TRAINING SETUP

MESHFREE-Team, February 2023

MESHFREE

Fraunhofer ITWM

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise I
  - Read-in of geometry including scaling
  - Geometry alias for no-slip wall
- Exercise II
  - Geometry aliases for inflow and outflow
  - Solver parameters and time control
- Exercise III
  - General saving parameters and save items
  - “integrated“ quantities
- Exercise IV
  - Equations and curves
  - Point cloud definition and alias variables

→ Result: USER_common_variables.dat (replacing default definitions in Ucv_DEFAULT.dat)

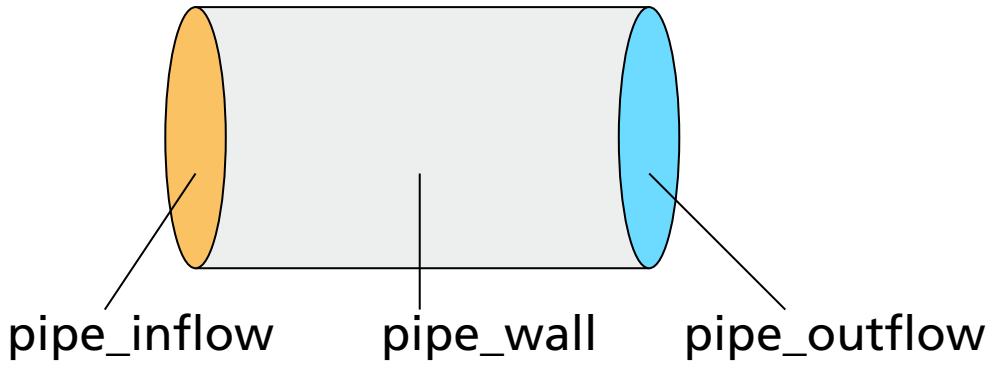
Exercise I

Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm
- Geometry parts of pipe labeled with unique names

Tasks:

1. Read-in the geometry
2. Scale the geometry to m (SI units in MESHFREE)



Exercise I – Task #1: Read-in the geometry

```
begin_boundary_elements{}
          include{/path/to/file/pipe.msh}
end_boundary_elements

CONTROL_StopAfterReadingGeometry = 1
```

Read-in of file

Stop simulation after read-in of file

MESHFREE

Fraunhofer ITWM

Exercise I – Task #2: Scale the geometry to m

```
begin_boundary_elements{}
          include{/path/to/file/pipe.msh} scale{0.001}
end_boundary_elements

CONTROL_StopAfterReadingGeometry = 1
```

Scaling (identical in all directions)

MESHFREE

Fraunhofer ITWM

Exercise I

Execute MESHFREE:

- Download the folder “TrainingSetup”.
- Add the lines of first Task #1 and later Task #2 in the empty file USER_common_variables.dat (with adapted path!).
- Run MESHFREE, e.g.

```
cd /path/to/input/files
export MESHFREE_LICENSE_FILE=/path/to/local/license/file.lcs
mpirun -n 2 /path/to/meshfree.x
```

Check the results:

- Start ParaView (see 10_ParaviewForMESHFREE.pdf).
- Open the results folder and then select the available BE-file.
- Check the scaling of the pipe by switching on “Axes Grid“.

Exercise I

```
begin_boundary_elements{}
          include{/path/to/file/pipe.msh} scale{0.001}
end_boundary_elements

#CONTROL_StopAfterReadingGeometry = 1
```

Comment after successful read-in and scaling

MESHFREE Fraunhofer ITWM

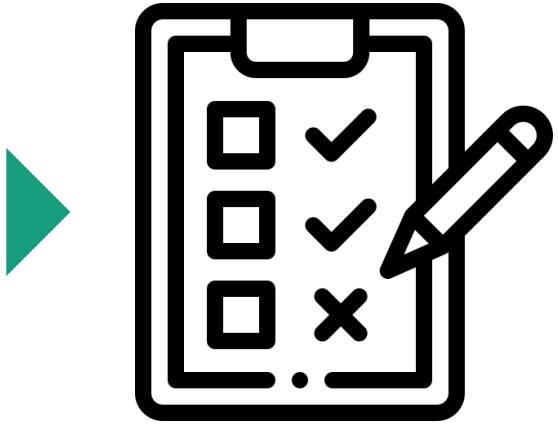
Exercise I – standard solution

Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm
- Geometry parts of pipe labeled with unique names

Tasks:

1. Read-in the geometry ✓
2. Scale the geometry to m (SI units in MESHFREE) ✓



MESHFREE Fraunhofer ITWM

Geometry – Additional options during read-in

Online documentation:

- GeometryManipulations
  - scale{}
  - rotate{}
  - offset{}
  - …
- GeometryRestrictions
- exportGeometry{}

→ Execution in given order, no commutativity!

Exercise I

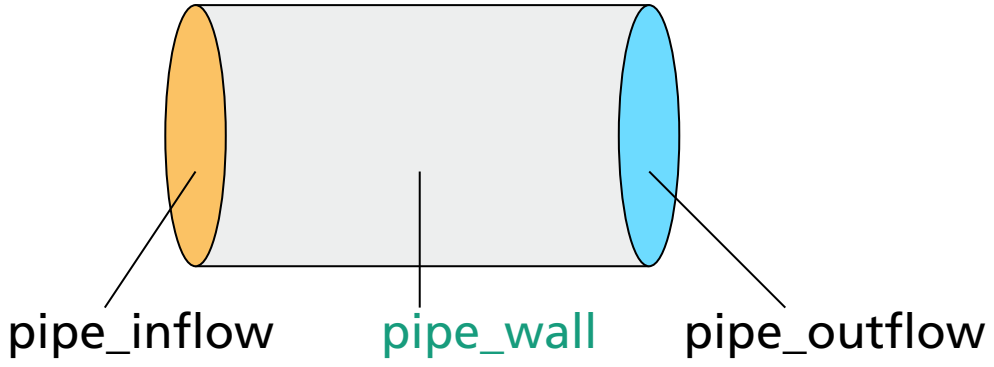
Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the scripting language.

Tasks:

3. Define the alias for the no-slip wall

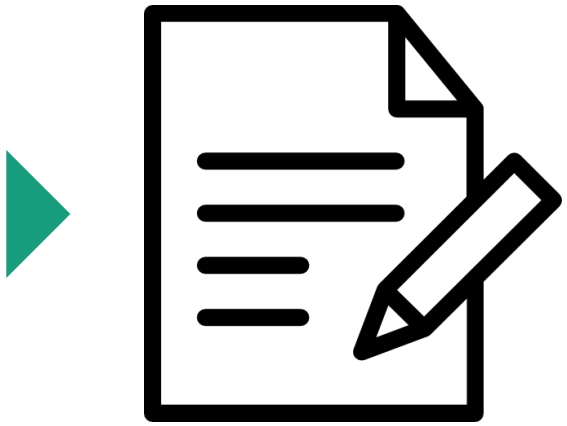


Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
   LHS = RHS
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element



Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = RHS
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element

MESHFREE

Fraunhofer ITWM

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase

MESHFREE Fraunhofer ITWM

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE$MV_vel$ … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement

User-defined movement:

```
MOVE($MV_vel$) = ( %MOVE_velocity%, 0.0, 0.0, 0.0 )
```

User definition = acronym	MESHFREE intrinsic definition = identifier

MESHFREE Fraunhofer ITWM

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement
- ACTIVE (initially and during simulation) – all the time

User-defined activation:

```
ACTIVE($ACT_all$) = ( %ACTIVE_init%, %ACTIVE_always% )
```

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement
- ACTIVE (initially and during simulation) – all the time
- MAT (seeding of points) – physical properties of water

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ … "
end_alias
```

User-defined material:

```
density($MAT_user$) = 1000.0              # density in kg/m³
cv($MAT_user$) = 1500.0                   # heat capacity in Nm/(kg*K)
lambda($MAT_user$) = 0.5                  # heat conductivity in W/(m*K)
eta($MAT_user$) = 1.0e-3                  # viscosity in Pa*s
mue($MAT_user$) = 0.0                     # shear modulus in Pa
sigma($MAT_user$) = 0.0                   # surface tension in N/m
gravity($MAT_user$) = (0.0, 0.0, -9.81) # gravity in m/s²
```

Rules:

- density, cv, lambda and eta must always be > 0.
- mue and sigma must be ≥ 0. If 0, the respective effects are neglected.
- Values are interpreted in SI units.

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement
- ACTIVE (initially and during simulation) – all the time
- MAT (seeding of points) – physical properties of water
- IDENT (organization of points) – follow physical behavior
  IDENT%IDENT_slip% → points move tangentially along the geometry but not in normal direction, no flow dynamic boundary condition!

MESHFREE Fraunhofer ITWM

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement
- ACTIVE (initially and during simulation) – all the time
- MAT (seeding of points) – physical properties of water
- IDENT (organization of points) – follow physical behavior
- TOUCH (wetting behavior of points) – contact at all times
  TOUCH%TOUCH_always% → boundary points always active

MESHFREE Fraunhofer ITWM

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

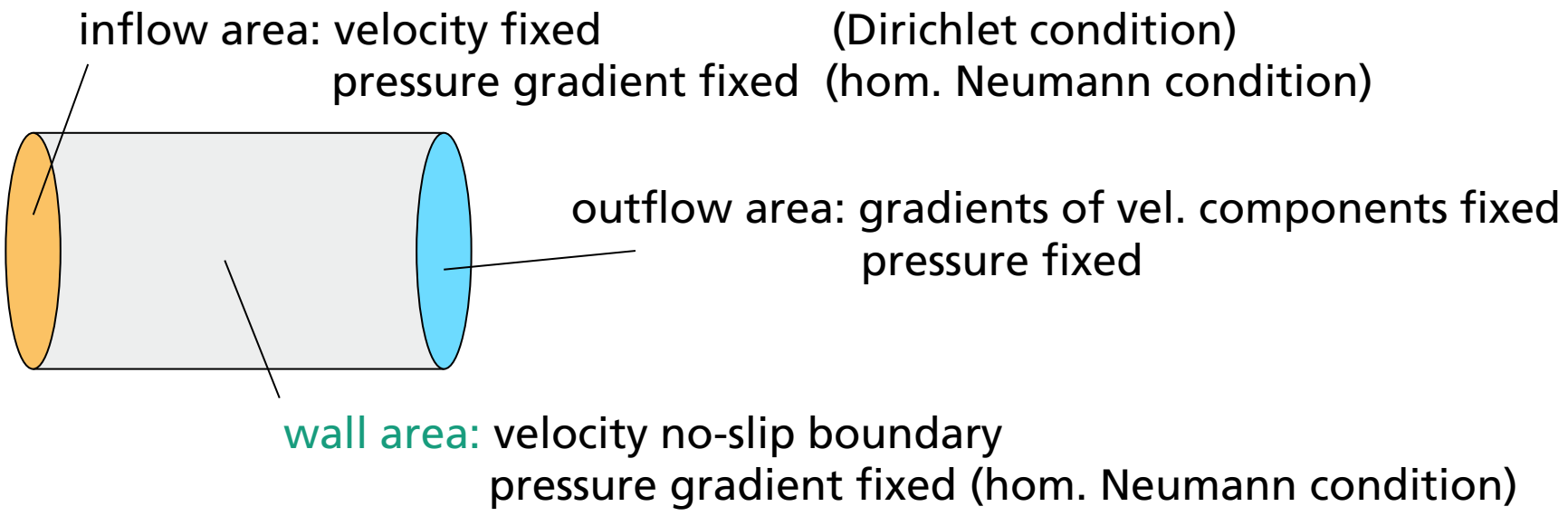
- CHAMBER (separate definitions for multiple phases) – one phase
- MOVE (time-dependent movement) – no movement
- ACTIVE (initially and during simulation) – all the time
- MAT (seeding of points) – physical properties of water
- IDENT (organization of points) – follow physical behavior
- TOUCH (wetting behavior of points) – contact at all times
- BC (velocity, hydrostatic pressure, dynamic pressure)

MESHFREE Fraunhofer ITWM

Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
end_alias
```

System to be modeled:



Exercise I – Task #3: Define the alias for the no-slip wall

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
end_alias
```

User-defined boundary conditions:

```
BC_v($BC_wall$) = ( %BND_wall_nosl% )    # no-slip velocity boundary condition
BC_p($BC_wall$) = ( %BND_wall% )         # classical wall boundary condition
                                         # for the hydrostatic pressure
BCON($BC_wall$,%ind_p_dyn%) = ( %BND_wall% )# classical wall boundary condition
                                            # for the dynamic pressure
```

The effect of the boundary condition identifiers depends on the variable they are used for. Even the number of arguments varies!

The split of the pressure into a hydrostatic and a dynamic part is described in DerivePoissonEquationForPressure.

Exercise I

Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the script language.

Tasks:

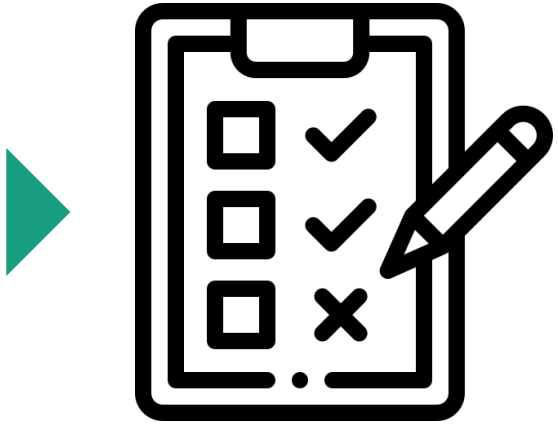
3. Define the alias for the no-slip wall

Exercise I

Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the script language.



Tasks:

3. Define the alias for the no-slip wall ✓

Exercise II

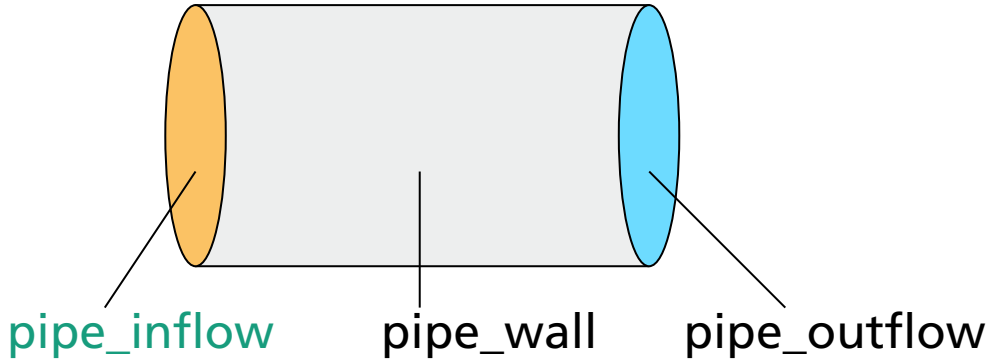
Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the script language.

Tasks:

3. Define the alias for the no-slip wall ✓
4. Define the alias for the inflow
5. Define the alias for the outflow



Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER, MOVE, ACTIVE, MAT

Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER, MOVE, ACTIVE, MAT
- IDENT – “closed“ inflow (no free surface)
  IDENT%IDENT_outflow% → outflow and inflow (adjacent to entirely filled interior domain) boundaries

Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER, MOVE, ACTIVE, MAT
- IDENT – “closed“ inflow (no free surface)
- TOUCH

Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
end_alias
```

LHS: Alias name/label of boundary element

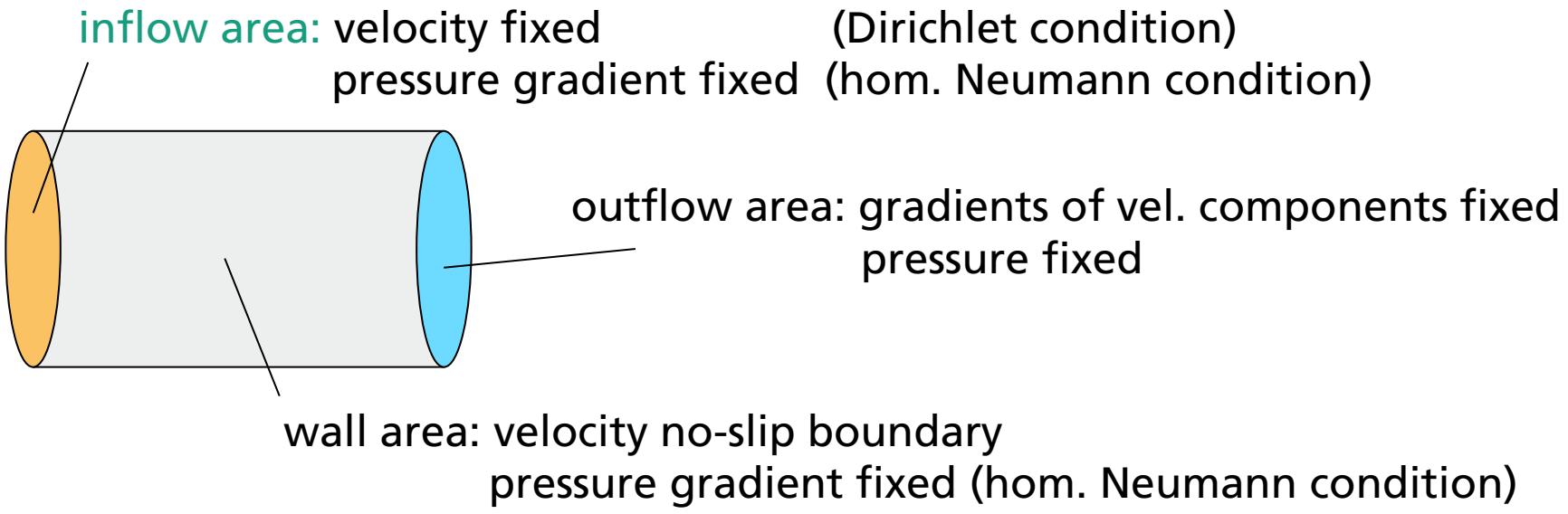
RHS: behavior of the boundary element → required definitions:

- CHAMBER, MOVE, ACTIVE, MAT
- IDENT – “closed“ inflow (no free surface)
- TOUCH
- BC – inflow with fixed velocity

Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
end_alias
```

System to be modeled:



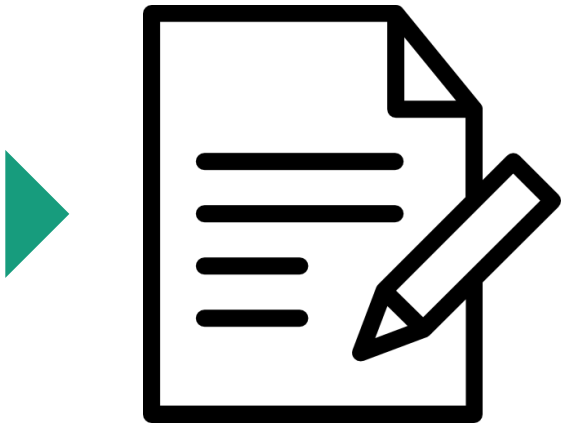
Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
end_alias
```

User-defined boundary conditions:

```
BC_v($BC_inflow$) = ( %BND_inflow%, 10.0 )# velocity inflow boundary condition
BC_p($BC_inflow$) = ( %BND_inflow% )      # classical inflow boundary condition
                                          # for the hydrostatic pressure
BCON($BC_inflow$,%ind_p_dyn%) = ( %BND_AVERAGE% ) # averaging boundary condition
                                                  # for the dynamic pressure
```

The effect of the boundary condition identifiers depends on the variable they are used for. Even the number of arguments varies!



Exercise II – Task #4: Define the alias for the inflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
end_alias
```

User-defined initial conditions (cf. inflow boundary condition):

```
INITDATA($MAT_user$,%ind_v(1)%) = 10.0
INITDATA($MAT_user$,%ind_v(2)%) = 0.0
INITDATA($MAT_user$,%ind_v(3)%) = 0.0
INITDATA($MAT_user$,%ind_p%) = 0.0
INITDATA($MAT_user$,%ind_p_dyn%) = 0.0
```

Exercise II

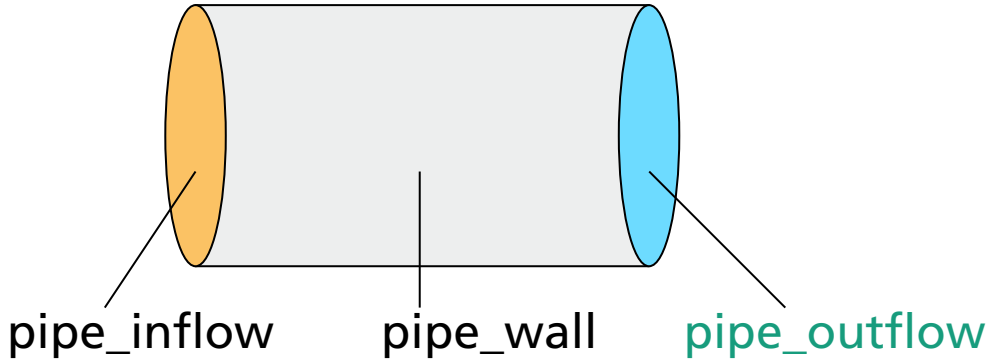
Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the script language.

Tasks:

3. Define the alias for the no-slip wall ✓
4. Define the alias for the inflow ✓
5. Define the alias for the outflow



Exercise II – Task #5: Define the alias for the outflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
 "pipe_outflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% … "
end_alias
```

LHS: Alias name/label of boundary element

RHS: behavior of the boundary element → required definitions:

- CHAMBER, MOVE, ACTIVE, MAT, IDENT, TOUCH

Exercise II – Task #5: Define the alias for the outflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
 "pipe_outflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_outflow$ "
end_alias
```

LHS: Alias name/label of boundary element

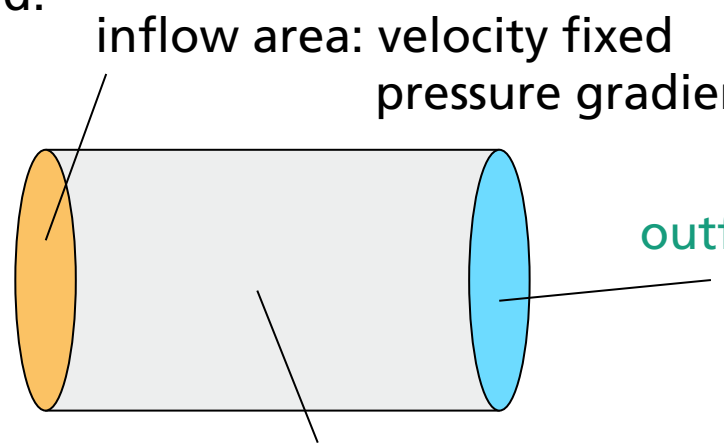
RHS: behavior of the boundary element → required definitions:

- CHAMBER, MOVE, ACTIVE, MAT, IDENT, TOUCH
- BC – outflow with fixed pressure

Exercise II – Task #5: Define the alias for the outflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
 "pipe_outflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_outflow$ "
end_alias
```

System to be modeled:



Exercise II – Task #5: Define the alias for the outflow

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ "
 "pipe_outflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_outflow$ "
end_alias
```

User-defined boundary conditions:

```
BC_v($BC_outflow$) = ( %BND_NEUMANN%, 0.0, 0.0, 0.0) # hom. Neumann condition
BC_p($BC_outflow$) = ( %BND_DIRICH%, 0.0 )          # hom. Dirichlet condition
                                                    # for the hydrostatic pressure
BCON($BC_outflow$,%ind_p_dyn%) = ( %BND_DIRICH%, 0.0 ) # hom. Dirichlet condition
                                                       # for the dynamic pressure
```

The effect of the boundary condition identifiers depends on the variable they are used for. Even the number of arguments varies!

Exercise II

Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the script language.

Tasks:

3. Define the alias for the no-slip wall ✓
4. Define the alias for the inflow
5. Define the alias for the outflow

Exercise II

Geometry

- Surface mesh of a pipe in .msh-format, constructed in mm ✓
- Geometry parts of pipe labeled with unique names

→ At read-in, MESHFREE creates for each part label a variable that has to be defined in the simulation script. The terminology for *variable* is *alias* in the script language.

Tasks:

3. Define the alias for the no-slip wall ✓
4. Define the alias for the inflow ✓
5. Define the alias for the outflow ✓

MESHFREE Fraunhofer ITWM

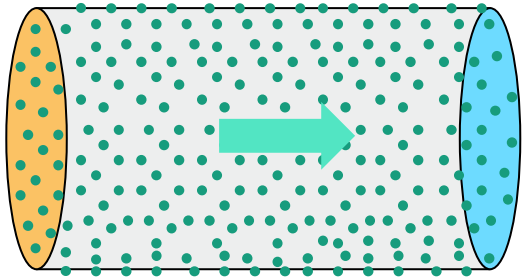
Exercise II

Solver parameters

- Incompressible 3D solver
- Lagrangian formulation (moving point cloud)
- Implicit time integration
- Only velocity, hydrostatic pressure, and dynamic pressure

Task:

6. Define the solver



MESHFREE Fraunhofer ITWM

Exercise II – Task #6: Define the solver

```
KOP(1) = RHS
```

- KOP (chamberwise definition of KindOfProblem) – one phase

Exercise II – Task #6: Define the solver

```
KOP(1) = LIQUID …
```

- KOP (chamberwise definition of KindOfProblem) – one phase
- LIQUID – incompressible solver

MESHFREE Fraunhofer ITWM

Exercise II – Task #6: Define the solver

```
KOP(1) = LIQUID LAGRANGE …
```

- KOP (chamberwise definition of KindOfProblem) – one phase
- LIQUID – incompressible solver
- LAGRANGE – moving point cloud

MESHFREE Fraunhofer ITWM

Exercise II – Task #6: Define the solver

```
KOP(1) = LIQUID LAGRANGE T:NONE …
```

- KOP (chamberwise definition of KindOfProblem) – one phase
- LIQUID – incompressible solver
- LAGRANGE – moving point cloud
- T:NONE – skip computation of temperature

MESHFREE Fraunhofer ITWM

Exercise II – Task #6: Define the solver

```
KOP(1) = LIQUID LAGRANGE T:NONE V:IMPLICIT …
```

- KOP (chamberwise definition of KindOfProblem) – one phase
- LIQUID – incompressible solver
- LAGRANGE – moving point cloud
- T:NONE – skip computation of temperature
- V:IMPLICIT – implicit time stepping

Exercise II – Task #6: Define the solver

```
KOP(1) = LIQUID LAGRANGE T:NONE V:IMPLICIT v--
```

- KOP (chamberwise definition of KindOfProblem) – one phase
- LIQUID – incompressible solver
- LAGRANGE – moving point cloud
- T:NONE – skip computation of temperature
- V:IMPLICIT – implicit time stepping
- v-- – segregated computation of velocity and pressure

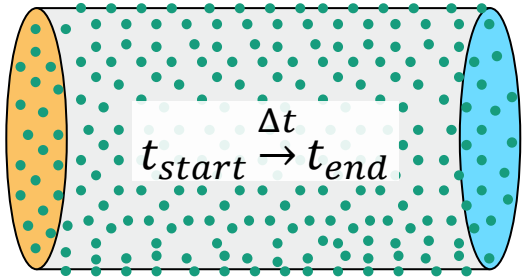
Exercise II

Time control

- Given start and end time of simulation
- Variable time step size

Tasks:

7. Define the start and end time
8. Define the time step size



Exercise II – Task #7: Define the start and end time

```
Tstart = 0
```

- Tstart (initial state INITDATA) – start at 0s

Exercise II – Task #7: Define the start and end time

```
Tstart = 0
Tend = 1
```

- Tstart (initial state INITDATA) – start at 0s
- Tend – end at 1s

Exercise II – Task #8: Define the time step size

```
DELT_dt_variable = 1
```

- DELT_dt_variable – 1 (variable time step size, respect CFL-condition)

Exercise II – Task #8: Define the time step size

```
DELT_dt_variable = 1
DELT_dt_start = 0.001
```

- DELT_dt_variable – 1 (variable time step size, respect CFL-condition)
- DELT_dt_start – fix starting time step (can not be determined automatically)
  → A small value avoids numerical instabilities!

Exercise II – Task #8: Define the time step size

```
DELT_dt_variable = 1
DELT_dt_start = 0.001
DELT_dt = 1.0
```

- DELT_dt_variable – 1 (variable time step size, respect CFL-condition)
- DELT_dt_start – fix starting time step (can not be determined automatically)
- DELT_dt – maximum time step size (overruling CFL-condition)

Exercise II

Solver parameters and time control

Tasks:

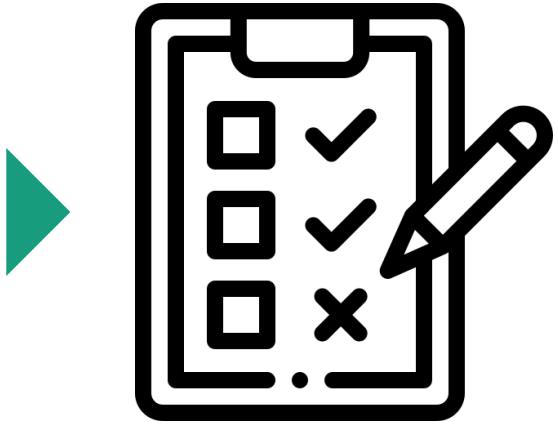
6. Define the solver
7. Define the start and end time
8. Define the time step size

Exercise II

Solver parameters and time control

Tasks:

6. Define the solver ✓
7. Define the start and end time ✓
8. Define the time step size ✓



Exercise III

Post-processing targets

- Velocity and pressure field on point cloud every 10 time steps
- Mass inside pipe, volume flow at inflow, and maximum hydrostatic pressure inside pipe at all times

Tasks:

9. Define the general saving parameters
10. Define the point cloud quantities to be saved

MESHFREE Fraunhofer ITWM

Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
```

- SAVE_path – absolute or relative path for the simulation results

Exercise III – Task #9: Define the general saving parameters

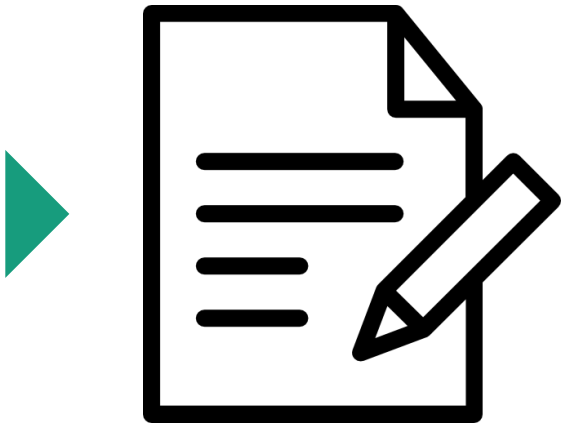
```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
```

- SAVE_path – absolute or relative path for the results
- SAVE_file – base file name for the results

Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
SAVE_format(1) = 'ENSIGHT6 BINARY N--T'
```

- SAVE_path – absolute or relative path for the results
- SAVE_file – base file name for the results
- SAVE_format – format for result files (point cloud + geometry)
  → Multiple formats triggered by indices >1.
  → Results in Ensight 6 binary format.
  → Four letter code: nodes and tetrahedra of point cloud



Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
SAVE_format(1) = 'ENSIGHT6 BINARY N--T'
SAVE_choose_meth = 'CONT'
```

- SAVE_choose_meth – saving mode wrt time steps ('CONT') or time ('TIME')

Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
SAVE_format(1) = 'ENSIGHT6 BINARY N--T'
SAVE_choose_meth = 'CONT'
SAVE_first(1) = 1
```

- SAVE_choose_meth – saving mode wrt time steps ('CONT') or time ('TIME')
- SAVE_first – start saving from first time step
  → Index not connected to index of SAVE_format!
  → Index allows for changing the saving frequency.

Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
SAVE_format(1) = 'ENSIGHT6 BINARY N--T'
SAVE_choose_meth = 'CONT'
SAVE_first(1) = 1
SAVE_interval(1) = 10
```

- SAVE_choose_meth – saving mode wrt time steps ('CONT') or time ('TIME')
- SAVE_first – start saving from first time step
- SAVE_interval – saving interval wrt time steps or time

```
SAVE_first(2) = 100       # save after time step 100
SAVE_interval(2) = 50     # save every 50th time step
```

Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
SAVE_format(1) = 'ENSIGHT6 BINARY N--T'
SAVE_choose_meth = 'CONT'
SAVE_first(1) = 1
SAVE_interval(1) = 10

restart = (0)
```

- restart – do not (0) or do (index > 0) use a previously generated restart file

Exercise III – Task #9: Define the general saving parameters

```
SAVE_path = './results'
SAVE_file = 'TrainingSetup'
SAVE_format(1) = 'ENSIGHT6 BINARY N--T'
SAVE_choose_meth = 'CONT'
SAVE_first(1) = 1
SAVE_interval(1) = 10

restart = (0)
restart_step_size = ( 20,%RESTART_sequence% )
```

- restart – do not (0) or do (index > 0) use a previously generated restart file for continuation
- restart_step_size – generation frequency and mode of restart files
  %RESTART_sequence% → generation of consecutively numbered restart files

Exercise III – Task #10: Define the point cloud quantities to be saved

```
SAVE_ITEM = (%SAVE_vector%, [Y%ind_v(1)%],[Y%ind_v(2)%],[Y%ind_v(3)%], "velocity")
```

LHS: SAVE_ITEM – point cloud item to be saved
RHS: specification of item

- %SAVE_vector% → vectorial quantity (followed by vector components)
- [Y%ind_...%] → inline equation wrt. point cloud quantity
- "…" → description text (displayed in post-processing tool)

Exercise III – Task #10: Define the point cloud quantities to be saved

```
SAVE_ITEM = (%SAVE_vector%, [Y%ind_v(1)%],[Y%ind_v(2)%],[Y%ind_v(3)%], "velocity")
SAVE_ITEM = (%SAVE_scalar%, [Y%ind_p%], "p_hyd")
SAVE_ITEM = (%SAVE_scalar%, [Y%ind_p_dyn%], "p_dyn")
```

LHS: SAVE_ITEM – point cloud item to be saved
RHS: specification of item

- %SAVE_scalar% → scalar quantity (followed by scalar value)
- [Y%ind_...%] → inline equation wrt. point cloud quantity
- "..." → description text (displayed in post-processing tool)

Exercise III

Post-processing targets

- Velocity and pressure field on point cloud every 10 time steps
- Mass inside pipe, volume flow at inflow, and maximum hydrostatic pressure inside pipe at all times

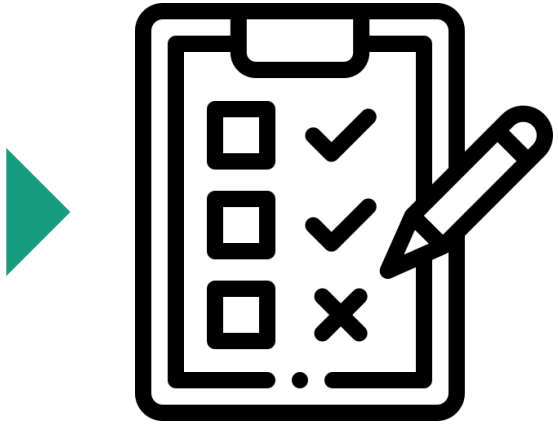
Tasks:

9. Define the general saving parameters
10. Define the point cloud quantities to be saved

Exercise III

Post-processing targets

- Velocity and pressure field on point cloud every 10 time steps
- Mass inside pipe, volume flow at inflow, and maximum hydrostatic pressure inside pipe at all times



Tasks:

9. Define the general saving parameters ✓
10. Define the point cloud quantities to be saved ✓

Exercise III

Post-processing targets

- Velocity and pressure field on point cloud every 10 time steps
- Mass inside pipe, volume flow at inflow, and maximum hydrostatic pressure inside pipe at all times

Tasks:

9. Define the general saving parameters ✓
10. Define the point cloud quantities to be saved ✓
11. Define the “integrated“ quantities to be saved

MESHFREE Fraunhofer ITWM

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = RHS
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration

MESHFREE Fraunhofer ITWM

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, … )
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration/reduction

- %INTEGRATION_INT% → volume integration

MESHFREE

Fraunhofer ITWM

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, [Y%ind_r%], … )
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration/reduction

- %INTEGRATION_INT% → volume integration
- […] → expression of integrand, here “density“

MESHFREE Fraunhofer ITWM

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, [Y%ind_r%], $MAT_user$, … )
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration/reduction

- %INTEGRATION_INT% → volume integration
- […] → expression of integrand, here “density“
- $...$ → material acronym to specify which points to consider (important for multiple phases)

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, [Y%ind_r%], $MAT_user$, %INTEGRATION_Header%, "mass" )
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration/reduction

- %INTEGRATION_INT% → volume integration
- […] → expression of integrand, here “density“
- $...$ → material acronym to specify which points to consider
- %INTEGRATION_Header% → header information
- “…“ → description text for header

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, [Y%ind_r%], $MAT_user$, %INTEGRATION_Header%, "mass" )
INTEGRATION($INT_flow$) = ( %INTEGRATION_BND%, [Y%ind_v(1)%], [Y%ind_v(2)%], [Y%ind_v(3)%], …
                            $PP_inflow$, %INTEGRATION_Header%, "flow_in")
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration/reduction

- %INTEGRATION_BND% → surface integration
- […], […], […] → vectorial expression of integrand, here “velocity“
- … → ContinuationLines (do not forget!)
- $...$ → post-processing acronym to specify which boundary points to consider

MESHFREE Fraunhofer ITWM

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, [Y%ind_r%], $MAT_user$, %INTEGRATION_Header%, "mass" )
INTEGRATION($INT_flow$) = ( %INTEGRATION_BND%, [Y%ind_v(1)%], [Y%ind_v(2)%], [Y%ind_v(3)%], …
                            $PP_inflow$, %INTEGRATION_Header%, "flow_in")
```

Alias definition:

```
begin_alias{}
 "pipe_wall" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_always% BC$BC_wall$ "
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_inflow$ …
                   POSTPROCESS$PP_inflow$ "
 "pipe_outflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_outflow% TOUCH%TOUCH_always% BC$BC_outflow$ "
end_alias
```

Exercise III – Task #11: Define the “integrated“ quantities to be saved

```
INTEGRATION($INT_mass$) = ( %INTEGRATION_INT%, [Y%ind_r%], $MAT_user$, %INTEGRATION_Header%, "mass" )
INTEGRATION($INT_flow$) = ( %INTEGRATION_BND%, [Y%ind_v(1)%], [Y%ind_v(2)%], [Y%ind_v(3)%], …
                            $PP_inflow$, %INTEGRATION_Header%, "flow_in")
INTEGRATION($INT_max$) = ( %MAXIMUM_INT%, [Y%ind_p%], $MAT_user$, %INTEGRATION_Header%, "max_p_hyd" )
```

LHS: INTEGRATION – integration/reduction result to be saved

- $INT…$ – user-defined integration acronym (for later reference)

RHS: specification of integration/reduction

- %MAXIMUM_INT% → maximum over all specified points
- […] → scalar functional, here “hydrostatic pressure“
- $...$ → material acronym to specify which points to consider

MESHFREE Fraunhofer ITWM

Exercise III – Task #11: Define the “integrated“ quantities to be saved

Saved integration results:

- Written to TimestepFile: pure ASCII file with the ending .timestep in the result folder
- Contains the INTEGRATION evaluations (columns) for each time step (rows)
  - 1st column: time
  - 2nd column: time step size
  - 3rd column: first user-defined INTEGRATION
  - …
- Corresponding header file with ending .timestep.header

Analysis e.g. with GNU Octave, gnuplot, or any spreadsheet (MS Excel), see also 11_GnuplotForMeshfree.pdf.

Exercise III

Post-processing targets

- Velocity and pressure field on point cloud every 10 time steps
- Mass inside pipe, volume flow at inflow, and maximum hydrostatic pressure inside pipe at all times

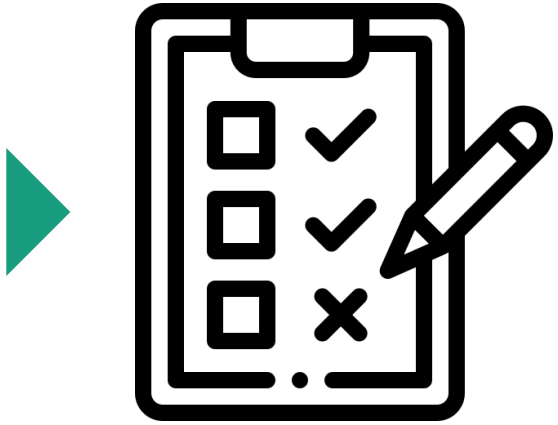
Tasks:

9. Define the general saving parameters ✓
10. Define the point cloud quantities to be saved ✓
11. Define the “integrated“ quantities to be saved

Exercise III

Post-processing targets

- Velocity and pressure field on point cloud every 10 time steps
- Mass inside pipe, volume flow at inflow, and maximum hydrostatic pressure inside pipe at all times



Tasks:

9. Define the general saving parameters ✓
10. Define the point cloud quantities to be saved ✓
11. Define the “integrated“ quantities to be saved ✓

Exercise IV

- Explicit equations used by reference
- Curve definitions
- Point cloud definition
- Parameter definitions with aliases

Tasks:

12. Replace an inline equation by referencing an explicit equation
13. Replace the inflow velocity by a time-dependent curve

Exercise IV – Task #12: Replace an inline equation by referencing an explicit equation

```
SAVE_ITEM = (%SAVE_scalar%, [Y%ind_p%], "p_hyd")  # inline equation
```

Exercise IV – Task #12: Replace an inline equation by referencing an explicit equation

```
SAVE_ITEM = (%SAVE_scalar%, [Y%ind_p%], "p_hyd")  # inline equation

begin_equation{$EQ_pressure$}
   Y%ind_p%
end_equation
```

- Equation environment – definition of equation (allows reuse of equation in different places)

MESHFREE Fraunhofer ITWM

Exercise IV – Task #12: Replace an inline equation by referencing an explicit equation

```
SAVE_ITEM = (%SAVE_scalar%, equn{$EQ_pressure$}, "p_hyd")   # inline equation

begin_equation{$EQ_pressure$}
   Y%ind_p%
end_equation
```

- Equation environment – definition of equation (allows reuse of equation in different places)
- Equation referencing – replaces inline equation

MESHFREE Fraunhofer ITWM

Exercise IV – Task #12: Replace an inline equation by referencing an explicit equation

```
SAVE_ITEM = (%SAVE_scalar%, equn{$EQ_kin_energy$}, "kinetic_energy")

begin_equation{$EQ_kin_energy$}
   0.5*Y%ind_r%*( Y%ind_v(1)%^2 + Y%ind_v(2)%^2 + Y%ind_v(3)%^2 )
end_equation
```

- Access to all point cloud quantities %ind_...%
- Many built-in functions and operators:
  - exp(…), abs(…), sqrt(…), cos(…), sin(…), min(…), max(…)
  - a^b, …
  - integ($...$) → `INTEGRATION($INT_mass$) = …`
  - if-then-else →

```
begin_equation{$EQ_filter$}
   if ( Y%ind_x(1)% > 0.0 ) :: 1.0
   else :: 0.0
   endif
end_equation
```

MESHFREE Fraunhofer ITWM

Exercise IV – Task #13: Replace the inflow velocity by a time-dependent curve

```
BC_v($BC_inflow$) = ( %BND_inflow%, 10.0 )          # velocity inflow boundary condition

INITDATA($MAT_user$,%ind_v(1)%) = 10.0
INITDATA($MAT_user$,%ind_v(2)%) = 0.0
INITDATA($MAT_user$,%ind_v(3)%) = 0.0
```


MESHFREE

Fraunhofer ITWM

Exercise IV – Task #13: Replace the inflow velocity by a time-dependent curve

```
BC_v($BC_inflow$) = ( %BND_inflow%, 10.0 )        # velocity inflow boundary condition

INITDATA($MAT_user$,%ind_v(1)%) = 10.0
INITDATA($MAT_user$,%ind_v(2)%) = 0.0
INITDATA($MAT_user$,%ind_v(3)%) = 0.0

begin_curve{$CV_inflow$}
   0.0   10.0
   0.1   11.0
   0.3   12.0
   1.0   20.0
end_curve
```

- Curve environment – definition of curves (allows reuse of curves in different places)

Exercise IV – Task #13: Replace the inflow velocity by a time-dependent curve

```
BC_v($BC_inflow$) = ( %BND_inflow%, curve{$CV_inflow$} )    # velocity inflow boundary condition

INITDATA($MAT_user$,%ind_v(1)%) = curve{$CV_inflow$}
INITDATA($MAT_user$,%ind_v(2)%) = 0.0
INITDATA($MAT_user$,%ind_v(3)%) = 0.0

begin_curve{$CV_inflow$}
   0.0   10.0
   0.1   11.0
   0.3   12.0
   1.0   20.0
end_curve
```

- Curve environment – definition of curves (allows reuse of curves in different places)
- Curve referencing (analogous to equation referencing) – replaces scalar value

MESHFREE Fraunhofer ITWM

Execise IV

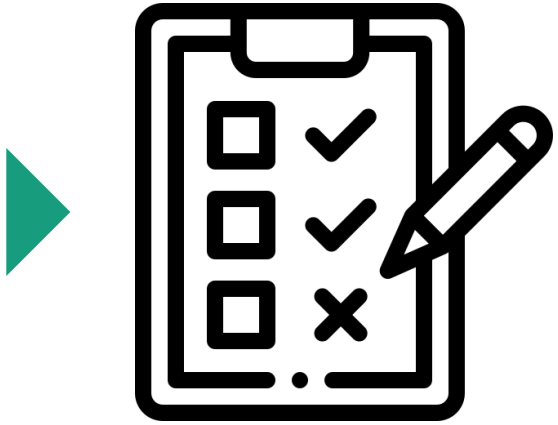
- Explicit equations used by reference
- Curve definitions
- Point cloud definition
- Parameter definitions with aliases

Tasks:

12. Replace an inline equation by referencing an explicit equation
13. Replace the inflow velocity by a time-dependent curve

Execise IV

- Explicit equations used by reference
- Curve definitions
- Point cloud definition
- Parameter definitions with aliases



Tasks:

12. Replace an inline equation by referencing an explicit equation ✓
13. Replace the inflow velocity by a time-dependent curve ✓

Execise IV

- Explicit equations used by reference
- Curve definitions
- Point cloud definition
- Parameter definitions with aliases

Tasks:

12. Replace an inline equation by referencing an explicit equation ✓
13. Replace the inflow velocity by a time-dependent curve ✓
14. Define a point cloud with constant density
15. Use an alias variable to define the maximum and minimum smoothing length

Exercise IV – Task #14: Define a point cloud with constant density

Smoothing length (interaction radius) = determines the average distance between points

```
USER_h_funct = 'CONS'
```

- Smoothing length strategy – constant

Exercise IV – Task #14: Define a point cloud with constant density

Smoothing length (interaction radius) = determines the average distance between points

```
USER_h_funct = 'CONS'
USER_h_max = 0.001
USER_h_min = 0.001
```

- Smoothing length strategy – constant
- Maximum and minimum smoothing length – identical

Exercise IV – Task #15: Use an alias variable to define the maximum and minimum smoothing length

```
USER_h_funct = 'CONS'
USER_h_max = 0.001
USER_h_min = 0.001
```


MESHFREE

Fraunhofer ITWM

Exercise IV – Task #15: Use an alias variable to define the maximum and minimum smoothing length

```
USER_h_funct = 'CONS'
USER_h_max = 0.001
USER_h_min = 0.001

begin_alias{}
 "h" = "0.001"
end_alias
```

- Alias environment – definition of geometry parts and user-defined variables (allows reuse in different places)

MESHFREE Fraunhofer ITWM

Exercise IV – Task #15: Use an alias variable to define the maximum and minimum smoothing length

```
USER_h_funct = 'CONS'
USER_h_max = &h&
USER_h_min = &h&

begin_alias{}
 "h" = "0.001"
end_alias
```

- Alias environment – definition of geometry parts and user-defined variables (allows reuse in different places)
- Alias referencing – string replacement
  &…& → usage in almost all places of the input file possible

Execise IV

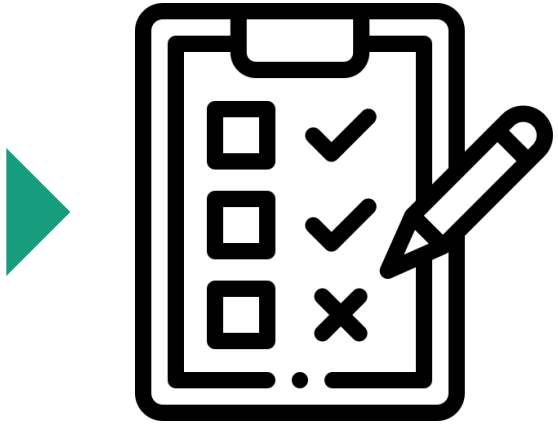
- Explicit equations used by reference
- Curve definitions
- Point cloud definition
- Parameter definitions with aliases

Tasks:

12. Replace an inline equation by referencing an explicit equation ✓
13. Replace the inflow velocity by a time-dependent curve ✓
14. Define a point cloud with constant density
15. Use an alias variable to define the maximum and minimum smoothing length

Execise IV

- Explicit equations used by reference
- Curve definitions
- Point cloud definition
- Parameter definitions with aliases



Tasks:

12. Replace an inline equation by referencing an explicit equation ✓
13. Replace the inflow velocity by a time-dependent curve ✓
14. Define a point cloud with constant density ✓
15. Use an alias variable to define the maximum and minimum smoothing length ✓

MESHFREE Fraunhofer ITWM

Advanced features

General remarks on the scripting language

- $...$, %...%, &...&, ...
- Selection
- include_Ucv{}

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise I
  - Read-in of geometry including scaling
  - Geometry alias for no-slip wall
- Exercise II
  - Geometry aliases for inflow and outflow
  - Solver parameters and time control
- Exercise III
  - General saving parameters and save items
  - “integrated“ quantities
- Exercise IV
  - Equations and curves
  - Point cloud definition and alias variables

→ Result: USER_common_variables.dat (replacing default definitions in Ucv_DEFAULT.dat)

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise V
  - Spherical refinement of point cloud attached to BND_point
  - Radial refinement of point cloud inside pipe

MESHFREE Fraunhofer ITWM

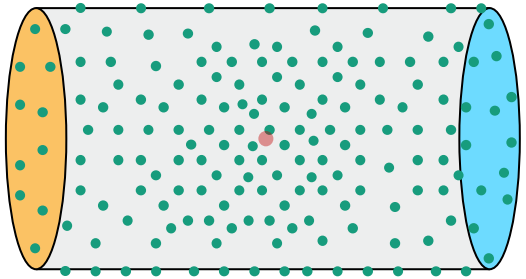
Execise V

Point cloud refinement

- Spherical attached to user-defined point
- Radial inside tube

Tasks:

16. Spherical refinement of point cloud attached to BND_point



Exercise V – Task #16: Spherical refinement of point cloud attached to BND_point

```
USER_h_funct = 'DSCR'

SMOOTH_LENGTH($SL_spherical$) = ( %H_spherical%, &h_min&, &L_min&, &dhdr&, &h_max& )

begin_alias{}
 "h_min" = "0.0005"
 "L_min" = "0.001"
 "dhdr"  = "0.1"
 "h_max" = "0.001"
end_alias

USER_h_max = &h_max&
USER_h_min = &h_min&

begin_boundary_elements{}
 BND_point &ALIAS_point&   0.005 0.0 0.0
end_boundary_elements

begin_alias{}
 "ALIAS_point" = "CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ SMOOTH_LENGTH$SL_spherical$"
end_alias
```

- Form a virtual ball of radius &L_min& around the BND_point
- Define a smoothing length of &h_min& inside the ball and a linear increase of the smoothing length outside of the ball to a maximum of &h_max& wrt. increase rate &dhdr&

Execise V

Point cloud refinement

- Spherical attached to user-defined point
- Radial inside tube

Tasks:

16. Spherical refinement of point cloud attached to BND_point

Tips:

- Save the smoothing length in a SAVE_ITEM, see %ind_h%.
- The number of point filling cycles can be controlled by SimCut.

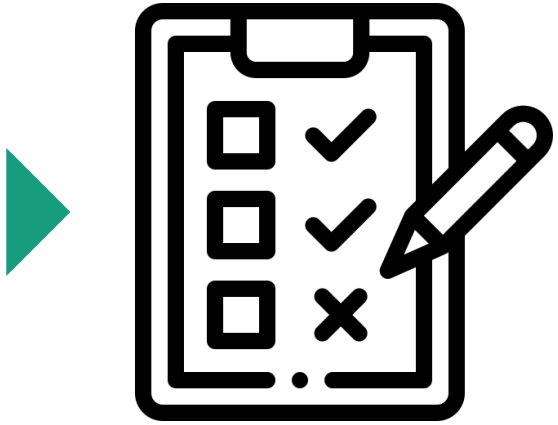
Execise V

Point cloud refinement

- Spherical attached to user-defined point
- Radial inside tube

Tasks:

16. Spherical refinement of point cloud attached to BND_point ✓



Further reference: Take a look at tut3d_04a.

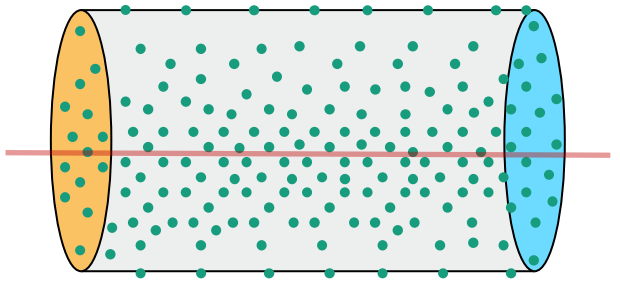
Execise V

Point cloud refinement

- Spherical attached to user-defined point
- Radial inside tube

Tasks:

16. Spherical refinement of point cloud attached to BND_point ✓
17. Radial refinement of point cloud inside pipe



MESHFREE Fraunhofer ITWM

Exercise V – Task #17: Radial refinement of point cloud inside pipe

```
USER_h_funct = 'DSCR'

SMOOTH_LENGTH($SL_radial$) = ( %H_radial%, &h_min&, &L_min&, &axis(1)&, &axis(2)&, &axis(3)&, &dhdr&, &h_max& )

begin_alias{}
 "h_min" = "0.0005"
 "L_min" = "0.001"
 "axis"  = "1.0,0.0,0.0"
 "dhdr"  = "0.1"
 "h_max" = "0.001"
end_alias

USER_h_max = &h_max&
USER_h_min = &h_min&

begin_boundary_elements{}
 BND_point &ALIAS_point&   0.005 0.0 0.0
end_boundary_elements

begin_alias{}
 "ALIAS_point" = "CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ SMOOTH_LENGTH$SL_radial$"
end_alias
```

- Form an infinitely long, virtual tube with radius &L_min& and axis (&axis_x&, &axis_y&, &axis_z&), attach it to the BND_point
- Define a smoothing length of &h_min& inside the tube and a linear increase of the smoothing length outside of the tube to a maximum of &h_max& wrt. radial increase rate &dhdr&

Execise V

Point cloud refinement

- Spherical attached to user-defined point
- Radial inside tube

Tasks:

16. Spherical refinement of point cloud attached to BND_point ✓
17. Radial refinement of point cloud inside pipe

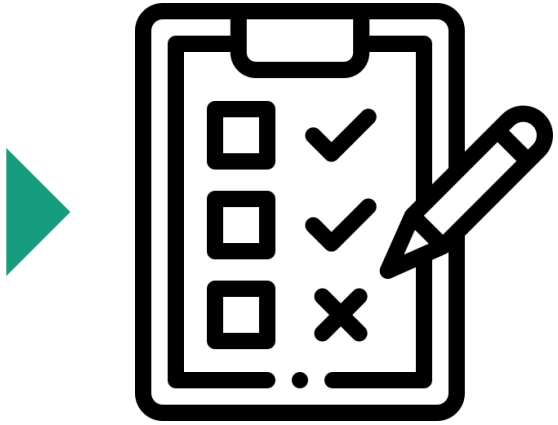
Execise V

Point cloud refinement

- Spherical attached to user-defined point
- Radial inside tube

Tasks:

16. Spherical refinement of point cloud attached to BND_point ✓
17. Radial refinement of point cloud inside pipe ✓



TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise V
  - Spherical refinement of point cloud attached to BND_point
  - Radial refinement of point cloud inside pipe

MESHFREE Fraunhofer ITWM

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise VI
  - Modify the pipe flow setup to an “open“ outflow
  - Modify the pipe flow setup to be only half-filled initially
  - Modify the pipe flow setup to a free inflow ignoring the wall and outflow

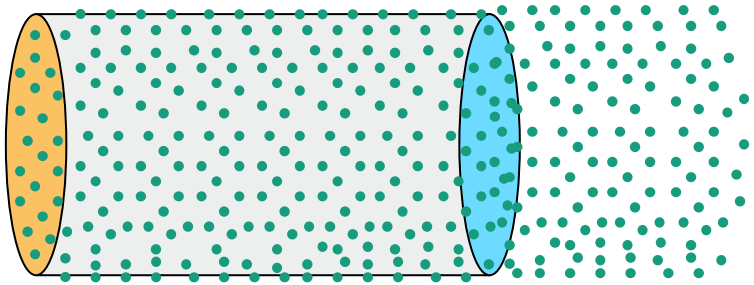
Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow

Tasks:

18. Modify the pipe flow setup to an “open“ outflow



Exercise VI – Task #18: Modify the pipe flow setup to an “open“ outflow

```
compute_FS = 'YES'   # switch on free surface detection
```

- Switch on free surface detection

Exercise VI – Task #18: Modify the pipe flow setup to an “open“ outflow

```
compute_FS = 'YES'   #switch on free surface detection

ACTIVE($ACT_init$) = ( %ACTIVE_init% )   # surface/boundary element is only active at initialization phase

# replace original "pipe_outflow" with free surface declaration (comment previous declaration)
begin_alias{}
 "pipe_outflow" = " CHAMBER1 ACTIVE$ACT_init$ MAT$MAT_user$ BC$free$ "
end_alias
```

- Switch on free surface detection
- Adapt the alias definition of the outflow

Exercise VI – Task #18: Modify the pipe flow setup to an “open“ outflow

```
compute_FS = 'YES'   #switch on free surface detection

ACTIVE($ACT_init$) = ( %ACTIVE_init% )   # surface/boundary element is only active at initialization phase

# replace original "pipe_outflow" with free surface declaration (comment previous declaration)
begin_alias{}
 "pipe_outflow" = " CHAMBER1 ACTIVE$ACT_init$ MAT$MAT_user$ BC$BC_free$ "
end_alias

# boundary conditions for free surface
BC_v($BC_free$)              = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p($BC_free$)              = ( %BND_free%, 0.0 )
BCON($BC_free$,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )

# fallback conditions for free surface
BC_v(0)              = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p(0)              = ( %BND_free%, 0.0 )
BCON(0,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )
```

- Switch on free surface detection
- Adapt the alias definition of the outflow
- Add boundary conditions for the free surface

Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow

Tasks:

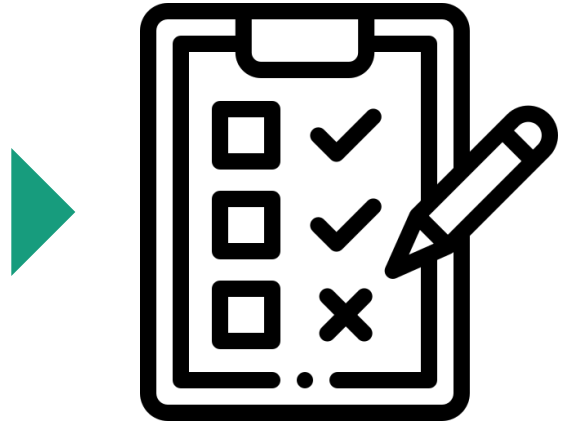
18. Modify the pipe flow setup to an “open“ outflow

Tip: Save the kind of boundary in a SAVE_ITEM, see %ind_kob%.

Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow



Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓

MESHFREE Fraunhofer ITWM

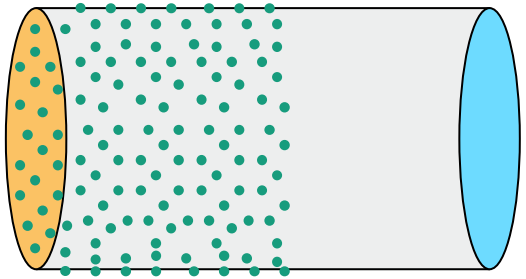
Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow

Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓
19. Modify the pipe flow setup to be only half-filled initially



MESHFREE Fraunhofer ITWM

Exercise VI – Task #19: Modify the pipe flow setup to be only half-filled initially

```
begin_boundary_elements{}
   include{pipe.msh} scale{0.001} ignore{"pipe_outflow"}
end_boundary_elements

compute_FS = 'YES'   # switch on free surface detection

ACTIVE($ACT_init$) = ( %ACTIVE_init% )   # surface/boundary element is only active at initialization phase

begin_boundary_elements{}
 BND_plane &ALIAS_plane&  0.005 0.0 0.0  -1.0 0.0 0.0
end_boundary_elements

# replace "pipe_wall" declaration (comment previous declaration)
begin_alias{}
 "ALIAS_plane"  = " METAPLANE1 CHAMBER1 ACTIVE$ACT_init$ MAT$MAT_user$ BC$BC_free$ "
 "pipe_wall"    = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_slip% TOUCH%TOUCH_liquid% BC$BC_wall$ "
end_alias

# boundary conditions for free surface
BC_v($BC_free$)              = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p($BC_free$)              = ( %BND_free%, 0.0 )
BCON($BC_free$,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )

# fallback conditions for free surface
BC_v(0)              = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p(0)              = ( %BND_free%, 0.0 )
BCON(0,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )
```

- Ignore the outflow and add a cutting plane for a half-filled state
- Adapt the declaration of the wall

Exercise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow

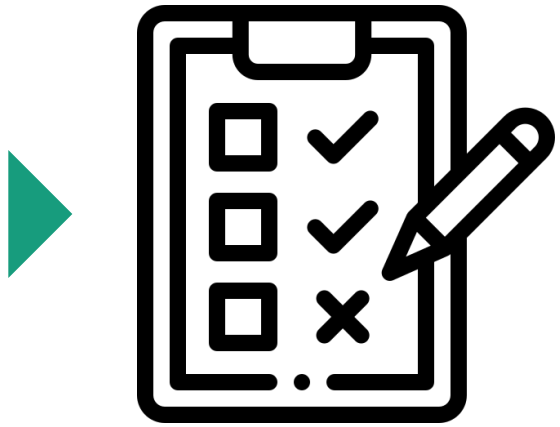
Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓
19. Modify the pipe flow setup to be only half-filled initially

Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow



Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓
19. Modify the pipe flow setup to be only half-filled initially ✓

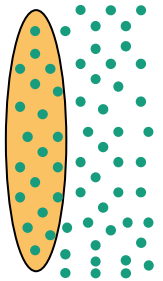
Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow

Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓
19. Modify the pipe flow setup to be only half-filled initially ✓
20. Modify the pipe flow setup to a free inflow ignoring the wall and outflow



MESHFREE Fraunhofer ITWM

Exercise VI – Task #20: Modify the pipe flow setup to a free inflow ignoring the wall and outflow

```
begin_boundary_elements{}
   include{pipe.msh} scale{0.001} ignore{"pipe_outflow","pipe_wall"}
end_boundary_elements

compute_FS = 'YES'   # switch on free surface detection

# replace "pipe_inflow" declaration (comment previous declaration)
begin_alias{}
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_inflow% TOUCH%TOUCH_always% BC$BC_inflow$ POSTPROCESS$PP_inflow$ "
end_alias

# boundary conditions for free surface
BC_v($BC_free$)              = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p($BC_free$)              = ( %BND_free%, 0.0 )
BCON($BC_free$,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )

# fallback conditions for free surface
BC_v(0)             = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p(0)             = ( %BND_free%, 0.0 )
BCON(0,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )
```

- Ignore the outflow and the wall
- Adapt the declaration of the inflow

Exercise VI – Task #20: Modify the pipe flow setup to a free inflow ignoring the wall and outflow

```
begin_boundary_elements{}
   include{pipe.msh} scale{0.001} ignore{"pipe_outflow","pipe_wall"}
end_boundary_elements

compute_FS = 'YES'    # switch on free surface detection

# replace "pipe_inflow" declaration (comment previous declaration)
begin_alias{}
 "pipe_inflow" = " CHAMBER1 MOVE-1 ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_inflow% TOUCH%TOUCH_always% BC$BC_inflow$ POSTPROCESS$PP_inflow$ "
end_alias

# boundary conditions for free surface
BC_v($BC_free$)              = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p($BC_free$)              = ( %BND_free%, 0.0 )
BCON($BC_free$,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )

# fallback conditions for free surface
BC_v(0)             = ( %BND_free%, 0.0, 0.0, 0.0, 0.3)
BC_p(0)             = ( %BND_free%, 0.0 )
BCON(0,%ind_p_dyn%) = ( %BND_free_implicit%, 0.0 )
```

```
COMP_FillEdges = 1    # add parameter in common_variables.dat
```

- Ignore the outflow and the wall
- Adapt the declaration of the inflow
- Activate filling of points on the edge of the inflow

Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow

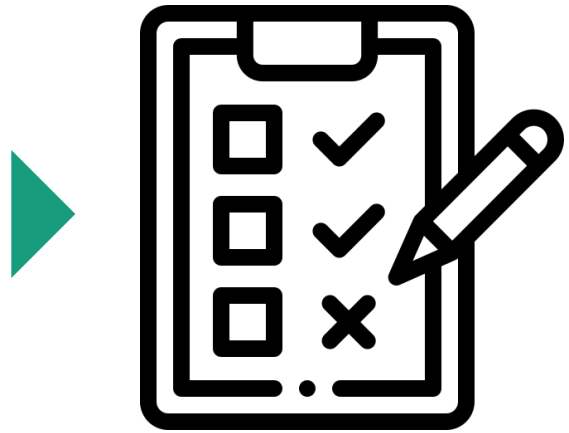
Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓
19. Modify the pipe flow setup to be only half-filled initially ✓
20. Modify the pipe flow setup to a free inflow ignoring the wall and outflow

Execise VI

Free surface flow

- “Open“ outflow
- Half-filled pipe
- Free inflow



Tasks:

18. Modify the pipe flow setup to an “open“ outflow ✓
19. Modify the pipe flow setup to be only half-filled initially ✓
20. Modify the pipe flow setup to a free inflow ignoring the wall and outflow ✓

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise VI
  - Modify the pipe flow setup to an “open“ outflow
  - Modify the pipe flow setup to be only half-filled initially
  - Modify the pipe flow setup to a free inflow ignoring the wall and outflow

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise VII
  - Move the free inflow with a constant velocity
  - Explore the tutorials and letter cases
  - Open topics

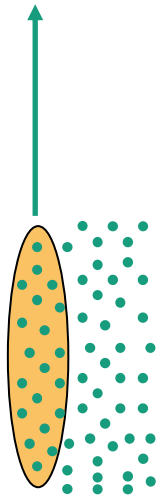
Execise VII

Movement

- Constant velocity

Tasks:

21. Move the free inflow with a constant velocity



Exercise VII – Task #21: Move the free inflow with a constant velocity

```
# adapt the movement of "pipe_inflow"
begin_alias{}
 "pipe_inflow" = " CHAMBER1 MOVE$MV_vel$ ACTIVE$ACT_all$ MAT$MAT_user$ IDENT%IDENT_inflow% TOUCH%TOUCH_always% BC$BC_inflow$ POSTPROCESS$PP_inflow$ "
end_alias

MOVE($MV_vel$) = ( %MOVE_velocity%, 0.0, 0.1, 0.0 )
```

- Adapt the movement of the inflow and define the movement with constant velocity

Execise VII

Movement

- Constant velocity

Tasks:

21. Move the free inflow with a constant velocity

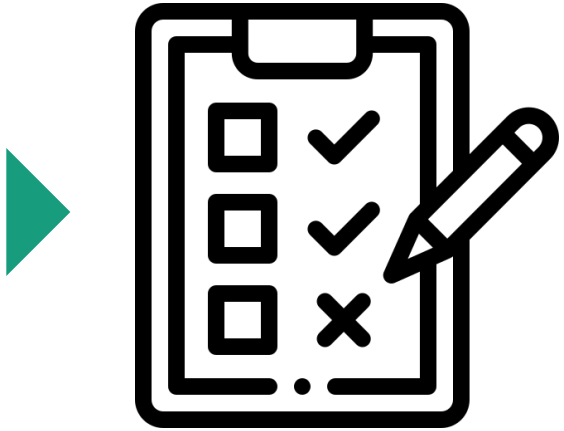
Execise VII

Movement

- Constant velocity

Tasks:

21. Move the free inflow with a constant velocity ✓



Execise VII

Explore the tutorials and letter cases

- Material properties
- …

Execise VII

Open topics

- …

TRAINING SETUP

Simulate a single-phase flow (velocity and pressure) in a fixed pipe

- Exercise VII
  - Move the free inflow with a constant velocity
  - Explore the tutorials and letter cases
  - Open topics

MESHFREE Fraunhofer ITWM