Guidelines for Assignment 2

Enginyeria Aeroespacial Computacional

Joaquín A. Hernández

May 11, 2020

1 Part 1 (basic)

1.1 Statement

The goal of this assignment is to develop a **Matlab program** able to solve any **two dimensional heat conduction problem** using **bilinear quadrilateral elements**.

- To test the performance of the program, consider the heat conduction problem depicted in Figure 1. The coordinates are given in meters. The conductivity matrix is isotropic, with $\boldsymbol{\kappa} = \kappa \begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}$, and $\kappa = 10\ W/°C$. The temperature $u = 0$ is prescribed along edges AB and AD. The heat fluxes $\boldsymbol{q} \cdot \boldsymbol{n} = 0$ and $\boldsymbol{q} \cdot \boldsymbol{n} = 20\ W/m$ are prescribed on edges BC and CD, respectively. A constant heat source $f = 4\ W/m^2$ is applied over the plate.

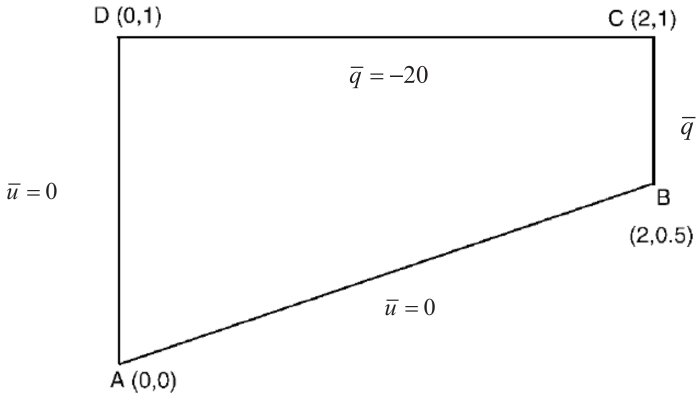


Figure 1

- To assess convergence upon mesh refinement, launch **4 different analyses** with increasing number of finite elements. In particular, use *structured* meshes with
  - $n_{el} = 1$ element
  - $n_{el} = 5 \times 5$ element
  - $n_{el} = 10 \times 10 = 100$ elements
  - $n_{el} = 30 \times 30 = 900$ elements

- For these three cases, plot the distribution of temperature along the edge DC in the same graph.

1.1.1 Programming tasks

- The requested matlab program is not to be developed from scratch. Rather, we recommend you to use the accompanying "template" code mainHEATC.m which incorporates already preprocess and postprocess facilities (using the commercial software GID). There are some parts missing in the code, and you have to complete them in order to generate the results. More specifically, you have to implement

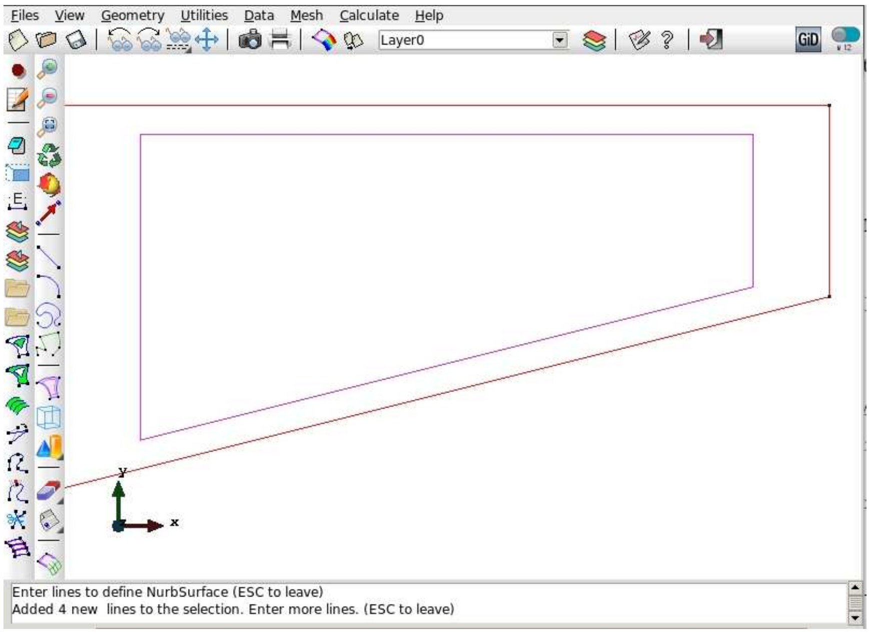
  - The assembly of the conductance matrix, as well as the shape function routines of bilinear quadrilateral elements (see video Video1_Kshape.avi).
  - The assembly of the vector of global source flux vector ($\boldsymbol{F}_s$), the solution of the final system of equations, and the computation of the heat flux vector at each Gauss point (see video Video2_FeqG.avi).

  The procedure for constructing the mesh using the finite element code is explained in what follows.

1.2 Guidelines for pre- and post-processing

1.2.1 Pre-process

1. Download and install GID from here (see video Video3_INSTALLGID.avi)
2. Open *gid* (see video Video4_GID_geom.avi)
3. Create new project (*Files>New*)
4. Save the project (*Files>Save*)
5. Create line (*Geometry>Create>Straight line*) (type its coordinates). Plot the lines A-B-C-D-A. When drawing line from D to A, right-click and select *Contextual>Join*.
6. Create the surface defined by the polygon A-B-C-D-A (Geometry>NURBS surface>By contour)



Enter lines to define NurbSurface (ESC to leave)
Added 4 new lines to the selection. Enter more lines. (ESC to leave)

Figure 2

7. Label surfaces and lines (for materials and boundary conditions, see video Video5_PROBLEMTYPE.avi).

(a) The first time you generate the geometry, the corresponding "Problem Type" has to be loaded. To do this, go to Data > Problem type > Load, and select the folder "PROBLEM_TYPE_simple".

(b) To identify surfaces with distinct material properties, go to Data > Material > Assign > Surfaces. Each selected surface will be associated with a number. This number is used in the matlab input data file to assign the corresponding conductivity

```
  % ------------------------------------------------------------
% 2. Material data.
% ---------------------------------------------------------------
imat =1 ; % Index material
PROPMAT(imat).kappa =  10  ; %  Conductivity  of material "imat" (
    ISOTROPIC)
```

(c) To identify lines with distinct boundary conditions, go to Data > Condition > (Line icon). Each selected line will be associated with a given number. This number will be used to impose both Dirichlet and Neumann Boundary Conditions in matlab's input data file

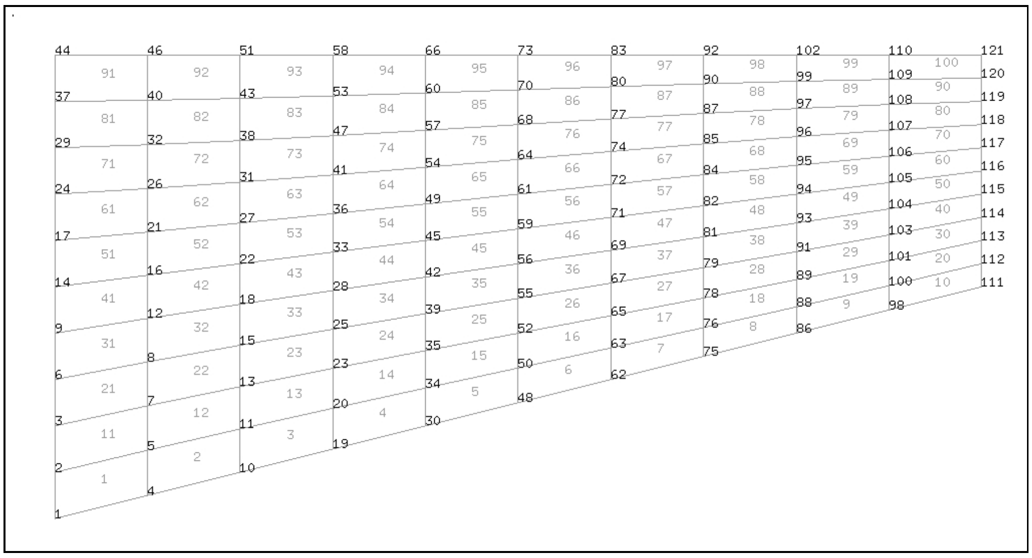
```
  % ------------------------------------------------------------
% 3. Dirichlet boundary conditions (prescribed temperature)
% -----------------------------------------------------------

icond = 1; % Number of condition
DIRICHLET(icond).NUMBER_LINE = 1 ;   % Number of line on which
    temperature  is prescribed
DIRICHLET(icond).PRESCRIBED_TEMPER = 0 ;  % (constant along the line)

icond = 2; % Number of condition
DIRICHLET(icond).NUMBER_LINE = 2;   % Number of line on which temperature
     is prescribed
DIRICHLET(icond).PRESCRIBED_TEMPER = 0 ;  % Prescribed temperature (
    constant along the line)

% -------------------------------------------------
% 4. Neumann Boundary conditions (prescribed flux)
% ------------------------------------------------
icond= 1 ;
NEUMANN(icond).NUMBER_LINE = 4 ;  % Line
NEUMANN(icond).PRESCRIBED_qBAR= -20 ;
```

8. Generation of the the **finite element mesh** (see video Video6_MESH.avi)

(a) Go to Mesh>Element Type>Quadrilateral. Select the surface ABCDA.

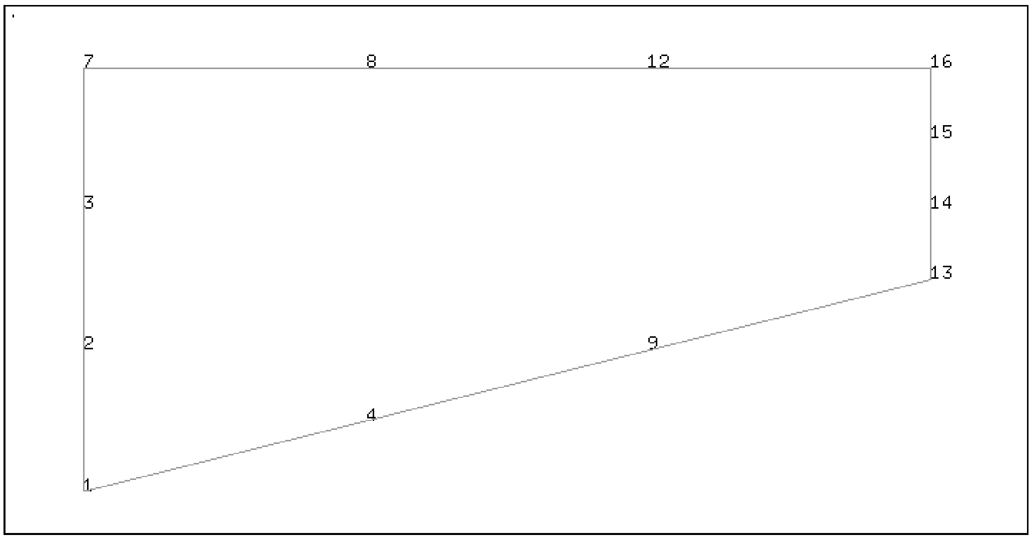
(b) Go to Mesh>Structured>Surfaces>Assign number of cells. Select surface, and then the size of the cells ($n_{el} = 10 \times 10$, type 10, and then select the lines forming the surface)

(c) Generate mesh for domain Ω : Mesh>Generate Mesh.



(d) Generation of mesh for the boundary Γ: Mesh > Create Boundary Mesh

(e) View mesh Γ: Mesh > View mesh boundary



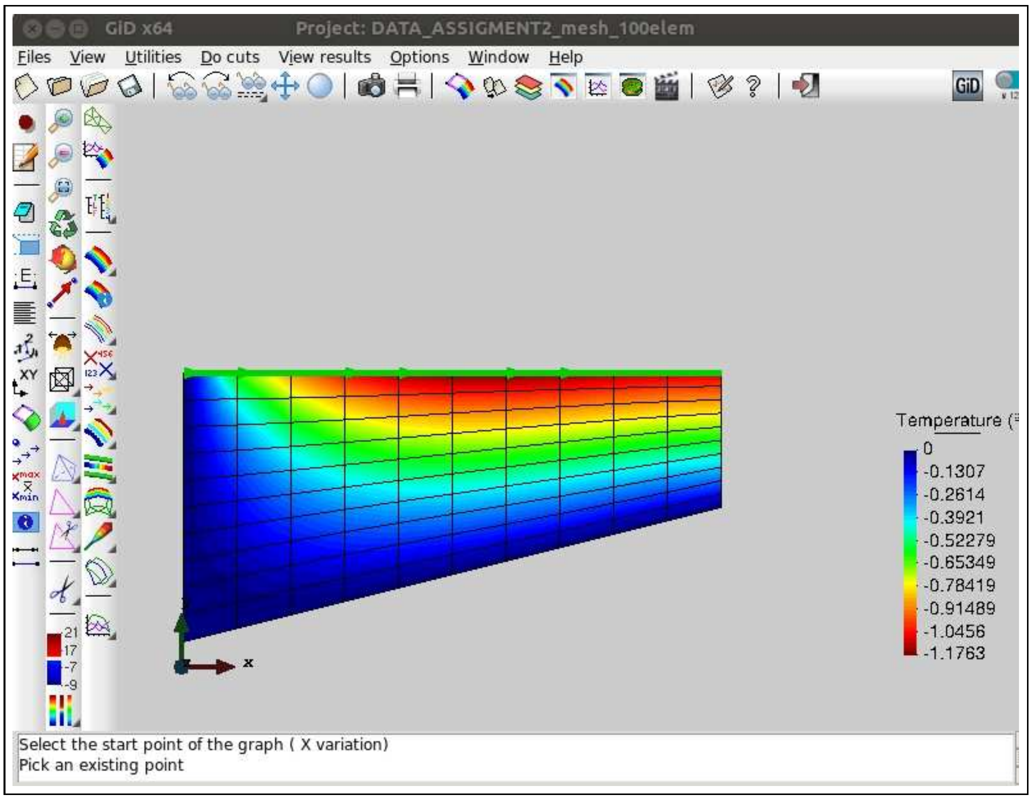
9. Export mesh and boundary conditions data (see video Video7_EXPORT.avi)
   - Files> Export > Gid mesh. Name of the mesh (the same used for the project, and in the folder where the main matlab script is located).
   - Files> Export > Calculation file
10. Change the data in the input matlab file (see video Video8_INPUT.avi).

1.2.2 Post-process (see video Video9_POST.avi)

1. Open gid
2. Go to postprocess: Files>Postprocess
3. Open the file whose name appears in the Matlab command window when running the FE program
4. To plot the distribution of temperatures, go to View results>Contour Fill>Temperature

5. To plot the distribution of temperatures along the top edge

- View results>Graphs>Line Graph>Set X axis> X variation
- View results>Graphs>Line Graph>Set X axis> Temperature



Select the start point of the graph (X variation)
Pick an existing point

You will be prompted for the starting and end points of the graph. Right-click on your mouse to select the option "Contextual > Join".

- To save the graph, go to Files>Export>Graph
- To retrieve the plotted graph anytime, just make Files>Import>Graph

2 Part 2 (advanced)

2.1 Convective heat transfer

2.1.1 Modified conductance matrix and nodal heat flux vector (see Video10_heat.avi)

The weak form of the prototypical linear heat conduction problem is given by (see page 18 of the slides):

$$\text{W.F.} \begin{cases} \text{Given } f : \Omega \to \mathbb{R},\ \bar{q} : \Gamma_b \to \mathbb{R}, \quad \text{find } u \in \mathcal{S} \ \text{ such that} \\ \displaystyle\int_\Omega \boldsymbol{\nabla} v^T \boldsymbol{\kappa} \boldsymbol{\nabla} u \, d\Omega = \int_\Omega v f \, d\Omega + \int_{\Gamma_b} v\bar{q} \, d\Gamma, \qquad \forall v \in \mathcal{V} \end{cases}$$

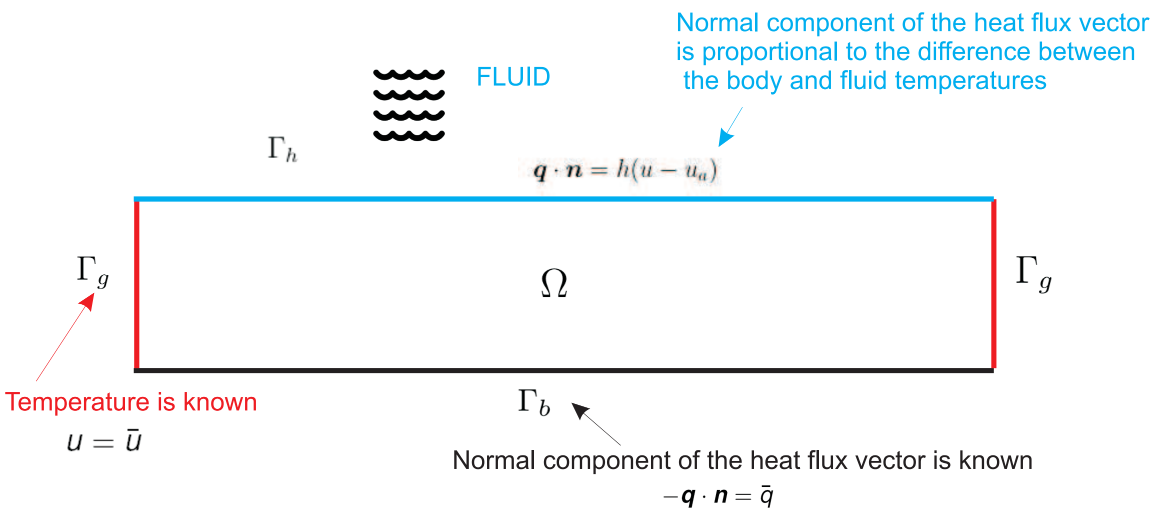

Normal component of the heat flux vector is proportional to the difference between the body and fluid temperatures
FLUID
Γ_h
$\boldsymbol{q}\cdot\boldsymbol{n} = h(u-u_a)$
Γ_g
Ω
Γ_g
Temperature is known
$u=\bar{u}$
Γ_b
Normal component of the heat flux vector is known
$-\boldsymbol{q}\cdot\boldsymbol{n}=\bar{q}$

Figure 3

This boundary value problem assumes that the boundary is split into a portion on which the temperature is known (Γ_g) and another portion on which the normal component of the heat flux vector is known (Γ_b). However, there are physical processes in which the conditions at the boundary cannot be described by either of these boundary conditions. This happens, for instance, when we have *convective heat transfer* between an external fluid and the body under study (see Figure 3). Mathematically, this condition is described in strong form as

$$\boldsymbol{q}\cdot\boldsymbol{n} = h(u-u_a) \qquad \text{on } \Gamma_h$$

where $u-u_a$ is the temperature difference between the wall and the fluid, and h is the convection heat transfer coefficient. Upon introduction of this boundary condition into the formulation of the problem, the corresponding expressions for the coefficient matrix $\boldsymbol{K}$ and the vector of external actions $\boldsymbol{F}$ change with respect to the prototypical problem. In particular, they adopt the forms

$$\boldsymbol{K} = \int_\Omega \boldsymbol{B}^T \boldsymbol{\kappa} \boldsymbol{B} \, d\Omega + \boldsymbol{K}_h$$

$$\boldsymbol{F} = \int_\Omega \boldsymbol{N}^T f \, d\Omega + \int_{\Gamma_b} \boldsymbol{N}^T \bar{q} \, d\Gamma + \boldsymbol{F}_h$$

Which are the expressions for $\boldsymbol{K}_h$ and $\boldsymbol{F}_h$? Explain the procedure to arrive at such expressions.

2.2 Time-dependent heat conduction

2.2.1 Formulation (see Video11_time.avi)

So far we have only addressed finite element analysis of heat conduction for *steady-state* conditions. No *time dependence* has been included in such analyses, as we have assumed conditions in which a *steady state* has been reached. Certainly, transient, time-dependent effects are often of paramount important, since such effects determine whether a steady state is achieved and, furthermore, what that steady state will be.

To account for transient effects, the strong form of the boundary value problem presented in Slide 14 of the theory document has to be modified as follows (the new terms are highlighted in blue):

$$
\boxed{
\begin{array}{l}
\text{Given } f:\Omega\times[0,T]\to\mathbb{R},\ \bar{u}:\Gamma_g\times[0,T]\to\mathbb{R},\ \bar{q}:\Gamma_b\times[0,T]\to\mathbb{R},\ \text{ find } u:\bar{\Omega}\times[0,T]\to\mathbb{R}\ \text{ such that} \\
\qquad \begin{array}{rcll}
\boldsymbol{\nabla}\cdot\boldsymbol{q}-f &=& \rho c\dfrac{\partial u}{\partial t}, & \text{in } \Omega\times[0,T] \\
u &=& \bar{u} & \text{on } \Gamma_g\times[0,T] \\
-\boldsymbol{q}\cdot\boldsymbol{n} &=& \bar{q} & \text{on } \Gamma_b\times[0,T] \\
u(\boldsymbol{x},0) &=& u_0(\boldsymbol{x}) & \text{on } \quad \bar{\Omega} \qquad \text{(initial condition)}
\end{array} \\
\text{where } \boldsymbol{q}=-\boldsymbol{\kappa}\cdot\boldsymbol{\nabla}u.
\end{array}
}
\tag{2.1}
$$

In the preceding equations, t is the time, $[0,T]$ represents the time domain, whereas $\rho(\boldsymbol{x})$ and $c(\boldsymbol{x})$ denote the density and the *specific heat* of the material, respectively. Problem 2.1 represents the prototypical second-order *parabolic* partial differential[1] equation (PDE). Notice that the only difference with respect to the steady state equation is the derivative with respect time appearing in the right-hand side of the differential equation. Since the equation is first-order in time, it requires the specification of the temperature of the body at time $t = 0$ (the initial condition $u(\boldsymbol{x}, 0) = u_0$). It can be shown that the variational expression corresponding to problem 2.1 is given

$$\boldsymbol{c}^T(\boldsymbol{M}\dot{\boldsymbol{d}} + \boldsymbol{K}\boldsymbol{d} - \boldsymbol{F}) = \boldsymbol{0} \tag{2.2}$$

where $\boldsymbol{M}$ is the *capacitance* matrix, whereas $\dot{\boldsymbol{d}}$ denotes the derivative with respect time of the nodal temperature vector $\boldsymbol{d}$. **Find the expression of the capacitance matrix in terms of the shape functions employed for the interpolation**.

[1]Without the time derivative, i.e., as in the problem studied in the theory document, the partial differential equation is called *elliptic*.