Numerical Methods for Partial Differential Equations
A.Y. 2022/2023

Laboratory 02
Finite Element method for the Poisson equation in 1D: convergence analysis

Exercise 1.

Let $\Omega = (0, 1)$. Let us consider the Poisson problem

$$\begin{cases} -\left(\mu(x)\, u'(x)\right)' = f(x) & x \in \Omega = (0,1) \qquad (1a) \\ u(0) = u(1) = 0 & \qquad (1b) \end{cases}$$

$\mu(x) = 1$ and $f(x) = 4\pi^2 \sin(2\pi x)$ for $x \in \Omega$.

1.1. Show that $u_{\text{ex}}(x) = \sin(2\pi x)$ is the exact solution to (1).

Solution. There holds $u_{\text{ex}}(0) = u_{\text{ex}}(1) = 0$, so that (1b) is satisfied. Moreover,

$$u'_{\text{ex}}(x) = 2\pi \cos(2\pi x)\ ,$$
$$(\mu(x)\, u'_{\text{ex}}(x))' = -4\pi^2 \sin(2\pi x) = -f(x)\ ,$$

so that also (1a) is satisfied.

1.2. Starting from the solution of the first laboratory, implement a method `double Poisson1D::compute_error(const VectorTools::NormType &norm_type) const` that computes the $L^2(\Omega)$ or $H^1(\Omega)$ norm (depending on the input argument) of the error between the computed solution and the exact solution:

$$e_{L^2} = \|u_h - u_{\text{ex}}\|_{L^2} = \sqrt{\int_0^1 |u_h - u_{\text{ex}}|^2\, dx}\ ,$$

$$e_{H^1} = \|u_h - u_{\text{ex}}\|_{H^1} = \sqrt{\int_0^1 |u_h - u_{\text{ex}}|^2\, dx + \int_0^1 |\boldsymbol{\nabla} u_h - \boldsymbol{\nabla} u_{\text{ex}}|^2\, dx}\ .$$

Solution. See the file `src/Poisson1D.cpp`.

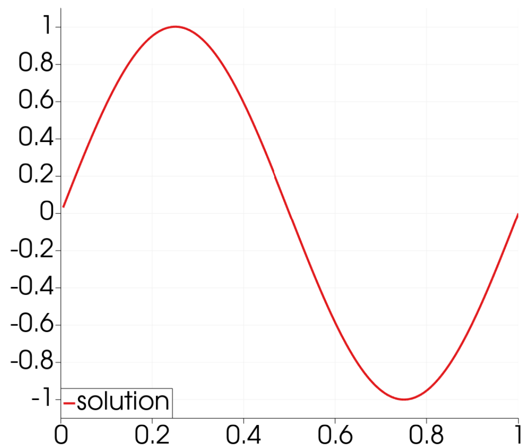


Figure 1: Solution u_h of Exercise 1.3, computed using $r = 1$ and $N + 1 = 160$.

1.3. With polynomial degree $r = 1$, solve the problem (1) with finite elements, setting $N + 1 = 10, 20, 40, 80, 160$. Compute the error in both the $L^2(\Omega)$ and $H^1(\Omega)$ norms as a function of the mesh size h, and compare the results with the theory.

Solution. See the file `src/lab-02.cpp` for the implementation. The solution for this test is plotted in Figure 1.

From the convergence theory, we expect that, if $u_{\text{ex}} \in H^{r+1}(\Omega)$ (in this case, $u_{\text{ex}} \in H^p$ for all $p = 1, 2, 3, \dots$),

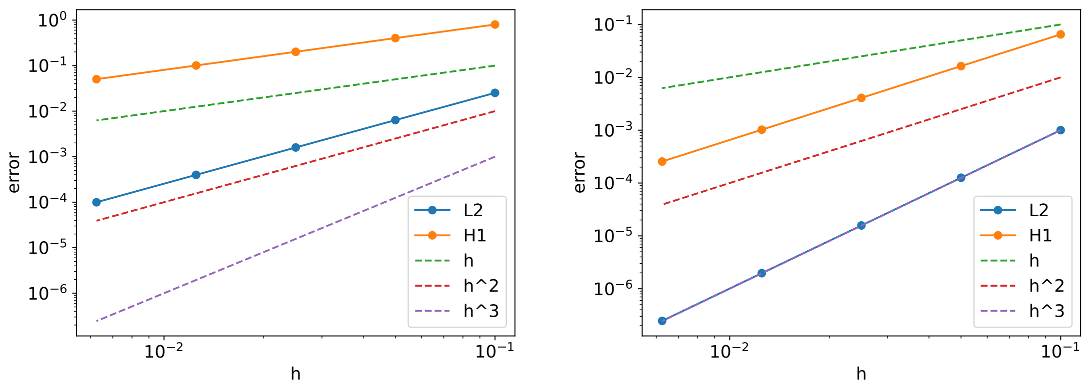
$$\begin{cases} e_{L^2} \leq C_1 h^{r+1} |u_{\text{ex}}|_{H^{r+1}} \ , \\ e_{H^1} \leq C_2 h^r |u_{\text{ex}}|_{H^{r+1}} \ , \end{cases}$$

where $h = 1/(N + 1)$ is the size of mesh elements. Therefore, we expect the error in the $L^2(\Omega)$ norm to converge to zero with order $r + 1 = 2$, and the error in the $H^1(\Omega)$ norm to converge to zero with order $r = 1$.

To verify this, we use the `deal.II` class `ConvergenceTable` to compute the estimated convergence order. Notice that the class works as expected as long as the mesh size between subsequent solutions is halved. We obtain the following results:

h	e_{L^2}	convergence order e_{L^2}	e_{H^1}	convergence order e_{H^1}
0.1	2.5199e-02	-	8.0096e-01	-
0.05	6.3529e-03	1.99	4.0231e-01	0.99
0.025	1.5916e-03	2.00	2.0139e-01	1.00
0.0125	3.9811e-04	2.00	1.0072e-01	1.00
0.00625	9.9539e-05	2.00	5.0364e-02	1.00

The estimated convergence orders for the two errors are in agreement with the theory.



(a) (b)

Figure 2: Exercises 1.3 and 1.4. Error in the L^2 and H^1 norms against h for $r = 1$ (a) and $r = 2$ (b).

We can draw the same conclusions by plotting the error against h in the log-log plane. Indeed, we want to assess whether

$$e_{L^2} \approx C_1 |u_{\text{ex}}|_{H^{r+1}} h^{r+1}$$

holds. By taking the logarithm of both sides, we get

$$\log e \approx \log\left(C_1 |u_{\text{ex}}|_{H^{r+1}}\right) + (r+1)\log h\ ,$$

so that the logarithm of the L^2 error is a linear function of the logarithm of h, with slope given by the convergence rate $r + 1 = 2$. Therefore, in the log-log plane, the plot of the L^2 error should be a line parallel to the one obtained by plotting the function h^{r+1}. Similar considerations hold for the H^1 error, which should be represented by a line of slope $r = 1$.

The errors against h are written to a file `convergence.csv`, which can be opened in any 2D graphing software (MATLAB, Python, ...). For convenience, you can find a Python script at `scripts/plot-convergence.py`, that can be called as `./plot-convergence.py convergence.csv` and produces a PDF file `convergence.pdf` containing the plot. The script requires that the package `matplotlib` is available: you can download and install it by running `pip install matplotlib`.

The result is displayed in Figure 2a. We can see how the error in the L^2 norm is represented by a straight line parallel to the one representing h^2, as expected, confirming that it tends to zero with order $r + 1 = 2$. Similarly, the H^1 error tends to zero with order $r = 1$ as expected.

1.4. Repeat the previous point setting $r = 2$.

Solution. By changing the polynomial degree to $r = 2$, we obtain the following convergence table:

h	e_{L^2}	convergence order e_{L^2}	e_{H^1}	convergence order e_{H^1}
0.1	1.0028e-03	-	6.5007e-02	-
0.05	1.2590e-04	2.99	1.6319e-02	1.99
0.025	1.5754e-05	3.00	4.0840e-03	2.00
0.0125	1.9698e-06	3.00	1.0213e-03	2.00
0.00625	2.4624e-07	3.00	2.5533e-04	2.00

The corresponding error plots are displayed in Figure 2b. From, both, we can observe the expected convergence rates: the L^2 error tends to zero with rate $r + 1 = 3$, and the H^1 error tends to zero with rate $r = 2$.

1.5. Let us now redefine the forcing term as

$$f(x) = \begin{cases} 0 & \text{if } x \leq \dfrac{1}{2}\,, \\ -\sqrt{x - \dfrac{1}{2}} & \text{if } x > \dfrac{1}{2}\,. \end{cases}$$

The exact solution in this case is

$$u_{\mathrm{ex}}(x) = \begin{cases} Ax & \text{if } x \leq \dfrac{1}{2}\,, \\ Ax + \dfrac{4}{15}\left(x - \dfrac{1}{2}\right)^{\frac{5}{2}} & \text{if } x > \dfrac{1}{2}\,, \end{cases}$$

$$A = -\frac{4}{15}\left(\frac{1}{2}\right)^{\frac{5}{2}}\,.$$

Check the convergence order of the finite element method in this case, with polynomial degrees $r = 1$ and $r = 2$. What can you observe?

Solution. The solution for this test is plotted in Figure 3. For $r = 1$, the convergence table reads

h	e_{L^2}	convergence order e_{L^2}	e_{H^1}	convergence order e_{H^1}
0.1	3.1998e-04	-	1.0198e-02	-
0.05	7.9874e-05	2.00	5.1018e-03	1.00
0.025	1.9899e-05	2.01	2.5514e-03	1.00
0.0125	4.9486e-06	2.01	1.2757e-03	1.00
0.00625	1.2281e-06	2.01	6.3788e-04	1.00

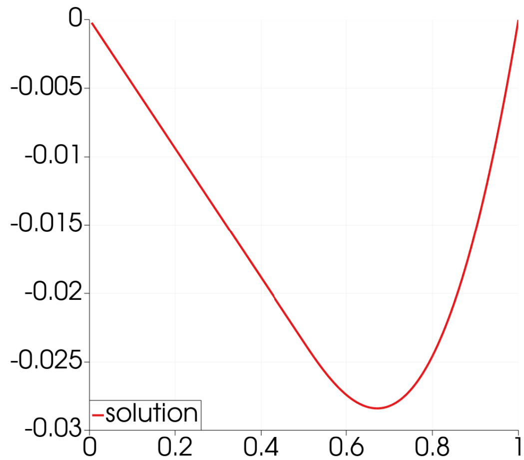


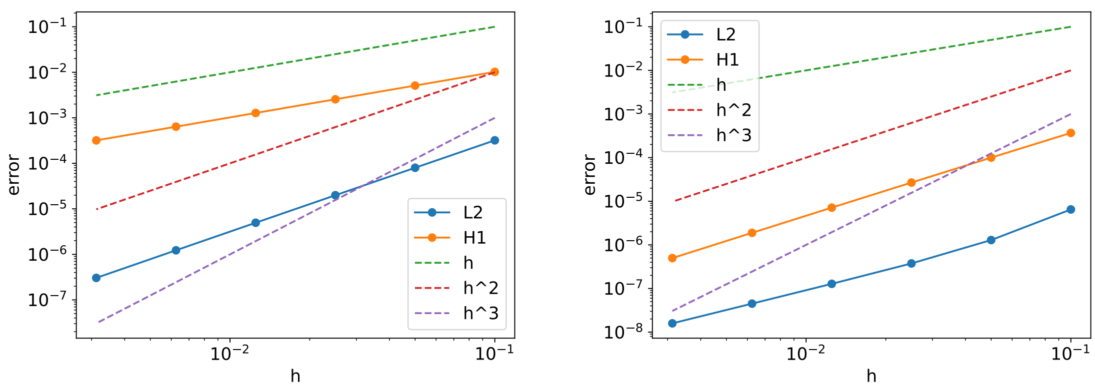
Figure 3: Solution u_h of Exercise 1.5, computed using $r = 1$ and $N + 1 = 160$.

and the associated convergence plot is displayed in Figure 4a. We observe once again the expected convergence rates.

If we set $r = 2$, however, we obtain the following convergence table:

h	e_{L^2}	convergence order e_{L^2}	e_{H^1}	convergence order e_{H^1}
0.1	6.4841e-06	-	3.6856e-04	-
0.05	1.2899e-06	2.33	9.9987e-05	1.88
0.025	3.7520e-07	1.78	2.6826e-05	1.90
0.0125	1.2792e-07	1.55	7.1384e-06	1.91
0.00625	4.4981e-08	1.51	1.8882e-06	1.92

and the convergence plot of Figure 4b. The convergence rates are not the optimal ones in this case. Indeed, the optimal convergence rates ($r + 1$ in the L^2 norm and r in the H^1 norm) can be observed only if the solution belongs to the space $H^{r+1}(\Omega)$. In this case, there holds $u \in H^2(\Omega)$ but $u \notin H^3(\Omega)$, so that the observed convergence rates, for the case $r = 2$, are slower than the optimal ones.



(a) (b)

Figure 4: Exercise 1.5. Error in the L^2 and H^1 norms against h, for $r = 1$ (a) and $r = 2$ (b).