Multiscale Modelling of Reinforced Concrete in OOFEM

Adam Sciegaj*

* Department of Mechanics of Materials and Structures
Faculty of Civil and Environmental Engineering
Gdańsk University of Technology
April 3, 2023

Abstract

This document summarises various aspects of multiscale modelling in the open-source C++ finite element code OOFEM developed in the course of my PhD project: *"Mutltiscale modelling of reinforced concrete structures"* carried out at Chalmers University of Technology during 2015–2020. Different types of boundary conditions, finite elements and material models used within the multiscale framework are presented and the FE^2 modelling technique is illustrated with numerous examples. Furthermore, this document also serves as a documentation to the `oofemTests` repository containing all examples.

Contents

1 Prerequisites

The user should be familiar with the format of OOFEM input file. A good starting point is to consult the official manuals (especially the *Input Data Format Specification*, *Element Library Manual*, and *Material Model Library Manual*) available at `https://oofem.org/doku.php?id=en:manual`. It is advised that the user first gets familiar with the OOFEM workflow (preprocessing, solving and postprocessing) on simple examples.

To make use of all features, the following is required:

- A working OOFEM installation. The modified source code can be found in the GitHub repository. All features (related to multiscale modelling of reinforced concrete) have been merged to the official OOFEM version, so that one could be used as well. To pass all tests, OOFEM should be compiled with PETSc (`USE_PETSC` flag), and Python support (`USE_PYTHON_EXTENSION` flag).
- CMake installation for compilation and running the tests.
- Some test cases require to use a PETSc solver instead of the default one. The development version of PETSc can be found at GitLab. The reader is referred to official PETSc website for more information on installation. The `release` branch should be cloned PETSc should be compiled with MUMPS solver. Use

  ```
  ./configure --download-mumps --download-scalapack --download-parmetis --download-metis
  --download-ptscotch
  ```

  when configuring to have PETSc installed with MUMPS. If OOFEM is not compiled with PETSc, the following test cases will fail:
  - `fe2Disp/planeStress/deepbeam04`
  - `rveDisp/3D/plateBending`
  - `rveDisp/3D/plateTension`
  - `rveDisp/3D/sm1`
  - `rveDisp/3D/sm2`
  - `rveDisp/3D/sm3`
- Python 3 installation. Necessary if the user wants to run an analysis on a single RVE with prescribed strain/slip history (i.e., this is not required for full FE^2 analysis). OOFEM should then be compiled with Python support, i.e., with the flag `USE_PYTHON_EXTENSION` set on. If OOFEM is not compiled with Python support, the following test cases will fail:
  - `fe2Disp/planeStress/fakefe2disp`
  - `fe2DispSlip/planeStress/fakefe2dispslip01`
  - `fe2DispSlip/planeStress/fakefe2dispslip02`
- ParaView (to be downloaded from official website and installed) can be used for postprocessing. The `*.vtu` and `*.pvd` files produced by the VTKXML output module can be used for visualizing results.
- SALOME (to be downloaded from official website) makes preprocessing much easier. Output from SALOME can be converted to OOFEM format using the `unv2oofem` tool. More information can be found in OOFEM wiki.

2 Test cases

A number of unit tests was developed during the project. Test cases aim to test not only each developed features, but also the FE^2 approach, which makes use of many components simultaneously. The unit tests are available in the GitHub repository `oofemTests` and are divided into following categories:

- `elements`: test of various 2D and 3D finite elements
- `fe2Disp`: tests of classical (plane stress) FE^2 analyses in 2D
- `fe2DispSlip`: tests of enhanced (macroscopic strain and slip fields) FE^2 analyses in 2D (in plane stress)
- `materials`: tests of material models
- `rveDisp`: tests on 2D and 3D reinforced concrete RVEs (plane stress, macroscopic Euler-Bernoulli beam, macroscopic Kirchhoff-Love plate)
- `rveDispSlip`: tests on 2D reinforced concrete RVEs (with boundary conditions prescribing both the strain, macroscopic reinforcement slip and slip gradient)

The easiest way to run all tests is to clone the repository and provide the path to the OOFEM executable (either `debug` or `release` version, although the `release` version is recommended for a speedy resolution) with CMake:

```
git clone https://gitub.com/adsci/oofemTests oofemTests
cd oofemTests
cmake -DOOFEM_PATH=/path/to/oofem .
```

After that, the test suite can be run with ctest, e.g.,

```
ctest -j 8
```

runs the tests suites in parallel using 8 jobs.

3 The RVE

The Representative Volume Element (RVE) is the main building block of multiscale model. In this project, the RVE considered the plain concrete solid, the reinforcement bars, and the bond between them. Both 2D and 3D models of the RVE were created, but only 2D RVEs were used in full FE^2 analyses. Below, the RVE models are briefly described.

3.1 2D

3.1.1 Model summary

In 2D, the RVE had a square shape with reinforcement bars arranged in a regular grid, see Figure 1. Depending on the size, either one, two or three reinforcing bars in each direction were considered [5, 3, 4]. Although in most cases both horizontal and vertical rebars were considered, [2] included examples with reinforcement only in one direction.

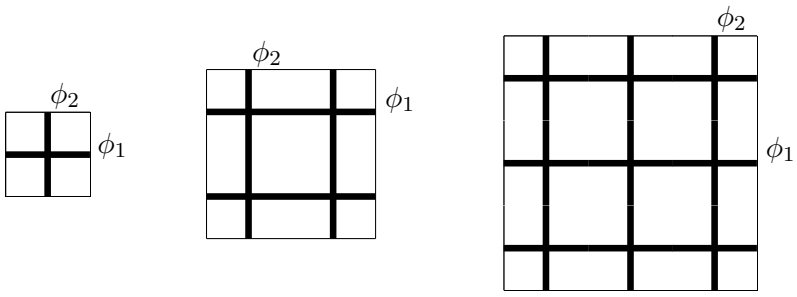

Figure 1: Examples of 2D reinforced concrete RVEs. From [5].

Sample input files with the 2D RVE can be found e.g., in `rveDisp/2D` directory. In particular, the tests `rveDisp01, rveDisp02, rveDisp03, rveDisp04` contain input files for the smallest RVE from Figure 1. In the same directory, Python scripts for creating the RVE mesh in SALOME and exporting it into OOFEM format are present (`rveDisp/2D/salomeAutoRVE`) The RVEs were modelled as follows:

- plane stress rectangular elements (`planestress2d`) were used for the concrete solid, e.g.,

```
planestress2d 1 nodes 4 3 46 58 13
```

- beam elements with linear shape function (`libeam2d`) were used for the reinforcement bars. Note that the keyword `xy` was specified to make sure that bending happens in-plane, e.g.,

```
libeam2d 101 nodes 2 124 125 crossSect 5 xy
```

- interface elements (`IntElLine1`) were placed between the coincident nodes of concrete and beam elements to describe the bond action (along the interface), e.g.,

```
IntElLine1 121 nodes 4 2 14 135 136  crossSect 7
```

- slip across the inerface (transverse slip) was prevented *strongly* by appropriate tyings between concrete and steel, e.g.,

```
node 122 coords 3 0.0 0.0 0.0 dofidmask 3 1 2 6 doftype 3 1 0 0
         mastermask 3 1 0 0
```

  The transverse constraint can also be applied *weakly*, see Section 5.3 for more information.
- a `SimpleCS` cross-section and `MazarsModel` material model was used for the plane-stress concrete elements
- a `LayeredCS` cross-section and `MisesMat` material model was used for the steel. Layered cross-section is needed if we want to use a plasticity model for the steel.
- an `InterfaceCS` cross-section and `bondceb` material was used to model the interface.

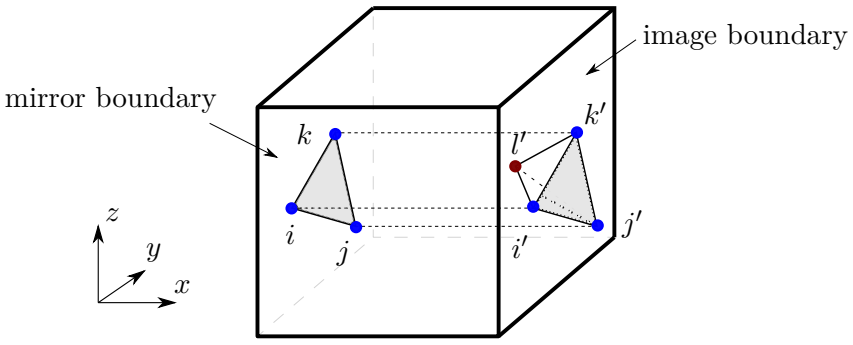

Figure 2: Image and mirror boundaries in the 3D RVE.

3.1.2 Analysis

The RVE model can either be used in the FE^2 setting, or off-line, for analyses on the subscale level. The application of the model to full FE^2 analysis requires the use of special boundary conditions, which allow for efficient homogenization of the subscale results, i.e., computing the homogenized work-conjugates (stress, transfer stress, reinforcement stress). For more information on this, the reader is referred to Sections 4 and 5. Those boundary conditions allow for efficient homogenization of the results, such as computing the homogenized stress tensor etc.

Off-line subscale analyses should follow the usual OOFEM convention, i.e., specifying analysis type and boundary conditions. Afterwards, the results can be visualized using the `vtkxml` export module.

3.1.3 Pertinent test cases

Following test cases can be studied for more insight:

- `materials/bondceb01` - simple pull-out test in 2D
- `materials/bondceb02` - simple pull-out test in 3D

3.2 3D

Modelling of the reinforced concrete RVE in 3D involves using 3D solid tetrahedra and beam elements. Between those, 3D node-to-node interface elements are placed. However, before the final model is described, some requirements of the mesh are presented.

3.2.1 Periodicity of the finite element mesh

A modelling paradigm has been chosen so that strongly periodic boundary conditions are implicitly used. This, in turn requires that the mesh is also periodic. To fulfil the periodicity requirements, all nodes present on the image boundary must have a counterpart on the mirror boundary, see Figure 2, where an example of periodicity (in x-direction) is shown for a solid tetrahedron element present on an image boundary.

Note that in general, the periodicity can extend in x-, y- and z-directions (depending on the type of the homogenization/mode of the macroscale - see Section 4) To ensure consistency between different modes, the following convention has been used to find the mirror boundary for a cube/cuboid RVE:

- We define right-handed Cartesian coordinate system with the x-axis to the right. The system is aligned with the cube/cuboid.
- All faces with outward-pointing normal unit vectors that are in *opposite* direction to the axes of coordinate system are considered *mirror faces*.
- If an edge is adjacent to *two mirror faces*, it is considered an *mirror edge*.
- The corner which belongs to *mirror edges* is considered to be the *mirror corner*.
- Mirror boundary is considered to be the *union* of the mirror faces, mirror edges and mirror corner.

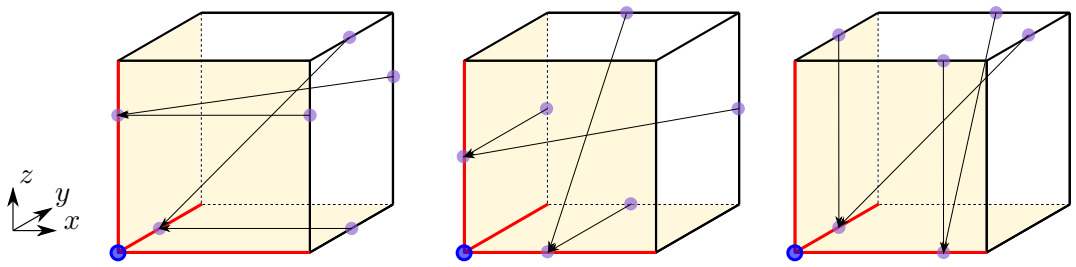

Figure 3: Periodic mapping of points lying on the edges of image boundary.

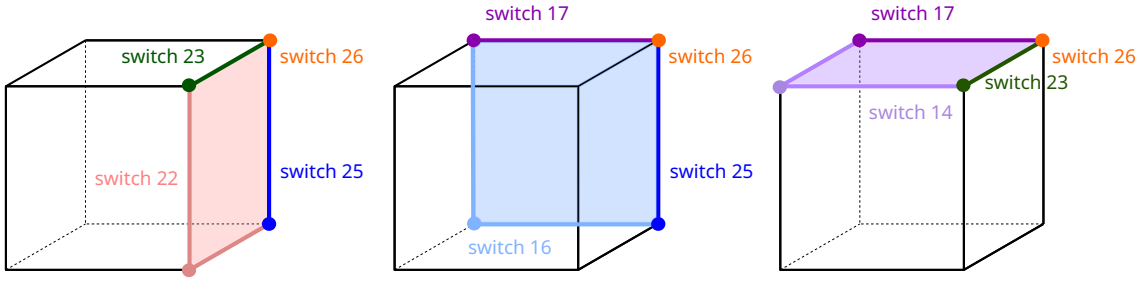

Figure 4: Switches used in periodic mapping (separately for x-, y- and z-direction, from left to right).

The periodic mapping $\varphi(\boldsymbol{x})$ maps any point outside the image boundary to a corresponding point on the image boundary. In case of points lying on the image faces, mapping is straightforward, i.e., a point on the opposite face is considered. In case of points on image edges, the point is mapped on the corresponding mirror edge with the same direction as the image edge. Furthermore, all image corners are mapped to the mirror corner. An example of edge mapping is shown in Figure 3, where the mirror boundary has been marked with light yellow (faces), red (edges) and blue (corner). To make the image more legible, the mapping has been split into three direction (x, y, and z, going from the left)

In order to use the mapping in OOFEM input files, it is necessary to be able to find a mirror node for any image node. To make that easier, a set of **switches** has been introduced. Switches are indicator variables which tell the program how to calculate the coordinates of the mirror node given an image node. This is done by specifying the offset between the image and mirror boundaries. Each switch consists of a triple of ternary values (x, y, z). Each value of the triple describes where the pertinent coordinate of the mirror node can be found:

- $x = 1$ means the x-coordinate of the image is to the right of the mirror. Accordingly, $x = -1$ means the x-coordinate of the image is to the left of the mirror. $x = 0$ means that the x-coordinates of the image and the mirror are the same.
- $y = 1$ means the y-coordinate of the image is behind the mirror. Accordingly, $y = -1$ means the y-coordinate of the image is before the mirror. $y = 0$ means that the y-coordinates of the image and the mirror are the same.
- $z = 1$ means the z-coordinate of the image is above the mirror. Accordingly, $z = -1$ means the z-coordinate of the image is below the mirror. $z = 0$ means that the z-coordinates of the image and the mirror are the same.

For example, if we consider the image face shown in Figure 2, we see that all points on it are situated to the right to the mirror boundary. At the same time, the y- and z-coordinates of the mirror nodes are the same as those of the image nodes. This corresponds to the indicator triple $(x = 1, y = 0, z = 0)$.

As each value in the triple is either -1, 0, or 1, there exist in total 27 different switches. However, as the triple $(0, 0, 0)$ is an identity mapping, we don't consider that case. The remaining 26 switches are are summarised in Table 1. To summarize, all switches used in mapping for points located on the image boundary are shown in Figure 4. For clarity, periodicity in each direction is shown separately.

Table 1: Switches used in periodic mapping.

Switch	x	y	z
1	-1	-1	-1
2	-1	-1	0
3	-1	-1	1
4	-1	0	-1
5	-1	0	0
6	-1	0	1
7	-1	1	-1
8	-1	1	0
9	-1	1	1
10	0	-1	-1
11	0	-1	0
12	0	-1	1
13	0	0	-1
14	0	0	1
15	0	1	-1
16	0	1	0
17	0	1	1
18	1	-1	-1
19	1	-1	0
20	1	-1	1
21	1	0	-1
22	1	0	0
23	1	0	1
24	1	1	-1
25	1	1	0
26	1	1	1

Table 2: Macroscopic modes and corresponding `dofidmask`s.

Mode	Macroscopic components	ndofs	dofidmask
Truss	$\bar{\varepsilon}_{xx}$	1	31
Membrane	$\bar{\varepsilon}_{xx}$, $\bar{\varepsilon}_{xy}$, $\bar{\varepsilon}_{yx}$, $\bar{\varepsilon}_{yy}$	4	31 38 39 32
3DVoigt	$\bar{\varepsilon}_{xx}$, $\bar{\varepsilon}_{yy}$, $\bar{\varepsilon}_{zz}$, $\bar{\gamma}_{yz}$, $\bar{\gamma}_{xz}$, $\bar{\gamma}_{xy}$	6	31 32 33 40 41 42
3D	$\bar{\varepsilon}_{xx}$, $\bar{\varepsilon}_{xy}$, $\bar{\varepsilon}_{xz}$, $\bar{\varepsilon}_{yx}$, $\bar{\varepsilon}_{yy}$, $\bar{\varepsilon}_{yz}$, $\bar{\varepsilon}_{zx}$, $\bar{\varepsilon}_{zy}$, $\bar{\varepsilon}_{zz}$	9	31 38 36 39 32 34 37 35 33
Beam	$\bar{\varepsilon}_{xx}$, $\bar{\varepsilon}_{zx}$,$\bar{\kappa}_{xx}$	3	31 37 43
Plate	$\bar{\varepsilon}_{xx}$, $\bar{\varepsilon}_{xy}$, $\bar{\varepsilon}_{yx}$, $\bar{\varepsilon}_{yy}$, $\bar{\varepsilon}_{zx}$, $\bar{\varepsilon}_{zy}$, $\bar{\kappa}_{xx}$, $\bar{\kappa}_{yy}$, $\bar{\kappa}_{xy}$, $\bar{\kappa}_{yx}$	10	31 38 39 32 37 35 43 44 49 50

3.2.2 RVE master node

Each RVE needs to have a master node. This special node acts as a link between the macroscale and the subscale. Boundary conditions can be set on the master node, i.e., prescribing certain components of macroscopic tensors. After solving the RVE problem, reactions in the master node correspond to the homogenized work-conjugates, e.g., stresses, normal forces and bending moments. The master node needs to fulfil following requirements requirements:

- the coordinates of the master node should be (L_x, L_y, L_z), i.e., length, width and height of the RVE, where L_i is the side length of the RVE in direction i. These values are used together with the switches to calculate the positions of the mirror nodes,
- no finite element should be explicitly connected to this node.
- the exception to the above are boundary elements (`ltrspaceboundary, libeam3dboundary` and derived classes), which have the master node as their fifth node (more on boundary elements in the following subsection),
- it has a `dofidmask` according to Table 2. More information on macroscale modes can be found in Section 4. An example definition is provided in Listing 1, where a master node with number 126 has been defined. The node has coordinates (0.1, 0.1, 0.1), meaning that the RVE is a cube with a side length of 0.1. `dofidmask` of this node specifies a full 3D mode, and boundary condition 2 is prescribed on each degree of freedom.

```
node 126 coords 3 1.000000e-01 1.000000e-01 1.000000e-01
         dofidmask 9 31 38 36 39 32 34 37 35 33 bc 9 2 2 2 2 2 2 2 2 2
```

Listing 1: Example of master node definition

3.2.3 Boundary elements

All element with at least one node located on the image boundary need to be boundary elements. Boundary elements are a special class of element, which will use the mirror nodes when assembling stiffness matrices and internal force vectors. At the moment, there are 2 classes that implement boundary elements:

- `ltrspaceboundary` and derived classes implement a boundary version of the `ltrspace` 4-node tetrahedral solid element,
- `libeam3dboundary` and derived classes implement a boundary version of the `libeam3d` 2-node beam element.

Boundary elements have the same number of nodes as the parent, plus the RVE master node. Also, a special `location` keyword needs to be used when defining boundary elements. Following the keyword, an integer array with switches corresponding to each node is placed. For example, Listing 2 defines a

Table 3: Boundary element names and pertinent test cases.

Mode	Element name	Test case
Truss	`ltrspaceboundarytruss`, `libeam3dboundarytruss`	`elements/ltrspaceboundary01`, `elements/libeam3dboundary01`
Membrane	`ltrspaceboundarymembrane`, `libeam3dboundarymembrane`	`elements/ltrspaceboundary02`, `elements/libeam3dboundary02`
3DVoigt	`ltrspaceboundaryvoigt`, `libeam3dboundaryvoigt`	`elements/ltrspaceboundary03`, `elements/libeam3dboundary03`
3D	`ltrspaceboundary`, `libeam3dboundary`	`elements/ltrspaceboundary04`, `elements/libeam3dboundary04`
Beam	`ltrspaceboundarybeam`, `libeam3dboundarybeam`	`elements/ltrspaceboundary05`, `elements/libeam3dboundary05`
Plate	`ltrspaceboundaryplate`, `libeam3dboundaryplate`	`elements/ltrspaceboundary06`, `elements/libeam3dboundary06`

couple of boundary elements. The `ltrspaceboundary` element with number 14 uses 4 normal nodes (99, 9, 45, 90) and the master node (126). `location` keyword specifies that node 9 has switch 17, node 45 has switch 14, and node 90 has switch 22. The first node (99) does not have a switch, which means it is not located on the image boundary. Similarly, the `libeam3dboundary` element with number 8 has 2 normal nodel (8, 5) and the master node (126). Note that the master node in an RVE is common for all boundary elements. Reference node 10 is used to orient the beam correctly in space (the reader is referred to *OOFEM Element Library Manual*).

```
# ltrspaceboundary element definition
ltrspaceboundary 14 nodes 5 99 9 45 90 126 location 4 0 17 14 22
# libeam3dboundary element definition
libeam3dboundary 8 nodes 3 8 5 126 refnode 10 crossSect 1 location 2 0 22
```

Listing 2: Example of boundary elements definition

Knowing the switches and the sizes of the RVE (indirectly through the coordinates of the master node), the element can compute coordinates of all mirror nodes. Those are later used to construct the stiffness matrix and internal force vectors, as described in more detail in [6]. For each parent element, six different types of boundary elements were implemented, depending on the macroscopic mode:

3.2.4 Model summary

A sample input file with the 3D RVE can be found e.g., in `rveDisp/3D` directory. In particular, the tests `beamTension, beamBending, plateTension, plateBending` contain input files for a complete reinforced concrete RVE. As a reminder, it is required that the RVE has a periodic mesh. More details on mesh generation can be found in [6]. The RVEs were modelled as follows:

- solid tetrahedral elements (`ltrspace`) were used for the concrete solid **not on the image boundary**, e.g.,

```
ltrspace 1481 nodes 4 1605 6933 1389 3801
```

- 3D beam elements with linear shape function (`libeam3d`) were used for the reinforcement bars not on the image boundary, e.g.,

```
libeam3d 44477 nodes 2 7041 7042 refnode 7069
```

- 3D node-to-node interface elements (`bondlink3d`) were placed between the nearest nodes of concrete and beam elements to describe the bond action (along the interface), e.g.,

```
bondlink3d 44506 nodes 2 7043 4719 dirvector 3 1.00e+00 0.00e+00 0.00e+00
                 length 9.946627e-03 length_end 4.913651e-02 diameter 1.60e-02
```

(`dirvector` defines a unit vector along the direction of the reinforcement bar with diameter specified by `diameter`. Parameter `length` defines the length of the interface pertinent to this element - it is a portion of the overall length of the rebar in the RVE)

- a `SimpleCS` cross-section and `idm1` material model was used for the solid concrete elements
- a `FiberedCS` cross-section and `MisesMat` material model was used for the steel. Fibered cross-section is needed if we want to use a plasticity model for the steel in 3D.
- an `InterfaceCS` cross-section and `linkslip` material was used to model the interface.

3.2.5 Analysis

The RVE model can only be used for analyses on the subscale level. Macroscopic quantities (such as stress tensor components, slopes or curvatures) can be prescribed on the RVE by applying a suitable boundary condition on the degree of freedom of the master node listed in Table 2. The effective (homogenized) quantities can then be obtained as reactions in the master node.

3.2.6 Pertinent test cases

Following test cases can be studied for more insight:

- `elements/bond_link_1` - test of `bondlink3d` element
- `elements/bond_link_2` - test of `bondlink3d` element
- `elements/bond_link_3` - test of `bondlink3d` element
- `elements/ltrspaceboundary01` - test of `ltrspaceboundarytruss` element
- `elements/ltrspaceboundary02` - test of `ltrspaceboundarymembrane` element
- `elements/ltrspaceboundary03` - test of `ltrspaceboundaryvoigt` element
- `elements/ltrspaceboundary04` - test of `ltrspaceboundary` element
- `elements/ltrspaceboundary05` - test of `ltrspaceboundarybeam` element
- `elements/ltrspaceboundary06` - test of `ltrspaceboundaryplate` element
- `elements/libeadm3dboundary01` - test of `libeam3dboundarytruss` element
- `elements/libeadm3dboundary02` - test of `libeam3dboundarymembrane` element
- `elements/libeadm3dboundary03` - test of `libeam3dboundaryvoigt` element
- `elements/libeadm3dboundary04` - test of `libeam3dboundary` element
- `elements/libeadm3dboundary05` - test of `libeam3dboundarybeam` element
- `elements/libeadm3dboundary06` - test of `libeam3dboundaryplate` element
- `materials/linkslip01` - test of `linkslip` material model

4 Macroscale modes

This section follows closely the description of different multiscale models for reinforced concrete structures given in [1], to which the reader is kindly referred. Instead of focusing on mathematical derivations, only summary of the information will be given here.

The basic layout of the FE^2 multiscale modelling technique is presented in Figure 5. For a given point in the large-scale model, the effective field (e.g., strain) is imposed on the RVE via suitable boundary conditions. Subsequently, the RVE problem is solved, and the subscale results are *homogenized* and brought back to the large-scale. Use of different types of boundary conditions on the RVE in OOFEM will be covered in Section 5, whereas this section will focus on the process of formatting the prolonged fields to be used in suitable boundary conditions. A few macroscale modes have already been presented in Table 2 according with the macroscopic components that are prolonged onto the RVE. Below, a few macroscale modes used within the project are presented in more detail.

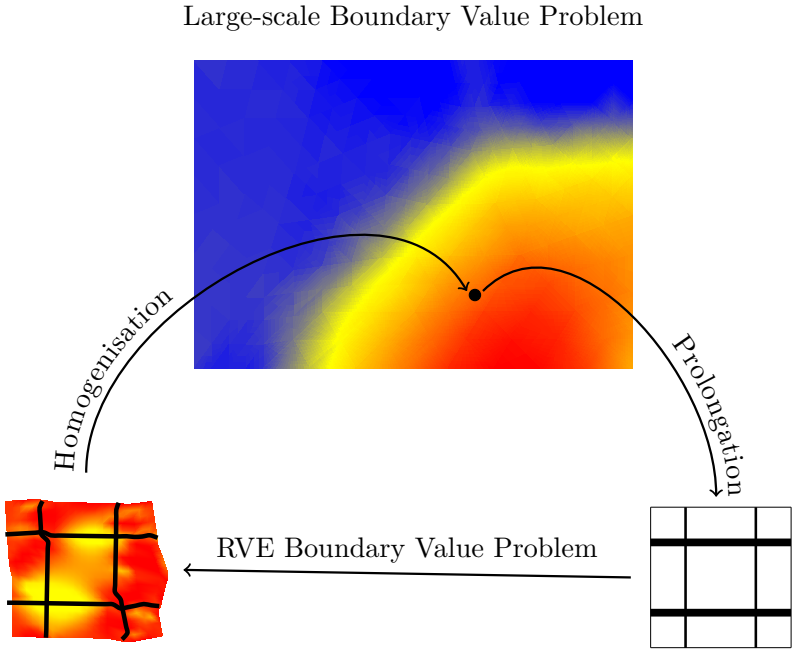

Figure 5: FE^2 scheme. From [1].

4.1 Plane stress

In plane stress, the unknown large-scale field is the displacement ($\bar{\boldsymbol{u}} = [\bar{u} \quad \bar{v}]$). This field is prolonged onto the RVE via the strain tensor (or displacement gradient):

$$\bar{\boldsymbol{\varepsilon}} = [\bar{\boldsymbol{u}} \otimes \nabla] = \begin{bmatrix} \bar{\varepsilon}_{xx} & \bar{\varepsilon}_{xy} \\ \bar{\varepsilon}_{yx} & \bar{\varepsilon}_{yy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \bar{u}}{\partial x} & \dfrac{\partial \bar{u}}{\partial y} \\ \dfrac{\partial \bar{v}}{\partial x} & \dfrac{\partial \bar{v}}{\partial y} \end{bmatrix} \tag{1}$$

All we need in the 2D plane stress setting is a way to prescribe a given strain tensor on the RVE. This is done via the keyword `dispGrad` used in boundary conditions (Section 5). The keyword is followed by the strain tensor $\bar{\boldsymbol{\varepsilon}}$ entered as a real matrix according to the OOFEM Input specification. Note that the matrix is specified in row major order, i.e., the columns are separated by spaces, and the rows are separated by semicolons. For example, the following strain tensor is expressed in the OOFEM format in Listing 6:

$$\bar{\boldsymbol{\varepsilon}} = \begin{bmatrix} 0.3 \cdot 10^{-6} & 0.4 \cdot 10^{-6} \\ 0.4 \cdot 10^{-6} & -0.5 \cdot 10^{-6} \end{bmatrix}$$

```
dispGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
```

Listing 3: OOFEM input format for $\bar{\boldsymbol{\varepsilon}}$.

The homogenized values (stress tensor) can be computed as the average stress over the volume of the RVE, see [5] for details on how to do it. This can be useful when studying a single RVE. However, it is not necessary to calculate the homogenized stresses explicitly in the FE^2 analysis, as this is handled automatically by the boundary conditions.

4.2 Euler-Bernoulli beam

For the Euler-Bernoulli beam problem on the macroscale, the unknown fields are the longitudinal displacement ($\bar{u}$) and the deflection ($\bar{w}$).The prolonged quantities are the longitudinal strain ($\bar{\varepsilon}_{xx}$), the slope ($\bar{\varepsilon}_{zx}$) and the curvature ($\bar{\kappa}_{xx}$). Those components are defined as:

$$\bar{\varepsilon}_{xx} = \frac{\partial \bar{u}}{\partial x} \tag{2}$$

$$\bar{\varepsilon}_{zx} = \frac{\partial \bar{w}}{\partial x} \tag{3}$$

$$\bar{\kappa}_{xx} = \frac{\partial^2 \bar{w}}{\partial x^2} \tag{4}$$

Unlike the 2D plane stress case, no suitable multiscale boundary conditions were explicitly implemented in OOFEM. This prevents running a full FE^2 analysis for macroscopic beam elements at the moment. However, the macroscopic fields can be directly prescribed on the RVE via suitable degrees of freedom of the master node (see Table 2). This way, the macroscopic quantities are prescribed using **strongly periodic boundary conditions** (see Section 5), and subscale analysis on the RVE level can still be performed. This can be done using the conventional `BoundaryCondition` class in OOFEM. For example, prescribing the macroscopic components corresponds to the input shown in Listing 4. In the following, node 9018 is the master node of the RVE:

$$[\bar{\varepsilon}_{xx} \quad \bar{\varepsilon}_{zx} \quad \bar{\kappa}_{xx}] = [0.3 \quad 0.0 \quad 1.0]$$

```
# set 1 contains the master node
BoundaryCondition 1 loadTimeFunction 1 dofs 3 31 37 43 values 3 0.3 0 1 set 1
Set 1 nodes 1 9018
```

Listing 4: OOFEM input format in case of an Euler-Bernoulli problem at macroscale.

The homogenized quantities (normal force, shear force and bending moment) can be obtained as reaction forces in the master node (on the pertinent degrees of freedom). Note that some rescaling with the RVE size might be necessary to get the actual values of the forces.

4.2.1 Pertinent test cases

Following test cases can be studied for more insight:

- `rveDisp/3D/macroBeam01` - prescribing macroscopic input on the 3D RVE (only solid elements)
- `rveDisp/3D/macroBeam02` - prescribing macroscopic input on the 3D RVE (only solid elements)
- `rveDisp/3D/macroBeam03` - prescribing macroscopic input on the 3D RVE (only solid elements)
- `rveDisp/3D/beamTension` - tension test on a 3D RVE (reinforced concrete). Presented in [6].
- `rveDisp/3D/beamBending` - bending test on a 3D RVE (reinforced concrete). Presented in [6].

4.3 Kirchhoff-Love plate

For the Kirchhoff-Love plate problem on the macroscale, the unknown fields are the longitudinal displacements $(\bar{u}, \bar{v})$ and the deflection $(\bar{w})$.The prolonged quantities are the membrane strains $(\bar{\varepsilon}_{xx}, \bar{\varepsilon}_{xy}, \bar{\varepsilon}_{yx}, \bar{\varepsilon}_{yy})$, the slopes $(\bar{\varepsilon}_{zx}, \bar{\varepsilon}_{zy})$ and the curvatures $(\bar{\kappa}_{xx}, \bar{\kappa}_{xy}, \bar{\kappa}_{yx}, \bar{\kappa}_{xx})$. Those components are defined as:

$$\begin{bmatrix} \bar{\varepsilon}_{xx} & \bar{\varepsilon}_{xy} \\ \bar{\varepsilon}_{yx} & \bar{\varepsilon}_{yy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \bar{u}}{\partial x} & \dfrac{\partial \bar{u}}{\partial y} \\ \dfrac{\partial \bar{v}}{\partial x} & \dfrac{\partial \bar{v}}{\partial y} \end{bmatrix} \tag{5}$$

$$\begin{bmatrix} \bar{\varepsilon}_{zx} & \bar{\varepsilon}_{zy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \bar{w}}{\partial x} & \dfrac{\partial \bar{w}}{\partial y} \end{bmatrix} \tag{6}$$

$$\begin{bmatrix} \bar{\kappa}_{xx} & \bar{\kappa}_{xy} \\ \bar{\kappa}_{yx} & \bar{\kappa}_{yy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial^2 \bar{w}}{\partial x^2} & \dfrac{\partial^2 \bar{w}}{\partial x \partial y} \\ \dfrac{\partial^2 \bar{w}}{\partial y \partial x} & \dfrac{\partial^2 \bar{w}}{\partial y^2} \end{bmatrix} \tag{7}$$

Unlike the 2D plane stress case, no suitable multiscale boundary conditions were explicitly implemented in OOFEM. This prevents running a full FE^2 analysis for macroscopic plate elements at the moment. However, the macroscopic fields can be directly prescribed on the RVE via suitable degrees of freedom of the master node (see Table 2). This way, the macroscopic quantities are prescribed using **strongly periodic boundary conditions** (see Section 5), and subscale analysis on the RVE level can still be performed. This can be done using the conventional `BoundaryCondition` class in OOFEM. For example, prescribing the macroscopic components corresponds to the input shown in Listing 5. In the following, node 2835 is the master node of the RVE:

$$[\bar{\varepsilon}_{xx}, \bar{\varepsilon}_{xy}, \bar{\varepsilon}_{yx}, \bar{\varepsilon}_{yy}, \bar{\varepsilon}_{zx}, \bar{\varepsilon}_{zy}, \bar{\kappa}_{xx}, \bar{\kappa}_{yy}, \bar{\kappa}_{xy}, \bar{\kappa}_{yx}] = [0.3, 0, 0, -0.5, 0, 0, 0.5, -1.0, 0, 0]$$

```
# set 1 contains the master node
BoundaryCondition 1 loadTimeFunction 1 dofs 10 31 38 39 32 37 35 43 44 49 50
                    values 10 0.3 0 0 -0.5 0 0 0.5 -1 0 0 set 1
Set 1 nodes 1 2835
```

Listing 5: OOFEM input format in case of an Kirchhoff-Love problem at macroscale.

The homogenized quantities (membrane forces, shear forces and bending moments) can be obtained as reaction forces in the master node (on the pertinent degrees of freedom). Note that some rescaling with the RVE size might be necessary to get the actual values of the forces.

4.3.1 Pertinent test cases

Following test cases can be studied for more insight:

- `rveDisp/3D/macroPlate01` - prescribing macroscopic input on the 3D RVE (only solid elements)
- `rveDisp/3D/macroPlate02` - prescribing macroscopic input on the 3D RVE (only solid elements)
- `rveDisp/3D/macroPlate03` - prescribing macroscopic input on the 3D RVE (only solid elements)
- `rveDisp/3D/plateTension` - tension test on a 3D RVE (reinforced concrete).
- `rveDisp/3D/plateBending` - bending test on a 3D RVE (reinforced concrete). Presented in [6].
- `rveDisp/3D/sm1` - uniaxial bending test SM1. Presented in [6].
- `rveDisp/3D/sm2` - uniaxial bending (with membrane loads) test SM2. Presented in [6].
- `rveDisp/3D/sm3` - biaxial bending test SM2. Presented in [6].

4.4 Maroscopic reinforcement slip

One enhancement that was introduced in the project at the macroscale (in plane stress) is the enrichment by the macroscopic reinforcement slip field, $\bar{\boldsymbol{s}} = [\bar{s}_x \quad \bar{s}_y]$. The idea behind introduction of the new macroscopic field was to allow for the transfer of reinforcement slip across macroscopic elements. The macroscopic slip represents the displacement of the reinforcement grid relative to the concrete, see Figure 6.

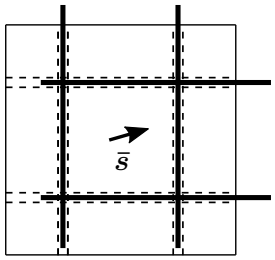

Figure 6: Physical interpretation of the macroscopic slip field. From [3].

Before (in plane stress case), only the displacement gradient was needed as input to the RVE problem. Now, due to the extra macroscopic field, two additional components are necessary. Therefore, the following are needed as the input for boundary conditions on the RVE problem:

- strain tensor (displacement gradient). This component is the same as previously.

$$\bar{\boldsymbol{\varepsilon}} = [\bar{\boldsymbol{u}} \otimes \nabla] = \begin{bmatrix} \bar{\varepsilon}_{xx} & \bar{\varepsilon}_{xy} \\ \bar{\varepsilon}_{yx} & \bar{\varepsilon}_{yy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \bar{u}}{\partial x} & \dfrac{\partial \bar{u}}{\partial y} \\ \dfrac{\partial \bar{v}}{\partial x} & \dfrac{\partial \bar{v}}{\partial y} \end{bmatrix} \tag{8}$$

- macroscopic slip field:

$$\bar{\boldsymbol{s}} = [\bar{s}_x \quad \bar{s}_y] \tag{9}$$

- slip gradient (gradient of the macroscopic slip field):

$$\bar{\boldsymbol{g}} = [\bar{\boldsymbol{s}} \otimes \nabla] = \begin{bmatrix} \bar{g}_{xx} & \bar{g}_{xy} \\ \bar{g}_{yx} & \bar{g}_{yy} \end{bmatrix} = \begin{bmatrix} \dfrac{\partial \bar{s}_x}{\partial x} & \dfrac{\partial \bar{s}_x}{\partial y} \\ \dfrac{\partial \bar{s}_y}{\partial x} & \dfrac{\partial \bar{s}_y}{\partial y} \end{bmatrix} \tag{10}$$

The corresponding work conjugates are:

- $\bar{\boldsymbol{\sigma}}$ - effective stress (conjugated with strain),
- $\bar{\boldsymbol{\tau}}_{\boldsymbol{b}}$ - effective transfer stress (conjugated with macroscopic slip)
- $\bar{\boldsymbol{\sigma}}_{\boldsymbol{s}}$ - effective reinforcement stress (conjugated with slip gradient)

For more details on the macroscopic slip, as well as for complete definitions and derivations of the aforementioned quantities, the reader is referred to [3].

All we need in the 2D plane stress setting (with macroscopic slip) is a way to prescribe a given strain, slip and slip gradient ($\bar{\boldsymbol{\varepsilon}}, \bar{\boldsymbol{s}}, \bar{\boldsymbol{g}}$) on the RVE. This is done via the keywords `dispGrad`, `slip` and `slipGrad` used in boundary conditions (Section 5). `dispGrad` keyword is followed by the strain tensor $\bar{\boldsymbol{\varepsilon}}$ entered as a real matrix. The slip vector $\bar{\boldsymbol{s}}$ entered as a real array follows the `slip` keyword. Last, the `slipGrad` keyword is followed by the slip gradient $\bar{\boldsymbol{g}}$ entered as a real matrix. For example, the following macroscopic input is expressed in the OOFEM format in **??**:

$$\bar{\boldsymbol{\varepsilon}} = \begin{bmatrix} 0.3 \cdot 10^{-6} & 0.4 \cdot 10^{-6} \\ 0.4 \cdot 10^{-6} & -0.5 \cdot 10^{-6} \end{bmatrix}, \quad \bar{\boldsymbol{s}} = \begin{bmatrix} 1 \cdot 10^{-5} & -2 \cdot 10^{-5} \end{bmatrix}, \quad \bar{\boldsymbol{g}} = \begin{bmatrix} 0.3 \cdot 10^{-6} & 0.4 \cdot 10^{-6} \\ 0.4 \cdot 10^{-6} & -0.5 \cdot 10^{-6} \end{bmatrix}$$

```
dispGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
slip 2 1e-5 -2e-5
slipGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
```

Listing 6: OOFEM input format for $(\bar{\boldsymbol{\varepsilon}}, \bar{\boldsymbol{s}}, \bar{\boldsymbol{g}})$.

The homogenized values (stress, transfer stress and reinforcement stress) can be computed as the average stress over the volume of the RVE, see [3] for details on how to do it. This can be useful when studying a single RVE. However, it is not necessary to calculate the homogenized stresses explicitly in the FE^2 analysis, as this is handled automatically by the boundary conditions.

In order to handle two separate macroscopic fields in multiscale modelling, special finite elements are needed. This is due to the fact that conventional plane stress elements have only two degrees of freedom per node (displacement), whereas now four degrees of freedom (two displacements and two slips) are needed. For this purpose, two new elements have been implemented:

- `qtrplstrslip` element, which is based on the `qtrplstr`, a 6-node isoparametric plane stress triangular element with quadratic shape functions (see Figure 7),
- `qplanestress2dslip` element, which is based on the `qplanestress2d`, an 8-node isoparametric plane stress quadrilateral element with quadratic shape functions (see Figure 8).

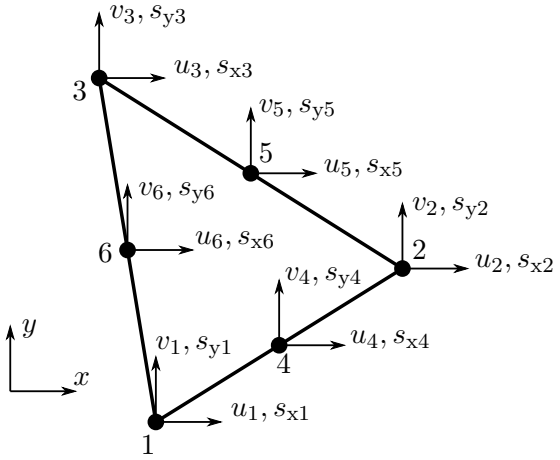

Figure 7: `qtrplstrslip` element. From [3].

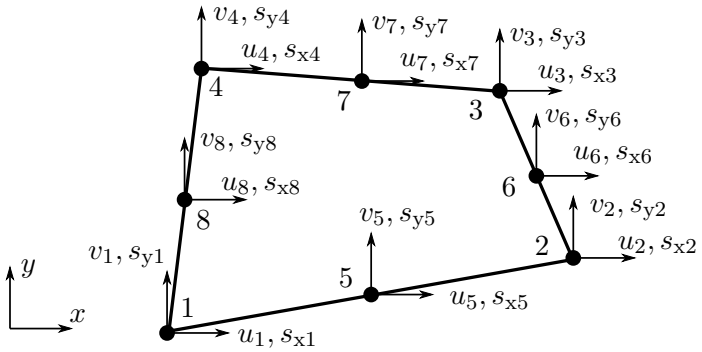

Figure 8: `qplanestress2dslip` element.

These elements have four degrees of freedom per node. The use syntax (and node numbering convention) of the elements is the same as the base elements, see *OOFEM Element Library Manual* for more information. An example use is as below:

```
qtrplstrslip 1 nodes 6 1 2 3 4 5 6 crossSect 1
qplanestress2dslip 2 nodes 8 1 2 3 4 5 6 7 8 crossSect 1 nip 9
```

The new degrees of freedom have received numbers `51` and `52`, for the slip in x- and y-directions, respectively. This allows boundary conditions to prescribe specific values of the macroscopic slip in certain nodes, e.g.

```
BoundaryCondition 3 loadTimeFunction 1 dofs 4 1 2 51 52
                    values 4 1e-4 -2e-4 1e-5 3e-5 set 3
```

prescribed the displacement ($u = 10^{-4}$, $v = -2 \cdot 10^{04}$) and macroscopic slip ($s_x = 10^{-5}$, $s_y = 3 \cdot 10^{-5}$) on set 3. It is noteworthy that the slip and displacement fields are uncoupled and separated when computing the stiffness matrix and internal force vectors. For more details on implementation, see [3] or the pertinent classes in OOFEM source code.

4.4.1 Pertinent test cases

Following test cases can be studied for more insight:

- `elements/qtrplstrslip01` - test of `qtrplstrslip` element
- `elements/qplanestress2dslip01` - test of `qplanestress2dslip` element

5 Subscale boundary conditions

Subscale boundary conditions (BC) allow for prescribing a macroscopic quantity (e.g., strain, slip or slip gradient) in such a way, that after solving the RVE problem, the homogenized quantity (averaged over the volume of the RVE) will be equal to the prescribed macroscopic one. The most popular types of boundary conditions used in multiscale modelling, are Dirichlet, Neumann, and Periodic boundary conditions, see Figure 9.

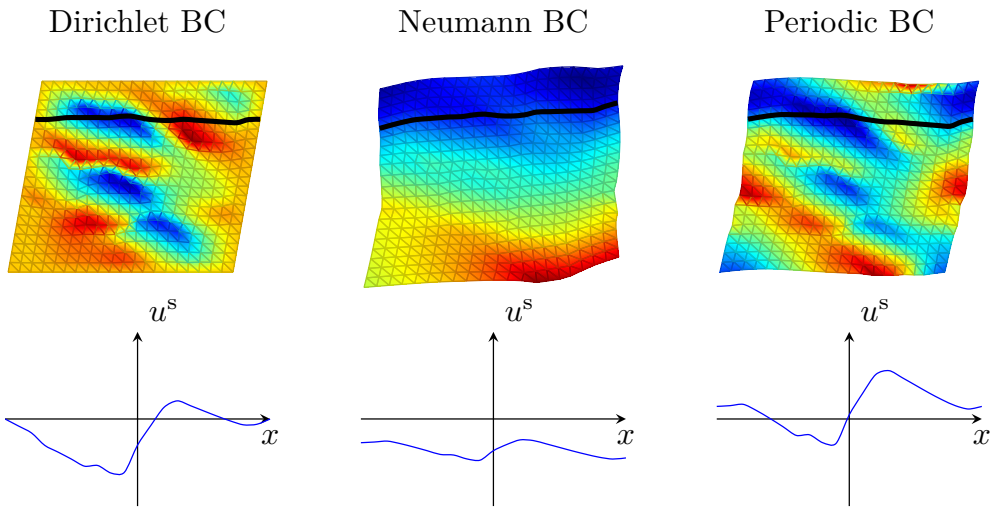

Figure 9: Different types of boundary conditions used on the RVE with the corresponding subscale fluctuation fields. From [1].

Those types differ in the way the macroscopic field is "prolonged" over the RVE, and the summary of differences can be found in [1]. The bottom line is, given the macroscopic field that are to be prescribed on the RVE and the type of boundary condition, a complete RVE boundary value problem can be solved. The details of the implementation are outside the scope of this document, thus the focus is put on how to use the boundary conditions in OOFEM input files.

For the 3D RVE, **only** (strongly) periodic boundary conditions were considered in the project. Those require a periodic mesh, and the use of boundary elements, as described in Section 3.2. No explicit boundary condition was implemented, as the macroscopic quantities are prescribed directly in the master node with conventional boundary conditions available in OOFEM.

In plane stress, **only** Dirichlet and Neumann type boundary conditions were implemented. Those can be used either in full FE^2 analyses (treated in Section 6), or separately for convenient setup and solution of RVE problems. The implemented boundary conditions are:

- `PrescribedDispSlipBCDirichletRC`, allowing for prescribing the macroscopic strain (optionally also macroscopic slip and slip gradient) on a reinforced concrete RVE using Dirichlet type boundary conditions
- `PrescribedDispSlipBCNeumannRC`, allowing for prescribing the macroscopic strain (optionally also macroscopic slip and slip gradient) on a reinforced concrete RVE using Neumann type boundary conditions

5.1 PrescribedDispSlipBCDirichletRC

The general syntax of `PrescribedDispSlipBCDirichletRC` boundary condition is the following:

PrescribedDispSlipBCDirichletRC	(num)(in)	\
	loadTimeFunction (in)	\
	dofs (ia)	\
	[ccoord (ra)]	\
	dispGrad (rm)	\
	[slip (ra)]	\
	[slipGrad (rm)]	\
	set (in)	\
	conboundset (in)	\
	[reinfxbound (in)]	\
	[reinfybound (in)]	

The PrescribedDispSlipBCDirichletRC keyword is followed by the integer number (number of the BC). Next, the loadTimeFunction corresponding to the BC is specified, usually a ConstantFunction. Subsequently, the degrees of freedom integer array is specified after the keyword dofs. As the RVE problem is a plane stress problem, we consider the translations (dofs 1 and 2) and, if the reinforcement is modelled with beam elements, an optional rotation in the xy plane (dof 6). The optional ccoord field specifies the coordinates of the centre of the RVE, and if not specified, it is set to the origin $(0, 0)$.

The macroscopic fields are specified by keywords dispGrad, slip, and slipGradient, as described in Section 4. Note that the dispGrad field is required, but the slip and slipGrad fields are optional. This makes it possible to use the same boundary condition for different types of macroscopic problems - with and without the macroscopic reinforcement slip field.

The conboundset is the elementboundaries set which makes up the boundary of the RVE (light red in Figure 10). This is used internally to compute the volume of the RVE when performing homogenization. The last two sets - reinfxbound and reinfybound (green and lightblue in Figure 10, respectively) - are required only if slip or slipGradient fields are present. These sets contain the boundary **nodes** of the reinforcement elements in x- and y-directions, respectively. Currently, only orthogonal reinforcement layout aligned with x and y axes is supported. The different types of the set are summarized in Figure 10.

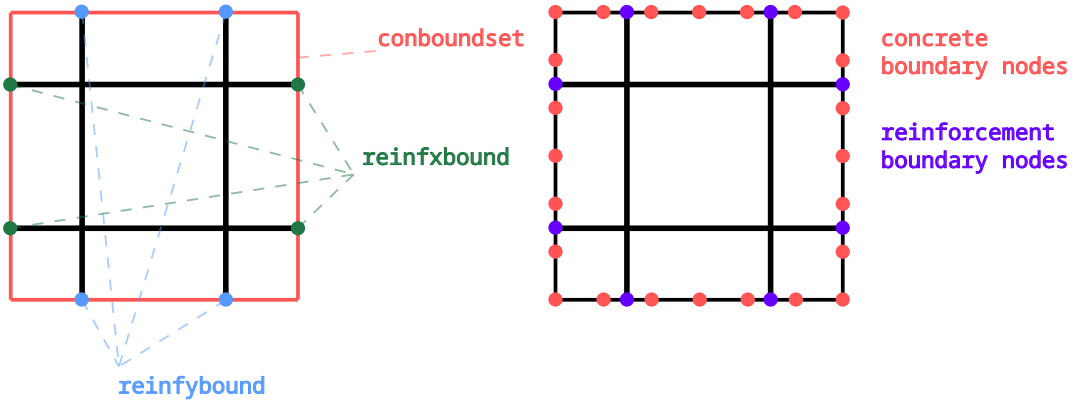

Figure 10: Summary of sets used in PrescribedDispSlipBCDirichletRC.

The set field specifies the set onto which the Dirichlet boundary condition is to be applied. This can be either a set of boundary nodes, or elementboundaries, depending on intended use. It is noteworthy, that this BC can be applied either on the concrete solid, reinforcement bars or on both at the same time. If the Dirichlet BC is only supposed to be prescribed only the concrete solid (not reinforcement), the set specified after the set keyword can be the same as the conboundset, which is a set of elementboundaries (light red in Figure 10). Alternatively, a set of all concrete nodes on the boundary of the RVE can be used (corresponds to concrete boundary nodes in Figure 10). This corresponds to Dirichlet-Neumann case described in Section 5.4.

It is also possible to apply this BC only on the reinforcement bars. In this case, the set of reinforcement boundary nodes (purple in Figure 10) should be specified after the set keyword. This would correspond to Neumann-Dirichlet case from Section 5.4, provided that an appropriate Neumann BC is used on the concrete boundary

Lastly, this BC can be applied on both concrete and reinforcement at the same time, corresponding to the Dirichlet-Dirichlet case from Section 5.4. To do this, the set specified after the `set` keyword should contain all the nodes at concrete and reinforcement boundary, i.e., this set should be a union of concrete boundary nodes and reinforcement boundary nodes from Figure 10.

5.2 PrescribedDispSlipBCNeumannRC

The general syntax of `PrescribedDispSlipBCNeumannRC` boundary condition is the following:

PrescribedDispSlipBCNeumannRC	(num)(in) \
	loadTimeFunction (in) \
	dofs (ia) \
	[ccoord (ra)] \
	dispGrad (rm) \
	[slip (ra)] \
	[slipGrad (rm)] \
	set (in) \
	[concretevolset (in)] \
	[rebarsets (ia)]

The `PrescribedDispSlipBCNeumannRC` keyword is followed by the integer number (number of the BC). Next, the `loadTimeFunction` corresponding to the BC is specified, usually a `ConstantFunction`. Subsequently, the degrees of freedom integer array is specified after the keyword `dofs`. As the RVE problem is a plane stress problem, we consider only the translations (dofs 1 and 2). This is due to the fact, that this BC prescribes the macroscopic strain only on the concrete volume, and not on the reinforcement. However, the optional macroscopic slip and slip gradients can be prescribed on the reinforcement bars. The optional `ccoord` field specifies the coordinates of the centre of the RVE, and if not specified, it is set to the origin $(0, 0)$.

The macroscopic fields are specified by keywords `dispGrad`, `slip`, and `slipGradient`, as described in Section 4. Note that the `dispGrad` field is required, but the `slip` and `slipGrad` fields are optional. This makes it possible to use the same boundary condition for different types of macroscopic problems - with and without the macroscopic reinforcement slip field.

The `set` field specifies the `elementboundaries` set of concrete boundaries (blue in Figure 11). The last two sets - `concretevolset` and `rebarsets` - are required only if `slip` or `slipGradient` fields are present. The `concretevolset` set contains the all concrete **elements** in the volume of the RVE (light red in Figure 11). The `rebarsets` field contains an integer array of reinforcement element sets, i.e., one set for all reinforcement bars going in the same direction (black in Figure 11). For the RVE presented in Figure 11, all elements of the reinforcement bars in x-direction should be grouped in one set (light orange in Figure 11). Similarly, all elements of the reinforcement bars in y-direction should be grouped in another set (light violet in Figure 11). Those sets should then be given in the `rebarsets` field. Currently, only orthogonal reinforcement layout aligned with x and y axes is supported.

The different types of the set are summarized in Figure 11.

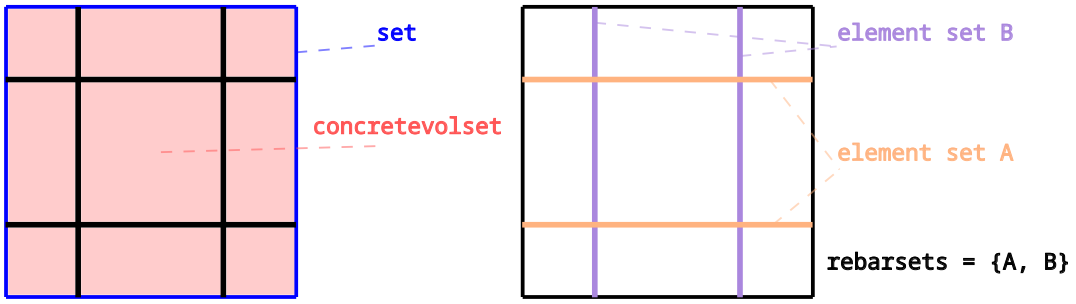

Figure 11: Summary of sets used in `PrescribedDispSlipBCNeumannRC`.

5.2.1 Pertinent test cases

Following test cases can be studied for more insight:

- `rveDisp/2D/prescribeddispbcneumann01` - presribing a displacement gradient on a minimal solid RVE (`planestress2d` elements)
- `rveDisp/2D/prescribeddispbcneumann02` - presribing a displacement gradient on a minimal solid RVE (`trplanestress2d` elements)
- `rveDisp/2D/prescribeddispbcneumann03` - presribing a displacement gradient on a minimal solid RVE (`qplanestress2d` elements)
- `rveDisp/2D/prescribeddispbcneumann04` - presribing a displacement gradient on a minimal solid RVE (`qtrplstr` elements)

5.3 TransverseReinforcementConstraint

The transverse constraint between the concrete and reinforcement (specifying that there is no slip in the transverse direction between the materials) can be satisfied either strongly or weakly. Strong constraint involved tying concrete and steel nodes and requiring that the displacements in the transverse direction is equal for both. This approach has been used in the project, as it was intuitive and did not require any additional implementation.

Another option is to satisfy the constraint *weakly*. To do that, a Lagrange multiplier, which corresponds to a distributed load, is introduced on the bar. To use that constraint, a boundary condition class `TransverseReinforcementConstraint` has been implemented with the following use syntax:

TransverseReinforcementConstraint	(num)(in) \
	loadTimeFunction (in) \
	steelElSet (in) \
	conElBoundSet (in)

The `TransverseReinforcementConstraint` boundary condition should be specified once for every single reinforcement bar. The `TransverseReinforcementConstraint` keyword is followed by the integer number (number of the BC). Next, the `loadTimeFunction` corresponding to the BC is specified, usually a `ConstantFunction`. `steelElSet` is a set of finite elements in the given reinforcement bar. `conElBoundSet` is a `elementboundaries` set of the boundaries of the plane stress concrete elements which are along the reinforcement bar, see Figure 12. In the figure, a distance between the concrete and reinforcement has been added for clarity. Note that in the model, those nodes can be either coincident or very close to each other.

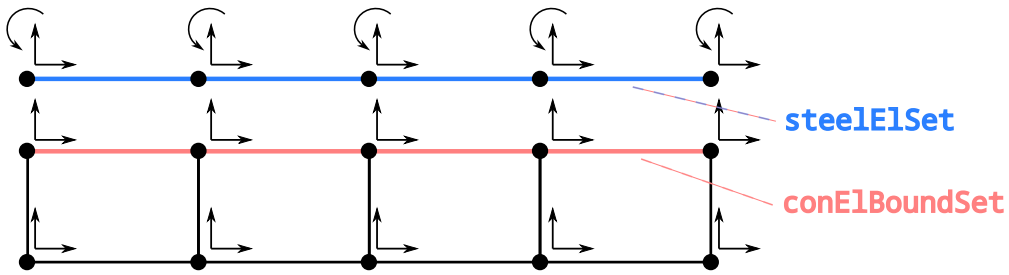

Figure 12: Summary of sets used in `TransverseReinforcementConstraint`.

The following excerpt from the `transversereinfconstraint` test illustrates the use of this boundary condition:

```
# Elements of single reinforcement bar (vertical)
Set 9 elementranges { (101 110) }
# Elements of single reinforcement bar (horizontal)
Set 10 elementranges { (111 120) }
#Concrete element boundary set along horizontal bar
Set 14 elementboundaries 20 55 3 60 3 65 3 70 3 75 3 80 3 85 3 90 3 95 3 100 3
#Concrete element boundary set along vertical bar
Set 15 elementboundaries 20 76 4 77 4 78 4 79 4 80 4 5 3 10 3 15 3 20 3 25 3
# transverse constraints - one per reinforcement bar
TransverseReinforcementConstraint 2 loadTimeFunction 1 steelElSet 10
                                    conElBoundSet 14
TransverseReinforcementConstraint 3 loadTimeFunction 1 steelElSet 9
                                    conElBoundSet 15
```

5.3.1 Pertinent test cases

The following test case can be studied for more insight:

- `rveDispSlip/2D/transversereinfconstraint` - weak transverse reinforcement constraint on a reinforced concrete RVE

5.4 Application examples

5.4.1 Macroscopic strain

As described in [5], different types of boundary conditions can be chosen to be applied on the concrete and reinforcement parts of the RVE. This gives four different possibilities, as shown in Figure 13.

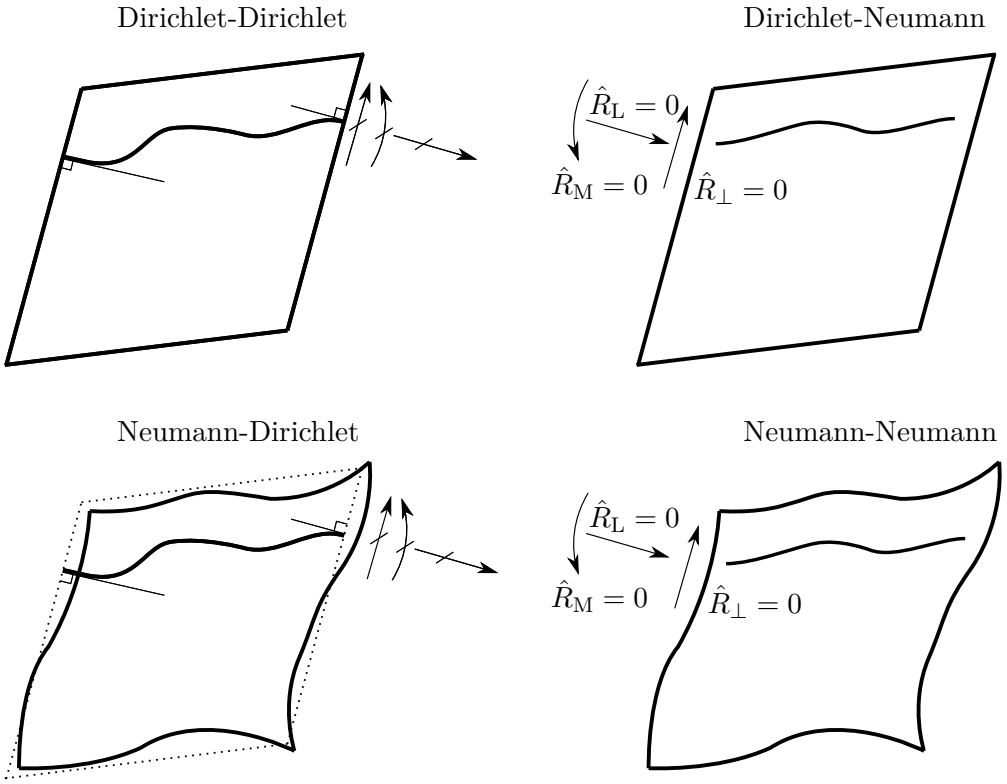

Figure 13: Combinations of boundary conditions used in a reinforced concrete RVE. From [5].

To summarize, the use of the pertinent boundary conditions has been demonstrated in Listing 7, which contains excerpts from test cases `rveDisp01-rveDisp04`. Note that the set numbers vary between the

examples, and the input files should be studied for more insight.

```
# Dirichlet-Dirichlet
PrescribedDispSlipBCDirichletRC 1 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-9 0.4e-9; 0.4e-9 -0.5e-9}
    set 5 conboundset 6

# Dirichlet-Neumann
PrescribedDispSlipBCDirichletRC 1 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-9 0.4e-9; 0.4e-9 -0.5e-9}
    set 6 conboundset 6

# Neumann-Dirichlet
PrescribedDispSlipBCNeumannRC 1 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-9 0.4e-9; 0.4e-9 -0.5e-9}
    set 5
PrescribedDispSlipBCDirichletRC 2 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-9 0.4e-9; 0.4e-9 -0.5e-9}
    set 6 conboundset 5

# Neumann-Neumann
PrescribedDispSlipBCNeumannRC 1 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-9 0.4e-9; 0.4e-9 -0.5e-9} set 5
```

Listing 7: Use of boundary condition for prescribing a macroscopic strain tensor.

The pertinent test cases are presented in the Table 4 below. The columns specify which type of boundary condition was used on the concrete and reinforcement part of the RVE.

Table 4: Test cases on a reinforced concrete RVE, prescribing a macroscopic strain tensor.

Test case	Concrete BC	Reinforcement BC
`rveDisp/2D/rveDisp01`	Dirichlet	Dirichlet
`rveDisp/2D/rveDisp02`	Dirichlet	Neumann
`rveDisp/2D/rveDisp03`	Neumann	Dirichlet
`rveDisp/2D/rveDisp04`	Neumann	Neumann

5.4.2 Macroscopic strain, slip and slip gradient

As previously alluded to, the implemented boundary conditions can also prescribe macroscopic slip and slip gradient. Similar as before, each macroscopic quantity can be prescribed either with Dirichlet, Neumann boundary condition, or not prescribed at all. To summarize, the use of the pertinent boundary conditions has been demonstrated in Listing 8, which contains excerpts from test cases `rveDispSlip07-rveDispSlip10`. Note that the set numbers vary between the examples, and the input files should be studied for more insight.

```
# Dirichlet-Dirichlet-Dirichlet
PrescribedDispSlipBCDirichletRC 1 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
    slip 2 1e-5 -2e-5 slipGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
    set 5 conboundset 6 reinfxbound 8 reinfybound 7

# Neumann-Neumann-Neumann
PrescribedDispSlipBCNeumannRC 1 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
    slip 2 1e-5 -2e-5 slipGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
    set 6 concretevolset 1 rebarsets 2 9 10

# Dirichlet-Neumann-Neumann
PrescribedDispSlipBCDirichletRC 1 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
    set 6 conboundset 6
PrescribedDispSlipBCNeumannRC 2 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 slip 2 1e-5 -2e-5
    slipGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6} set 6 concretevolset 1
    rebarsets 2 9 10

# Neumann-Dirichlet-Dirichlet
PrescribedDispSlipBCNeumannRC 1 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6} set 6
PrescribedDispSlipBCDirichletRC 2 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 slip 2 1e-5 -2e-5
    slipGrad 2 2 {0.3e-6 0.4e-6; 0.4e-6 -0.5e-6}
    set 15 conboundset 6 reinfxbound 8 reinfybound 7
```

Listing 8: Use of boundary condition for prescribing macroscopic strain, slip and slip gradient.

The pertinent test cases are presented in the Table 5 below. The columns specify which type of boundary condition was used to prescribe a specific macroscopic component.

Table 5: Test cases on a reinforced concrete RVE, prescribing macroscopic strain, slip and slip gradient.

Test case	$\bar{\boldsymbol{\varepsilon}}$	$\bar{\boldsymbol{s}}$	$\bar{\boldsymbol{g}}$
`rveDispSlip/2D/rveDispSlip01`	-	Dirichlet	-
`rveDispSlip/2D/rveDispSlip02`	-	-	Dirichlet
`rveDispSlip/2D/rveDispSlip03`	-	Neumann	-
`rveDispSlip/2D/rveDispSlip04`	-	-	Neumann
`rveDispSlip/2D/rveDispSlip05`	-	Dirichlet	Dirichlet
`rveDispSlip/2D/rveDispSlip06`	-	Neumann	Neumann
`rveDispSlip/2D/rveDispSlip07`	Dirichlet	Dirichlet	Dirichlet
`rveDispSlip/2D/rveDispSlip08`	Neumann	Neumann	Neumann
`rveDispSlip/2D/rveDispSlip09`	Dirichlet	Neumann	Neumann
`rveDispSlip/2D/rveDispSlip10`	Neumann	Dirichlet	Dirichlet

6 FE^2 analysis

In an FE^2 analysis, an RVE problem is solved for each integration point in the macroscopic model, see Figure 5. The macroscopic fields are prescribed on the RVE via suitable boundary conditions (see Section 5), and the effective work conjugates (e.g., stresses) are homogenized and brought back to the macroscale. The analysis then continues. As such, the multiscale method can be viewed as a complex material model, which for given input (e.g., strain tensor) provides the output (e.g., stress tensor). However, apart from the macroscopic response, we also have access to subscale results, and can study in detail the progression of damage in individual RVEs.

The RVE input file must fulfil certain requirements:

- it must have a `PrescribedDispSlip*` boundary condition,
- the `PrescribedDispSlip*` must be the first boundary condition,
- the fields `dispGrad`, `slip`, and `slipGrad` must be specified. However, as the macroscopic components will be propagated directly from the macroscopic problem (and are not known in advance), the values for those input fields should be 0, e.g.,

```
PrescribedDispSlipBCDirichletRC 1 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0 0; 0 0} slip 2 0 0
    slipGrad 2 2 {0 0; 0 0} set 5 conboundset 6 reinfxbound 8 reinfybound 7
```

6.1 StructSlipFE2Material

In OOFEM, the FE^2 analysis is possible with the `StructSlipFE2Material` material model assigned to macroscopic finite element. This material requires the RVE model, given in the form of OOFEM input file. Each `StructSlipFE2Material` material requires one RVE model, but several different RVEs can be used throughout the macroscopic model (as in [2]). In that case, different `StructSlipFE2Material` models should be assigned to different finite elements in the macroscopic problem. The syntax of the model is as follows:

```
StructSlipFE2Material (num)(in) \
filename (s) \
[export_all_gps] \
[output_selected_el_gps (ia)] \
[use_num_tangent] \
[use_ext_stiffness] \
[dSdE (rm)] \
[dBSdE (rm)] \
[dRSdE (rm)] \
[dSdS (rm)] \
[dBSdS (rm)] \
[dRSdS (rm)] \
[dSdG (rm)] \
[dBSdG (rm)] \
[dRSdG (rm)]
```

The `StructSlipFE2Material` keyword is followed by the integer number (number of the material). Next, and most important the name of the input file for the RVE model is specified after the `filename` keyword. The rest of the keywords is optional. The minimal syntax is illustrated below:

```
StructSlipFE2Material 1 d 1.0 filename fe2disprve1x1dd.in.rve
```

In the above, the RVE model is stored in the file `fe2disprve1x1dd.in.rve`, located in the same directory as the macroscopic model. If output modules are specified in the RVE file (e.g., `vtkxml` or `matlab` output modules), only the results for one RVE (last one when iterating over model Gauss points) will be created by default (with the output file suffex with `_el1_gp1`). This behaviour can be changed by using the `export_all_gps` flag:

```
StructSlipFE2Material 1 d 1.0 filename fe2disprve1x1dd.in.rve export_all_gps
```

This way, the results from all Gauss points will be stored in separate files. Note that this will produce a large amount of files, make sure that there is enough space for them. An alternative is to output results only from selected Gauss points. This can be done by specyfing the integer array of finite element and Gauss point numbers after the keyword `output_selected_el_gps`, e.g., the following syntax

```
StructSlipFE2Material 1 d 1.0 filename fe2disprve1x1dd.in.rve
                      output_selected_el_gps 6 20 3 21 1 23 4
```

will output the results from Gauss point 3 in element 20, Gauss point 1 in element 21, and Gauss point 4 in element 23.

When solving nonlinear problems, Newton-Raphson iterations are usually used in OOFEM. Computing of tangent stiffness matrix is then handled by the material model directly. In case of the RVE problem, this should be implemented by the corresponding boundary conditions. By default, the material will use the tangent implemented by the boundary condition. However, as this is usually a complex task, the extact computation of the tangent is often omitted. In this case, we can either compute the tangent numerically, or supply a value directly (useful for e.g., modified Newton iterations with linear stiffness). To compute the tangent numerically, the `use_num_tangent` should be specified:

```
StructSlipFE2Material 1 d 1.0 filename fe2disprve1x1dd.in.rve use_num_tangent
```

Note that this will result in the model computing the tangent stiffness via numerical perturbation (i.e., solving a new RVE problem for each component of the tangent). This means solving many RVE problems for each Gauss point in each element. The overall computation time is therefore increased substantially.

Another option is for the user to provide the tangent stiffness directly (e.g., by performing the numerical perturbation on a single RVE off-line before the analysis). To do this, the `use_ext_stiffness` flag should be used, followed by the tangent stiffness matrices pertinent to the problem. After the keyword, the tangents pertinent to the macroscopic problem should be specified as matrices. Note that the plane stress problem requires only one tangent, whereas the addition of macroscopic reinforcement slip field increases the number of required tangents to 9. The required tangent keywords, their mathematical expression and their dimensions are summarized in Table 6. In plane stress, only the `dsde` tangent is necessary, see Listing 9 and 10.

Table 6: Tangent stifness matrices that can be supplied to `StructSlipFE2Material` material.

keyword	symbol	dim	keyword	symbol	dim	keyword	symbol	dim
dSdE	$\frac{\bar{\boldsymbol{\sigma}}}{\bar{\boldsymbol{\varepsilon}}}$	3×3	dSdS	$\frac{\bar{\boldsymbol{\sigma}}}{\bar{\boldsymbol{s}}}$	3×2	dSdG	$\frac{\bar{\boldsymbol{\sigma}}}{\bar{\boldsymbol{g}}}$	3×4
dBSdE	$\frac{\bar{\boldsymbol{\tau}}_{\boldsymbol{b}}}{\bar{\boldsymbol{\varepsilon}}}$	2×3	dBSdS	$\frac{\bar{\boldsymbol{\tau}}_{\boldsymbol{b}}}{\bar{\boldsymbol{s}}}$	2×2	dBSdG	$\frac{\bar{\boldsymbol{\tau}}_{\boldsymbol{b}}}{\bar{\boldsymbol{g}}}$	2×4
dRSdE	$\frac{\bar{\boldsymbol{\sigma}}_{\boldsymbol{s}}}{\bar{\boldsymbol{\varepsilon}}}$	4×3	dRSdS	$\frac{\bar{\boldsymbol{\sigma}}_{\boldsymbol{s}}}{\bar{\boldsymbol{s}}}$	4×2	dRSdG	$\frac{\bar{\boldsymbol{\sigma}}_{\boldsymbol{s}}}{\bar{\boldsymbol{g}}}$	4×4

```
# macroscopic plane stress problem
StructSlipFE2Material 1 d 1.0 filename fe2disprve1x1dd.in.rve use_ext_stiffness
                      dSdE 3 3 {36.2507e+09   6.9323e+09      2.9505e+02;
                                6.9323e+09    34.9136e+09     -2.3936e+03;
                                2.9505e+02    -2.3936e+03     13.8658e+09}
```

Listing 9: Syntax for providing tangent stiffness for macroscopic plane stress problem

```
# macroscopic plane stress with reinforcement slip field
StructSlipFE2Material 1 d 1.0 filename fe2dispsliprve1x1dd.in.rve use_ext_stiffness
            dSdE 3 3 { 3.6250e+10  6.9323e+09  2.9505e+02;
                    6.9323e+09  3.4913e+10  -2.3936e+03;
                    2.9505e+02  -2.3936e+03  1.3865e+10}
            dBSdE 2 3 { 7.5564e+05  1.4917e+05  -4.6064e+07;
                    -2.7273e+05  -1.3649e+06  2.2506e+06 }
            dRSdE 4 3 { 1.5706e+09  3.4606e+05  -9.6819e+02;
                    -6.8920e+04  2.5083e+08  -3.1767e+02;
                        0.0000e+00  0.0000e+00  0.0000e+00;
                        0.0000e+00  0.0000e+00  0.0000e+00 }
            dSdS 3 2 { 7.5564e+05  -2.7273e+05;
                    1.4917e+05  -1.3649e+06;
                    -4.6064e+07  2.2506e+06 }
            dBSdS 2 2 {  7.9177e+10  4.7826e+03;
                        4.7826e+03  2.6249e+10  }
            dRSdS 4 2 { -5.4548e+03  1.2422e+05;
                        -2.2920e+04  4.0332e+03;
                        0.0000e+00  0.0000e+00;
                        0.0000e+00  0.0000e+00 }
            dSdG 3 4 {  1.5706e+09  -6.8920e+04  0.0000e+00  0.0000e+00;
                        3.4606e+05  2.5083e+08  0.0000e+00  0.0000e+00;
                    -9.6819e+02  -3.1767e+02  0.0000e+00  0.0000e+00  }
            dBSdG 2 4 { -5.4548e+03  -2.2920e+04  0.0000e+00  0.0000e+00;
                        1.2422e+05  4.0332e+03  0.0000e+00  0.0000e+00 }
            dRSdG 4 4 { 1.8773e+09  9.6766e+04  0.0000e+00  0.0000e+00;
                        9.6766e+04  3.6845e+08  0.0000e+00  0.0000e+00;
                        0.0000e+00  0.0000e+00  0.0000e+00  0.0000e+00;
                        0.0000e+00  0.0000e+00  0.0000e+00  0.0000e+00 }
```

Listing 10: Syntax for providing tangent stiffness for macroscopic plane stress with effective reinforcement slip field

6.2 PrescribedDispSlipMultiple

As already mentioned in the beginning of Section 6, the `PrescribedDispSlip*` boundary condition should be specified as the first one in the RVE model. Implicitly, the displacement (and potentially slip) gradient will be passed to this boundary condition in the course of the multiscale analysis. However, if more than one boundary conditions need to be specified on parts of the RVE (e.g., in the Neummann-Dirichlet case), the second boundary condition will not be automatically used by the code. For this reason, a `PrescribedDispSlipMultiple` wrapper, allowing for use of multiple boundary conditions in the RVE problem, was implemented. The syntax of the wrapper is the following:

PrescribedDispSlipMultiple	(num)(in)	\
	bcs (ia)	\
	loadTimeFunction (in)	

The boundary conditions which are to be used for the RVE are specified in the integer array after the `bcs` keyword. The `PrescribedDispSlipMultiple` wrapper acts as the one required `PrescribedDispSlip*` boundary conditionin the RVE. The pertinent gradients (and fields) are then autmatically passed to the boundary conditions specified by the wrapper. For instance, a set of two boundary conditions (numbered 2 and 3) can be used in the FE^2 analysis using the following syntax:

```
PrescribedDispSlipMultiple 1 bcs 2 2 3 loadTimeFunction 1
PrescribedDispSlipBCNeumannRC 2 loadTimeFunction 1 dofs 2 1 2
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0 0; 0 0} set 5
PrescribedDispSlipBCDirichletRC 3 loadTimeFunction 1 dofs 3 1 2 6
    ccoord 3 0.0 0.0 0.0 dispGrad 2 2 {0 0; 0 0} set 6 conboundset 5
```

The following test case can be studied for more insight:

- `fe2Disp/planeStress/deepbeam03` - FE^2 analysis of a deep beam using Neumann-Dirichlet boundary conditions, from [5].

6.3 UserDefDirichletBC

Sometimes, the full FE^2 analysis is just too much, as only results from a single RVE are necessary, e.g., to study crack growth. However, it's usually not possible to predict and describe the strain evolution with a single `loadTimeFunction`. Furthermore, the evolution of individual tensor components adds complexity to this task. To overcome this limitation, a way of prescribing a given strain (or potentially also slip) history on a single RVE is needed. To do this, a mock FE^2 analysis can be used. In the mock analysis, only one macroscopic element can be defined.

The challenge now is to prescribe deformation history on the macroscopic nodes, such that the strains ($\boldsymbol{\varepsilon}$) computed at the locations of the chosen Gauss point are equal to the given strain history. In case of macroscopic slip field, we also need to prescribe a slip history, so that the given slip ($\boldsymbol{s}$) and slip gradient ($\boldsymbol{g}$) will be propagated into the Gauss point. This can be done in many ways, and the parent macroscopic elements chosen in the project are shown in Figure 14.

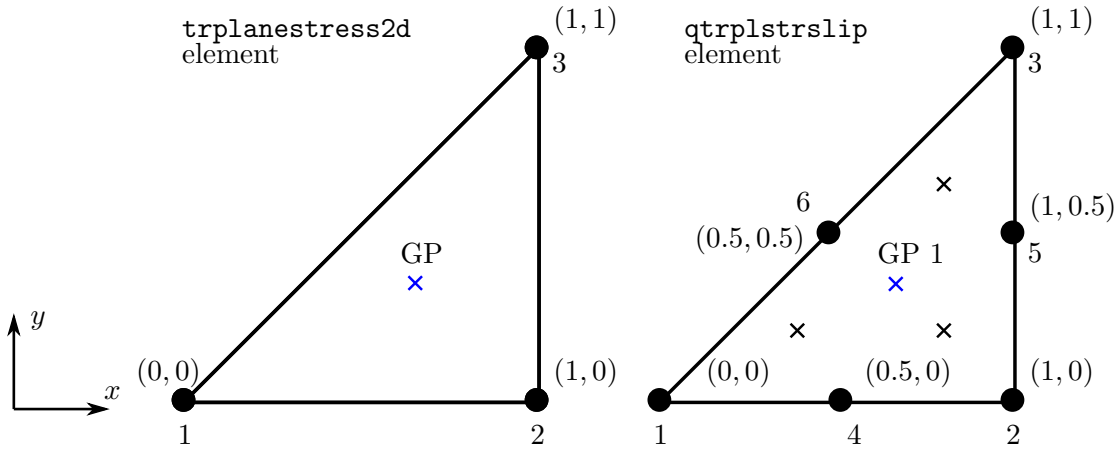

Figure 14: Parent macroscopic elements for mock FE^2 analyses.

By prescribing specific displacement (and slip) values in specific nodes, the given strain (as well as slip and slip gradient) values will be propagated into the Gauss point (marked in blue). The choices made in the project are summarized in Table 7 for the `trplanestress2d` element, and in Table 8 for the `qtrplstrslip` element.

Table 7: Prescribed displacement values in the nodes of `trplanestress2d` element.

Node	u	v
1	0	0
2	$\bar{\varepsilon}_{xx}$	0
3	$\bar{\varepsilon}_{xx} + \bar{\gamma}_{xy}$	$\bar{\varepsilon}_{yy}$

Table 8: Prescribed displacement and slip values in the nodes of `qtrplstrslip` element.

Node	u	v	s_x	s_y
1	0	0	$-2\bar{g}_{xx} - \bar{g}_{xy} - 3\bar{s}_x$	$-2\bar{g}_{yx} - \bar{g}_{yy} - 3\bar{s}_y$
2	$3\bar{\varepsilon}_{xx}$	$3\bar{\gamma}_{xy} + 3\bar{\varepsilon}_{xx}$	$\bar{g}_{xx} - \bar{g}_{xy} - 3\bar{s}_x$	$\bar{g}_{yx} - \bar{g}_{yy} - 3\bar{s}_y$
3	0	$3\bar{\gamma}_{xy} + 3\bar{\varepsilon}_{xx} + 3\bar{\varepsilon}_{yy}$	$\bar{g}_{xx} + 2\bar{g}_{xy} - 3\bar{s}_x$	$\bar{g}_{yx} + 2\bar{g}_{yy} - 3\bar{s}_y$
4	0	0	0	0
5	0	0	0	0
6	0	0	0	0

Prescribing the displacement and slip values given above ensures that the strain, slip and slip gradients in the marked Gauss point are:

$$\begin{aligned}\bar{\boldsymbol{\varepsilon}} &= [\bar{\varepsilon}_{xx}, \quad \bar{\varepsilon}_{yy}, \quad \bar{\gamma}_{xy}] \\ \bar{\boldsymbol{s}} &= [\bar{s}_x, \quad \bar{s}_y] \\ \bar{\boldsymbol{g}} &= [\bar{g}_{xx}, \quad \bar{g}_{yy}, \quad \bar{g}_{xy}, \quad \bar{g}_{yx}]\end{aligned}$$

To prescribe a specific time-history value in a node, the `UserDefDirichletBC` boundary condition can be used. This boundary condition allows the user to write a Python function named `giveUserDefBC`, which returns the value for a given node at a given time and for a specific degree of freedom. The use syntax of `UserDefDirichletBC` is:

UserDefDirichletBC	(num)(in)	\
	`loadTimeFunction` (in)	\
	`filename` (s)	

The most important input field is the `filename`, which specifies the Python file with the function `giveUserDefBC`. In the following example, three different `UserDefDirichletBC` boundary conditions are used in three different nodes. The corresponding functions are located in the files `bcnode1.py`, `bcnode2.py`, and `bcnode3.py`.

```
# node definition
node 1 coords 3 0 0 0 dofidmask 4 1 2 51 52 bc 4 1 1 1 1
node 2 coords 3 1 0 0 dofidmask 4 1 2 51 52 bc 4 2 2 2 2
node 3 coords 3 1 1 0 dofidmask 4 1 2 51 52 bc 4 3 3 3 3

# bc definition
UserDefDirichletBC 1 loadTimeFunction 1 filename bcnode1
UserDefDirichletBC 2 loadTimeFunction 1 filename bcnode2
UserDefDirichletBC 3 loadTimeFunction 1 filename bcnode3
```

The `giveUserDefBC` function takes the node coordinates, the degree of freedom and the time step as arguments and returns a single value which is to be prescribed in the node. The function prototype is given below:

```
1 def giveUserDefBC(iCoords, iDofNum, iTime):
2     prescribedValue = ...
3     return prescribedValue
```

For example implementations, the pertinent Python files of the test cases `fakefe2*` can be consulted. The implemented function reads input from a text file. This is of course, only one possible solution, the freedom is left to the user. It is noteworthy, that for the `UserDefDirichletBC` boundary condition to work, OOFEM must be compiled with Python support, i.e., the flag `USE_PYTHON_EXTENSION` must be on during CMake configuration. Following test cases can be studied for more insight:

- `fe2Disp/planeStress/fakefe2disp` - prescribed strain history on an RVE with Dirichlet-Dirichlet boundary conditions. Presented in [5] and [3].

- `fe2DispSlip/planeStress/fakefe2dispslip01` - prescribed strain, slip and slip gradient history on an RVE with Dirichlet-Dirichlet-Dirichlet boundary conditions. Presented in [3] and [4].
- `fe2DispSlip/planeStress/fakefe2dispslip02` - prescribed strain, slip and slip gradient history on an RVE with Dirichlet-Neumann-Neumann boundary conditions. Presented in [4].

6.4 Application examples

6.4.1 Macroscopic strain

Following test cases can be studied for more insight:

- `fe2Disp/planeStress/deepbeam01` - FE^2 analysis of a reinforced concrete deep beam with Dirichlet-Dirichlet subscale boundary conditions. Presented in [5].
- `fe2Disp/planeStress/deepbeam02` - FE^2 analysis of a reinforced concrete deep beam with Dirichlet-Neumann subscale boundary conditions. Presented in [5].
- `fe2Disp/planeStress/deepbeam03` - FE^2 analysis of a reinforced concrete deep beam with Neumann-Dirichlet subscale boundary conditions. Presented in [5].
- `fe2Disp/planeStress/deepbeam04` - FE^2 analysis of a reinforced concrete deep beam with Neumann-Neumann subscale boundary conditions. Presented in [5].
- `fe2Disp/planeStress/aci-i` - FE^2 analysis of the ACI-I beam with Dirichlet-Dirichlet subscale boundary conditions and multiple RVEs over the macroscopic domains. Presented in [2].
- `fe2Disp/planeStress/stm-m` - FE^2 analysis of the STM-M beam with Dirichlet-Dirichlet subscale boundary conditions and multiple RVEs over the macroscopic domains. Presented in [2].
- `fe2Disp/planeStress/wt-4` - FE^2 analysis of the WT4 beam with Dirichlet-Dirichlet subscale boundary conditions and multiple RVEs over the macroscopic domains. Presented in [2].

6.4.2 Macroscopic strain, slip and slip gradient

Following test cases can be studied for more insight:

- `fe2DispSlip/planeStress/deepbeam01` - FE^2 analysis of a reinforced concrete deep beam with Dirichlet-Dirichlet-Dirichlet subscale boundary conditions. Presented in [3].
- `fe2Disp/planeStress/deepbeam02` - FE^2 analysis of a reinforced concrete deep beam with Dirichlet-Neumann-Neumann subscale boundary conditions. Presented in [4].

References

[1] Adam Sciegaj. *Multiscale Modelling of Reinforced Concrete Structures*. PhD Thesis. Chalmers University of Technology, 2020. ISBN: 978-91-7905-238-6. URL: https://research.chalmers.se/publication/515377/file/515377_Fulltext.pdf.

[2] Adam Sciegaj and Alexandre Mathern. "Two-scale modelling of reinforced concrete deep beams: Choice of unit cell and comparison with single-scale modelling". In: *Advances in Engineering Materials, Structures and Systems: Innovations, Mechanics and Applications*. Ed. by A. Zingoni. CRC Press, 2019, pp. 251–256. DOI: 10.1201/9780429426506.

[3] Adam Sciegaj et al. "A multiscale model for reinforced concrete with macroscopic variation of reinforcement slip". In: *Computational Mechanics* 63.2 (2019), pp. 139–158. DOI: 10.1007/s00466-018-1588-3.

[4] Adam Sciegaj et al. "On a volume averaged measure of macroscopic reinforcement slip in two-scale modeling of reinforced concrete". In: *International Journal for Numerical Methods in Engineering* 121.8 (2020), pp. 1822–1846. DOI: 10.1002/nme.6288.

[5] Adam Sciegaj et al. "Two-scale finite element modelling of reinforced concrete structures: Effective response and subscale fracture development". In: *International Journal for Numerical Methods in Engineering* 114.10 (2018), pp. 1074–1102. DOI: 10.1002/nme.5776.

[6] Adam Sciegaj et al. "Upscaling of three-dimensional reinforced concrete representative volume elements to effective beam and plate models". In: *International Journal of Solids and Structures* 202 (2020), pp. 835–853. DOI: 10.1016/j.ijsolstr.2020.07.006.