User's Manual for the Function `matfem`. A MATlab function for the Finite Element Method

The MATMOL Group (2009)

Introduction

The Finite Element Method (FEM) is a technique for computing the numerical solution of a partial differential equations. In this toolbox we consider equations with the general form:

$$da\frac{\partial z}{\partial t} + \overrightarrow{\nabla}\cdot(\overrightarrow{v}z) = \overrightarrow{\nabla}\cdot\left(\kappa(z)\overrightarrow{\nabla}z\right) + f(z) \tag{1}$$

and with boundary conditions (BC)

$$\overrightarrow{n}\cdot\left(\kappa(z)\overrightarrow{\nabla}z\right) + qz = g\Big|_{\mathscr{B}}. \tag{2}$$

where $\overrightarrow{\nabla}$ is the gradient operator and $\overrightarrow{n}$ is a unitary vector pointing outwards the spatial domain. It is important to remark that BC (2), typically known as Robin BC, can represent other type of BC. For instance, by setting $q = 0$ and $g = 0$ the classical Neumann BC are recovered, this is $\overrightarrow{n}\cdot\overrightarrow{\nabla}z = 0\Big|_{\mathscr{B}}$. For obtaining Dirichlet BC a large value of q is chosen, for instance $q = 1000$, in such case g will be chosen as $g = qz^*$ where z^* is the value of the field in the boundary. Since q and g are very large, the term $\overrightarrow{n}\cdot\left(\kappa(z)\overrightarrow{\nabla}z\right)$ can be neglected as compared with them, and thus Eqn (2) can be approximated by $qz = g$ which is equivalent to $z = z^*$.

In the FEM, the spatial domain is discretized into a number of finite elements. Through this discretization, the original PDE (1) with BC (2) is approximated by a number of ordinary differential equations (ODE). It is not the aim of this report to give an introduction to the FEM, the reader interested in a deeper insight in the FEM is referred to the literature (Reddy, 1993; Zienkiewicz et al., 2005; Pozrikidis, 2005; García, 2008; Vilas, 2008). The system of ODE can be expressed in matrix form as:

$$da\mathcal{MM}\frac{dZ}{dt} + (\kappa\mathcal{DM} + v\mathcal{CM} + q\mathcal{BM})\,Z = \mathcal{F} + \mathcal{G} \tag{3}$$

where $\mathcal{MM}$, $\mathcal{DM}$, $\mathcal{CM}$, $\mathcal{BM}$ are, respectively, the mass, diffusion, convection and homogeneous boundary matrices. Z is the discretized version of the field z as indicated in Figure 1. Multiplying

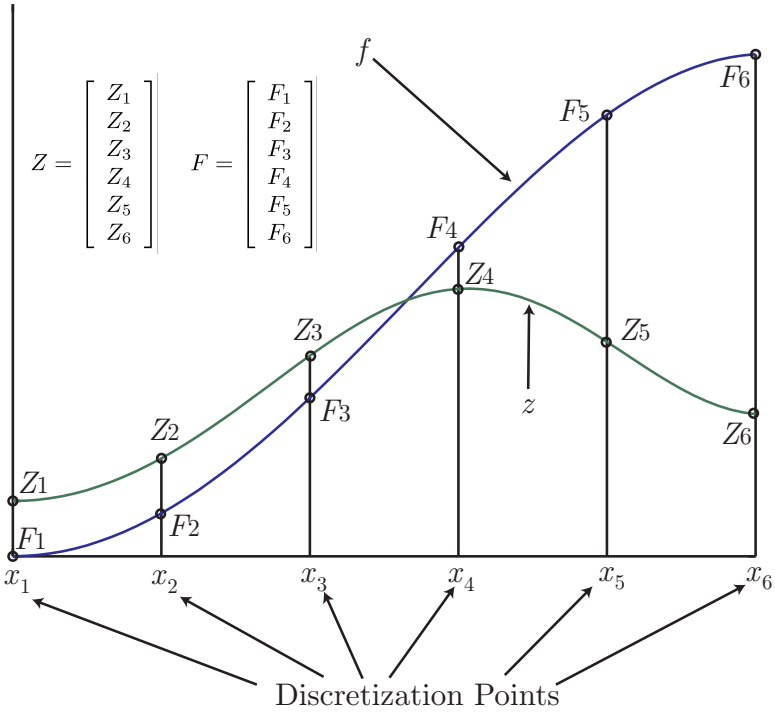

Figure 1: Discretization of field z and function f.

Eqn (3) by $\mathcal{M}\mathcal{M}^{-1}$ and rearranging the terms one has that:

$$da\frac{dZ}{dt} = -\mathcal{M}\mathcal{M}^{-1}\left(\kappa\mathcal{D}\mathcal{M} + v\mathcal{C}\mathcal{M} + q\mathcal{B}\mathcal{M}\right)Z + \mathcal{M}\mathcal{M}^{-1}\mathcal{F} + \mathcal{M}\mathcal{M}^{-1}\mathcal{G} \tag{4}$$

It should be remarked that $F = \mathcal{M}\mathcal{M}^{-1}\mathcal{F}$ corresponds to the value of the discretized version of the nonlinear function f as indicated in Figure 1. Finally, $\mathcal{G}$ is a vector with all the elements zero except in the boundary points (in 1D problems these correspond with the first and last points) in which it takes the value of g in Eqn (2). In the same way, matrix $\mathcal{B}\mathcal{M}$ is a matrix of zeros except in the boundary (for 1D problems $\mathcal{B}\mathcal{M}(1,1)$ and $\mathcal{B}\mathcal{M}(N,N)$). In these points it can be:

- 1 for Robin BC
- 0 for Neumann BC
- 10^5 for Dirichlet BC[1].

Another advantage of the FEM matrices is that they can be employed for approximating spatial derivatives and integrals by algebraic operations as indicated in Table 1.

The objective of section is to explain how to use the `matfem` function for obtaining these matrices.

[1]It should be highlighted that the only requirement on the value of $\mathcal{B}\mathcal{M}$ at a Dirichlet boundary condition is that it has to be large. In the matlab function it has been fixed to 10^5 but other $(500, 1000, 2300, 1e6, ...)$ could be employed.

	Continuous	Discrete
1	$\int_{\mathscr{V}} g(\boldsymbol{\xi}) f(\boldsymbol{\xi}) d\boldsymbol{\xi}$	$\mathcal{G}^T \mathcal{M} \mathcal{M} \mathcal{F}$
2	$\int_{\mathscr{V}} g(\boldsymbol{\xi}) \frac{\partial f(\boldsymbol{\xi})}{\partial \xi_i} \, \mathrm{d}\boldsymbol{\xi}$	$\mathcal{G}^T \mathcal{C} \mathcal{M}_i \mathcal{F}$
3	$\int_{\mathscr{V}} g(\boldsymbol{\xi}) \Delta f(\boldsymbol{\xi}) \, \mathrm{d}\boldsymbol{\xi}$	$-\mathcal{G}^T \left(\mathcal{D} \mathcal{M} + \mathcal{B} \mathcal{M} \right) \mathcal{F}$
4	$\frac{\partial}{\partial \xi_i}$	$\mathcal{M} \mathcal{M}^{-1} \mathcal{C} \mathcal{M}_i$
5	$\frac{\partial^2}{\partial \xi_1^2} + \frac{\partial^2}{\partial \xi_2^2} + \frac{\partial^2}{\partial \xi_3^2}$	$-\mathcal{M} \mathcal{M}^{-1} \left(\mathcal{D} \mathcal{M} + \mathcal{B} \mathcal{M} \right)$

Table 1: Relationships between the continuous spatial derivatives and integrals and their discrete counterparts using the FEM matrices. Vectors $\mathcal{G}$ and $\mathcal{F}$ are the FEM discretized versions of continuous functions $g(\boldsymbol{\xi})$ and $f(\boldsymbol{\xi})$.

The Function `matfem`

This function was constructed from the free library `FSELIB` (see Pozrikidis (2005) for details). The original library consists of a number of Matlab© functions which allow us to carry out different steps of the FEM: from spatial domain discretization (using different techniques) to the computation of the FEM matrices (using different boundary conditions, basis functions,...). The `matfem` presented in this report is limited to the 1D case using first and second order polynomials[2] as basis functions and the user is responsible for defining the spatial domain discretization. Nevertheless, as an advantage, this function is more transparent to the user than the `FSELIB` library. This will allow the user to obtain the FEM matrices and to deal with the boundary conditions in a more systematic way. The way of calling the function is

```
[fem_im] = matfem(xe, bc_x0, bc_xL,fem_om)
```

where the input and output parameters are

```
Input parameters:
xe:       Coordinates of the spatial discretization points.
bc_x0:    Type of boundary conditions at the beginning of the domain.
bc_xL:    Type of boundary conditions at the end of the domain.
fem_om:   This variable indicates the FEM matrices that the user wants to
          compute. The matrix is indicated by means of a string: 'MM' for
          the mass matrix, 'DM' for the diffusion matrix, 'CM' for the
          convection matrix and 'BM' for the boundary matrix. The last
          element of this variable indicates the type of basis functions
```

[2]For constructing these, new discretization points are considered. These points are located at the middle of each finite element.

```
          employed: 'lin' for linear and 'quad' for quadratic. In the case
          of linear basis functions the last element can be omitted.

The boundary type can be 'dir' (Dirichlet boundary conditions), 'neu'
(Neumann boundary conditions) or 'rob' (Robin boundary conditions)

Output parameters:
fem_im:   The output parameters are the matrices indicated in fem_om.
          They must be sorted following the same sequence used in fem_om.
          If quadratic elements are chosen, it may result convenient to
          recover the mesh with the interior points. This is done by
          adding a new output variable.
```

A simple example

Compute the mass and the boundary matrices, using linear basis functions, with Dirichlet boundary conditions in the first point and Neumann in the last point. The spatial domain goes from 0 to π and 21 spatial discretization points are required. In Matlab© this is carried out by:

```
xe                = linspace(0,pi,21);  % Spatial discretization
[mass, boundary] = matfem(xe, 'dir', 'neu','MM','BM');
```

This is equivalent to

```
xe                = linspace(0,pi,21);  % Spatial discretization
[mass, boundary] = matfem(xe, 'dir', 'neu','MM','BM','lin');
```

When quadratic basis functions are chosen, a new input parameter is introduced, so that:

```
xe                    = linspace(0,pi,21);  % Spatial discretization
[mass, boundary,xex] = matfem(xe, 'dir', 'neu','MM','BM', 'quad');
```

Note that an extra output parameter (`xex`) is obtained. `xex` contains the extended mesh with the interior points. For details about this, the reader is referred to the bibliography (Reddy, 1993; Zienkiewicz et al., 2005; Pozrikidis, 2005).

Examples

In this section some problems will be employed to illustrate the use of the `matfem` function.

A simple diffusion problem: FEM solution

Use the `matfem` function to compute the numerical solution of equation:

$$\frac{\partial z}{\partial t} = \kappa \frac{\partial^2 z}{\partial x^2}; \quad \kappa = 0.1 \tag{5}$$

with boundary conditions

$$z(0,t) = 3, \quad \frac{\partial z(L,t)}{\partial x} = 0. \tag{6}$$

These are Dirichlet in the first point and Neumann homogeneous in the last point. The spatial domain is $\mathscr{V} = \{x/x \in [0,\pi]\}$. The initial conditions are chosen as $z(x,0) = -x^2 + 2\pi x + 3$ and the number of discretization points will be $N = 21$. Since no convection is included in the formulation, the FEM matrices to be computed are: $\mathcal{MM}$, $\mathcal{DM}$ and $\mathcal{BM}$ which is done by:

```
L       = pi;                    % Length of the spatial domain
ndisc   = 21;                    % Number of discretization points
xe      = linspace(0,L,ndisc);   % Spatial discretization


% Call to the function which computes the FEM matrices
[MM, DM, BM] = matfem(xe, 'dir', 'neu','MM','DM','BM');
inv_MM       = inv(MM);
```

Note that $\mathcal{BM}(1,1) = 10^5$ (Dirichlet boundary conditions) and $\mathcal{BM}(N,N) = 0$ (Neumann homogeneous boundary conditions). Now we have to define the vector $\mathcal{G}$. Since the boundary conditions are constant, this vector can be defined in the main program. In the first point we have Dirichlet BC so, according to the explanation given in the introduction, the value of g in this point must be $g = qz^*$, thus $\mathcal{G}(1) = 3\mathcal{BM}(1,1)$. In the last point we have $g = 0$. The rest of the code is standard and it is included in `Ex1.m` and `ode_ex1.m`.

When quadratic basis functions are considered, the code is modified so that

```
% Call to the function which computes the FEM matrices
[MM, DM, BM,xe] = matfem(xe, 'dir', 'neu','MM','DM','BM','quad');
ndisc           = length(xe);
inv_MM          = inv(MM);
```

Note that now the number of discretization points (`ndisc`) is recomputed for taking into account the interior points. The rest of the code is the same as in the linear case and it is included in `Ex1_quad.m`.

Reaction-Diffusion-Convection problem: FEM solution

In this example reaction and convection terms are considered. The model equations are

$$\frac{\partial z}{\partial t} = \kappa \frac{\partial^2 z}{\partial x^2} - v\frac{\partial z}{\partial x} + f(z); \quad f(z) = z - z^3 \tag{7}$$

where $\kappa = 0.5$ and $v = 0.01$, with the usual Danckwerts boundary conditions

$$\vec{\mathbf{n}} \cdot \kappa \vec{\boldsymbol{\nabla}} z \Big|_{x=0} = v(z_{in} - z|_{x=0}) \iff \kappa \left. \frac{\partial z}{\partial x} \right|_{x=0} = v(z|_{x=0} - z_{in}) \tag{8}$$

$$\vec{\mathbf{n}} \cdot \vec{\boldsymbol{\nabla}} z \Big|_{x=L} = 0, \tag{9}$$

with z_{in} being the concentration in the inlet stream ($z_{in} = 100sin(7t) + 50$). The spatial domain is $\mathscr{V} = \{x/x \in [0, \pi]\}$. The initial conditions are $z(x, 0) = (-x^2 + 2\pi x + 2\pi k/v + 50)/150$. Regarding the spatial discretization, in this example, a variable element size will be employed. In this regard, from $x = 0$ to $x = L/3$, 11 elements points will be employed. In the rest of the domain 10 elements will be employed. This discretization can be carried out, for instance, as follows:

```
L       = pi;                   % Length of the spatial domain
ndisc   = 21;                   % Number of discretization points
n1      = 11;                   % First part of the reactor
n2      = ndisc - n1 + 1;       % Second part of the reactor
xe1     = linspace(0,L/3,n1);   % First part of the reactor
xe2     = linspace(L/3,L,n2);   % Second part of the reactor
xe      = [xe1 xe2(2:end)];     % Total spatial discretization
```

In this case, the FEM matrices are computed as follows:

```
% Call to the function which computes the FEM matrices
[MM, DM, CM, BM] = matfem(xe, 'rob', 'neu','MM','DM', 'CM', 'BM');
inv_MM = inv(MM);      % This will be employed later
G = zeros(ndisc , 1); % For the boundary conditions
```

Note that, in this example, the boundary conditions are not constant so they cannot be defined in the main program. The dimensions of matrix $\mathcal{G}$ can be defined in the main file as shown above but it has to be filled in the function where the ODEs as indicated below:

```
function dz = ode_ex2(t, z, Sp_oper, G, inv_MM, v)


% Boundary conditions
zin   = 100*sin(7*t)+50; % Inlet concentration
G(1)  = v*zin;              % Boundary condition in the first point
GG    = inv_MM*G;
```

GG in this code corresponds to $\mathcal{M}\mathcal{M}^{-1}\mathcal{G}$ in Eqn (4). The rest of the code is included in Ex2.m and ode_ex2.m.

When quadratic basis functions are considered, the code is modified so that

```
[MM, DM, CM, BM, xe] = matfem(xe, 'rob', 'neu','MM','DM', 'CM',...
                                        'BM', 'quad');
ndisc                = length(xe);
```

The rest of the code is the same as in the linear case and it is included in Ex2_quad.m.

The Burgers equation: FEM solution

The well known Burgers equation presents the following form:

$$\frac{\partial z}{\partial t} = \frac{\partial(-0.5z^2)}{\partial x} + \mu\frac{\partial^2 z}{\partial x^2} = -z\frac{\partial z}{\partial x} + \mu\frac{\partial^2 z}{\partial x^2} \tag{10}$$

The spatial domain is $\mathscr{V} = \{x/x \in [0,1]\}$ and Dirichlet boundary conditions are considered at both sides of the domain. The value for the boundary and initial conditions is obtained through the analytical solution given by:

$$z = \frac{0.1ea + 0.5eb + ec}{ea + eb + ec},$$

with:

$$ea = \exp\left(\frac{0.05}{\mu}(x - 0.5 + 4.95t)\right); \quad eb = \exp\left(\frac{0.25}{\mu}(x - 0.5 + 0.75 * t)\right).$$

$$ec = \exp\left(\frac{0.5}{\mu}(x - 0.375)\right).$$

These equations are were written into the burgers_exact.m function which is included in the toolbox. The number of discretization points will be $N = 201$:

```
%... spatial grid
x0 = 0.0;
xL = 1.0;
n  = 201;
dx = (xL-x0)/(n-1);
x  = [x0 : dx : xL]';
```

The FEM matrices are computed as follows:

```
[MM, DM, CM, BM]    = matfem(x, 'dir', 'dir', 'MM', 'DM', 'CM', 'BM');
inv_MM              = inv(MM);
G                   = zeros(n , 1); % For the boundary conditions
Lapl_op             = -inv_MM*DM;   % Laplacian operator
```

Note that $\mathcal{BM}(1,1) = \mathcal{BM}(N,N) = 10^5$, since Dirichlet boundary conditions are considered at both sides. In this case we decide to compute together the homogeneous part of the boundary -this is, the term qz in Eqn (2)- and the inhomogeneous part -this is, the term g in Eqn (2)-. Thus in the code where the ODEs are defined we have to add:

```
%... boundary conditions at z = 0
zin     = burgers_exact(x0,t);
G(1)    = BM(1,1)*(z(1) - zin);
%... boundary conditions at z = zL
zout    = burgers_exact(xL,t);
G(n)    = BM(n,n)*(z(n) - zout);
GG      = -inv_MM*G;
```

The main difference with respect to the other examples of this manual is the form of the convective term. For defining it, we use the relations of Table 1 and compute the gradient operator as:

```
Grad_op = inv_MM*CM;    % Gradient operator
```

The convective term is contructed, in the script were the ODEs are defined, as:

```
fx     = Grad_op*(0.5*z.^2);
```

The rest of the code is standard and it is included in the scripts `burgers_main_FEM.m` and `burgers_pde_FEM.m`.

When quadratic basis functions are considered, the code is modified so that

```
% Call to the matfem function for computing the FEM matrices
[MM, DM, CM, BM,x] = matfem(x, 'dir', 'dir', 'MM', 'DM', 'CM',...
 'BM', 'quad');
n                  = length(x);
```

The rest of the code is the same as in the linear case (see the `burgers_main_FEM_quad.m` matlab script).

References

García, M. R. (2008). *Identification and Real Time Optimisation in the Food Processing and Biotechnology Industries.* PhD thesis, University of Vigo, Spain. Available online at http://digital.csic.es/handle/10261/4662.

Pozrikidis, C. (2005). *Introduction to Finite and Spectral Element Methods using Matlab*. Chapman & Hall/CRC.

Reddy, J. N. (1993). *An Introduction to the Finite Element Method*. McGraw-Hill, 2nd edition.

Vilas, C. (2008). *Modelling, Simulation and Robust Control of Distributed Processes: Application to Chemical and Biological Systems*. PhD thesis, University of Vigo, Spain. Available online at http://digital.csic.es/handle/10261/4236.

Zienkiewicz, O. C., Taylor, R. L., and Zhu, J. Z. (2005). *The Finite Element Method: Its Basis & Fundamentals*. Elsevier, Amsterdam, 6^{th} edition.