PX920 Workshop: **Implementation of Periodic Boundary Conditions (PBCs)**

Łukasz Figiel

WMG, University of Warwick

PBCs via Multiple Point Constraints

Consider a periodic cell (RVE) bounded by the boundaries Γ_1 is coupled with Γ_2, while Γ_3 is coupled with Γ_4.

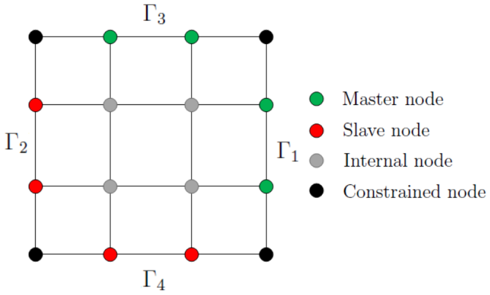

Figure: RVE with matching meshes on the opposite sides.

The nodes within the RVE are divided into **master** (m), **slave** (s), **internal** (*free*) and **corner** (*constrained*) nodes.

PBCs via MPCs

Multiple Point Constraints (**MPCs**) is a common of way of enforcing nodal constraints, and applying periodic boundary conditions. The *slave* DOFs are related to the *master* DOFs through constraint equations, and the original system of equations is modfied, so the slave nodes are eliminated.

The constraint equations can be written as

$$\begin{bmatrix} \boldsymbol{G}_m & \boldsymbol{G}_s \end{bmatrix} \begin{bmatrix} \boldsymbol{d}_m \\ \boldsymbol{d}_s \end{bmatrix} = \mathbf{0} \tag{1}$$

where $\boldsymbol{d}_m$ and $\boldsymbol{d}_s$ are the vectors containing master and slave DOFs, while $\boldsymbol{G}_m$ and $\boldsymbol{G}_s$ are matrices containing constants (typically 1 or -1 as coefficients in front of master and slave DOFs).

PBCs via MPCs

Solving for the slave DOFs

$$\boldsymbol{d}_s = -\boldsymbol{G}_s^{-1}\boldsymbol{G}_m\boldsymbol{d}_m \tag{2}$$

which gives the complete expression of DOFs in terms of master DOFs, and $\boldsymbol{G}_m$ and $\boldsymbol{G}_s$ are matrices

$$\begin{bmatrix} \boldsymbol{d}_m \\ \boldsymbol{d}_s \end{bmatrix} = \begin{bmatrix} \boldsymbol{I} \\ -\boldsymbol{G}_s^{-1}\boldsymbol{G}_m \end{bmatrix} \boldsymbol{d}_m \implies \boldsymbol{d} = \boldsymbol{T}\boldsymbol{d}_m \tag{3}$$

where $\boldsymbol{d}$ is the vector of global DOF, $\boldsymbol{T}$ is the transformation matrix, and $\boldsymbol{I}$ is the identity matrix.

PBCs via MPCs

The global system of FE equations is given by

$$\boldsymbol{K}\boldsymbol{d} = \boldsymbol{r} \tag{4}$$

Substitution of $\boldsymbol{d}$ from the second term in (3) into (4), and further pre-multiplication by $\boldsymbol{T}^{\mathrm{T}}$ yields

$$\boldsymbol{T}^{\mathrm{T}}\boldsymbol{K}\boldsymbol{T}\boldsymbol{d}_m = \boldsymbol{T}^{\mathrm{T}}\boldsymbol{r} \implies \boldsymbol{K}_m\boldsymbol{d}_m = \boldsymbol{r}_m \tag{5}$$

Solution of (5) for $\boldsymbol{d}_m$ can be used to calculate the total nodal displacement vector $\boldsymbol{d}$ in (3).

Constraint equations with linearly dependent DOFs

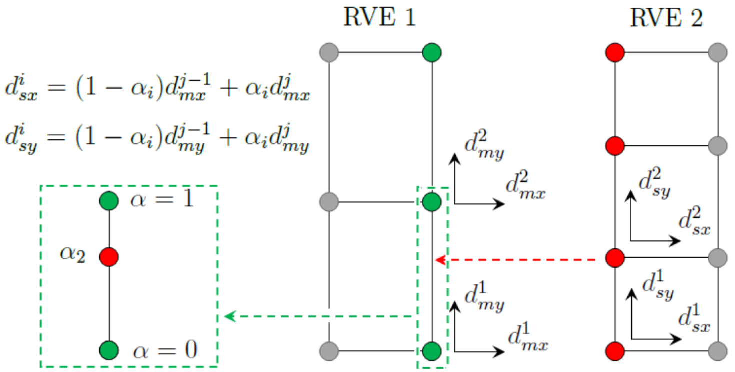

Figure: The slave DOF is linearly related to the two adjacent master DOFs.

MPCs for non-matching grids

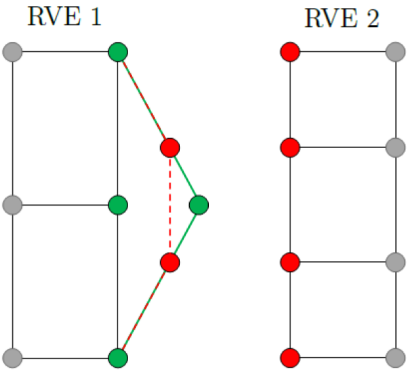

Figure: RVE deformation for non-matching grids.

Transformation matrix T - pseudo-code (Part 1)

The T-matrix

- id1 = node numbers for boundary 1
- id3 = node numbers for boundary 3
- ids2 = slave node numbers for boundary 2 (excluding corner nodes)
- ids4 = slave node numbers for boundary 4 (excluding corner nodes)
- idm = master node numbers (excluding corner nodes, including internal nodes)
- slavedofs = slave DOFs
- masterdofs = master DOFs

G = zeros(2∗number of slave nodes , 2∗number of RVE nodes)
FOR i = 1 **TO** i = number of boundary 4 slave nodes **DO**
  G(2∗i−1 , 2∗ids4(i)−1) = −1
  G(2∗i , 2∗ids4(i)) = −1
  - Call mpc_alpha.m to get **alpha** and **masternode** = the upper masternode where alpha is equal 1

  G(2∗i−1 , 2∗id3(masternode)−1) = alpha
  G(2∗i−1 , 2∗id3(masternode−1)−1) = 1 − alpha
  G(2∗i , 2∗id3(masternode)) = alpha
  G(2∗i , 2∗id3(masternode−1)) = 1 − alpha
END DO

Transformation matrix T - pseudo-code (Part 2)

```
c = number of boundary 4 slave nodes
FOR i = 1 TO i = number of boundary 2 slave nodes DO
    G(2*(i + c)-1 , 2*ids2(i)-1) = -1
    G(2*(i + c) , 2*ids2(i)) = -1
    • Call mpc_alpha.m to get alpha and masternode = the upper masternode
      where alpha is equal 1
    G(2*(i + c)-1 , 2*id1(masternode)-1) = alpha
    G(2*(i + c)-1 , 2*id1(masternode-1)-1) = 1 - alpha
    G(2*(i + c) , 2*id1(masternode)) = alpha
    G(2*(i + c) , 2*id1(masternode-1)) = 1 - alpha
END DO

Gs = G(: , slavedofs)
Gm = G(: , masterdofs)
T = zeros(2*number of RVE nodes , 2*number of master nodes)
T(masterdofs , :) = identity matrix
T(slavedofs , :) = -inv(Gs) * Gm
```

- Eliminate the columns in T corresponding to the constrained corner DOFs