Università
della
Svizzera
italiana

**Facoltà
di scienze
informatiche**

**Software Atelier: Differential Equations** **Academic Year 2016/2017**

Instructor: Dr. Drosos Kourounis TA: Hardik Kothari

# Assignment 4 - Finite Element Solution of Poisson's equation in 3D

Due date: Monday 7 November 2016, 10:30

### Solution of Poisson's equation

We seek the discrete solution of Poisson's equation

$$
\begin{aligned}
-\nabla^2 u(x,y,z) &= f(x,y,z), \quad \in \Omega \\
\frac{\partial u}{\partial n} &= 0, \quad y = 1, z = 1, \\
u(x,y,z) &= u_0(x,y,z), \text{otherwise},
\end{aligned}
$$

where $\Omega = [0,1]^3$. We discretize the domain $\Omega$ using a quadrilateral $N_x \times N_y \times N_z$ grid of trilinear elements.

1. Find the analytical expression of $f(x,y)$ so that the exact solution of the PDE is

$$u_0(x,y) = x\, e^{-(y-1)^2(z-1)^2}.$$

2. Verify that the exact solution satisfies the homogeneous Neumann conditions at $y = 1$ and at $z = 1$.

3. Implement the solution in MATLAB following the steps described below.

### 1. Mesh generation

Modify the mesh generation routine you already have, so that it assumes three input arguments for the cube dimensions $L_x, L_y, L_z$ and three for the grid size $N_x, N_y, N_z$. It outputs the hexahedral mesh of the cube.

### 2. Finite element assembly

- Compute by hand the mass matrix and the Laplacian on the Hexahedron $[0,1]^3$.
- Compute the mass and Laplacian matrices on the single-element grid consisting of the hexahedron you used in the previous step using your code and verify that previous steps produces the same matrices.

## 3. Local operators

Modify the following functions to obtain the local operators for hexahedral trilinear elements.

$$M_e = \int_{V_e} N_i N_j dV_e$$

```
function I = Mass2DSymbolic()
    xi  = sym('xi', 'real');  eta = sym('eta', 'real');
    dx  = sym('dx', 'real');  dy  = sym('dy', 'real');

    c=[-1 -1; 1 -1; 1  1; -1  1];

    J(1,1) = 0.5*dx; J(2,2) = 0.5*dy;

    for i=1:4
        N(i) = 1/4*( 1+c(i,1)*xi )*( 1+c(i,2)*eta );
    end

    F   = det(J)*N'*N;
    M   = int(int(F, 'xi', -1, 1), 'eta', -1, 1);
end
```

$$K_e = \int_{V_e} \nabla N_i \cdot \nabla N_j dV_e$$

```
function I = Laplace2DSymbolic()
    xi  = sym('xi', 'real'); eta = sym('eta', 'real');
    dx  = sym('dx', 'real'); dy  = sym('dy', 'real');

    c=[-1 -1; 1 -1; 1  1; -1  1];
    J(1,1) = 0.5*dx; J(2,2) = 0.5*dy;

    for i=1:4
        N(i) = 1/4*( 1+c(i,1)*xi )*( 1+c(i,2)*eta );
    end

    Nx  = diff(N, 'xi'); Ny  = diff(N, 'eta');
    dN = [Nx; Ny];
    Jd  = inv(J) * dN;
    F   = det(J)*Jd'*Jd;
    I   = int(int(F, 'xi', -1, 1), 'eta', -1, 1);
end
```

## 4. Linear Solver

1. Use the direct sparse solver implemented in MATLAB (backslash) to solve the linear system for $h = \delta z = \delta y = \delta x = 1/N$ for $N = 10, 20, 40, 80, 160$. Create a table listing the running time of the direct sparse solver. Use the superfast assembly routine you were provided.

2. For each $N$ of the previous question, compute the time needed for the assembly, and the solution of the linear system and plot them using the logarithmic scale on both axes.

## 5. Convergence study

Perform a study of the convergence. Compute the $L^2$ and $H^1$ norms of the error and plot it **using logarithmic scale on both axes** as a function of $h = \delta z = \delta y = \delta x = 1/N$ for $N = 10, 20, 40, 80, 160$. Note that the discrete $L^2$ and $H^1$ norms are easily obtained from

$$
\begin{aligned}
\|u - u_h\|_{L^2} &= \sqrt{(u - u_h)^T M (u - u_h)} \\
\|u - u_h\|_{H^1} &= \sqrt{(u - u_h)^T M (u - u_h) + (u - u_h)^T K (u - u_h)},
\end{aligned}
$$

where $u_h$ is the vector of the discrete finite element solution you obtained by solving the linear system $Ku_h = b$ and $u$ is the vector of the exact values of the solution (recall that the exact solution is given by $u_0(x, y, z)$) evaluated at the grid-points.

## 6. Visualization

Visualize the solution using (ViSiT) for three values of $N$, $N = 10, 80, 160$ and also visualize the exact solution at the end (as the fourth plot) for comparison.