Tutorial: Rotating Drum

This tutorial uses the discrete element method to simulate a rotating drum. The drum is held at a fixed hot temperature of 600K and particles are initially placed in the drum with a temperature of 300K. As the drum rotates, the particles slide along the surface of the wall and are warmed by conduction. Radiative heat transfer is neglected, and heat is conducted through the particle-wall contact area as well as through the thin layer of interstitial gas between the wall and adjacent particles.

This tutorial demonstrates:

1. Simulating particle-wall and particle-fluid-wall heat transfer
2. Using terms that correct unphysically large contact areas that may arise if artificial collision softening is employed
3. Outputting the boundary heat flux

Geometry:

The rotating drum is simulated in 3-D and the depth of the domain is approximately 8 particle diameters. The dimensions of the computational domain and drum diameter are shown in Table 1. When using cut-cells and the discrete element method, the cut surfaces need to be represented using a stl file (geometry.stl in this tutorial).

Table 1. Description of domain geometry

D_{cyl}	= 0.60 m		
X_{length}	= 0.65 m	$NCell_x$	= 30
Y_{length}	= 0.65 m	$NCell_y$	= 30
Z_{length}	= 0.04 m	$NCell_z$	= 4

Simulation Conditions:

The drum is initially filled with 20,000 particles as shown in Figure 1. The particles are initialized with a temperature of 300 K. The walls of the drum are held at a fixed temperature of 600 K and the front and back walls of the domain are free-slip adiabatic boundaries.

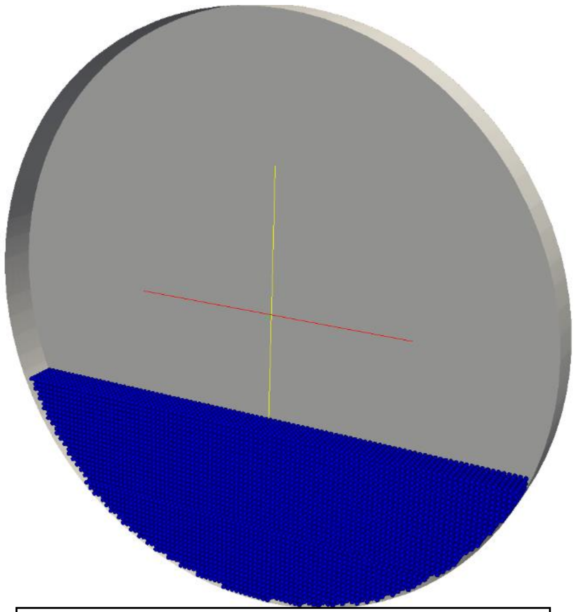
Figure 1. Particles inside the drum

The geometry is held fixed in time, and the rotation of the drum is generated by modifying the gravity vector. The rotational speed (RPM) of the drum is specified in “mfix.dat” using the C(1) constant, and a baseline value of 20 RPM is used in this tutorial.

1 - Setting up the MFIX solver (mfix.dat)

The rotating drum is modeled as a granular system and the background fluid is not simulated. The solids phase is solved via the discrete element method, so the continuum momentum equations are turned off for both the solids and gas phases. Heat transfer in the system is solved by turning on the energy equation. The species equations are turned off because chemical reactions do not occur. The "run control" section of mfix.dat is:

```
! Run control
!                          (gas)      (solid)
MOMENTUM_X_EQ(0:1) =       .FALSE.    .FALSE.    ! Turn off momentum equations for
MOMENTUM_Y_EQ(0:1) =       .FALSE.    .FALSE.    ! gas and solids phases for pure
MOMENTUM_Z_EQ(0:1) =       .FALSE.    .FALSE.    ! granular DEM

SPECIES_EQ(0:1) =          .FALSE.    .FALSE.    ! Turn off species equations

ENERGY_EQ =                .TRUE.                ! Turn on energy equation
```

2 - Creating the domain geometry

The cylinder is modeled using cut-cells. The cut surfaces must be defined via an STL file when using the discrete element method. In addition, DEM simulations with cut-cells must be 3-dimensional. Small cells are not removed from the domain so that the boundary appears smooth, and this is done by setting TOL_SMALL_CELL to zero.

```
!  Geometry section
!  (domain dimensions)                (number of cells in each direction)
XLENGTH  =               0.65         IMAX  =    30
YLENGTH  =               0.65         JMAX  =    30
ZLENGTH  =               0.04         KMAX =     4

CARTESIAN_GRID =         .TRUE.       ! Use cut-cells
USE_STL =                .TRUE.       ! DEM requires STL files to be used (reads geometry.stl)
DIM_FACETS_PER_CELL =    24           ! May have to increase this value depending on the geometry
TOL_SMALL_CELL =         0.0          ! Do not remove small cells to get a smooth boundary

TX_STL =                 0.325        ! Shift cylinder to center of domain
TY_STL =                 0.325
OUT_STL_VALUE =          1.0          ! Internal domain

STL_BC_ID =              5            ! Use BC 5 for drum
```

3 – Setting solids and gas phase properties

The solids and gas phase properties are defined in mfix.dat. To model conductive heat transfer, the solids thermal conductivity (K_s0) needs to be specified. Radiative heat transfer is not included in these simulations, and radiation is turned off by setting "DES_em" to zero. The gas density (RO_g0) is set to zero for a pure granular system. Note, however, that the thermal conductivity of the gas is specified because that value is used when computing particle-fluid-particle and particle-fluid-wall heat transfer.

```
!  Solids Phase Properties

MMAX                    1           ! Number of solids phases
D_p0(1) =               0.005       ! Particle diameter [mm]
RO_s0(1)  =             2500.0      ! Particle density [kg/m3]
C_ps0(1) =              840.0       ! Specific heat [J/kg-K]
K_s0(1) =               1.05        ! Thermal conductivity [W/m-K] – Conduction is modeled
DES_em(1) =             0.00        ! Thermal emissivity set to zero.  Radiation is neglected

! Gas phase Properties
RO_g0 =                 0.00        ! Setting rho to zero because simulation is pure granular
Mu_g0 =                 1.8d-5      ! viscosity of air [Pa-s] (not used)
C_pg0 =                 1020.0      ! Specific heat of air [J/kg-K]
K_g0 =                  0.0372      ! Thermal conductivity of air [W/m-K]. This is used when
                                    ! computing conduction through interstitial gas
```

4 – Generating the initial particles

The drum is initially filled with 20,000 particles that are read from the particle_input.dat file. The particle_input.dat file was generated using a slightly modified version of the DES particle generator in the ./mfix/tools directory. The particle generator (DESParticleGen.f) was modified to ensure that all new particles are located inside of the drum. The file "particle_input.dat" must be located in the run directory and "PARTICLES" must be set to 20,000 in mfix.dat.

5 – DEM properties

The default linear-spring-dashpot collision model is used in this tutorial.

```
!  Discrete Element Model

PARTICLES =               20000      ! Read 20,000 particles from particle_input.dat
! Collision properties
KN =                      1.0D2      ! normal collision spring constant
MEW =                     0.10       ! Friction coefficient
KN_W =                    1.0D2      ! particle-wall normal collision spring constant
MEW_W =                   0.10       ! Friction coefficient for particle-wall contacts
DES_EN_INPUT =            0.90       ! Restitution coefficient (particle-particle)
DES_EN_WALL_INPUT =       0.90       ! Restitution coefficient (particle-wall)
! DES Conduction parameters
FLPC =                    0.20       ! Particle fluid lens thickness divided by particle radius
DES_MIN_COND_DIST         5.75E-8    ! Assumed minimum gap distance between particles and the wall
```

6 – Correction terms for artificial softening

For soft-sphere discrete-particle simulations, it is computationally expensive to resolve the true collision time because doing so would require a restrictively small numerical time step. To improve the computational speed, it is common to increase the 'softness' of the material to artificially increase the collision time, but doing so affects the heat transfer by increasing the contact area and collision time. Correction terms that depend on the true material properties have been developed to reduce errors associated with artificial softening. To use these correction terms, the following keywords must be specified in mfix.dat.

```
! Correction Terms for Conduction

E_YOUNG_ACTUAL(1) =       70.0D9    ! [Pa] – The true Young's modulus of particles
EW_YOUNG_ACTUAL =         70.0D9    ! [Pa] – The true Young's modulus of the walls
v_poisson_actual(1) =     0.30      ! True poisson ratio for particles
vw_poisson_actual =       0.30      ! True poisson ratio for wall material
```

7 – Writing user defined functions

User defined functions are used to rotate the gravity vector and store time-averaged values for the particle-wall heat conductive heat flux. The UDFs for this tutorial are found in "usr0.f, usr1_des.f, usr_mod.f". These files need to be placed in the compile directory.

8 – Rotating the gravity vector

The rotational speed is set by defining the constant C(1) in "mfix.dat". The gravity vector is rotated using "usr1_des.f", which is called every solids time step.

```
! … Code in usr1_des.f
RPM = C(1)
ANGLE = time * RPM * 2.0D0*Pi/60.0D0
GRAVITY_X = sin(ANGLE)*GRAVITY
GRAVITY_Y = -cos(ANGLE)*GRAVITY
GRAV(1) = GRAVITY_X
GRAV(2) = GRAVITY_Y        ! GRAV() is used by DES solver
```

9 – Computing Time-Averaged Particle-Wall Heat Flux

In many heat transfer applications, the heat transfer to and from the wall is a desired output from the simulation. To compute the average heat transfer, the following UDFs are written.

Step 1 (usr_mod.f): The array "DES_QwFlux_AVG(:,:)" is created to store the average particle-wall heat flux. The variable "QW_SAMPLE_TIME" keeps track of the size of the averaging window.

Step 2 (usr0.f): Usr0 is called once at the beginning of the simulation. The usr0 subroutine is used to allocate the new arrays and initialize values. The array for the average particle-wall flux is allocated by:

```
! usr0.f
        ALLOCATE(DES_QwFlux_Avg (DIMENSION_3, DIMENSION_M))
        DES_QwFlux_Avg(:,:) = ZERO
        QW_SAMPLE_TIME = ZERO
```

Step 3 (usr1_des.f): The subroutine usr1_des is called every solids time step and is used to keep a running average of the particle-wall heat flux. The particle-wall heat flux is computed during each solids time step in the subroutine "CALC_DEM_THERMO_WITH_WALL_STL". The particle wall heat flux is saved in the variable "DES_QW_Cond". The particle-wall heat flux is averaged over the time between output samples in the following user defined function:

```
! usr1_des.f

DO IJK = IJKSTART3, IJKEND3                 ! Loop over fluid cells
      IF(.NOT.FLUID_AT(IJK))CYCLE           ! Skip blocked cells
      DO M=MMAX+1, DES_MMAX+MMAX   ! Loop over DEM solids phases
! Compute running average
            DES_QWFlux_AVG(IJK,M) = (DES_QWFlux_AVG(IJK,M)*QW_SAMPLE_TIME+&
&                 DESQw_COND(IJK,M)*DTSOLID)/(QW_SAMPLE_TIME+DTSOLID)
       ENDDO
 ENDDO
 ! Increment sample time counter
 QW_SAMPLE_TIME+=DTSOLID
```

To write the output data, the DES_QWFlux_AVG(:,:) array is copied into the DEBUG_CG(:,:) array. In addition, QW_SAMPLE_TIME and the average heat flux arrays are reset after the output file is written.

To output DEBUG_CG field data for the 1st solids phase, VTK_DEBUG(1,1) is set to .TRUE. in mfix.dat. The output VTU files should now have data called "DEBUG_CG" which corresponds to the particle-wall heat flux.

10 – Compiling and Running

Download and extract the current release of MFIX. To compile and run, make sure that the following files are located in the run directory:

1. mfix.dat
2. particle_input.data
3. geometry.stl
4. usr_mod.f
5. usr0.f
6. usr1_des.f

Navigate to your run directory and configure/make via:

```
>$  sh{mfix_source_dir}/configure_mfix --dmp FC=mpif90 FCFLAGS="-O3"
>$  make
```

MFIX should now compile with the user defined functions. To run mfix on 4 processors,

```
>$ mpirun –n 4 ./mfix
```

The rotational speed of the drum can be modified by changing C(1) in mfix.dat.

11 – Sample outputs

Particles are colored by temperature and are shown in Figure 2 at different times. The drum rotates clockwise and particles are warmed as they contact the wall. The particles circulate upward near the wall and then avalanche down the top face of the particle pile. The warmed particles tend to be located near the wall and along the top surface where the particles avalanche down the pile. As the drum rotates, the particles mixing distributes heat throughout the drum.

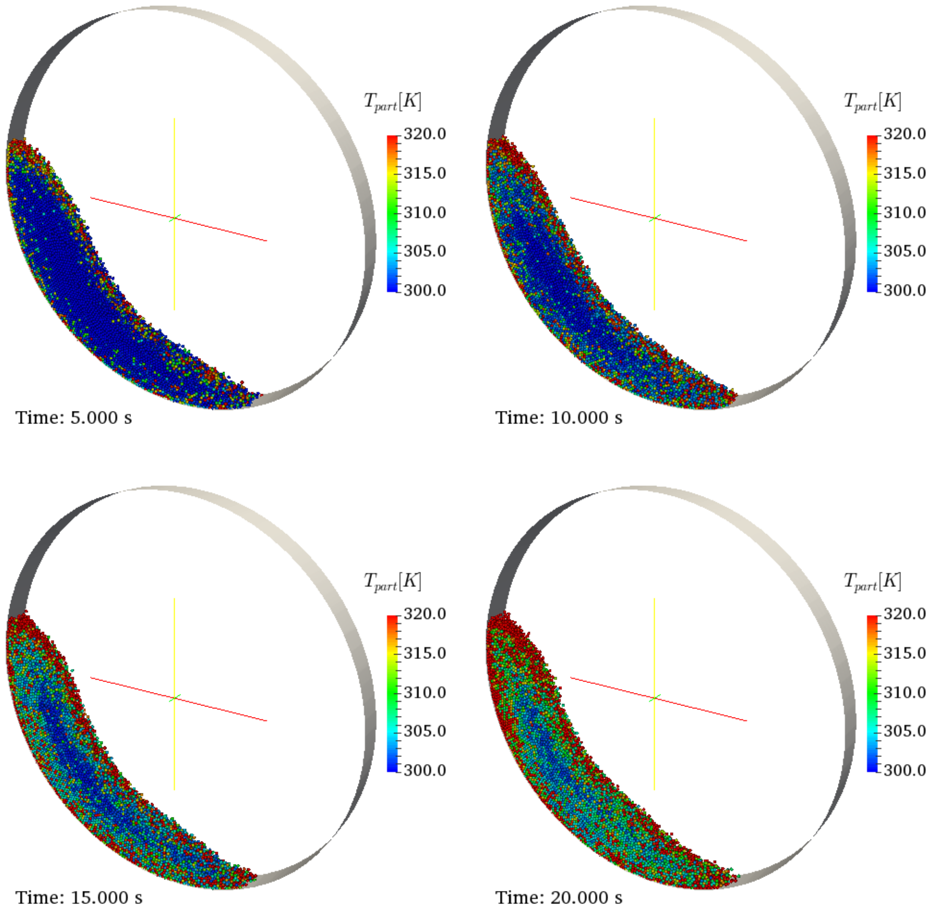

Figure 2. Snapshots showing particles colored by temperature at different times in the drum.

The conductive heat flux is shown in Figure 3 after the drum rotates for 10 seconds. The heat flux is large where particles maintain good thermal contact with the wall and zero in void regions. The heat flux is slightly smaller near the front and back walls of the domain, and this may occur because the proximity to such boundaries affects the particle packing.

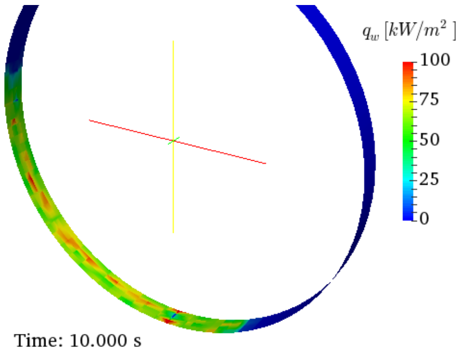

Figure 3. Conductive heat flux from the wall to particles