Master's Degree in
Numerical Methods in
Engineering

Computational Mechanical Tools

Assignment 3:

PDE-Toolbox

Authors:
Ivan Pérez

Professor:
Amir Abdollahi

Academic Year 2019-2020

Contents

Abstract

In the first place, it has been studied the evolution of logarithmic error against the logarithmic element mesh size, resulting in a linear dependency between them. Next, it has been studied the error for different end times. The error stabilizes for times over 10 seconds. Finally, it has been simplified the parabolic original PDE to an elliptic PDE (not time-dependent) for times over 10 seconds and studied its error. The new elliptic PDE converges as quickly to the same error of the original more complex PDE, but with much less computational effort.

1 Introduction

In this assignment it is required to use *MATLAB: PDE-Toolbox* module to solve a given partial differential equation. It will be necessary to define a geometric domain of one by one (no units specified) as well as meshing inside with triangular elements.

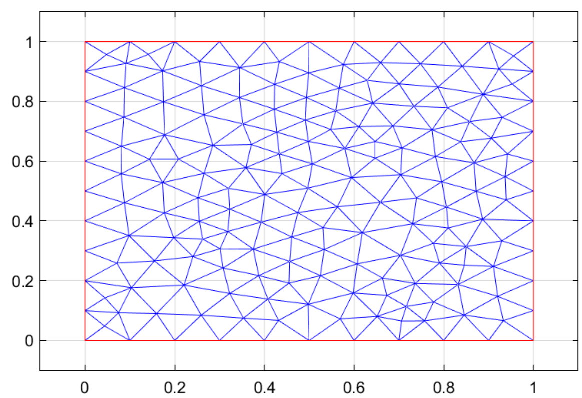


Figure 1: Triangular mesh of domain $\Omega = [0, 1]^2$
Level 0. Elements = 324

The following step is to define the boundary conditions and the governing PDE problem equation. *PDE-Toolbox* can handle all kind of PDE by introducing the proper factors that multiply each term.

$-\nabla \cdot (c\nabla u) + au = f$ Elliptic PDE	$d\frac{\partial u}{\partial t} - \nabla \cdot (c\nabla u) + au = f$ Parabolic PDE
$d\frac{\partial^2 u}{\partial t^2} - \nabla \cdot (c\nabla u) + au = f$ Hyperbolic PDE	$-\nabla \cdot (c\nabla u) + au = \lambda du$ Generalized Eigen Value Problem PDE

2 Problem Equation

The problem governing equation is presented below with its source term:

$$u_t - \Delta u = f \tag{1}$$

$$f(x, y, t) = -3e^{-3t} \tag{2}$$

We consider the initial value condition at $t = 0$ and the rest of boundary conditions:

$$u(x, y, t = 0) = x^2 + xy - y^2 + 1 \tag{3}$$

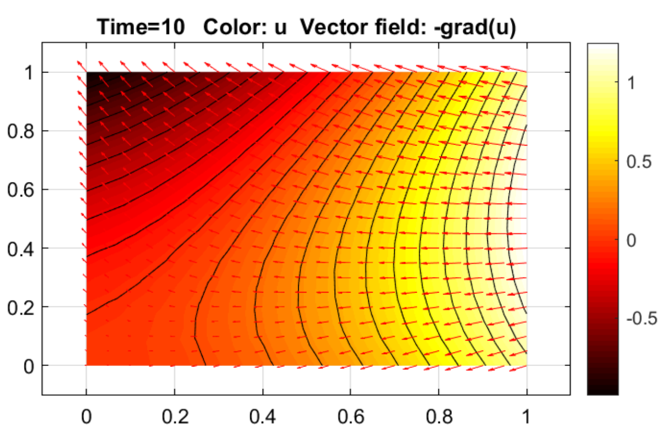
$$\begin{array}{l|l} u_n(x = 0, y, t) = -y & u(x, y = 0, t) = x^2 + e^{-3t} \\ u_n(x = 1, y, t) = 2 + y & u_n(x, y = 1, t) = x - 2 \end{array}$$

The analytical solution of the problem has been provided in order to measure the error on the next section:

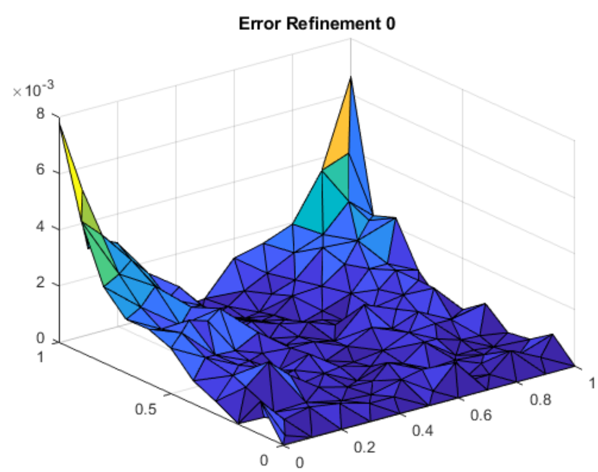
$$u(x, y, t) = x^2 + xy - y^2 + e^{-3t} \tag{4}$$

3 Error Study Approaches

Using *PDE-Toolbox* the problem equation presented on section 2 has been computed. Below, the temperature field distribution through the domain is presented:



Temperature distribution



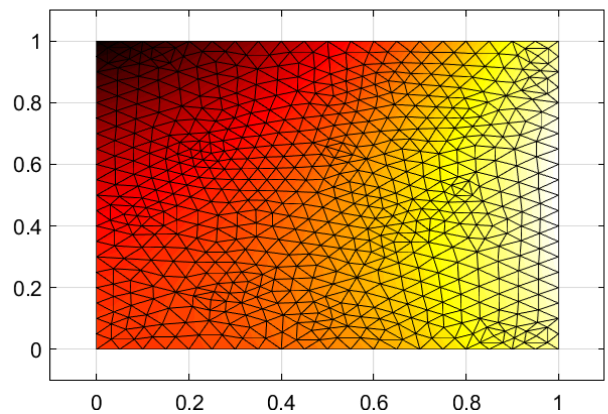
Error distribution

Table 1: Temperature and Error distribution on Domain Ω at $T_{end} = 10s$.
Mesh base size: 0 Level.

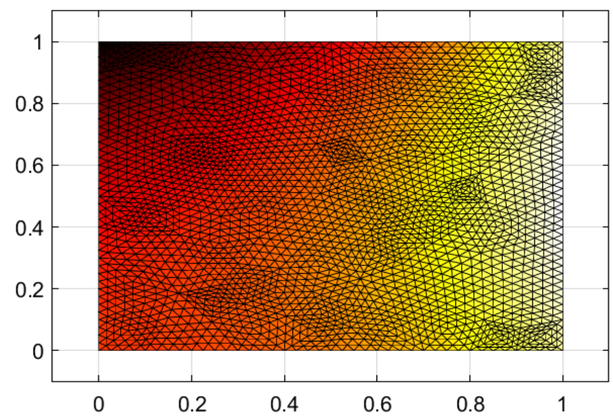
3.1 Dependence of Element Mesh Size

Firstly it is assessed the influence on error due to the refinement of the mesh or the average element size of the mesh.

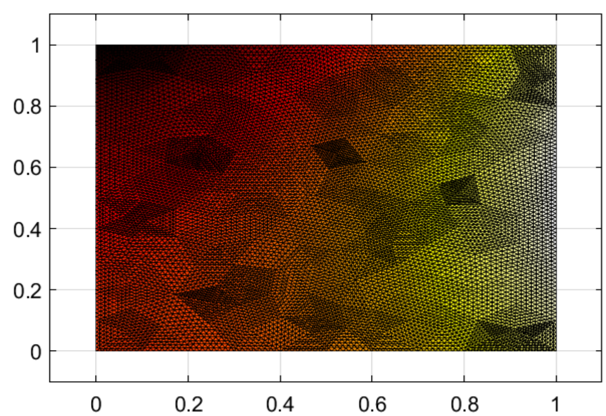
The mesh has been refined up to 4 Levels. Each level increases by 2 the number of elements in each boundary. Since it is a squared domain, the number of elements increases by 4 times per each level iteration.



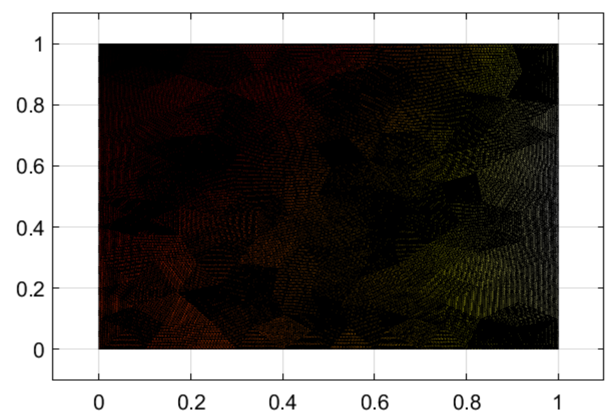
Level 1. Elements = 1296.



Level 2. Elements = 5184



Level 3. Elements = 20736.



Level 4. Elements = 82944

Figure 2: Refinement levels and its number of elements

Table of the element size and its maximum error:

	Elements	$\overline{h}_e$	Max. Error
Level 0	324	0.0786	$7.628e^{-3}$
Level 1	1296	0.0393	$2.162e^{-3}$
Level 2	5184	0.0196	$6.051e^{-4}$
Level 3	20736	0.0098	$1.681e^{-4}$
Level 4	82944	0.0049	$4.602e^{-5}$

Below, it has been plotted the relation between the error and the element size:

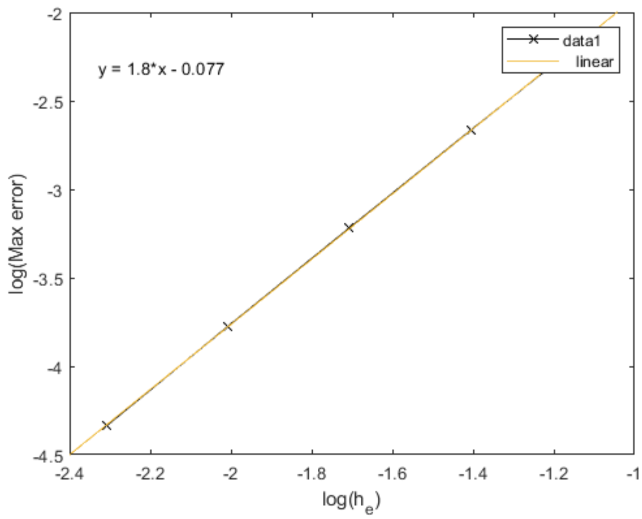


Figure 3: Error vs Element size

As it can be appreciated in the linear regression equation of the figure 3, the slope constant of the error evolution is 1.8.
For instance, to reduce the error from 10^{-2} to 10^{-4}, it is needed increase element resolution by more than 10 times. In particular, increasing by 10 times the element resolution at the boundaries implies having 100 times more elements. Thus, it is needed more than 100 times the original computational effort.

3.2 Modifying the Final Time

Secondly, the error is evaluated in front different end-times to see which is the influence of time.

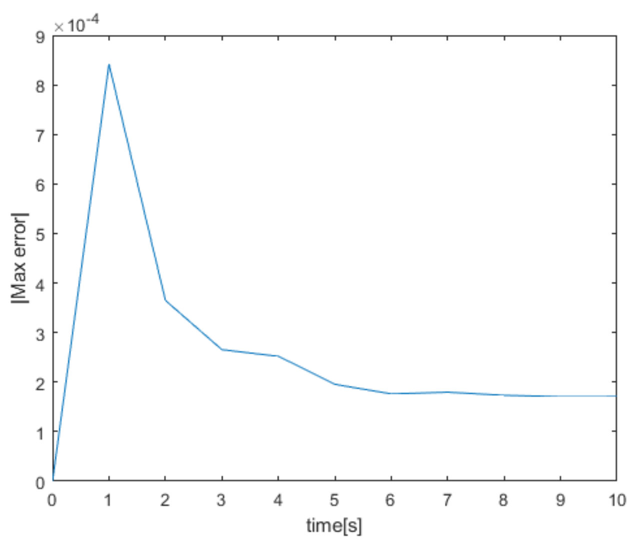


Figure 4: Error vs Final time

As it can be appreciated in the linear regression equation of the figure 4, the converges and for times above 10 seconds. It means it has been reached the steady-state solution and for times further than 10 seconds it is possible to not take into account the temporal terms from the equation.

The next section will explore if removing temporal terms is possible without compromising the error.

3.3 Removing the Time-Dependence

Finally, the time-dependence has been removed from our problem governing equation and evaluated the error against the original one.

The new governing equation is:

$$-\Delta u = 0 \tag{5}$$

Initial value condition at $t = 0$ and the rest of boundary conditions:

$$u(x, y, t = 0) = x^2 + xy - y^2 + 1 \tag{6}$$

$$\begin{array}{l|l} u_n(x = 0, y, t) = -y & u(x, y = 0, t) = x^2 \\ u_n(x = 1, y, t) = 2 + y & u_n(x, y = 1, t) = x - 2 \end{array} \tag{7}$$

Then, the error of the original parabolic equation and the new elliptic equation (no time-dependent) has been plotted for several final times:

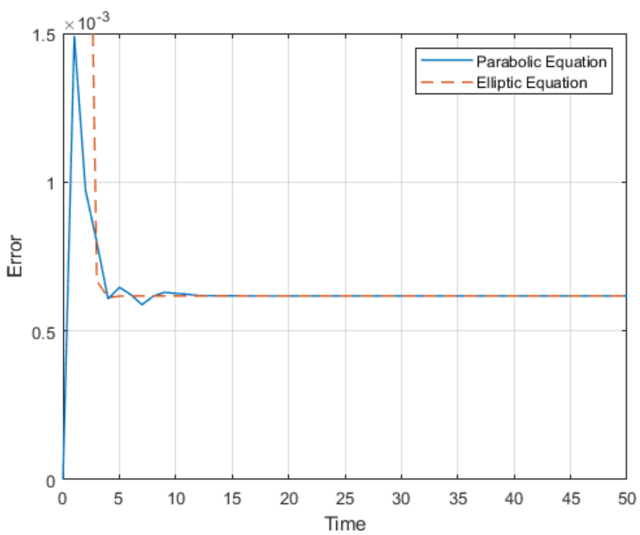


Figure 5: Error vs Final time

There is no difference on the results on steady-state problems. It is a major improvement on computational effort since it is not necessary anymore to solve the problem for several time steps.

Thus, if the temporal dependence is negligible, it is possible to get more accurate results of a problem in the steady-state range by focusing only on increasing mesh resolution while still being affordable in terms of CPU-time.

4 Conclusions

Several conclusion arise from the results

- Regarding mesh refinement or element size, the logarithmic error decreases linearly with the mesh size with a slope of approximately 1.8. Thus, mesh refinement is welcome to improve the accuracy of the solution but a very high cost in CPU-time.
- Regarding our time-dependence problem, it has been demonstrated that after 10 seconds it almost reaches the time equilibrium and the error does not vary noticeably after the mentioned time. So that, when $t \to \infty$ the equation is not time-dependant. In fact, after 10 seconds it can be considered that the system reaches the steady-state.
- To calculate further times, the elliptic not time-dependent PDE version of the original equation can be used matching the same error in results. Thus, elliptic PDE can be used instead of the original with the same level of confidence for steady-state problems, which improves hugely the computational time.