32x32, leading to h = 3.12e-2. In (a) is the contour plot of the numerical approximation $u_{h,1}$ obtained using equation (14), (b) is the contour plot of the numerical approximation $u_{h,2}$ obtained using equation (19), (c) is the contour plot of the exact solution u_e. In (d) is the contour plot of the error in $u_{h,1}$ calculated by doing $|u_{h,1} - u_e|$, (e) is the contour plot of the error in $u_{h,2}$ calculated by doing $|u_{h,2} - u_e|$ and (f) is the difference between the results of the two methods.

We can use it to calculate the L^2-error and residual by doing

$$L_1^2 = \sqrt{\int_\Omega |u_{h,1}(x,y) - u_e(x,y)|^2 dxdy}, \tag{44}$$

$$L_2^2 = \sqrt{\int_\Omega |u_{h,2}(x,y) - u_e(x,y)|^2 dxdy}, \tag{45}$$

$$L^2 = \sqrt{\int_\Omega |u_{h,1}(x,y) - u_{h,2}(x,y)|^2 dxdy}. \tag{46}$$

I calculated it for four mesh sizes: 16x16, 32x32, 64x64 and 128x128; and for four pronominal orders: 1, 2, 3, 4. The result for this is displayed in Table 1.

Looking at Table 1 one can see that a few combinations of {h,p} refinement lead to L^2-error values that are very close. For example, the combinations {1.56e-2, 2} (mesh 64x64) and {6.25e-2,3} (mesh 16x16) both lead to errors $L_1^2 = L_2^2 \sim 10^{-9}$, this are highlighted in green in Table 1. Additionally the combinations {7.81e-3, 2} (mesh 128x128) and {3.12e-2,3} (mesh 32x32) both lead to errors $L_1^2 = L_2^2 \sim 10^{-10}$, this are highlighted in orange in Table 1. This is a result of the fact that both refinements in h and p lead to an increase in the precision, hence decreasing the error, and therefore, when variated simultaneously, it is natural to appear {h,p} combinations that are equivalent among themselves.

Moreover, we can also see that observe that for p = 3 and 4 the L^2-errors, L_1^2 and L_2^2, are starting to get to the same order of magnitude to the residual L^2. Furthermore, L_2^2 for p =4, seems to start to oscillate instead of monotonically decrease as we refine the mesh. This can indicate that the method reached the limit of precision, and probably we wont be able to decrease L_2^2 inly by refining {h,p}.

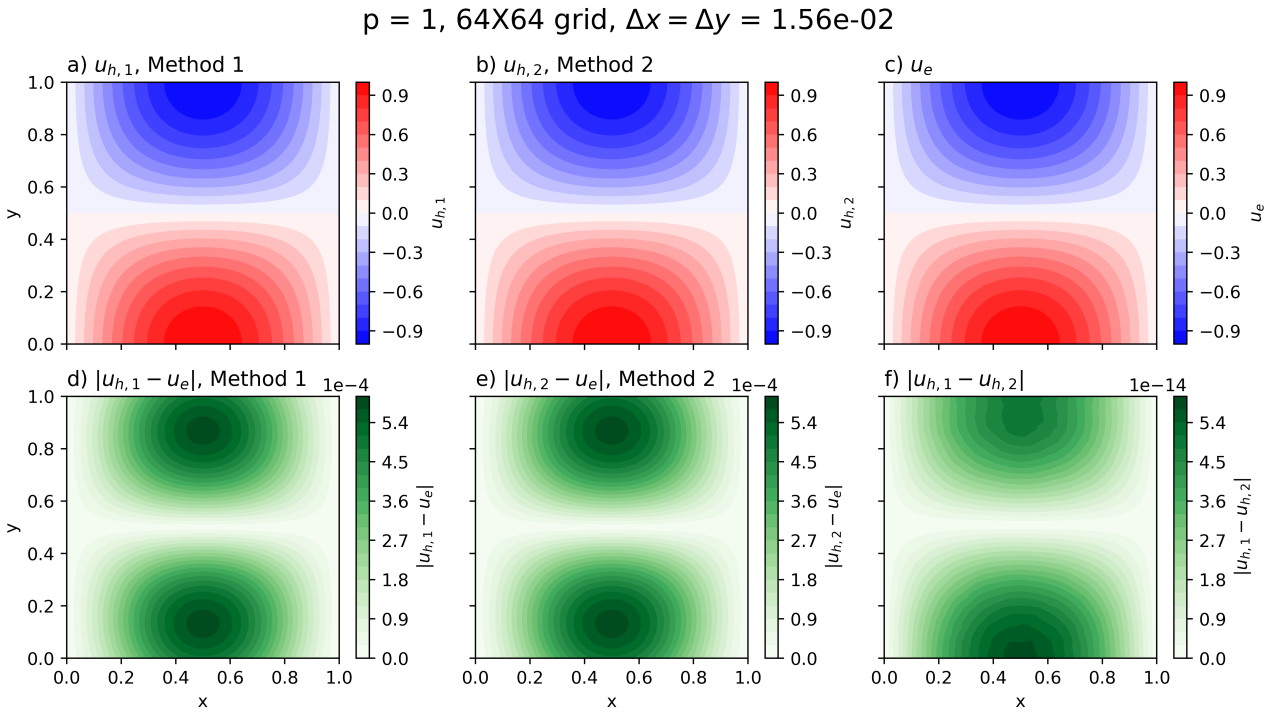

Figure 4: Solutions of the system in 1(a-d), using pronominal order $p = 1$, and mesh size 64x64, leading to h = 1.56e-2. In (a) is the contour plot of the numerical approximation $u_{h,1}$ obtained using equation (14), (b) is the contour plot of the numerical approximation $u_{h,2}$ obtained using equation (19), (c) is the contour plot of the exact solution u_e. In (d) is the contour plot of the error in $u_{h,1}$ calculated by doing $|u_{h,1} - u_e|$, (e) is the contour plot of the error in $u_{h,2}$ calculated by doing $|u_{h,2} - u_e|$ and (f) is the difference between the results of the two methods.

p	Mesh Size	h	L_1^2	L_2^2	L^2
1	16x16	6.25e-02	1.59e-03	1.59e-03	1.36e-15
1	32x32	3.12e-02	4.01e-04	4.01e-04	6.16e-15
1	64x64	1.56e-02	1.00e-04	1.00e-04	2.79e-14
1	128x128	7.81e-03	2.51e-05	2.51e-05	8.70e-14
2	16x16	6.25e-02	1.04e-06	1.04e-06	6.28e-11
2	32x32	3.12e-02	6.53e-08	6.53e-08	2.85e-11
2	64x64	1.56e-02	4.08e-09	4.08e-09	3.04e-12
2	128x128	7.81e-03	2.55e-10	2.55e-10	2.36e-12
3	16x16	6.25e-02	3.38e-09	3.38e-09	3.05e-11
3	32x32	3.12e-02	1.06e-10	1.06e-10	1.15e-11
3	64x64	1.56e-02	5.35e-12	3.36e-12	4.25e-12
3	128x128	7.81e-03	2.19e-12	2.67e-12	1.33e-12
4	16x16	6.25e-02	3.86e-11	2.67e-11	2.83e-11
4	32x32	3.12e-02	1.15e-11	4.32e-13	1.15e-11
4	64x64	1.56e-02	4.94e-12	4.74e-13	4.91e-12
4	128x128	7.81e-03	2.45e-12	1.89e-12	1.21e-12

Table 1: L^2-error and residual for different mesh refinement and p-refinements.

QUESTION 5

Explain how the above first two or three steps are implemented in Firedrake, also by adding clear comments to your code.

Solution:

For Method 1 ($u_{h,1}$) it was used Equation (14) from step 1, in the line

```
solve(a == L, u_1, solver_parameters={'ksp_type': 'cg', 'pc_type': 'none'}, bcs=[bc_x0,bc_x1])
```

with

$$a = \nabla u \nabla w dx dy \tag{47}$$

defined in the code as

```
a = (inner(grad(u),grad(v)))*dx # Step 2/3: The weak form first term
```

and

$$L = f w dx dy \tag{48}$$

defined in the code as

```
L = (f*v)*dx
```

Note that although I used w to designate the test function, the in the code it is called v, but it is suppose to be the same test function.

For Method 2 ($u_{h,2}$) it was used Equation (19) from step 1, in the line

```
solve(F == 0, u_2, bcs=[bc_x0, bc_x1])
```

where

$$F = \frac{dJ}{du} \tag{49}$$

defined in the code as

```
F = derivative(Ju, u_2, du=v)
```

with

$$J = \frac{1}{2}|\nabla u|^2 - uf \tag{50}$$

defined in the code as

```
Ju = (0.5*inner(grad(u_2),grad(u_2)) - u_2*f)*dx
```

Steps 2 and 3 are done internally by Firedrake and not explicitly implemented.

References

[1] Jeroen van Kan et al. *Numerical Methods in Scientific Computing*. VSSD, 2005, p. 279.