The MFiX 20.4 User Guide

Release 20.4-dev

MFS Development Group

Feb 11, 2021

GETTING STARTED

1 About **1**
1.1 MFiX . . . 1
1.2 Development state of MFiX models . . . 1
1.3 GUI . . . 3
1.4 Support Forum . . . 3
1.5 User contribution . . . 4

2 Getting Started **5**
2.1 Windows Installation . . . 5
2.2 MFiX Installation on Mac . . . 7
2.3 Linux Installation . . . 10
2.4 Pip Installation . . . 13

3 Tutorials **15**
3.1 First Tutorial . . . 15
3.2 Two Dimensional Fluid Bed, Two Fluid Model (TFM) . . . 18
3.3 Two Dimensional Fluid Bed, Discrete Element Model (DEM) . . . 31
3.4 Three Dimensional Single phase flow over a sphere . . . 40
3.5 Three Dimensional Fluidized Bed . . . 50
3.6 Three Dimensional DEM Hopper . . . 60
3.7 DEM Granular Flow Chutes . . . 66
3.8 SMS meshing workflow, cyclone, Discrete Element Model (DEM) . . . 72
3.9 DEM Initial Conditions . . . 91
3.10 Advanced Tutorials . . . 99

4 Model Guide **103**
4.1 Model Setup . . . 103
4.2 Geometry . . . 105
4.3 Mesh . . . 109
4.4 Regions . . . 110
4.5 Fluid . . . 112
4.6 Solids . . . 114
4.7 Scalars . . . 123
4.8 Initial Conditions . . . 125
4.9 Boundary Conditions . . . 131
4.10 Point Sources . . . 132
4.11 Monitors . . . 132
4.12 Internal Surfaces . . . 139
4.13 Chemical Reactions . . . 140

5 Building the Solver **157**
5.1 Building Custom Interactive Solver . . . 157
5.2 Building a Batch Solver . . . 158
5.3 Build from Source (for Developers) . . . 160

6 Running the Solver **165**
6.1 Default Interactive Solver . . . 165
6.2 Running Custom Interactive Solver . . . 166
6.3 Running a Batch Solver . . . 166

7 Visualization **169**
7.1 Visualization in GUI . . . 169
7.2 Post Processing . . . 169

8 Reference **175**
8.1 GUI Reference . . . 175
8.2 User-Defined Functions . . . 190
8.3 Keywords . . . 193
8.4 Frequently Asked Questions . . . 335
8.5 How to setup PIC simulations . . . 343

9 Running MFiX on Joule **349**
9.1 Running the GUI . . . 349
9.2 Building Solver with GCC . . . 349
9.3 Building Solver with Intel Fortran Compiler . . . 350
9.4 Building Solver from Source Tarball . . . 350

Bibliography **351**

CHAPTER

ONE

ABOUT

1.1 MFiX

MFiX is an open-source multiphase flow solver and is free to download and use. A one-time, no-cost registration is required prior to downloading the source code. To register, go to https://mfix.netl.doe.gov/ and click on the "Register" button in the upper right corner. Once you have read the notice, you can submit your application by clicking on "REGISTER." After your application has been reviewed and accepted, you will receive an email notification and instructions on how to download the code. Please allow for 2-3 business days for your registration to be processed.

Potential users may find reviewing the Frequently Asked Questions section of the MFiX website useful before downloading the code.

1.2 Development state of MFiX models

MFiX provides a suite of models that treat the carrier phase (typically the gas phase) and disperse phase (typically the solids phase) differently. Their current state of development is summarized in the tables below.

Symbol description for the following tables:

Symbol	Description
●	Implemented and fully tested
○	Implemented with limited testing
□	Not tested or status unknown
†	Models not extended to DMP-parallel are only available for serial runs
‡	Models not extended to SMP-parallel are available for SMP runs but do not scale with thread count

1.2.1 MFiX-TFM (Two-Fluid Model)

MFiX-TFM (Two-Fluid Model) is an Eulerian-Eulerian model, which supports a broad range of capabilities for dense, reacting, multiphase flows by representing the fluid and solids as interpenetrating continua. This is the most mature MFiX model and is capable of modeling multiphase reactors ranging in size from benchtop to industry-scale. Approximation of the solid phase as a continuum typically allows for faster simulation time than Lagrangian techniques; however, it also introduces the need for accurate mathematical models to capture realistic solids phase behavior. This includes transport properties, heterogeneous reaction kinetics, and constitutive relations for interaction between fluid and solid phases, e.g., solids phase drag and interphase heat transfer.

Feature	Serial	†DMP	‡SMP
Momentum Equations	●	●	●
Energy Equations	●	●	●
Species Equations	●	●	●
Chemical Reactions	●	●	
Cartesian cut-cell	●	●	◻

1.2.2 MFiX-DEM (Discrete Element Model)

MFiX-DEM (Discrete Element Model) is an Eulerian-Lagrangian model that treats the fluid phase as a continuum and models the individual particles of the solid phase. This is a relatively new variation on MFiX. While the treatment of individual particles can provide higher fidelity over a broad range of flow regimes (from dilute to packed), it becomes very challenging (in terms of computational resources) when dealing with very large numbers of particles for large-scale simulations. These large-scale applications will require high performance computing (HPC) resources and large amounts of computer time. Code optimization and speed up are critical research fronts to support industrial scale applications.

Feature	Serial	†DMP	‡SMP
Momentum Equations	●	●	●
Energy Equations	●	●	
Species Equations	●	●	
Chemical Reactions	●	●	
Cartesian cut-cell	○	○	

1.2.3 MFiX-CGP (Coarse Grained Particle)

MFiX-CGP (Coarse Grained Particle) is an Eulerian-Lagrangian model similar to DEM, where particles are grouped into larger coarse-grained particles (CGP). The collisions between CGPs are resolved the same way as in DEM. This modeling approach is more affordable than DEM since fewer CGPs need to be tracked. This modeling approach falls between DEM and PIC in terms of computational resources.

Feature	Serial	†DMP	‡SMP
Momentum Equations	●	●	●
Energy Equations	●	●	
Species Equations	●	●	
Chemical Reactions	●	●	
Cartesian cut-cell	○	○	

1.2.4 MFiX-PIC (Multiphase Particle in Cell)

MFiX-PIC (Multiphase Particle in Cell) is another Eulerian-Lagrangian model that represents the fluid as a continuum while using "parcels" to represent groups of real particles with similar physical characteristics. The MFiX-PIC approach offers reduced computational cost over MFiX-DEM as there are typically fewer parcels to track and parcel collisions are not resolved. However, the added modeling approximations influence the overall accuracy of the method. Development, validation, and optimization of modeling approximations are critical research fronts.

Feature	Serial	†DMP	‡SMP
Momentum Equations	●	●	○
Energy Equations	●	●	
Species Equations	●	●	
Chemical Reactions	●	●	
Cartesian cut-cell	●	●	

1.2.5 MFiX-Hybrid (Eulerian-Lagrangian-Eulerian)

MFiX-Hybrid (Eulerian-Lagrangian-Eulerian) is a blend of MFiX-TFM and MFiX-DEM that represents the fluid as a continuum and models solids as either a continuous phase (TFM) or discrete particles (DEM). This technique is presently restricted to solving only the momentum equations to yield hydrodynamic predictions. This model is still in its infancy and has seen only limited testing.

Feature	Serial	†DMP	‡SMP
Momentum Equations	○	○	○
Energy Equations			
Species Equations			
Chemical Reactions			
Cartesian cut-cell	○	○	○

Note: The hybrid model is not currently supported and users should use caution if they decide to explore the hybrid model. Creating MFiX-Hybrid models in the GUI is not supported.

1.3 GUI

The MFiX GUI is a front-end tool that allows users to quickly set-up MFiX models, run the developed models, and provide post-processing tools with the goal of making MFiX easy to use.

The GUI is written in Python using the cross-platform Qt framework. The Anaconda platform is used for MFiX dependencies, which allows us to support MFiX on Linux, Windows, and Mac. The Visualization Toolkit (VTK) is used to visualize and manipulate the input geometry.

1.4 Support Forum

When your subscription to MFiX is accepted, you are automatically added to the MFiX forum, where important announcements about MFiX are shared with the MFiX community. If you are already an MFiX member, use you current credentials to participate in the forum.

The Forum is a discussion platform for MFiX, Nodeworks and Tracker software. Developers and users can post topics (discussion threads) to various categories for each software.

Users are encouraged to participate in the discussion and share their work (input files, udf's, images or small animations showing simulation results) to benefit the entire MFiX community.

The support forum is located at https://mfix.netl.doe.gov/forum. Click on the topics of interest to you.

Support forum etiquette:

1) Do not post offensive or inappropriate material. Stay courteous and respectful at all times.
2) Please allow sufficient time (say 2 to 3 business days) for MFiX developers and users to reply before posting unanswered questions again.
3) Post questions in the appropriate forum category. Do not send requests or emails directly to MFiX developers or other users unless a prior arrangement has been made. This ensures questions and answers are archived, and it allows the entire MFiX community to engage. Additionally, follow-up questions should occur in the same thread.
4) Do not ask for a copy of a reference, e.g., a journal article. Do not post copyrighted material.
5) Prior to posting questions regarding MFiX installation or compilation issues, please search the forum to see if your question has already been answered.
6) When posting a question to the forum, provide a complete description of the issue you encountered. It may be useful to provide the following information in your post:
    a. MFiX version you are trying to install or run
    b. Some details on your operating system environment (for Linux: copy and paste the output of the command "uname –a", Linux distribution name and version also)
    c. Your compiler name and version number (e.g. ifort -v will give the version number for Intel Fortran compiler)
    d. Output for your $PATH environment (in csh type echo $PATH)
    e. Your MPI library name and version number (if compilations problem with DMP mode encountered but make sure you can compile and run a simple hello world type MPI program with your current installation) Also please provide hardware details such as number of cores per socket in your system (or send the output for "cat /proc/cpuinfo" and how many cores you are trying to utilize.

Note: Common reasons you may not receive an answer to your request

1. Your question has already been answered and is available in the archive.
2. You did not provide sufficient description of your problem (saying "It doesn't work" is not useful).
3. Your question is outside the scope of the MFiX forum.

1.5 User contribution

If you wish to contribute to the development of MFiX, please contact the MFiX team at admin@mfix.netl.doe.gov. We are looking for simulation results (figures, animations, input files, user-defined subroutines), and new models that could benefit the entire MFiX community. If you have written or know any publication that uses MFiX, please let us know and we will post the citation on the website (https://mfix.netl.doe.gov/publications). Proper credit will be given to all contributors.

These steps have been broken out by platform:

CHAPTER

TWO

GETTING STARTED

This section explains how to install the Anaconda MFiX packages on Linux, Mac, or Windows. There are four steps overall to the MFiX install process:

1. Create an account at https://mfix.netl.doe.gov
2. Download and install Anaconda
3. Create a new Conda environment
4. Install MFiX in the Conda environment

Step 4 requires registration and a login account. MFiX is an open-source multiphase flow solver and is free to download and use. A one-time no-cost registration is required prior to downloading the software. To register, go to https://mfix.netl.doe.gov/register and submit your account information. Your application will be manually reviewed and accepted, so please allow for 2-3 business days for your registration to be processed. You will receive a confirmation email when you can login.

Note: The MFiX solver can still be used from a source tarball (as in previous releases). See *Build from Source (for Developers)* for more information.

For further details, see the install section for your platform.

2.1 Windows Installation

MFiX has been developed and tested on the following Windows versions:

- Windows 10 (64-bit)

If you have an issue running MFiX works on your system, ask for help at *Support Forum*.

2.1.1 Install Anaconda

Download the 64-bit, Python 3, Windows version of Anaconda (~500 MB download) or Miniconda (~50 MB download).

For instance, with the Anaconda installer:

1. Double-click on the downloaded file. (At the time of this writing, it would be `Anaconda3-2020.02-Windows-x86_64.exe`)
2. **Uncheck** "Add Anaconda to my PATH environment variable"
3. **Check** "Register Anaconda as my default Python"

Anaconda is now installed. To verify, open the Start Menu, and run `Anaconda Prompt`↵.

2.1.2 Install MFiX

After installing Anaconda, install MFiX with the following steps:

1. Open the Anaconda Prompt (Start Menu, type anaconda, enter)
2. Browse to MFiX Download (requires registration and login)
3. Copy the conda command.
4. Paste it in the Anaconda Prompt and hit enter.

Note: If you already installed MFiX, you may see the following message. Select `y` to confirm.

```
WARNING: A conda environment already exists
Remove existing environment (y/[n])? y
```

MFiX will be installed in a new conda environment. The process will take a few minutes to complete.

2.1.3 Run MFiX

Open the Anaconda prompt by opening the Start Menu, type "Anaconda Prompt", and selecting the Anaconda Prompt. Then run the following commands:

1. Run `conda activate mfix-20.4`
2. Run `mfix`↵

Your prompt should look something like this:

```
C:\> conda create -n mfix-20.4 mfix=20.4 -c https://mfix.netl.doe.gov/....

C:\> conda env list
# conda environments:
#
base        *  C:\Users\user\anaconda3
mfix-20.4      C:\Users\user\anaconda3\envs\mfix-20.4

C:\> conda activate mfix-20.4

C:\(mfix-20.4)> conda env list
# conda environments:
#
base           C:\Users\user\anaconda3
mfix-20.4   *  C:\Users\user\anaconda3\envs\mfix-20.4

C:\> mfix
```

You are now ready to proceed to the *First Tutorial* and *Tutorials*.

2.1.4 Deactivate Conda Environment

After using MFiX you can just `exit` to leave the Anaconda Prompt. However, if you need to deactivate the `mfix-20.4` conda environment, you can do so with:

```
C:\(mfix-20.4)> deactivate
C:\>
```

This returns to the base conda environment.

2.1.5 Uninstall MFiX

To uninstall MFiX from a conda environment:

```
C:\(mfix-20.4)> conda uninstall mfix
```

To remove the conda environment (if you have the environment activated, deactivate it first):

```
C:\(mfix-20.4)> conda deactivate
C:\> conda env remove -n mfix-20.4
```

To uninstall Anaconda entirely, go to "Add or remove programs" under Windows Control Panel, and uninstall Anaconda.

Note: To learn more about managing conda environments, visit the conda documentation .

2.2 MFiX Installation on Mac

As of this release, MFiX has been tested on the latest version of MacOS at the time of this release, 10.14 (Mojave). Earlier of MacOS are likely to work. If you have an issue running MFiX on your Mac, ask for help at *Support Forum*.

2.2.1 Install Anaconda

Download the Python 3, Mac version of Anaconda (~500 MB download) or Miniconda (~50 MB download). Either the Graphical Installer or Command Line Installer would work.

For instance, with the Anaconda Graphical Installer:

1. Open up Finder and navigate to the Download directory.
2. Double-click the downloaded file. (At the time of this writing, it would be `Anaconda3-2019.03-MacOSX-x86_64.pkg` for Graphical Installer).
3. Follow the installation guide, using the default settings.

2.2.2 Install MFiX

After installing Anaconda, install MFiX:

1. Open a Terminal (Type "⌘-Spacebar", type "terminal")
2. Browse to MFiX Download (requires registration and login)
3. Copy the conda command
4. Paste it in the Terminal.

Note: If you already installed MFiX, you may see the following message. Select `y` to confirm.

```
WARNING: A conda environment already exists
Remove existing environment (y/[n])? y
```

MFiX will be installed in a new conda environment. The process will take a few minutes to complete.

2.2.3 Install Solver Build Dependencies

Build dependencies are needed for building a *Custom Solver*.

Building the MFiX solver requires:

- Fortran 2003 compiler (GFortran 4.8 or later)
- CMake
- For DMP support, an MPI implementation (such as OpenMPI)

For building with other compilers, or for building with DMP, see *Building Custom Interactive Solver*.

Homebrew is the recommended way to install MFiX build dependencies. Go to the Homebrew website and follow the installation instructions.

Once Homebrew is installed, install the build dependencies:

1. Open a Terminal (Type "⌘-Spacebar", type "terminal")
2. Run `brew install gcc cmake open-mpi`

2.2.4 Run MFiX

Open Terminal (if not open already), and run the following commands:

1. Run `conda activate mfix-20.4`
2. Run `mfix`↵

Note: If you are using a shell different from bash, switch to `bash` before running `conda activate`

Your prompt should look something like this:

```
> conda env list
# conda environments:
#
base        *  /Users/user/anaconda3
mfix-20.4      /Users/user/anaconda3/envs/mfix-20.4

> conda activate mfix-20.4
(mfix-20.4) > conda env list
# conda environments:
#
base           /Users/user/anaconda3
mfix-20.4   *  /Users/user/anaconda3/envs/mfix-20.4

(mfix-20.4) > mfix
```

You are now ready to proceed to the *First Tutorial* and *Tutorials*.

Note: Activating a conda environment sets certain environment variables such as PATH in the current shell. It does not create a new shell session.

You will need to activate the environment every time before running MFiX.

2.2.5 Deactivate Conda Environment

After using MFiX you can just `exit` to leave the Terminal session. However, if you need to deactivate the `mfix-20.4` conda environment, you can do so with:

```
(mfix-20.4)> deactivate
>
```

This returns to the base conda environment.

2.2.6 Uninstall MFiX

To uninstall MFiX from a conda environment:

```
(mfix-20.4)> conda uninstall mfix
```

To remove the conda environment (if you have the environment activated, deactivate it first):

```
(mfix-20.4)> conda deactivate
> conda env remove -n mfix-20.4
```

To uninstall Anaconda entirely, remove the Anaconda directory. By default, `~/anaconda3` in your home directory. From the Terminal:

```
> cd ~
> rm -rf anaconda3/
```

Note: To learn more about managing conda environments, visit the conda documentation .

2.3 Linux Installation

MFiX has been developed and tested on the following linux distributions:

- Ubuntu 18.10
- CentOS 7

Other recent releases of Linux are likely to work. If you have an issue running MFiX works your distro, ask for help at *Support Forum*.

2.3.1 Install Anaconda

Download the 64-bit, Python 3, Linux version of Anaconda (~500 MB download) or Miniconda (~50 MB download).

For instance, with the Anaconda installer:

1. Open a terminal
2. `cd` to the directory of the downloaded installer (for example `~/Downloads`)
3. Run the downloaded installer (At the time of this writing, it would be `sh Anaconda3-2019.03-Linux-x86_64.sh`↵.
4. You will be asked to review the License, hit enter
5. Hit the "space bar" until you get to the end of the license agreement.
6. Agree to the license by typing `yes` and hitting enter
7. When prompted for an installation location, hit enter to use the default.
8. When prompted to add the conda bin directory to `PATH`, enter `yes` and hit enter.

Anaconda is now installed. In order for the changes to take effect, close the terminal and open a new one. Verify that your `PATH` was updated by running `conda` in the new terminal.

2.3.2 Install MFiX

After installing Anaconda, install MFiX with the following steps:

1. Open the Terminal application for your Linux desktop
2. Browse to MFiX Download (requires registration and login)
3. Copy the conda command.
4. Paste it in the Terminal.

Note: If you already installed MFiX, you may see the following message. Select `y` to confirm.

```
WARNING: A conda environment already exists
Remove existing environment (y/[n])? y
```

MFiX will be installed in a new conda environment. The process will take a few minutes to complete.

2.3.3 Install Solver Build Dependencies (optional)

Build dependencies are needed for building a *custom interactive solver*. If you only use the default solver, you can skip this step.

Building the MFiX solver requires:

- Fortran 2003 compiler (GFortran 4.8 or later)
- GNU Make
- CMake
- For DMP support, an MPI implementation (such as OpenMPI)

For building with other compilers, or for building with DMP, see *Building Custom Interactive Solver*.

Installing GCC and Make through your system package manager (such as `apt` or `yum`) is recommended. If you don't have root access on your system, you alternatively could install GCC and GNU Make through Anaconda with:

1. Open a terminal
2. Run `conda activate mfix-20.3` to activate the mfix environment
3. Run `conda install gcc make` to install GCC and GNU Make

2.3.4 Install DMP/MPI Solver Build Dependencies

To build and run MFiX with DMP, you will need an MPI implementation installed, such as OpenMPI.

To install OpenMPI on Ubuntu/Debian derived distributions:

```
$ sudo apt install openmpi-bin libopenmpi-dev
```

To install OpenMPI on Fedora/RHEL/Centos derived distributions:

```
$ sudo yum install openmpi-devel
$ module load mpi
```

2.3.5 Run MFiX in Conda Environment

To run MFiX:

1. Open a terminal
2. Run `conda activate mfix-20.4`
3. Run `mfix` to start MFiX

Note: If you are using a shell different from bash, switch to `bash` before activating the environment.

Your prompt should look something like this:

```
> conda env create -n mfix-20.4 mfix/mfix-20.4-linux64

> conda env list
# conda environments:
#
```

(continues on next page)

(continued from previous page)

```
base         *  /home/user/anaconda3
mfix-20.4       /home/user/anaconda3/envs/mfix-20.4

> conda activate mfix-20.4
(mfix-20.4) > conda env list
# conda environments:
#
base            /home/user/anaconda3
mfix-20.4    *  /home/user/anaconda3/envs/mfix-20.4

(mfix-20.4) > mfix
```

You are now ready to proceed to the *First Tutorial* and *Tutorials*.

Note: Activating a conda environment sets certain environment variables such as PATH in the current shell. It does not create a new shell session.

You will need to activate the environment every time before running MFiX.

2.3.6 Deactivate Conda Environment

After using MFiX you can just `exit` to leave the terminal session. However, if you need to deactivate the `mfix-20.4` conda environment, you can do so with:

```
(mfix-20.4)> deactivate
>
```

This returns to the base conda environment.

2.3.7 Uninstall MFiX

To uninstall MFiX from a conda environment:

```
(mfix-20.4)> conda uninstall mfix
```

To remove the conda environment (if you have the environment activated, deactivate it first):

```
(mfix-20.4)> conda deactivate
> conda env remove -n mfix-20.4
```

To uninstall Anaconda entirely, remove the Anaconda directory. By default, `~/anaconda3` in your home directory. From the terminal:

```
> cd ~
> rm -rf anaconda3/
```

Note: To learn more about managing conda environments, visit the conda documentation .

2.4 Pip Installation

As of MFiX 20.1, Python wheels are available for installation with Pip. (Installation via the Conda packages will continue to be supported.)

The Pip package includes the MFiX GUI and the MFiX solver source. It does not include a local copy of the MFiX documentation or a binary version of the solver. If you install MFiX with Pip, you will need to *build a solver yourself*.

Python 3.7 is required, because at the time of this writing, VTK packages for Python 3.8 are not available from PyPI.

The Pip package should work on any platform, so long you have Python 3.7, CMake, GNU Make, and a Fortran compiler.

2.4.1 Install MFiX

Make sure Python 3.7 or later is installed, either from your system package manager or from Python.org.

1. Browse to MFiX Download (requires registration and login)
2. Go to the Source tab
3. Download either the source tarball or the wheel (`*.whl`) file.
4. Install the downloaded file with pip.

```
> python3 -m pip install mfix-20.4.tar.gz
#  <or>
> python3 -m pip install mfix-20.4-py3-none-any.whl
```

MFiX runs the same whether installed from a source tarball or wheel. The source tarball is better for looking at the source. The wheel file installs slightly faster.

Warning: The install command assumes your intended Python command is `python`. If `python` on your system is Python 2.x or for any other reason is not your intended Python version, then edit the pasted command in your shell to start with `python3`, `python3.7`, or whatever the full path is to your intended Python distribution to remove any uncertainty.

2.4.2 Install Solver Build Dependencies (optional)

The MFiX wheel is a pure Python package, and does not include a solver binary. To actually run the solver, you will need:

- Fortran 2003 compiler (GFortran 4.8 or later recommended)
- GNU Make
- CMake
- For DMP support, an MPI implementation (such as OpenMPI)

Installing the compiler, CMake, and Make through your system package manager (such as `apt` or `yum`) is recommended.

Note: If your system package manager has an old version of CMake, you can install a recent version with `python -m pip install cmake`.

On Windows, the Scoop tool can be used to install build dependencies for MFiX.

```
> scoop install gcc  # install full MinGW installation, including MinGW Make
> scoop install cmake
```

2.4.3 Run MFiX

To run MFiX:

1. Open a terminal
2. Run `mfix`⏎ to start MFiX

You are now ready to proceed to the *First Tutorial* and *Tutorials*.

2.4.4 Uninstall MFiX

To uninstall MFiX:

```
> pip uninstall mfix
```

CHAPTER

THREE

TUTORIALS

These are detailed step-by-step tutorials for setting up projects in MFiX from scratch.

3.1 First Tutorial

This section illustrates how to launch the GUI, open a project, run it, and visualize the results.

This section assumes MFiX is already installed. To install MFiX, see the *Setup Guide*.

It is also assumed that the terminal shell on Linux and Mac is `bash`. If this is not the case, type `bash` at the prompt before activating the environment.

Everything in this section applies to all platforms (Linux, macOS, Windows) unless otherwise noted.

3.1.1 Starting the MFiX GUI

Starting the GUI requires first to activate the environment (to select a given installed version of MFiX):

On Windows, open the Anaconda Prompt and activate the *mfix environment*

On Linux open a terminal and activate the *mfix environment* from `bash`.

On macOS open a terminal and activate the *mfix environment* from `bash`.

Next, type `mfix` at the prompt. The two steps are shown below for each Operating System:

```
# Windows
conda activate mfix-20.4
(mfix-20.4) C:\> mfix

# Linux or Mac
conda activate mfix-20.4
(mfix-20.4) > mfix
```

If this is the first time opening the GUI, the main menu will automatically open. Otherwise, the previous project will automatically open.

3.1.2 Creating a Project

- Click (*New*)

A list of project templates is displayed.

The templates are tagged with the following icons. (Some templates have more than one icon).

Icon	Description
	Single phase Model
	Two Fluid Model (TFM)
	Particle in Cell Model (PIC)
	Discrete Element Model (DEM)
	Hybrid Model (TFM + DEM)
	Cartesian cut-cell (complex) geometry
	Chemistry

Note: The blank template is the default, but it won't run without customization. It requires a pressure outflow, among other things.

- Filter the templates by de-selecting the (single phase), and (chemistry) icons
- Double-click on `hopper_3d (3D DEM granular flow hopper)` to create a new project
- Enter a project name (or keep the existing name) and browse to a location for the new project.
- Click Ok

A new project directory will be created in the selected directory, with the name being the project name. Here, a DEM Hopper simulation setup has been loaded and is ready to run. You should see the model geometry in the "model" window (top-right).

3.1.3 Running the Solver

- Click the (Start) button to open the *Run Dialog*.
- Click Ok to use the default mfixsolver: `[default]/mfixsolver`

The solver will begin to run, with output displayed in the *Terminal window*. There is a progress bar along the bottom of the screen where it says *MFiX running: elapsed time*.

3.1.4 View results

Results can be viewed, and plotted, while the simulation is running.

- Select the `VTK` results tab. The visibility and representation of the `*.vtk` files can be controlled with the menu on the side.
- Change frames with the , , , and buttons
- Click the button to play the available vtk files.
- Change the playback speed under the section on the sidebar.

3.1.5 Pausing/Unpausing

- Click the (pause) button to pause the solver.

  `MFiX paused` will be displayed in the *Terminal window*. The solver process still exists, but is waiting for you to unpause it.

  While the solver is paused, you can change the project. For instance, on the *Run* Pane you could change `stop time` to another value to have the solver run for a different amount of time.
- Click the (run) button to unpause the solver and continue running.

3.1.6 Stopping/Resuming

- Click the (stop) button to stop the solver.

`MFiX stopped` will be displayed in the *Terminal window*. The solver process has ended, with the output files still in the project directory.

You can then resume the solver by:

- Clicking the (run) button to open the *Run Dialog*.
- Check `Restart` and select `Resume` from the drop down list.
- Click Ok to use the default mfixsolver.

The solver process will start and continue from the point at which the output files were last written.

3.2 Two Dimensional Fluid Bed, Two Fluid Model (TFM)

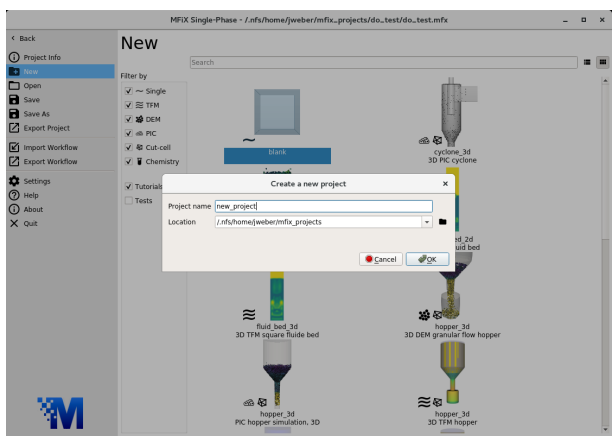
This tutorial shows how to create a two dimensional fluidized bed simulation using the two fluid model. The model setup is:

Property	Value
geometry	10 cm x 30 cm
mesh	20 x 60
solid diameter	200 microns (200×10^{-6} m)
solid density	2500 kg/m^2
gas velocity	0.25 m/s
temperature	298 K
pressure	101325 Pa

3.2.1 Create a new project

- (`fig_new_project`): On the main menu select `New project`
- Create a new project by double-clicking on "Blank" template.
- Enter a project name and browse to a location for the new project.
- When prompted to enable SMS workflow, answer *No*, we will use the standard workflow for this tutorial.

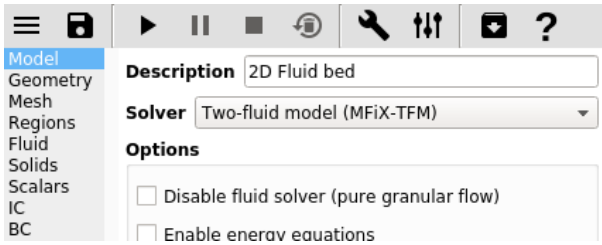
Note: A new project directory will be created in the location directory, with the name being the project name.



3.2.2 Select model parameters

(`fig_model_param`): On the `Model` pane:

- Enter a descriptive text in the `Description` field
- Select "Two-Fluid Model (MFiX-TFM)" in the `Solver` drop-down menu.

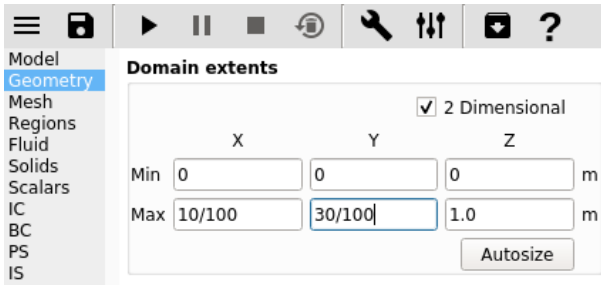


3.2.3 Enter the geometry

On the `Geometry` pane:

- Select the `2 Dimensional` checkbox
- Enter `10/100` meters for the maximum x value
- Enter `30/100` meters for the maximum y value

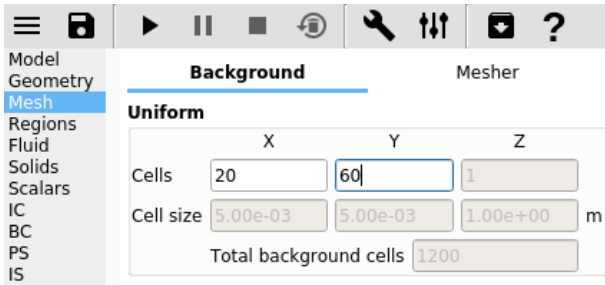
Note: We could have entered 0.1 and 0.3 to define the domain extents, but this example shows that simple mathematical expressions (+,-,*,/) are allowed.



3.2.4 Enter the mesh

On the `Mesh` pane, `Background` sub-pane:

- Enter `20` for the x cell value
- Enter `60` for the y cell value

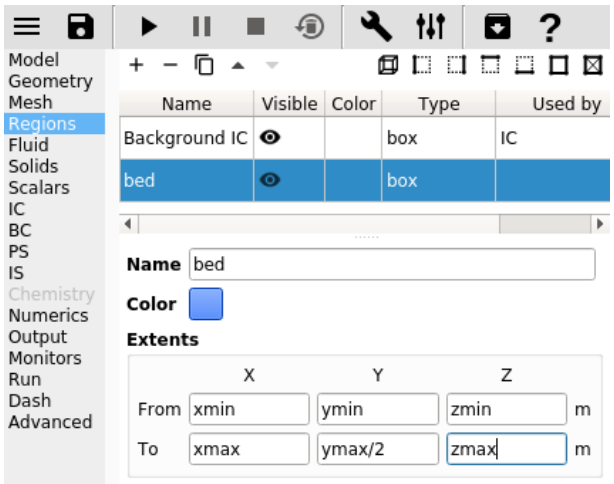


3.2.5 Create regions for initial and boundary condition specification

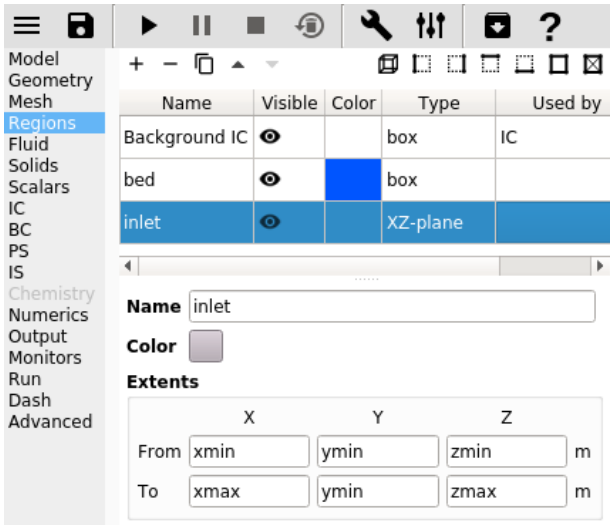
Select the `Regions` pane. By default, a region that covers the entire domain is already defined. This is typically used to initialize the flow field and visualize the results.

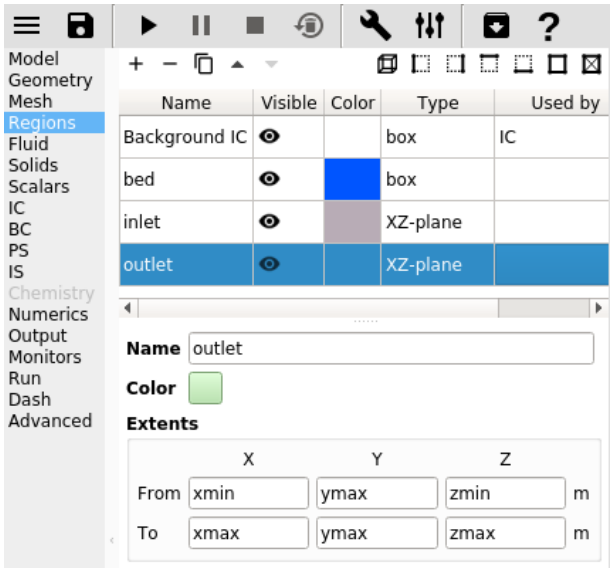
- click the button to create a new region to be used for the bed initial condition.
- Enter a name for the region in the `Name` field ("bed")
- Change the color by pressing the `Color` button
- Enter `xmin` or `min` in the `From X` field
- Enter `xmax` or `max` in the `To X` field
- Enter `ymin` or `min` in the `From Y` field
- Enter `ymax/2` or `max/2` in the `To Y` field
- Enter `zmin` or `min` in the `From Z` field
- Enter `zmax` or `max` in the `To Z` field

Note: Here we could have entered numerical values for the coordinates, but it is recommended to use parameters (xmin, xmax etc.) when possible. These parameters will update automatically if the "Domain Extents" change.



- Click the button to create a new region with the `From` and `To` fields already filled out for a region at the bottom of the domain, to be used by the gas inlet boundary condition. `From Y` and `To Y` should equal `ymin`, defining an XZ-plane.
- Enter a name for the region in the `Name` field ("inlet")
- Click the button to create a new region with the `From` and `To` fields already filled out for a region at the top of the domain, to be used by the pressure outlet boundary condition. `From Y` and `To Y` should equal `ymax`, defining an XZ-plane.
- Enter a name for the region in the `Name` field ("outlet")

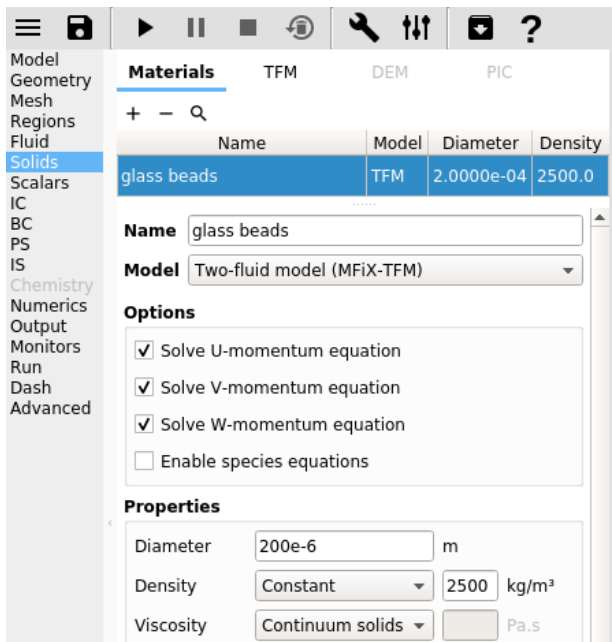

Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Name
Visible
Color
Type
Used by
Background IC
box
IC
bed
box
inlet
XZ-plane
Name
inlet
Color
Extents
X
Y
Z
From
xmin
ymin
zmin
m
To
xmax
ymin
zmax
m


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Name
Visible
Color
Type
Used by
Background IC
box
IC
bed
box
inlet
XZ-plane
outlet
XZ-plane
Name
outlet
Color
Extents
X
Y
Z
From
xmin
ymax
zmin
m
To
xmax
ymax
zmax
m

3.2.6 Create a solid

On the `Solids` pane:

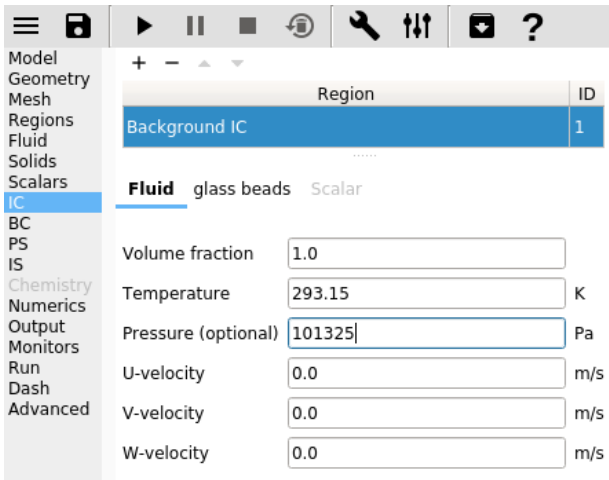
- Click the button to create a new solid
- Accept the radial distribution setting (Carnahan-Starling)
- Enter a descriptive name in the `Name` field ("glass beads")
- Keep the model as "Two-Fluid Model (MFiX-TFM)")
- Enter the particle diameter of `200e-6` m in the `Diameter` field
- Enter the particle density of `2500` kg/m^2 in the `Density` field

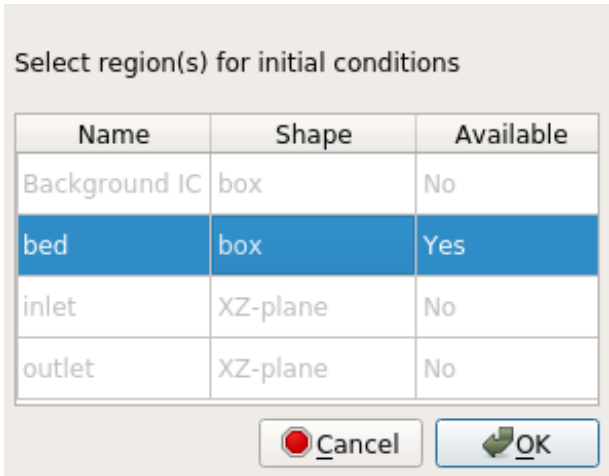


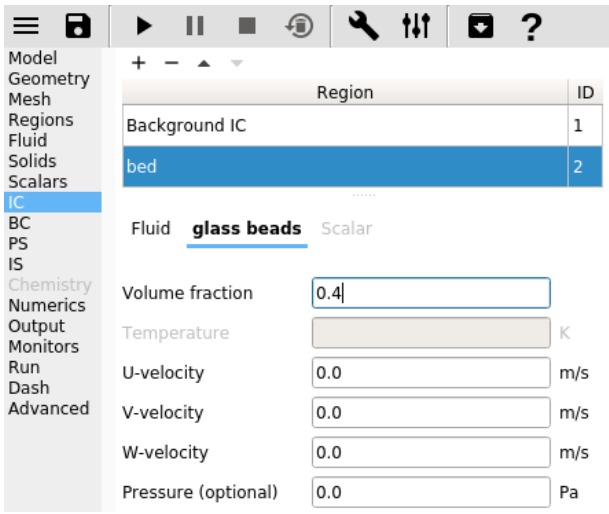
3.2.7 Create Initial Conditions

On the `Initial conditions` pane:

- Select the already populated "Background IC" from the region list. This will initialize the entire flow field with air.
- Enter `101325` Pa in the `Pressure (optional)` field
- Create a new Initial Condition by pressing the button
- Select the bed region created previously for the bed Initial Condition ("bed" region) and click the `OK` button.
- Select the solid (named previously as "glass beads") sub-pane and enter a volume fraction of `0.4` in the `Volume Fraction` field. This will fill the bottom half of the domain with glass beads.


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Region
ID
Background IC
1
Fluid
glass beads
Scalar
Volume fraction
1.0
Temperature
293.15
K
Pressure (optional)
101325
Pa
U-velocity
0.0
m/s
V-velocity
0.0
m/s
W-velocity
0.0
m/s

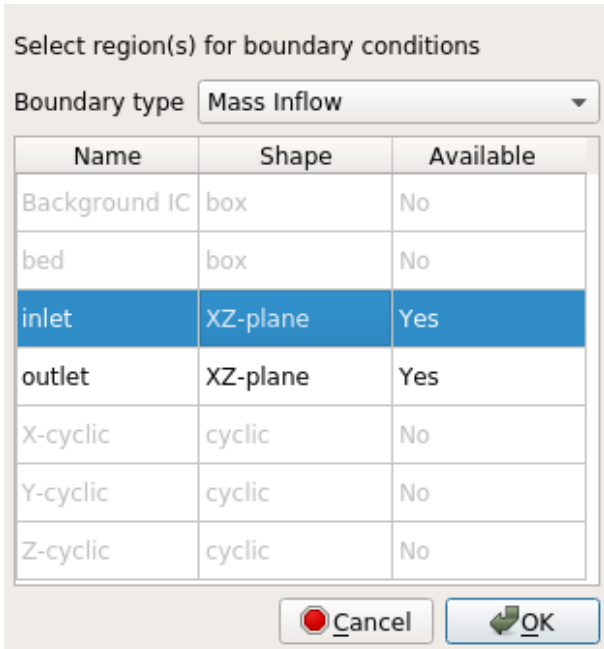

Select region(s) for initial conditions
Name
Shape
Available
Background IC
box
No
bed
box
Yes
inlet
XZ-plane
No
outlet
XZ-plane
No
Cancel
OK


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Region
ID
Background IC
1
bed
2
Fluid
glass beads
Scalar
Volume fraction
0.4
Temperature
K
U-velocity
0.0
m/s
V-velocity
0.0
m/s
W-velocity
0.0
m/s
Pressure (optional)
0.0
Pa

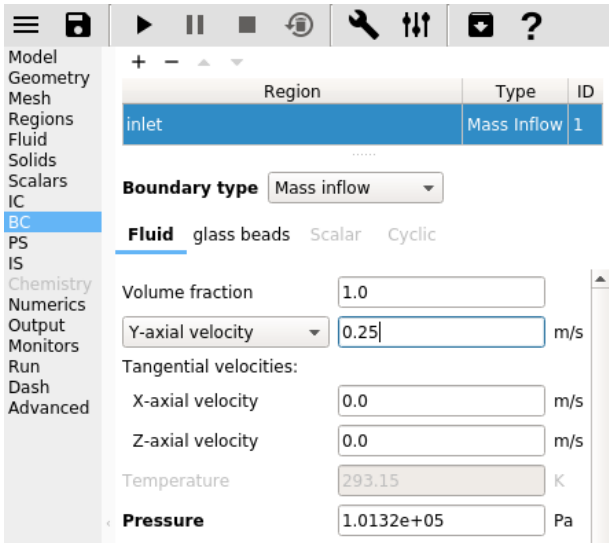
3.2.8 Create Boundary Conditions

On the `Boundary conditions` pane:

- Create a new Boundary condition by clicking the button
- On the `Select Region` dialog, select "Mass Inflow" from the `Boundary type` drop-down menu
- Select the "inlet" region and click `OK`



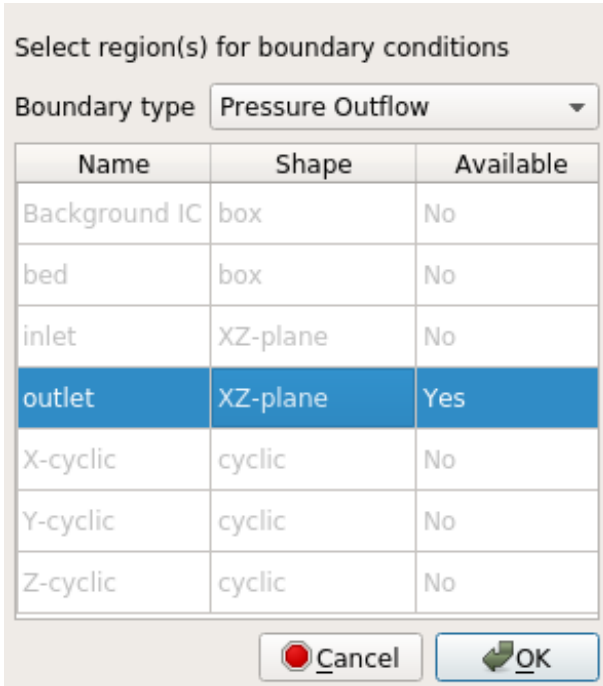
- On the "Fluid" sub-pane, enter a velocity in the `Y-axial velocity` field of "0.25" m/s



- Create another Boundary condition by clicking the button
- On the `Select Region` dialog, select "Pressure Outflow" from the `Boundary type` combo-box

- Select the "outlet" region and click `OK`

Note: The default pressure is already set to 101325 Pa, no changes need to be made to the outlet boundary condition.



Note: By default, boundaries that are left undefined (here the left and right planes) will behave as No-Slip walls.

3.2.9 Change numeric parameters

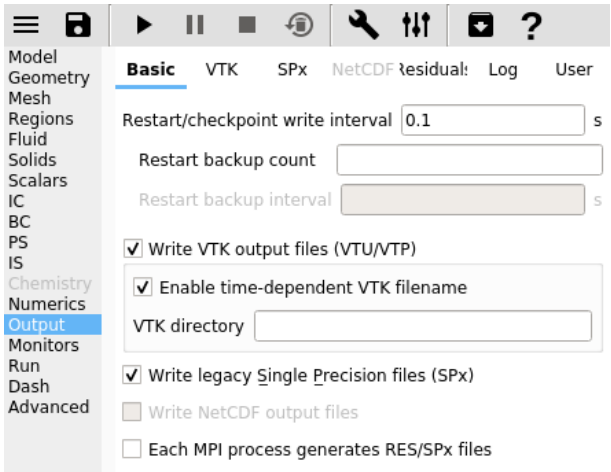
On the `Numerics` pane, `Residuals` sub-pane:

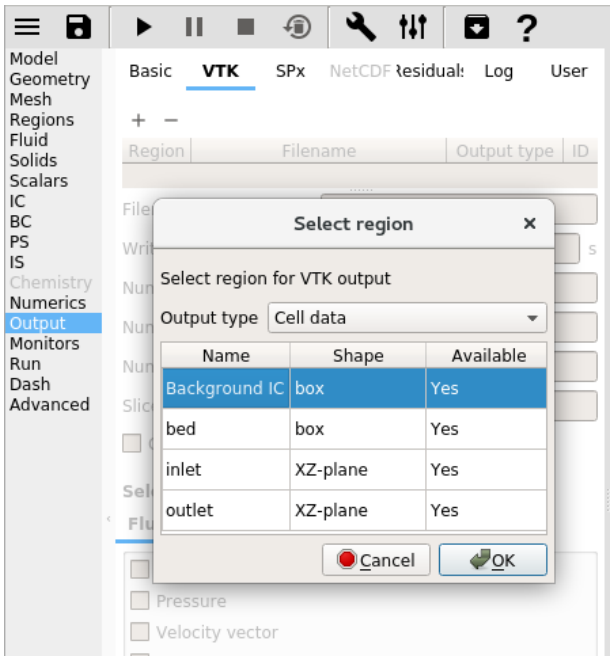
- Enter `0` in the `Fluid Normalization` field.

3.2.10 Select output options

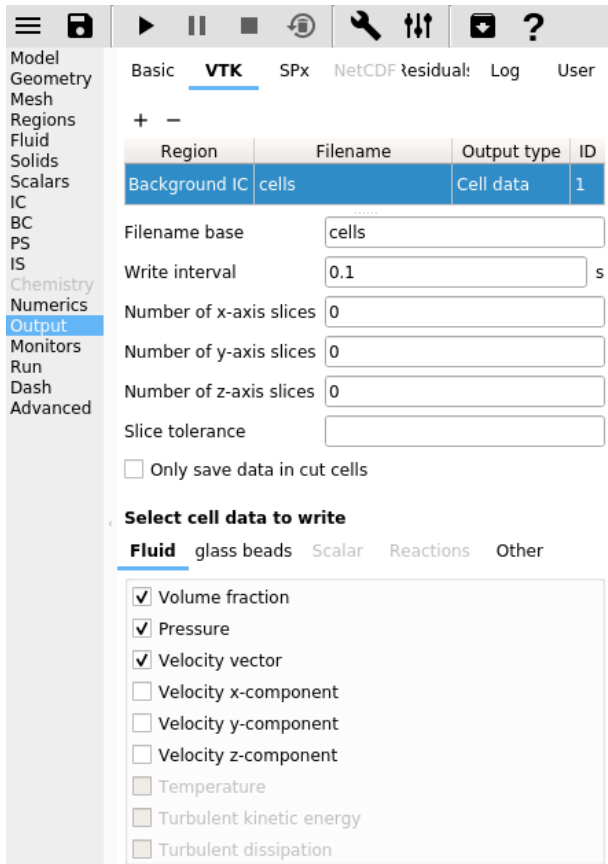
On the `Output` pane:

- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` checkbox
- Select the `VTK` sub-pane
- Create a new output by clicking the button
- Select the "Background IC" region from the list to save all the cell data
- Click `OK` to create the output
- Enter a base name for the `*.vtu` files in the `Filename base` field
- Change the `Write interval` to `0.1` seconds


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Basic
VTK
SPx
NetCDF
Residuals
Log
User
Restart/checkpoint write interval
0.1
s
Restart backup count
Restart backup interval
s
Write VTK output files (VTU/VTP)
Enable time-dependent VTK filename
VTK directory
Write legacy Single Precision files (SPx)
Write NetCDF output files
Each MPI process generates RES/SPx files


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Basic
VTK
SPx
NetCDF
Residuals
Log
User
Region
Filename
Output type
ID
Select region
Select region for VTK output
Output type
Cell data
Name
Shape
Available
Background IC
box
Yes
bed
box
Yes
inlet
XZ-plane
Yes
outlet
XZ-plane
Yes
Cancel
OK
Pressure
Velocity vector

- Select the `Volume fraction`, `Pressure`, and `Velocity vector` check-boxes on the `Fluid` sub-sub-pane



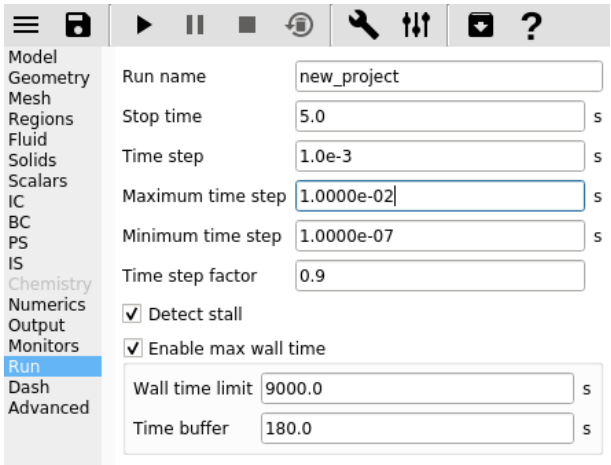
3.2.11 Change run parameters

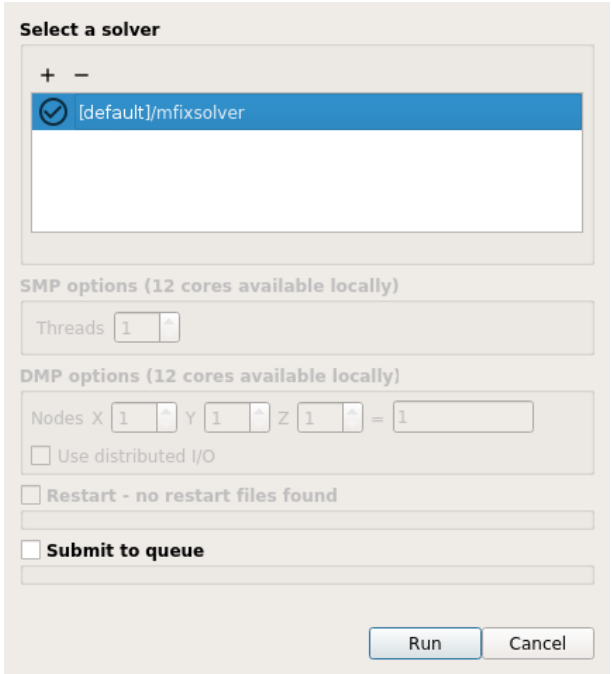
On the `Run` pane:

- Change the `Time step` to `1e-3` seconds
- Change the `Maximum time step` to `1e-2` seconds

3.2.12 Run the project

- Save project by clicking the button
- Run the project by clicking the button
- On the `Run Solver` dialog, select the executable from the combo-box
- Click the `Run` button to actually start the simulation

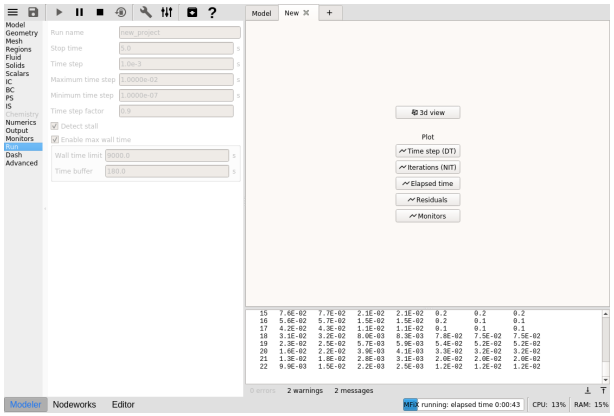

Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Run name
new_project
Stop time
5.0
s
Time step
1.0e-3
s
Maximum time step
1.0000e-02
s
Minimum time step
1.0000e-07
s
Time step factor
0.9
Detect stall
Enable max wall time
Wall time limit
9000.0
s
Time buffer
180.0
s


Select a solver
[default]/mfixsolver
SMP options (12 cores available locally)
Threads
1
DMP options (12 cores available locally)
Nodes X
1
Y
1
Z
1
=
1
Use distributed I/O
Restart - no restart files found
Submit to queue
Run
Cancel

3.2.13 View results

Results can be viewed, and plotted, while the simulation is running.

- Create a new visualization tab by pressing the next to the *Model* tab
- Select an item to view, such as plotting the time step (dt) or click the `3D view` button to view the vtk output files.

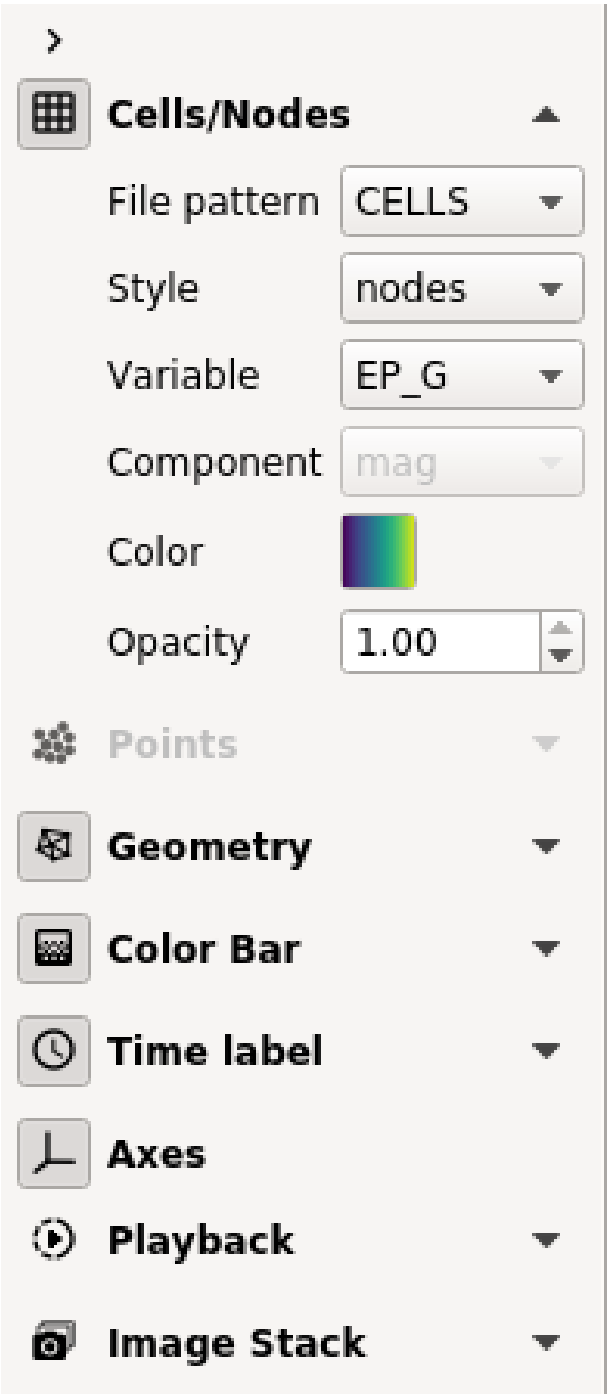


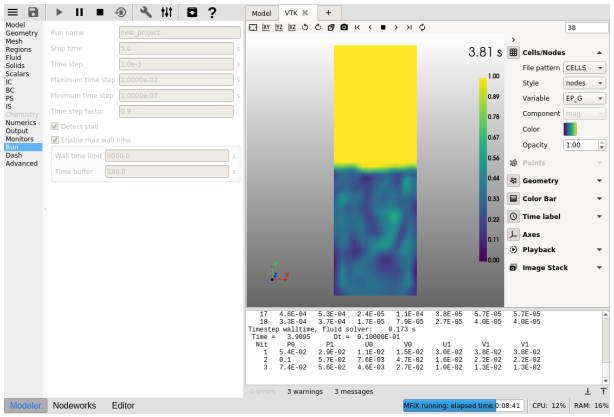
- On the `VTK` results tab, the visibility and representation of the `*.vtk` files can be controlled with the menu on the side.
- Change frames with the , , , and buttons
- Click the button to play the available vtk files.
- Change the playback speed under the section on the sidebar.

3.2.14 Increase grid resolution

For this simulation, increasing the grid resolution will better resolve the bubbles (at the expense of a slower simulation time). To do this:

- Click the button and delete all simulation files.
- On the `Mesh` pane, change the `X Cells` to 80 and the `Y Cells` to 180
- On the `Output` pane, `VTK` sub-pane, change the write interval to `0.01`
- Save and run the project.


Cells/Nodes
File pattern
CELLS
Style
nodes
Variable
EP_G
Component
mag
Color
Opacity
1.00
Points
Geometry
Color Bar
Time label
Axes
Playback
Image Stack



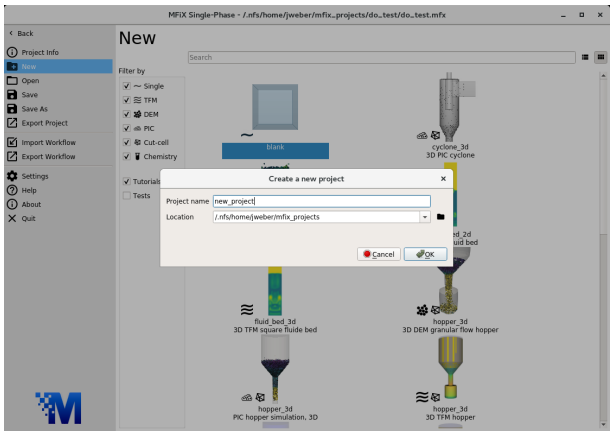
3.3 Two Dimensional Fluid Bed, Discrete Element Model (DEM)

This tutorial shows how to create a two dimensional fluidized bed simulation using the Discrete Element Model. The model setup is:

Property	Value
geometry	15 cm x 90 cm x 0.4 cm
mesh	15 x 45 x 1
solid diameter	4000 microns (4000×10^{-6} m)
solid density	2700 kg/m^2
gas velocity	42.0 m/s
temperature	298 K
pressure	101325 Pa

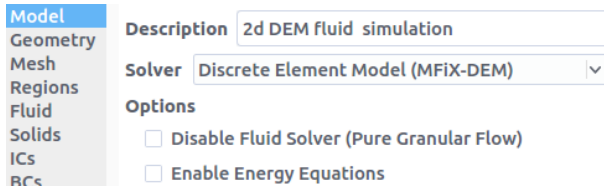
3.3.1 Create a new project

- On the main menu, select `New project`
- Create a new project by double-clicking on "Blank" template.
- Enter a project name and browse to a location for the new project.
- When prompted to enable SMS workflow, answer *No*, we will use the standard workflow for this tutorial.



3.3.2 Select model parameters

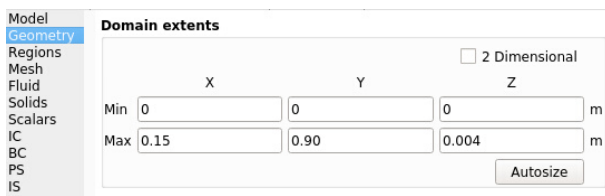
- On the `Model` pane, enter a descriptive text in the `Description` field
- Select "Discrete Element Model (MFiX-DEM)" in the `Solver` drop-down menu.



3.3.3 Enter the geometry

On the `Geometry` pane:

- Enter `0.15` meters for the maximum x value
- Enter `0.90` meters for the maximum y value
- Enter `0.004` meters for the maximum z value

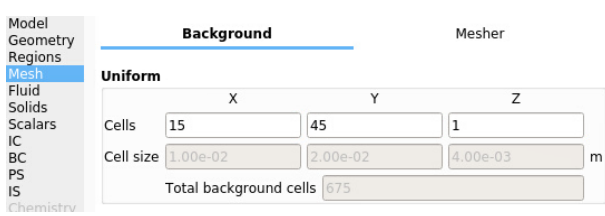


3.3.4 Enter the mesh

On the `Mesh` pane, `Background` sub-pane:

- Enter `15` for the x cell value
- Enter `45` for the y cell value
- Enter `1` for the z cell value

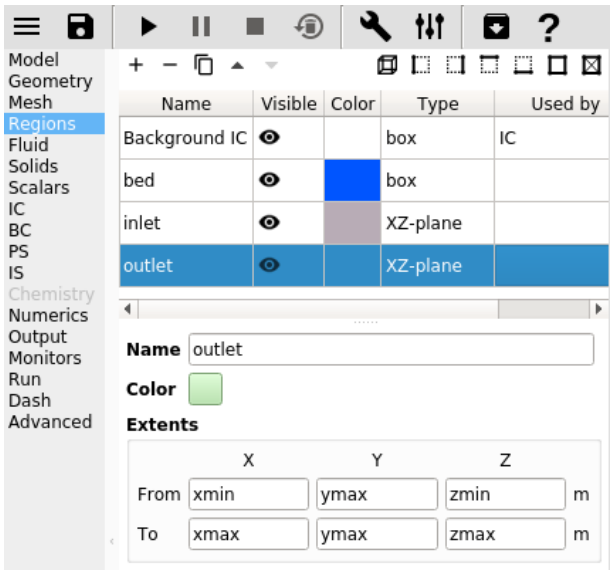
Note: Since there is only one cell in the Z direction, this model is effectively a 2D simulation.



3.3.5 Create regions for initial and boundary condition specification

On the `Regions` pane:

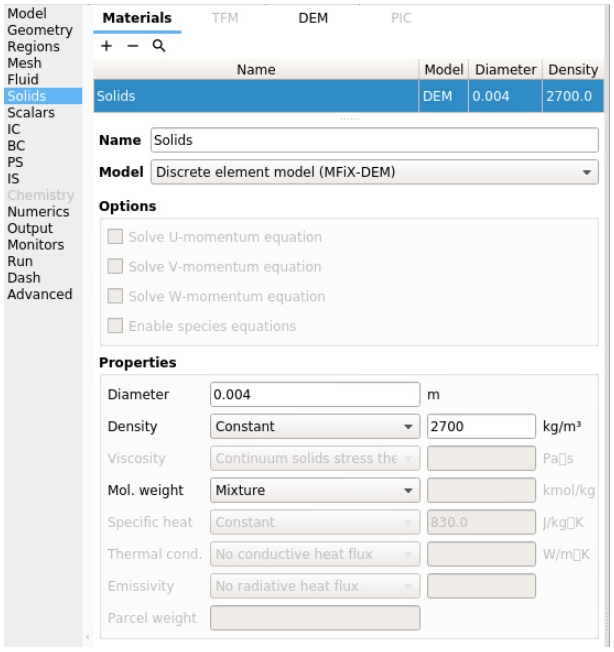
- click the button to create a new region that covers the entire domain to be used for the bed initial condition.
- Enter a name for the region in the `Name` field ("bed")
- Change the `To Y` field to be "ymax/2"
- Click the button to create a new region to be used by the gas inlet boundary condition.
- Enter a name for the region in the `Name` field ("inlet")
- Enter `0.07` in the `From X` field and `0.08` in the `To Y` field.
- Click the button to create a new region to be used by pressure outlet boundary condition.
- Enter a name for the region in the `Name` field ("outlet")

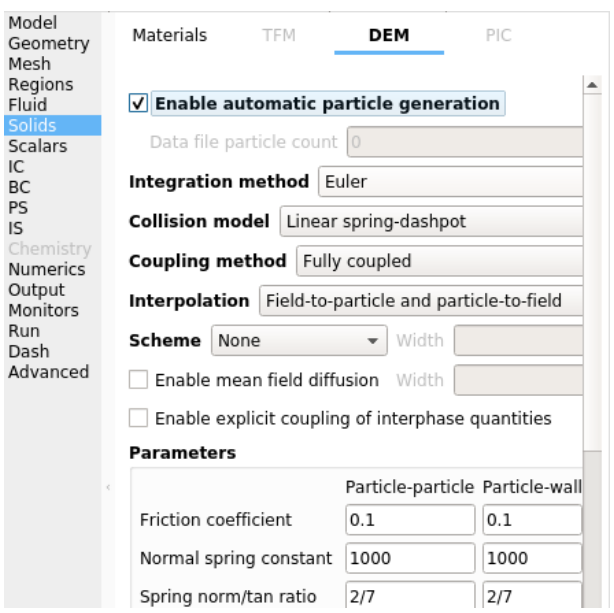


3.3.6 Create a solid

On the `Solids` pane, `Materials` sub-pane:

- Click the button to create a new solid
- Enter a descriptive name in the `Name` field ("solids")
- Enter the particle diameter of `0.004` m in the `Diameter` field
- Enter the particle density of `2700` kg/m^2 in the `Density` field
- Select the `Solids` pane, `DEM` sub-pane
- Check the `Enable automatic particle generation` checkbox, so that the bed Initial Condition, defined later, will be filled with solids

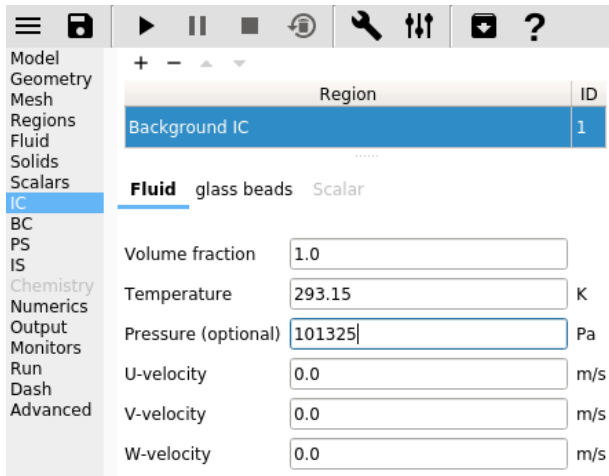

Model
Geometry
Regions
Mesh
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Materials
TFM
DEM
PIC
Name
Model
Diameter
Density
Solids
DEM
0.004
2700.0
Name
Solids
Model
Discrete element model (MFiX-DEM)
Options
Solve U-momentum equation
Solve V-momentum equation
Solve W-momentum equation
Enable species equations
Properties
Diameter
0.004
m
Density
Constant
2700
kg/m³
Viscosity
Continuum solids stress the
Pa□s
Mol. weight
Mixture
kmol/kg
Specific heat
Constant
830.0
J/kg□K
Thermal cond.
No conductive heat flux
W/m□K
Emissivity
No radiative heat flux
Parcel weight


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Materials
TFM
DEM
PIC
Enable automatic particle generation
Data file particle count
0
Integration method
Euler
Collision model
Linear spring-dashpot
Coupling method
Fully coupled
Interpolation
Field-to-particle and particle-to-field
Scheme
None
Width
Enable mean field diffusion
Width
Enable explicit coupling of interphase quantities
Parameters
Particle-particle
Particle-wall
Friction coefficient
0.1
0.1
Normal spring constant
1000
1000
Spring norm/tan ratio
2/7
2/7

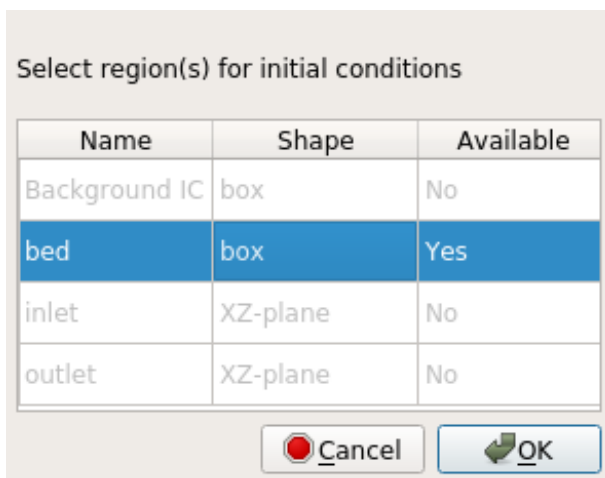
3.3.7 Create Initial Conditions

On the `Initial conditions` pane:

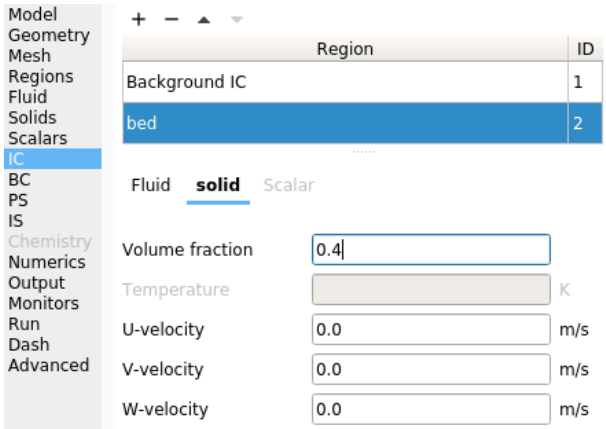
- Select the already populated "Background IC" from the region list. This will initialize the entire flow field with air.
- Enter `101325` Pa in the `Pressure (optional)` field



- Create a new Initial Condition by pressing the button
- Select the bed region created previously for the bed Initial Condition ("bed" region) and click the `OK` button.



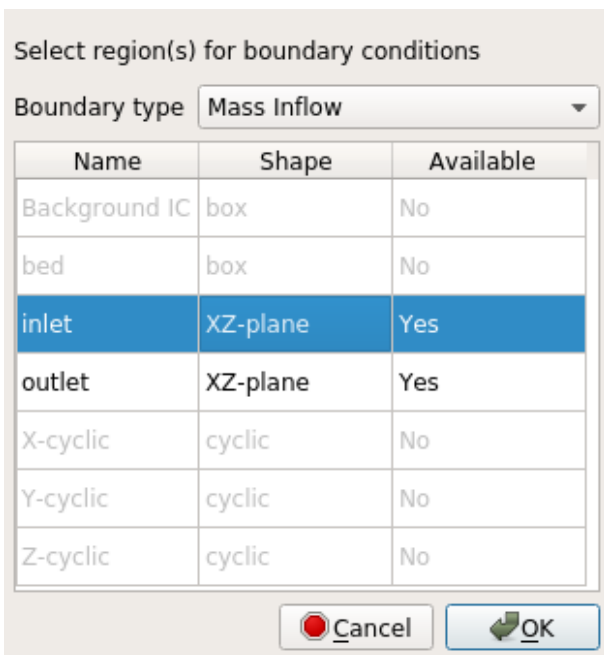
- Select the solid (named previously as "solid") sub-pane and enter a volume fraction of `0.4` in the `Volume Fraction` field. This will fill the bottom half of the domain with solids.
- Note the estimated number of particles and inventory (around 3,200 particles or 0.3 kg). When running DEM simulations on a single core machine, it is recommended to stay below 100,000 particles to get reasonable run times.



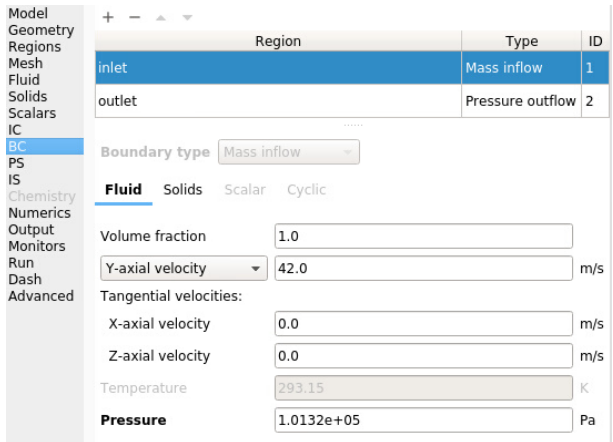
3.3.8 Create Boundary Conditions

On the `Boundary conditions` pane:

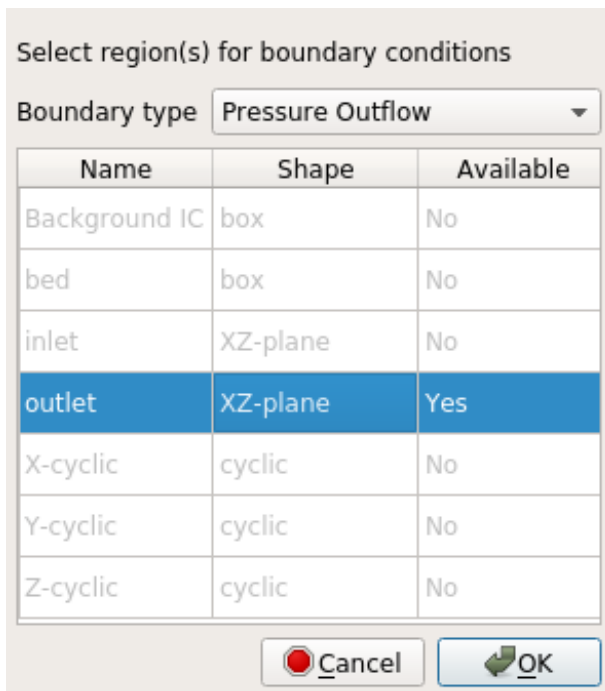
- Create a new Boundary condition by clicking the button
- On the `Select Region` dialog, select "Mass Inflow" from the `Boundary type` drop-down menu
- Select the "inlet" region and click `OK`



- On the "Fluid" sub-pane, enter a velocity in the `Y-axial velocity` field of "42" m/s
- Create another Boundary condition by clicking the button
- On the `Select Region` dialog, select "Pressure Outflow" from the `Boundary type` combo-box
- Select the "outlet" region and click `OK`



Note: The default pressure is already set to 101325 Pa, no changes need to be made to the outlet boundary condition.

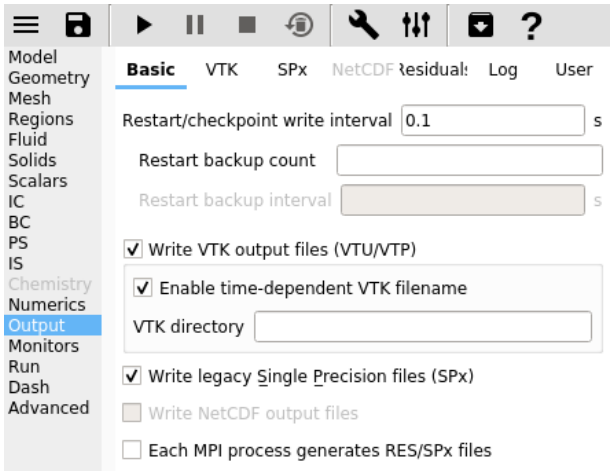


Note: By default, boundaries that are left undefined (here the left, right, front, and back planes) will behave as No-Slip walls.

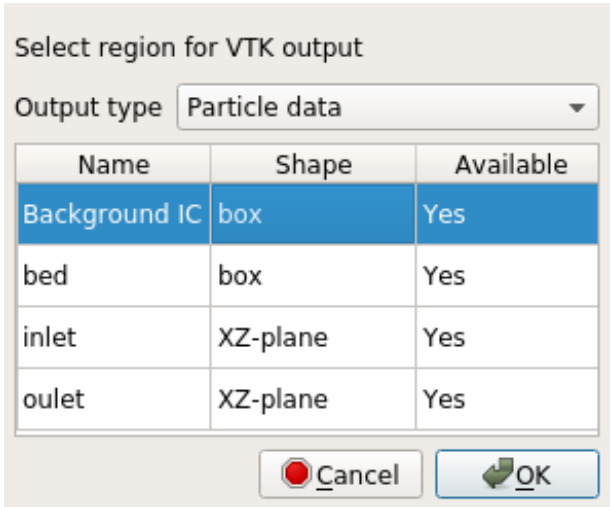
3.3.9 Select output options

On the `Output` pane:

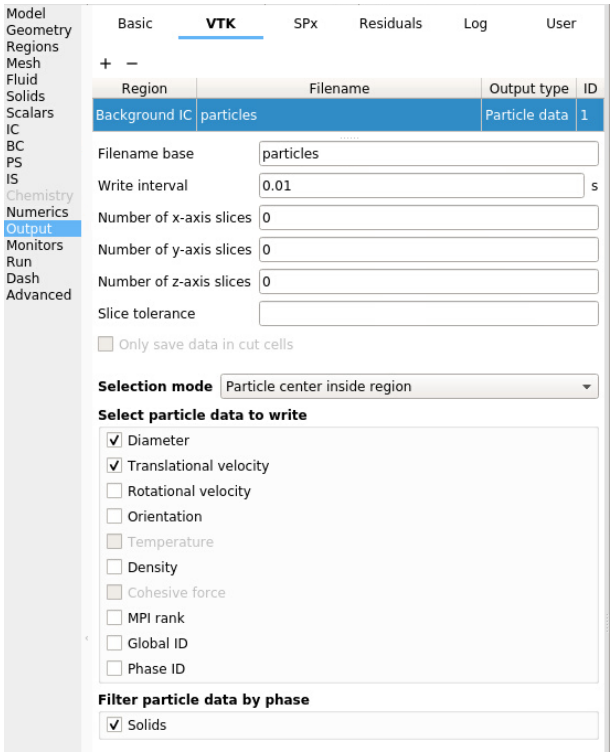
- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` checkbox



- Select the `VTK` sub-pane
- Create a new output by clicking the button
- Select "Particle Data" from the 'Output type' drop-down menu.
- Select the "Background IC" region from the list to save all the particle data
- Click `OK` to create the output



- Enter a base name for the `*.vtu` files in the `Filename base` field ("particles")
- Change the `Write interval` to `0.01` seconds
- Select the `Diameter` and `Translational Velocity` check-boxes



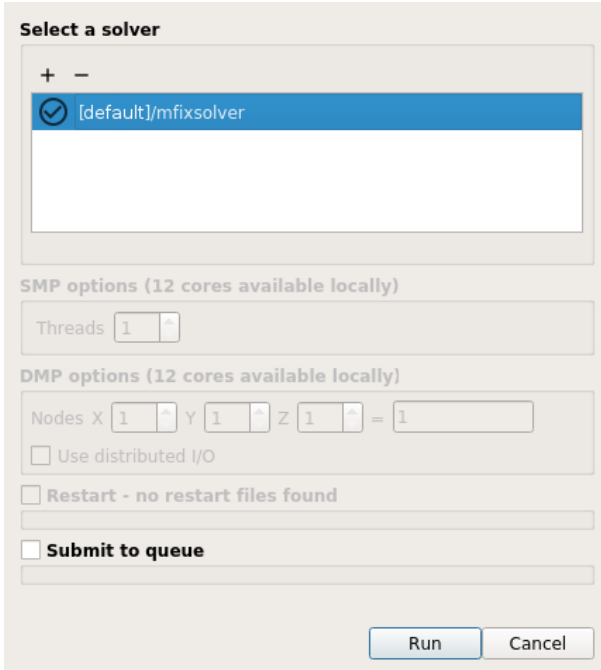
3.3.10 Change run parameters

On the `Run` pane:

- Change the `Stop Time step` to `1.0` seconds
- Change the `Time step` to `1e-2` seconds
- Change the `Maximum time step` to `1e-2` seconds

3.3.11 Run the project

- Save project by clicking the button
- Run the project by clicking the button
- On the `Run` dialog, select the executable from the combo-box
- Click the `Run` button to actually start the simulation



3.3.12 View results

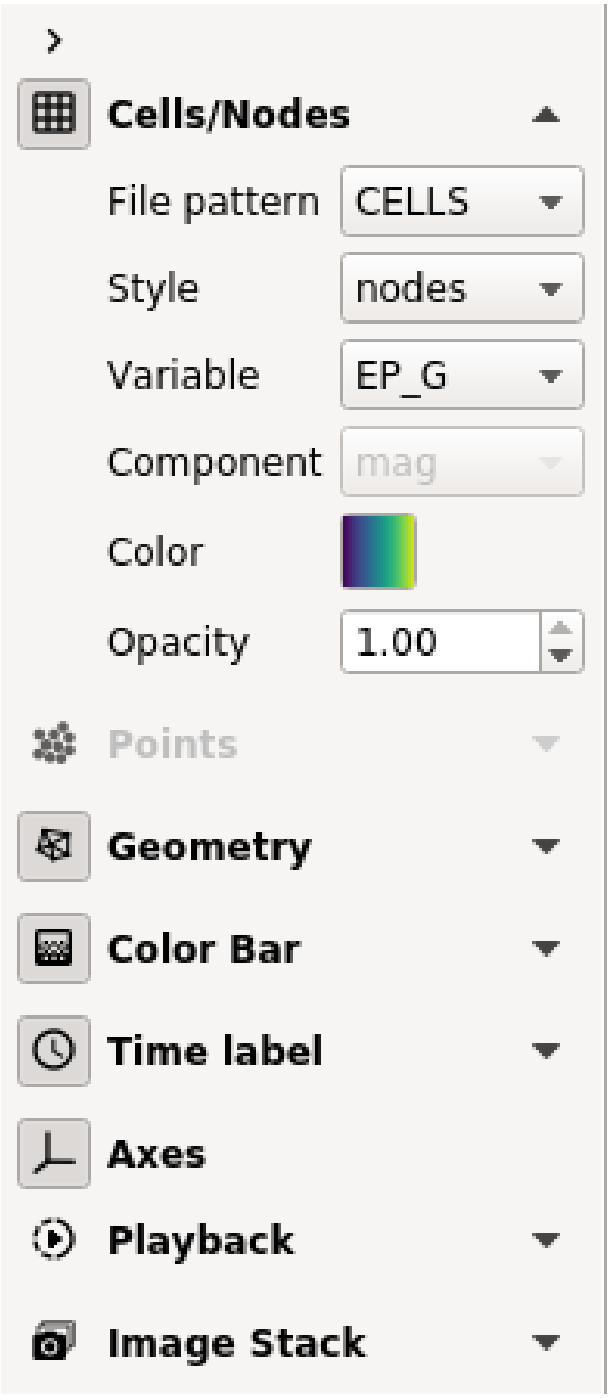
Results can be viewed, and plotted, while the simulation is running.

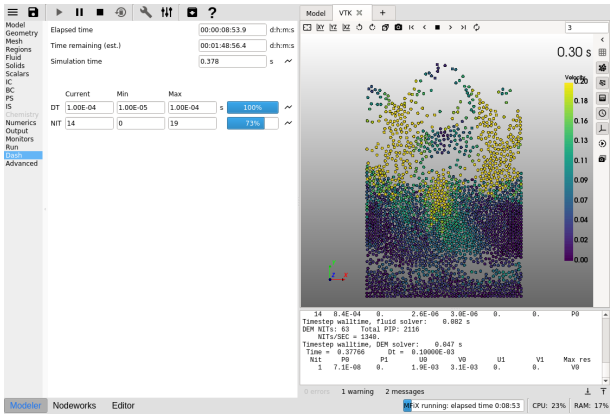
- Create a new visualization tab by pressing the next to the *Model* tab
- Select an item to view, such as plotting the time step (dt) or click the `3D view` button to view the vtk output files.
- On the `VTK` results tab, the visibility and representation of the `*.vtk` files can be controlled with the menu on the side.
- Change frames with the , , , and buttons
- Click the button to play the available vtk files.
- Change the playback speed under the section on the sidebar.

3.4 Three Dimensional Single phase flow over a sphere

This tutorial shows how to create a three dimensional single phase flow over a sphere.

Property	Value
geometry	60 cm 20 cm x 20 cm
mesh	30 x 10 x 10
gas velocity	1 m/s
temperature	298 K
pressure	101325 Pa

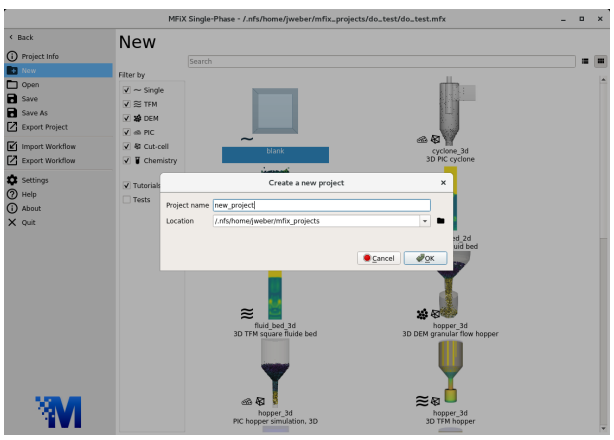

Cells/Nodes
File pattern
CELLS
Style
nodes
Variable
EP_G
Component
mag
Color
Opacity
1.00
Points
Geometry
Color Bar
Time label
Axes
Playback
Image Stack



3.4.1 Create a new project

- On the main menu, select `New project`
- Create a new project by double-clicking on "Blank" template.
- Enter a project name and browse to a location for the new project.
- When prompted to enable SMS workflow, answer *No*, we will use the standard workflow for this tutorial.

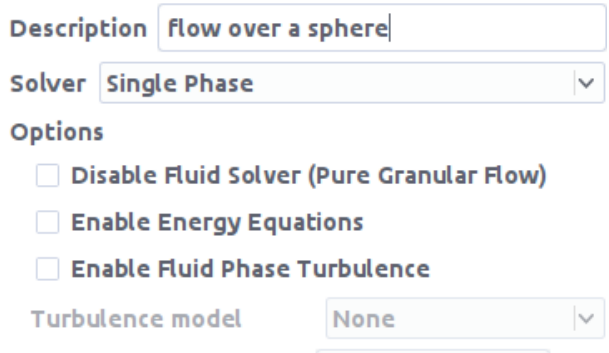
Note: A new project directory will be created in the location directory, with the name being the project name.



3.4.2 Select model parameters

On the `Model` pane:

- Enter a descriptive text in the `Description` field
- Select "Single Phase" in the `Solver` drop-down menu.

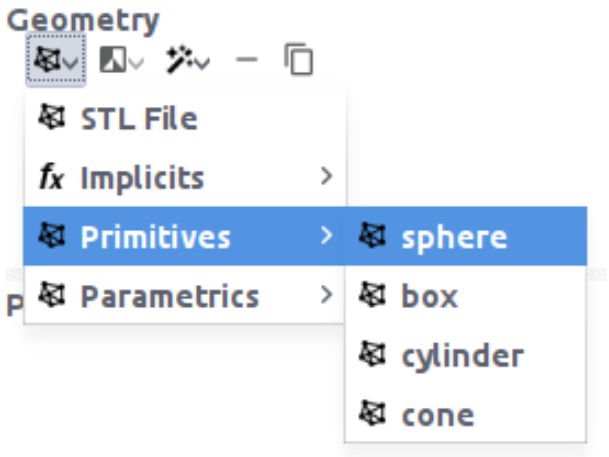


3.4.3 Enter the geometry

On the `Geometry` pane enter the domain extents:

- `60/100` meters for the maximum x value
- `20/100` meters for the maximum y value
- `20/100` meters for the maximum z value

Next, add a sphere by clicking the button -> primitives -> sphere. This adds a sphere constructed of triangles (STL) to the project.

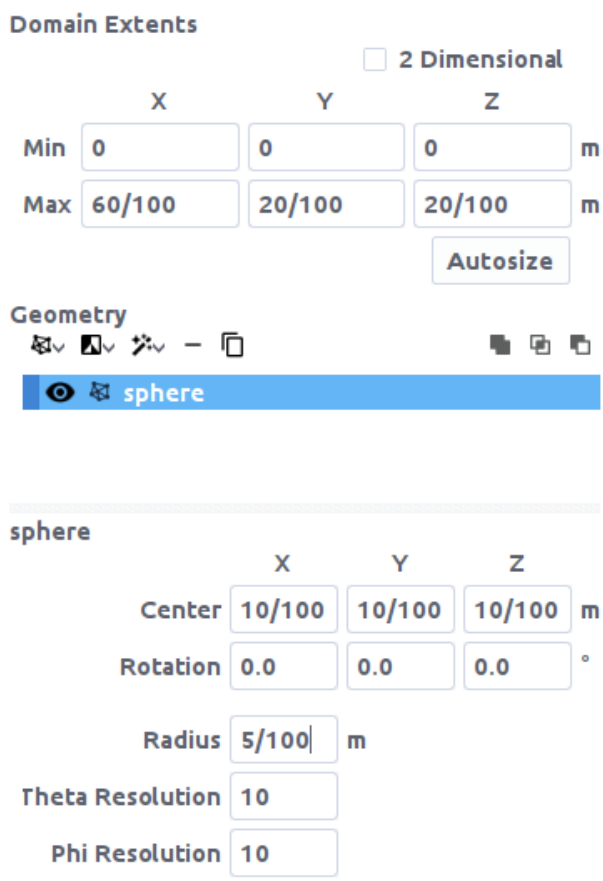


Change the center position and radius of the sphere so that it is located in the domain by entering the following:

- `10/100` for the center X position
- `10/100` for the center Y position
- `10/100` for the center Z position

Change the radius of the sphere by entering:

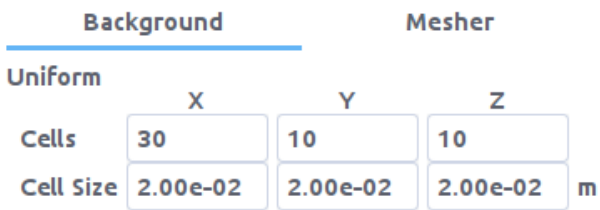
- `5/100` for the radius.


Domain Extents
2 Dimensional
X
Y
Z
Min
0
0
0
m
Max
60/100
20/100
20/100
m
Autosize
Geometry
sphere
sphere
X
Y
Z
Center
10/100
10/100
10/100
m
Rotation
0.0
0.0
0.0
°
Radius
5/100
m
Theta Resolution
10
Phi Resolution
10

3.4.4 Enter the mesh

On the `Mesh` pane, `Background` sub-pane:

- Enter `30` for the x cell value
- Enter `10` for the y cell value
- Enter `10` for the z cell value



3.4.5 Create regions for initial and boundary condition specification

Select the `Regions` pane. By default, a region that covers the entire domain is already defined.

A region for the sphere is needed to apply a wall boundary condition to:

- click the button to create a region that encompasses the entire domain
- change the name of the region to a descriptive `name` such as "sphere"
- Check the `Select Facets (STL)` check-box to turn the region into a STL region. The facets of the sphere should now be selected.

Create a region to apply a mass inflow boundary condition to:

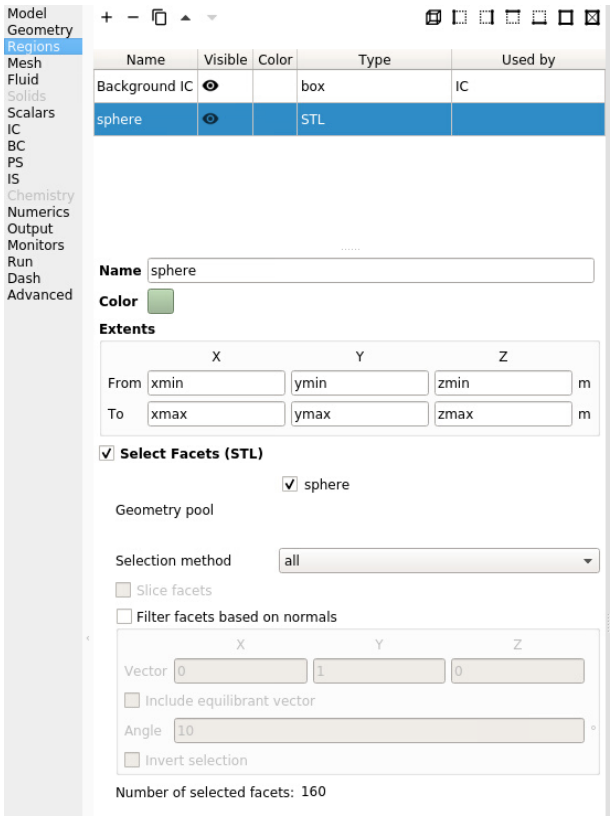
- Click the
- Enter a name for the region in the `Name` field ("inlet")

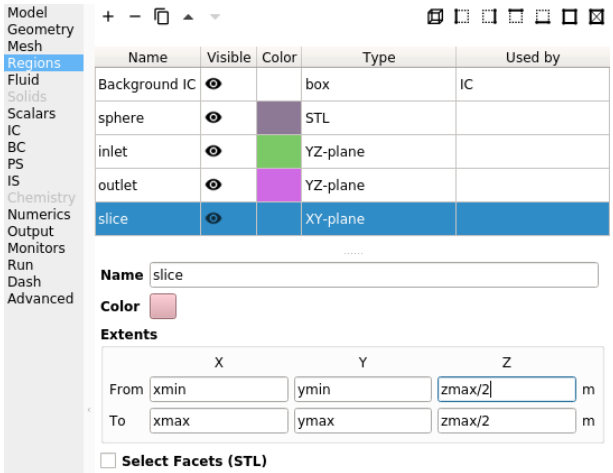
Create a region to apply a pressure outlet boundary condition to:

- Click the
- Enter a name for the region in the `Name` field ("outlet")

Finally, create a slice through the center to use as a vtk output region:

- Click the
- Enter a name for the region in the `Name` field ("slice")
- Enter `zmax/2` in both the `From Z` and `To Z` fields to move the region to the center of the domain


Model
Geometry
Regions
Mesh
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Name
Visible
Color
Type
Used by
Background IC
box
IC
sphere
STL
Name
sphere
Color
Extents
X
Y
Z
From
xmin
ymin
zmin
m
To
xmax
ymax
zmax
m
Select Facets (STL)
sphere
Geometry pool
Selection method
all
Slice facets
Filter facets based on normals
Vector
0
1
0
Include equilibrant vector
Angle
10
Invert selection
Number of selected facets: 160


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Name
Visible
Color
Type
Used by
Background IC
box
IC
sphere
STL
inlet
YZ-plane
outlet
YZ-plane
slice
XY-plane
Name
slice
Color
Extents
X
Y
Z
From
xmin
ymin
zmax/2
m
To
xmax
ymax
zmax/2
m
Select Facets (STL)

3.4.6 Create Initial Conditions

- Select the `Initial conditions` pane
- Select the already populated "Background IC" from the region list. This will initialize the entire flow field with air.
- Enter `101325` Pa in the `Pressure (optional)` field

3.4.7 Create Boundary Conditions

Select the `Boundary conditions` pane and create a wall boundary condition for the sphere by:

- clicking the button
- On the `Select Region` dialog, select "No Slip Wall" from the `Boundary type` combo-box
- Select the "sphere" region and click `OK`

Add a mass inlet boundary condition by:

- clicking the button
- On the `Select Region` dialog, select "Mass Inflow" from the `Boundary type` combo-box
- Select the "inlet" region and click `OK`
- On the "Fluid" sub-pane, enter a velocity in the `X-axial velocity` field of `1.0` m/s

Finally, create a pressure outlet boundary condition by:

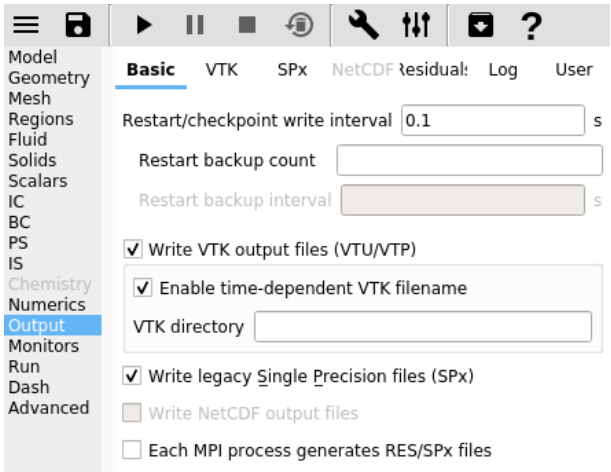
- clicking the button
- On the `Select Region` dialog, select "Pressure Outflow" from the `Boundary type` combo-box
- Select the "outlet" region and click `OK`

Note: The default pressure is already set to 101325 Pa, no changes need to be made to the outlet boundary condition.

3.4.8 Select output options

On the `Output` pane:

- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` checkbox
- Select the `VTK` sub-pane
- Create a new output by clicking the button
- Select the "slice" region from the list to save cell data at a slice through the domain
- Click `OK` to create the output
- Change the `Write interval` to "0.01" seconds
- Select the `Pressure`, `Velocity vector`, `Velocity x-component`, `Velocity y-component`, and `Velocity z-component` check-boxes on the `Fluid` sub-sub-pane



3.4.9 Change run parameters

On the `Run` pane:

- Change the `Stop Time step` to `2.0` seconds
- Change the `Time step` to `1e-2` seconds
- Change the `Maximum time step` to `1e-2` seconds

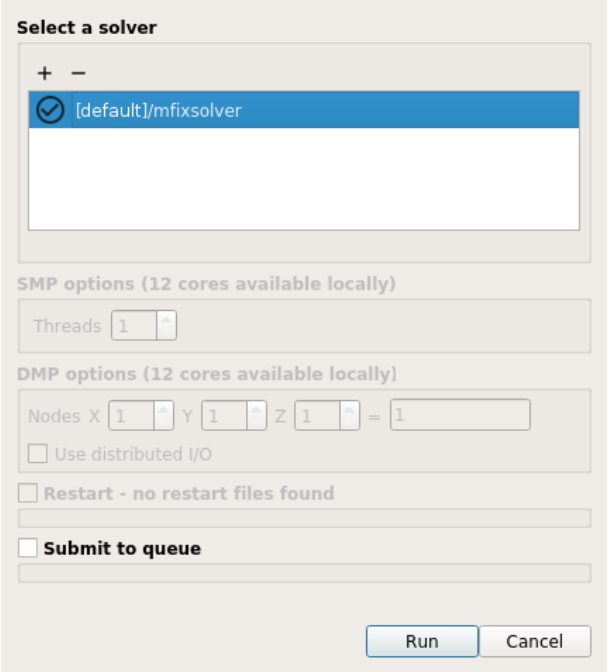
3.4.10 Run the project

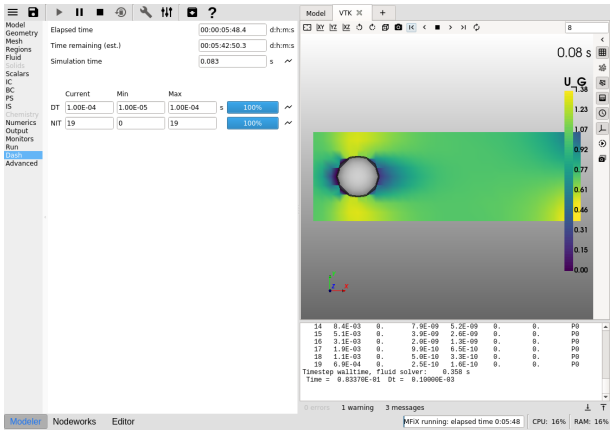
- Save project by clicking the button
- Run the project by clicking the button
- On the `Run Solver` dialog, select the executable
- Click the `Run` button to actually start the simulation

3.4.11 View results

Results can be viewed, and plotted, while the simulation is running.

- Create a new visualization tab by pressing the next to the *Model* tab
- Select an item to view, such as plotting the time step (dt) or click the `3D view` button to view the vtk output files.
- On the `VTK` results tab, the visibility and representation of the `*.vtk` files can be controlled with the menu on the side.


Select a solver
+ −
[default]/mfixsolver
SMP options (12 cores available locally)
Threads 1
DMP options (12 cores available locally)
Nodes X 1 Y 1 Z 1 = 1
Use distributed I/O
Restart - no restart files found
Submit to queue
Run
Cancel


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Tools
Advanced
Elapsed time 00:00:05:48.4 d:h:m:s
Time remaining (est.) 00:05:42:50.3 d:h:m:s
Simulation time 0.083 s
Current Min Max
DT 1.00E-04 1.00E-05 1.00E-04 s 100%
NIT 19 0 19 100%
Model
VTK
0.08 s
U G
1.36
1.23
1.07
0.92
0.77
0.61
0.46
0.31
0.15
0.00
14 0.4E-03 0. 7.9E-09 5.2E-09 0. 0. P0
15 5.1E-03 0. 3.9E-09 2.6E-09 0. 0. P0
16 3.1E-03 0. 2.0E-09 1.3E-09 0. 0. P0
17 1.9E-03 0. 9.9E-10 6.5E-10 0. 0. P0
18 1.1E-03 0. 5.0E-10 3.3E-10 0. 0. P0
19 6.9E-04 0. 2.5E-10 1.6E-10 0. 0. P0
Timestep walltime, fluid solver: 0.358 s
Time = 0.83370E-01 Dt = 0.10000E-03
1 warning
3 messages
Modeler
Nodeworks
Editor
MFIX running: elapsed time 0:05:48
CPU: 16%
RAM: 16%

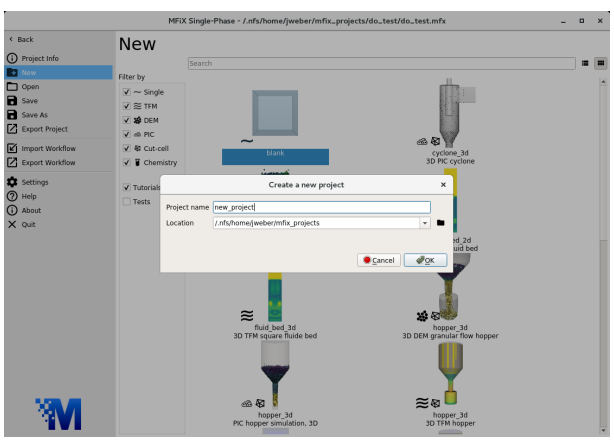
3.5 Three Dimensional Fluidized Bed

This tutorial shows how to create a three dimensional fluidized bed simulation using the two fluid model and the discrete element model (DEM). The model setup is:

Property	Value
geometry	10 cm diameter x 40 cm
mesh	20 x 60 x 20
solid diameter	200 microns (200×10^{-6} m)
solid density	2500 kg/m^2
gas velocity	0.25 m/s
temperature	298 K
pressure	101325 Pa

3.5.1 Create a new project

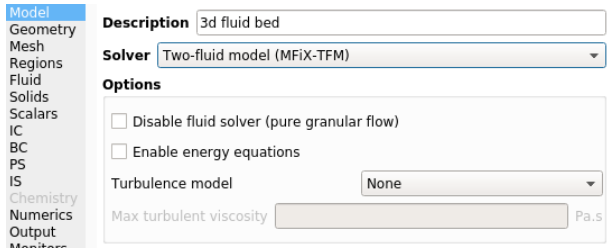
- (`fig_new_project`): On the main menu, select `New project`
- Create a new project by double-clicking on "Blank" template.
- Enter a project name and browse to a location for the new project.
- When prompted to enable SMS workflow, answer *No*, we will use the standard workflow for this tutorial.



3.5.2 Select model parameters

(`fig_model_param`): On the `Model` pane:

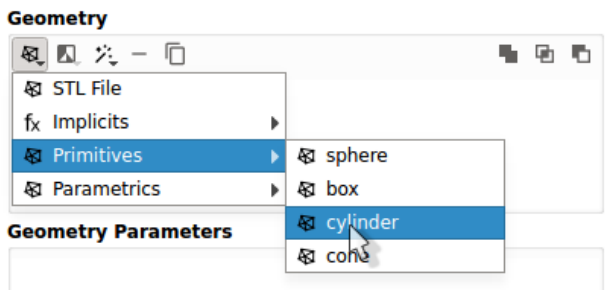
- Enter a descriptive text in the `Description` field
- Select "Two-Fluid Model (MFiX-TFM)" in the `Solver` drop-down menu.



3.5.3 Enter the geometry

On the `Geometry` pane:

- Create the cylindrical geometry by pressing the `Add Geometry` button -> `Primitives` -> `cylinder`



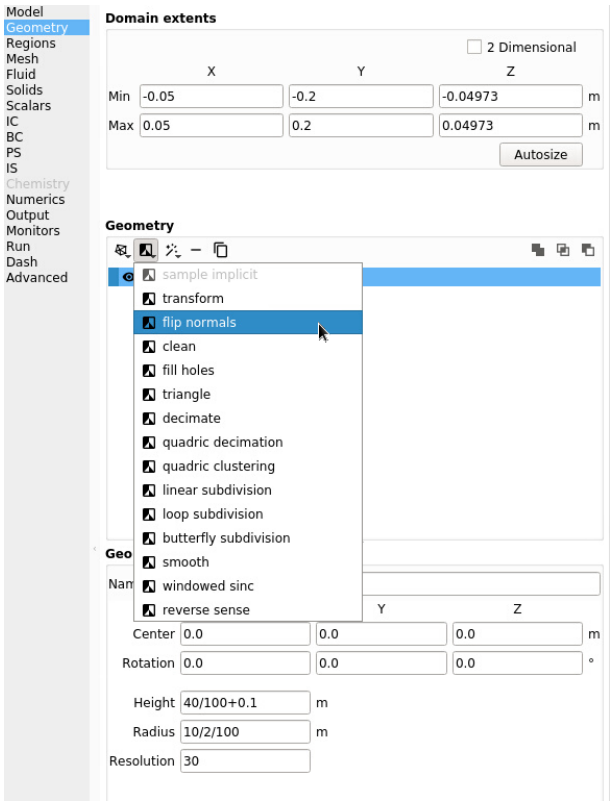
- Enter `40/100` meters for the cylinder height
- Enter `10/2/100` meters for the cylinder radius
- Enter `30` for the cylinder resolution
- Press the autosize button to fit the domain extents to the geometry
- Extend the height of the cylinder by adding `0.1` meters. This will hang the stl file outside of the domain, allowing for a sharp and clean cut.
- Flip the normals by clicking the button and selecting the `flip normals` filter

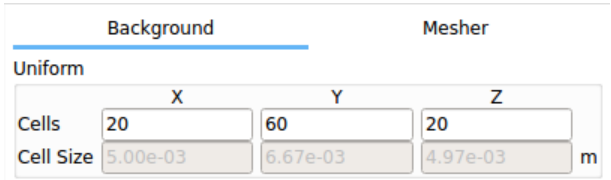
3.5.4 Enter the mesh

On the `Mesh` pane:

- On the `Background` sub-pane
- Enter `20` for the x cell value
- Enter `60` for the y cell value
- Enter `20` for the z cell value

Note: This is a fairly coarse grid for a TFM simulation. After completing this tutorial, try increasing the grid resolution to better resolve the bubbles.

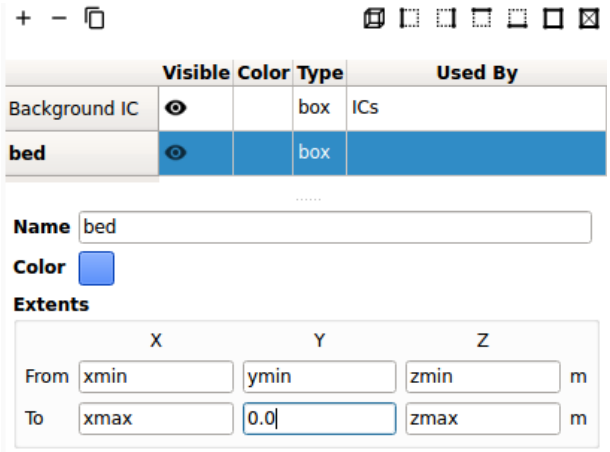

Model
Geometry
Regions
Mesh
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Domain extents
2 Dimensional
X
Y
Z
Min -0.05 -0.2 -0.04973 m
Max 0.05 0.2 0.04973 m
Autosize
Geometry
sample implicit
transform
flip normals
clean
fill holes
triangle
decimate
quadric decimation
quadric clustering
linear subdivision
loop subdivision
butterfly subdivision
smooth
windowed sinc
reverse sense
Center 0.0 0.0 0.0 m
Rotation 0.0 0.0 0.0 °
Height 40/100+0.1 m
Radius 10/2/100 m
Resolution 30


Background
Mesher
Uniform
X
Y
Z
Cells 20 60 20
Cell Size 5.00e-03 6.67e-03 4.97e-03 m

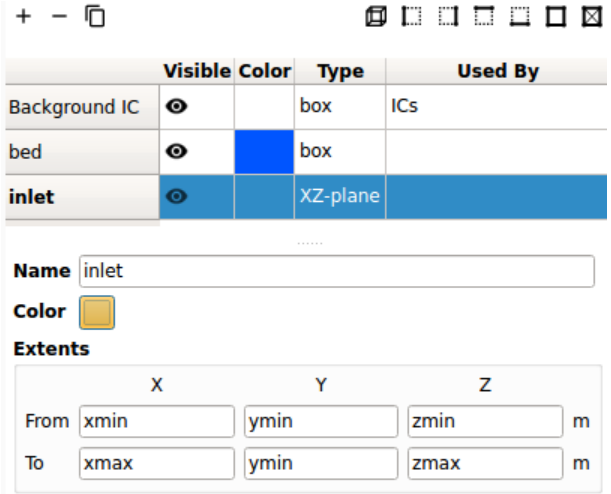
3.5.5 Create regions for initial and boundary condition specification

Select the `Regions` pane. By default, a region that covers the entire domain is already defined. This is typically used to initialize the flow field and visualize the results.

- click the button to create a new region to be used for the bed initial condition.
- Enter a name for the region in the `Name` field ("bed")
- Change the color by pressing the `Color` button
- Enter `0` in the `To Y` field

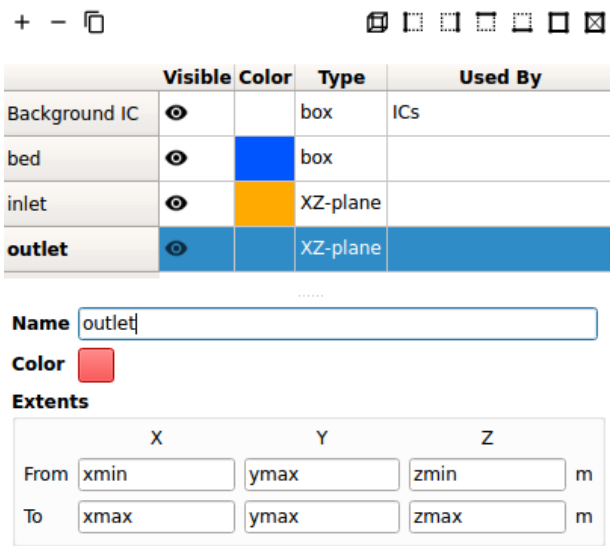


- Click the button to create a new region to be used by the gas inlet boundary condition.
- Enter a name for the region in the `Name` field ("inlet")



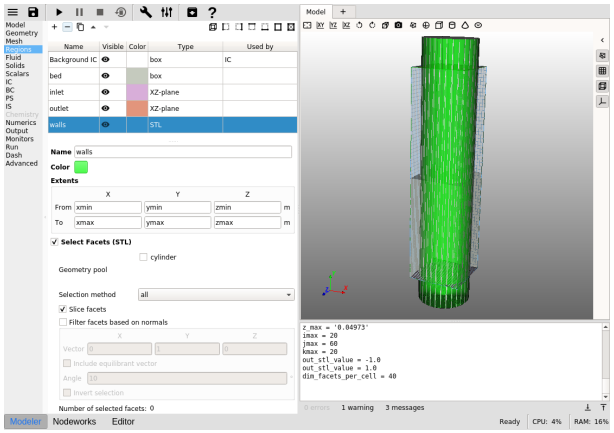
- Click the button to create a new region to be used by the pressure outlet boundary condition.
- Enter a name for the region in the `Name` field ("outlet")

- Click the button to create a new region to be used to select the walls.

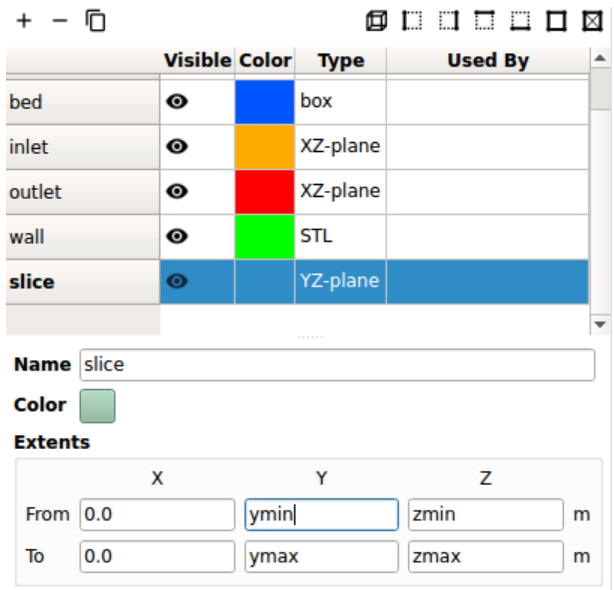


- Enter a name for the region in the `Name` field ("walls")

All the facets of the cylinder should now be selected. Since the cylinder is outside the domain extents, normal cells (i.e. not cut-cells) will be placed at the outlet and inlet. This allows for the standard boundary conditions to be applied.



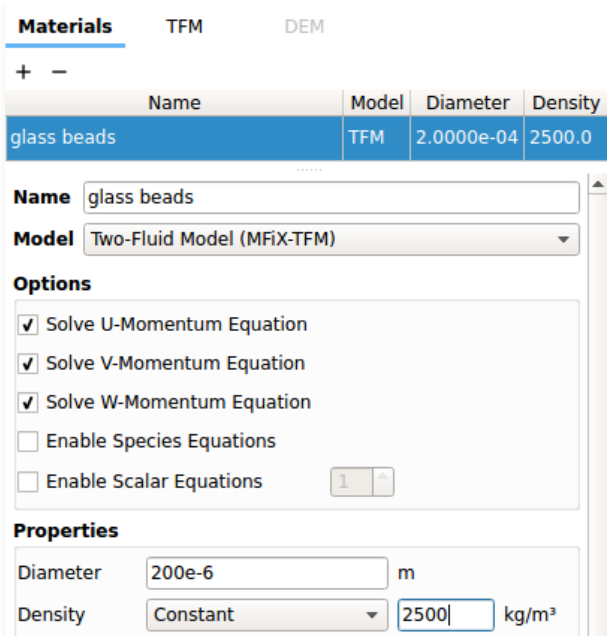
- Click the button to create a new region to be used to save a slice of cells at the center of the domain.
- Enter a name for the region in the `Name` field ("slice")
- Enter `0` in the `From X` and `To X` fields



3.5.6 Create a solid

On the `Solids` pane:

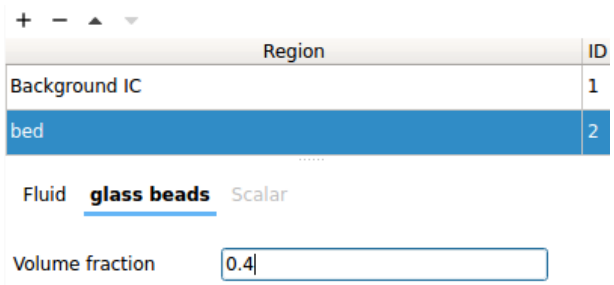
- Click the button to create a new solid
- Enter a descriptive name in the `Name` field ("glass beads")
- Accept the radial distribution setting (Carnahan-Starling)
- Enter the particle diameter of `200e-6` m in the `Diameter` field
- Enter the particle density of `2500` kg/m^2 in the `Density` field



3.5.7 Create Initial Conditions

On the `Initial conditions` pane:

- Select the already populated "Background IC" from the region list. This will initialize the entire flow field with air.
- Enter `101325` Pa in the `Pressure (optional)` field
- Create a new Initial Condition by pressing the button
- Select the region created previously for the bed Initial Condition ("bed" region) and click the `OK` button.
- Select the solid (named previously as "glass beads") sub-pane and enter a volume fraction of `0.4` in the `Volume Fraction` field. This will fill the bottom half of the domain with glass beads.



3.5.8 Create Boundary Conditions

On the `Boundary conditions` pane:

- Create a new Boundary condition by clicking the button
- On the `Select Region` dialog, select "Mass Inflow" from the `Boundary type` drop-down menu
- Select the "inlet" region and click `OK`
- On the `Fluid` sub-pane, enter a velocity in the `Y-axial velocity` field of `0.25` m/s
- Create another Boundary condition by clicking the button
- On the `Select Region` dialog, select "Pressure Outflow" from the `Boundary type` combo-box
- Select the "outlet" region and click `OK`

Note: The default pressure is already set to 101325 Pa, no changes need to be made to the outlet boundary condition.

- Create another Boundary condition by clicking the button
- On the `Select Region` dialog, select "No Slip Wall" from the `Boundary type` combo-box
- Select the "wall" region and click `OK`

3.5.9 Change numeric parameters

On the `Numerics` pane, `Residuals` sub-pane:

- Enter `0` in the `Fluid Normalization` field.

3.5.10 Select output options

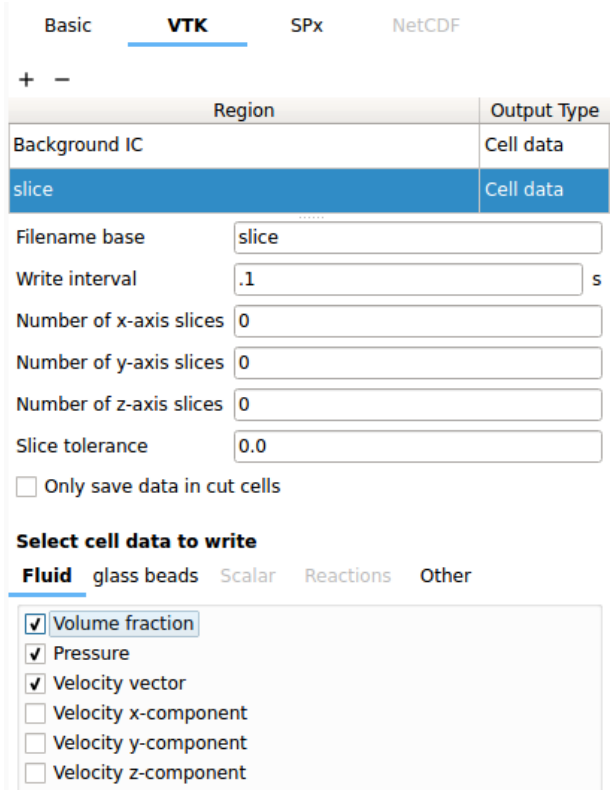
On the `Output` pane:

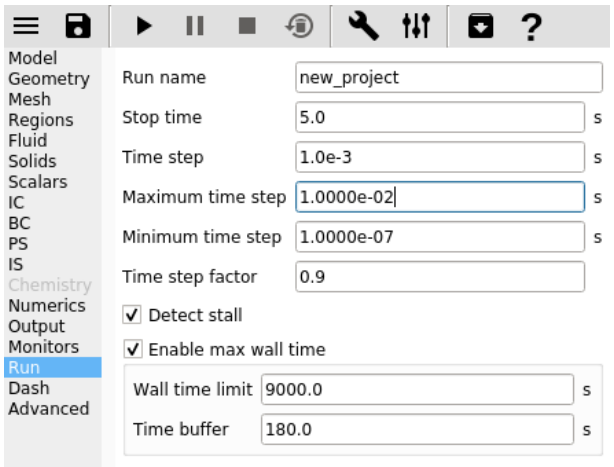
- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` checkbox
- Select the `VTK` sub-pane
- Create a new output by clicking the button
- Select the "Background IC" region from the list to save all the cell data
- Click `OK` to create the output
- Enter a base name for the `*.vtu` files in the `Filename base` field
- Change the `Write interval` to `0.01` seconds
- Select the `Volume fraction`, `Pressure`, and `Velocity vector` check-boxes on the `Fluid` tab
- Create another output by clicking the button
- Select the "Slice" region from the list to save all the cell data
- Click `OK` to create the output
- Enter a base name for the `*.vtu` files in the `Filename base` field
- Change the `Write interval` to `0.01` seconds
- Select the `Volume fraction`, `Pressure`, and `Velocity vector` check-boxes on the `Fluid` tab

3.5.11 Change run parameters

On the `Run` pane:

- Change the `Stop Time` to `1.0` seconds
- Change the `Time step` to `1e-3` seconds
- Change the `Maximum time step` to `1e-2` seconds


Basic
VTK
SPx
NetCDF
+ −
Region
Output Type
Background IC
Cell data
slice
Cell data
Filename base
slice
Write interval
.1
s
Number of x-axis slices
0
Number of y-axis slices
0
Number of z-axis slices
0
Slice tolerance
0.0
Only save data in cut cells
Select cell data to write
Fluid
glass beads
Scalar
Reactions
Other
Volume fraction
Pressure
Velocity vector
Velocity x-component
Velocity y-component
Velocity z-component


Model
Geometry
Mesh
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Run name
new_project
Stop time
5.0
s
Time step
1.0e-3
s
Maximum time step
1.0000e-02
s
Minimum time step
1.0000e-07
s
Time step factor
0.9
Detect stall
Enable max wall time
Wall time limit
9000.0
s
Time buffer
180.0
s

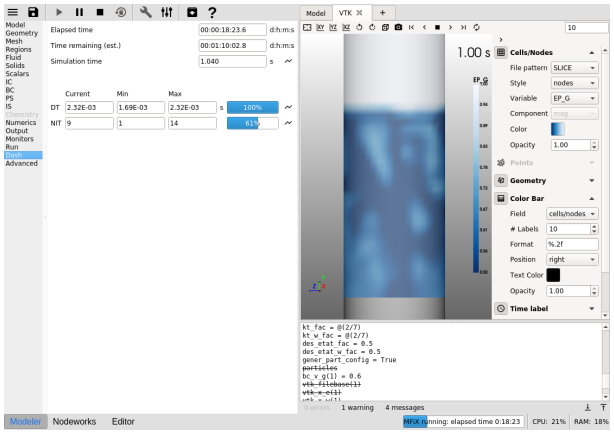
3.5.12 Run the project

- Save project by clicking button
- Run the project by clicking the button
- On the `Run` dialog, select the default executable from the list
- Click the `Run` button to actually start the simulation

3.5.13 View results

Results can be viewed, and plotted, while the simulation is running.

- Create a new visualization tab by pressing the next to the *Model* tab
- Select an item to view, such as plotting the time step (dt) or click the `3D view` button to view the vtk output files.
- On the `VTK` results tab, the visibility and representation of the `*.vtk` files can be controlled with the menu on the side.

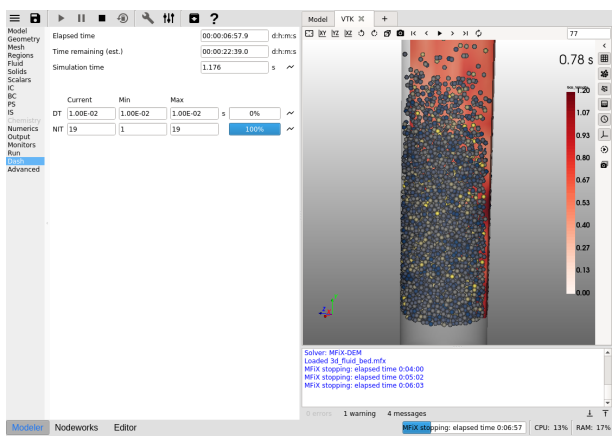


3.5.14 Convert this project into a 3D DEM simulation

- Click the button and delete all simulation files.
- Close the VTK window
- On the `Model` pane, change the solver to `MFiX-DEM`
- On the `Mesh` pane, coarsen the grid to *15*, *45*, and *15* cells in the in the x, y, and z direction, respectfully.

Note: The grid resolution needs to be coarser because we are drastically increasing the particle diameter below. The fluid grid cell size has to be bigger than the particle size.

- On the `Solids` pane, change that particle diameter to `5.0000e-03` to get a more reasonable particle count for tutorial purposes.
- On the `Solids` pane, `DEM` sub-pane: - check the `Enable automatic particle generation` - enter a value of `15` in the `Search grid partitions`, `KMAX` field.
- On the `Boundary Conditions` pane, change the `inlet Y-axial velocity` to `0.6` m/s
- On the `Boundary Conditions` pane, delete the `walls` boundary condition and re-add it to write the correct wall parameters for the DEM simulation.
- On the `Output` pane, `VTK` sub-pane:
  - Delete the `all` or `Background_IC` output
  - Create a new output, change the `Output type` to `Particle data` and select the `Background_IC` region.
  - Change the write frequency to `0.01`
  - Select the `Diameter` and `Translational Velocity` data
- Run the simulation
- Create a VTK window to visualize the data. It will automatically show the slice (cell data) and the particles.



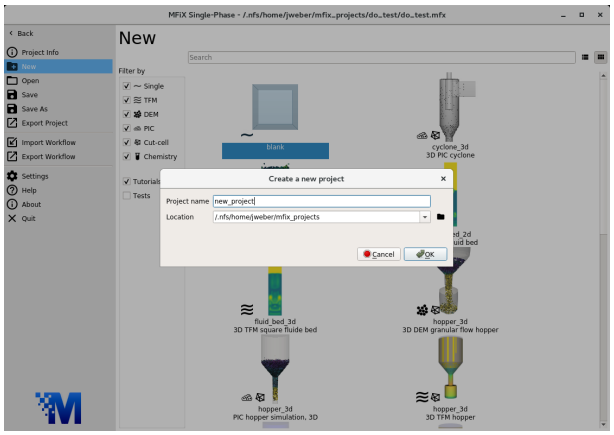
3.6 Three Dimensional DEM Hopper

This tutorial shows how to create a three dimensional granular flow DEM simulation. The model setup is:

Property	Value
geometry	5 cm diameter hopper
mesh	10 x 25 x 10
solid diameter	0.003 m
solid density	2500 kg/m^2
gas velocity	NA
temperature	298 K
pressure	NA

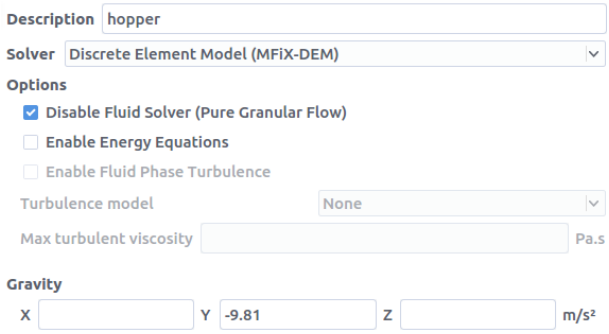
3.6.1 Create a new project

- On the main menu, select `New project`
- Create a new project by double-clicking on "Blank" template.
- Enter a project name and browse to a location for the new project.
- When prompted to enable SMS workflow, answer *No*, we will use the standard workflow for this tutorial.



3.6.2 Select model parameters

- On the `Model` pane, enter a descriptive text in the `Description` field
- Select "Discrete Element Model (MFiX-DEM)" in the `Solver` drop-down menu
- Check the `Disable Fluid Solver (Pure Granular Flow)` check-box.

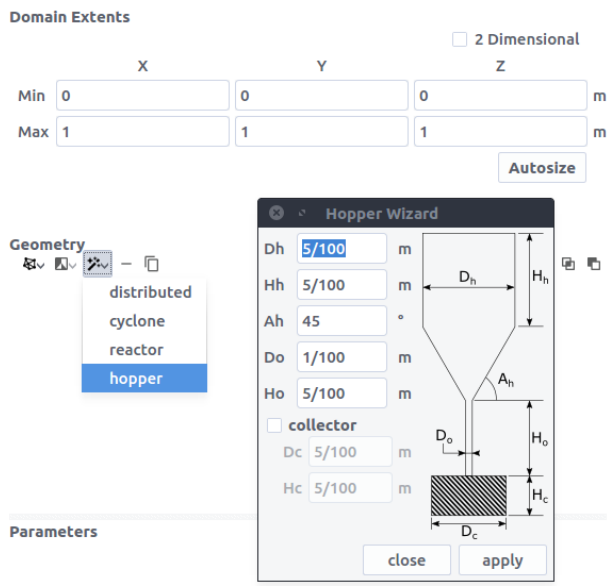


3.6.3 Enter the geometry

On the `Geometry` pane:

- Select the wizard menu and select the `hopper` wizard
- Select `apply` to build the hopper stl file
- Press the `Autosize` button to fit the domain extents to the geometry

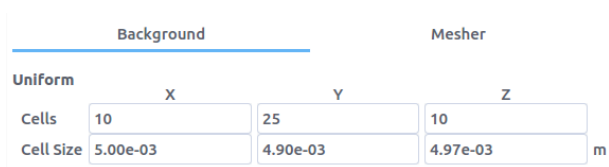
- Press the Reset View icon on the top-left corner of the Model window



3.6.4 Enter the mesh

On the `Mesh` pane, `Background` sub-pane:

- Enter `10` for the x cell value
- Enter `25` for the y cell value
- Enter `10` for the z cell value



On the `Mesh` pane, `Mesher` sub-pane:

- Enter `0.0` in the `Facet Angle Tolerance`

3.6.5 Create regions for initial and boundary condition specification

Select the `Regions` pane. By default, a region that covers the entire domain is already defined.

A region for the hopper walls (STL) is needed to apply a wall boundary condition to:

- click the button to create a region that encompasses the entire domain
- change the name of the region to a descriptive `name` such as "walls"
- Check the `Select Facets (STL)` check-box to turn the region into a STL region. The facets of the hopper should now be selected.

In order to allow the particles to leave the domain through the bottom, the facets at the bottom of the hopper need to be deselected. There are three ways to accomplish this:

1. Change the `Y Min` domain extent value on the geometry pane by a small amount so that those facets fall outside the simulation domain (for example, change the value to `-0.0974`).
2. Change the `Y From` region extent value on the regions pane for this STL region by a small amount so that those facets fall outside the region (for example, change the value to `ymin+0.0001`)
3. Or, use the `Filter facets based on normals` option on the region pane.

For this example, use option 3 to filter the facets that have normals pointed in the `Y` direction by:

- Select `box` from the `Selection method` drop-down.
- Uncheck `Slice facets`.
- Check the `Filter facets based on normals` check-box
- Enter a vector pointed in the positive `Y` direction by entering `0`, `1`, and `0` in the x, y, z fields, respectively
- Check the `Include equilibrant vector` to include facets with normals in the opposite direction of the vector (-x, -y, -z)
- Enter an angle of `10` in the `angle` field to include facets with normals within 10 degrees of the filter vector
- Check the `Invert Selection` check box to de-select facets that fall along the filter vector and select all the other facets

All the facets, except for the facets at the top and bottom, of the hopper should now be selected.

Create a region to apply a pressure outlet boundary condition to allow particles to leave the domain:

- Click the
- Enter a name for the region in the `Name` field ("outlet")

Finally, create a region to initialize the solids:

- Click the
- Enter a name for the region in the `Name` field ("solids")
- Enter `ymin/3` in the `From Y` field
- Enter `0` in the `To Y` field

3.6.6 Create a solid

On the `Solids` pane

- Click the button to create a new solid
- Enter a descriptive name in the `Name` field ("solids")
- Keep the model as "Discrete Element Model (MFiX-DEM)")
- Enter the particle diameter of `0.003` m in the `Diameter` field
- Enter the particle density of `2500` kg/m^2 in the `Density` field

In the `DEM` sub-pane, check `Enable automatic particle generation` check-box

3.6.7 Create Initial Conditions

On the `Initial conditions` pane

- Click the button to create a new initial condition
- On the `Select Region` dialog, select the "solids" region and click `OK`
- On the `solids` sub-pane, enter `0.4` in the `Volume fraction` field

3.6.8 Create Boundary Conditions

Select the `Boundary conditions` pane and create a wall boundary condition for the hopper by:

- clicking the button
- On the `Select Region` dialog, select "No Slip Wall" from the `Boundary type` drop-down menu
- Select the "walls" region and click `OK`

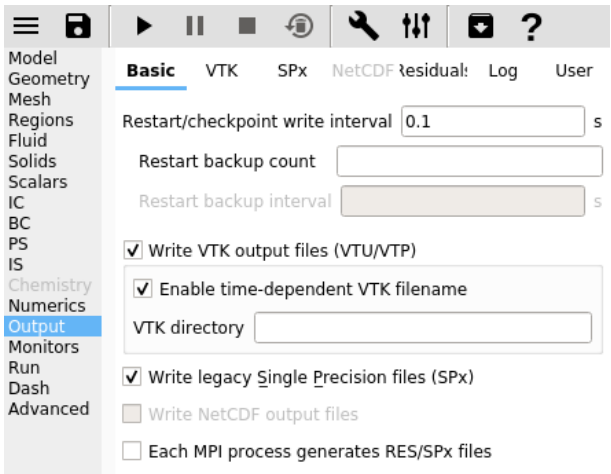
Finally, create a pressure outlet boundary condition by:

- clicking the button

- On the `Select Region` dialog, select "Pressure Outflow" from the `Boundary type` combo-box
- Select the "outlet" region and click `OK`

3.6.9 Select output options

- Select the `Output` pane
- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` checkbox



- Select the `VTK` sub-pane
- Create a new output by clicking the button
- On the "select region" dialog, select "particle data" from the `output type` drop-down menu
- Select the "Background IC" region from the list to save particle data over the entire domain
- Click `OK` to create the output
- Change the `Write interval` to `0.01` seconds
- Select the `Diameter` and `Translational velocity` check-boxes

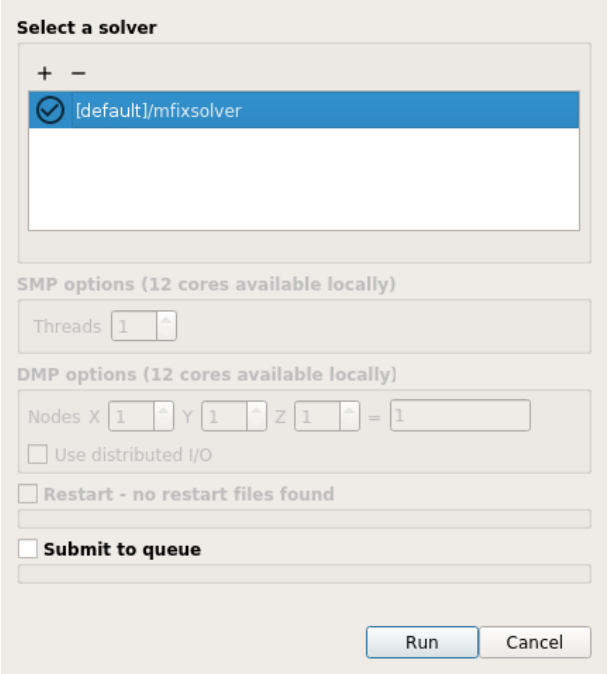
3.6.10 Change run parameters

On the `Run` pane:

- Change the `Stop Time` to `2.0` seconds
- Change the `Time step` to `1e-3` seconds
- Change the `Maximum time step` to `1e-2` seconds

3.6.11 Run the project

- Save project by clicking button
- Run the project by clicking the button
- On the `Run` dialog, select the default solver
- Click the `Run` button to actually start the simulation



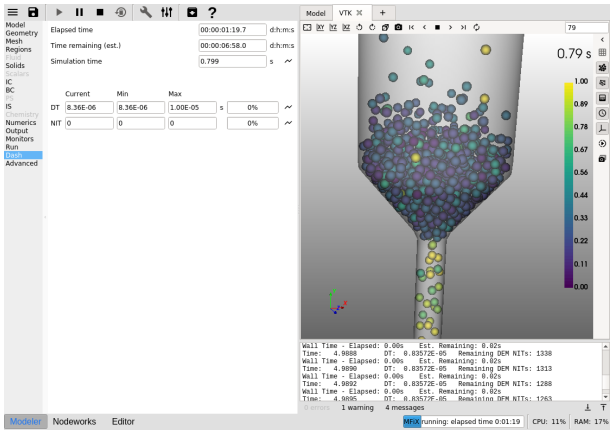
3.6.12 View results

Results can be viewed, and plotted, while the simulation is running.

- Create a new visualization tab by pressing the in the upper right hand corner.
- Select an item to view, such as plotting the time step (dt) or click the `VTK` button to view the vtk output files.

3.7 DEM Granular Flow Chutes

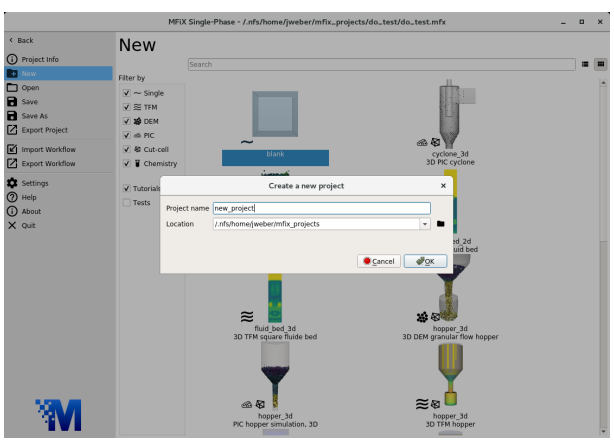
This tutorial shows how to create a three dimensional pure granular flow simulation from a series of cylindrical chutes. The model setup is:



Property	Value
geometry	1.75 m x 0.94 m x 0.2 m
mesh	20 x 20 x 10
solid diameter	12.5 mm (0.0125 m)
solid density	2500 kg/m^2
temperature	298 K
pressure	101325 Pa

3.7.1 Create a new project

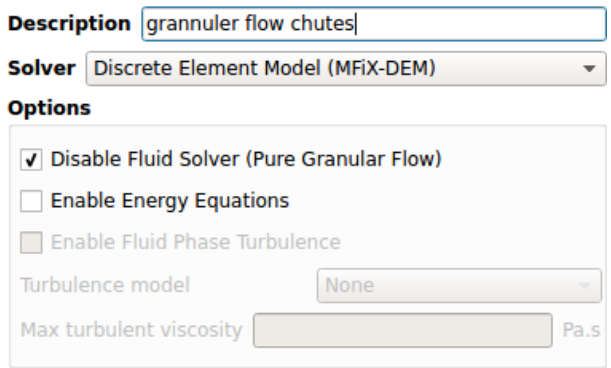
- (`fig_new_project`): On the main menu, select `New project`
- Create a new project by double-clicking on "Blank" template.
- Enter a project name and browse to a location for the new project.
- When prompted to enable SMS workflow, answer *No*, we will use the standard workflow for this tutorial.



3.7.2 Select model parameters

(`fig_model_param`): On the `Model` pane:

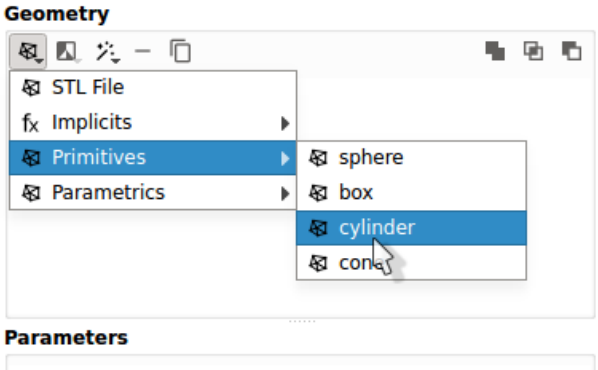
- Enter a descriptive text in the `Description` field
- Select "Discrete Element Model (MFiX-DEM)" in the `Solver` drop-down menu.
- Check the `Disable Fluid Solver` checkbox.



3.7.3 Enter the geometry

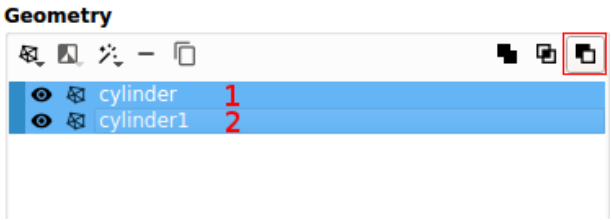
On the `Geometry` pane:

- (`fig_add_cylinder`): Create the cylindrical geometry by pressing -> `primitives` -> `cylinder`
  - Enter `0.1` meters for the cylinder radius
  - Enter `30` for the cylinder resolution

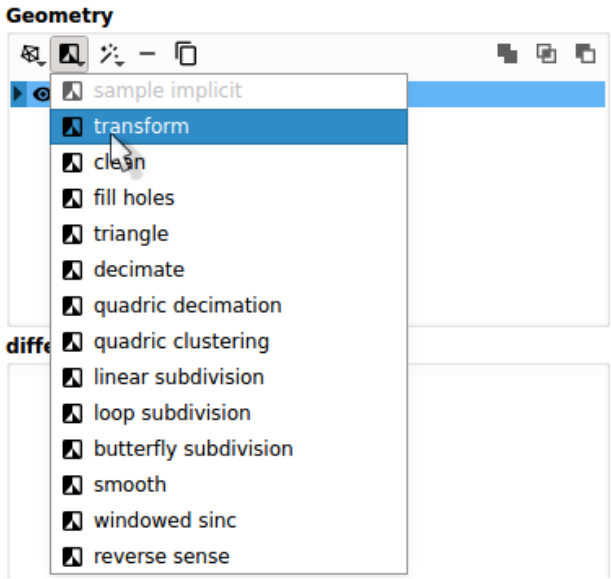


- Create another cylindrical geometry by pressing -> `primitives` -> `cylinder`
  - Enter `1.2` meters for the cylinder height
  - Enter `0.08` meters for the cylinder radius
  - Enter `30` for the cylinder resolution
- (`fig_difference`): Subtract the smaller radius cylinder from the larger radius cylinder by

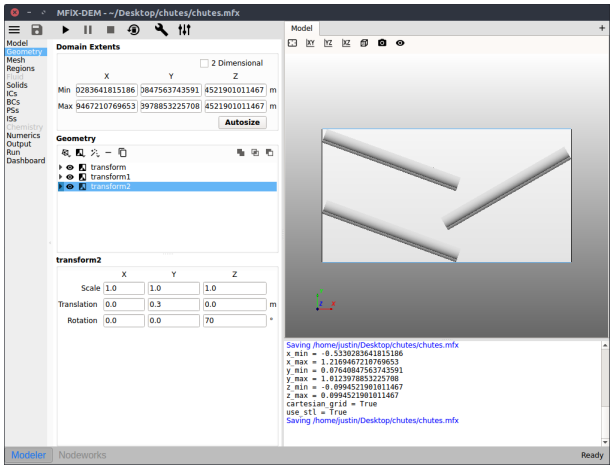
  - Select the larger radius cylinder (named `cylinder`)
  - While holding the `ctrl` key, select the smaller radius cylinder (named `cylinder1`)
  - Press the button



- Slice the tube in half by:
  - Add a box: -> `primitives` -> `box`
  - Change the `Center X` value to `0.5` m
  - Change the `Y Length` to `1.2` m
  - Select the tube (named `difference`)
  - While holding the `ctrl` key, select the box (named `box`)
  - Press the button
- Rotate the chute by:
  - Select the chute (named `difference1`)
  - (`fig_add_transform`): Add a filter: -> `transform`
  - Enter a value of `0.8` m in the `Translate Y` field.
  - Enter a value of `70` in the `Rotation Z` field.



- Copy the chute by pressing the button with the chute selected (named `transform`)
  - Enter a value of `0.8` in the `Translate X` field
  - Enter a value of `0.6` in the `Translate Y` field
  - Enter a value of `120` in the `Rotation Z` field
- Copy the chute again by pressing the button with the chute selected (named `transform1`)
  - Enter a value of `0.0` in the `Translate X` field
  - Enter a value of `0.3` in the `Translate Y` field
  - Enter a value of `70` in the `Rotation Z` field
- Press the `Autosize` button to fit the domain extents to the geometry



3.7.4 Enter the mesh

On the `Mesh` pane:

- On the `Background` sub-pane
- Enter `20` for the x cell value
- Enter `20` for the y cell value
- Enter `10` for the z cell value

3.7.5 Create regions for initial and boundary condition specification

Select the `Regions` pane.

- Create a region to be used for the wall boundary condition
  - click the button to create a new region
  - Enter a name for the region in the `Name` field ("walls")
  - Change the color by pressing the `Color` button

  - Check the `Select Facets (STL)` checkbox
- Create a region to be used as the mass inflow boundary condition
  - Click the button to create a new region
  - Enter a name for the region in the `Name` field ("inlet")
  - Enter a value of `-0.4` m in the `From X` field
  - Enter a value of `-0.2` m in the `To X` field

3.7.6 Create a solid

On the `Solids` pane:

- Click the button to create a new solid
- Enter a descriptive name in the `Name` field ("glass beads")
- Enter the particle diameter of `1/80` m in the `Diameter` field
- Enter the particle density of `2500` kg/m^2 in the `Density` field

On the `DEM` sub pane:

- check the `Enable automatic particle generation` checkbox and keep defaults values for all other settings.

3.7.7 Create Initial Conditions

On the `Initial conditions` pane leave the default initial condition.

3.7.8 Create Boundary Conditions

On the `Boundary conditions` pane:

- Create a new Boundary condition by clicking the button
- On the `Select Region` dialog, select "Mass Inflow" from the `Boundary type` drop-down menu
- Select the "inlet" region and click `OK`
- On the `glass beads` sub-pane:
  - Enter a value of `0.1` in the `volume fraction` field.
  - Enter a velocity in the `Y-axial velocity` field of `-0.1` m/s
- Create another Boundary condition by clicking the button
- On the `Select Region` dialog, select "No Slip Wall" from the `Boundary type` combo-box
- Select the "walls" region and click `OK`

3.7.9 Select output options

On the `Output` pane:

- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` check box
- Select the `VTK` sub-pane
- Create a new output by clicking the button
- Select `Particle Data` in the `Output type` combo box.
- Select the "Background IC" region from the list to save all the cell data
- Click `OK` to create the output
- Enter a base name for the `*.vtp` files in the `Filename base` field
- Change the `Write interval` to `0.1` seconds
- Select the `Diameter` and `Translational Velocity` check boxes.

3.7.10 Run the project

- Save project by clicking button
- Run the project by clicking the button
- On the `Run` dialog, select the default executable from the list
- Click the `Run` button to actually start the simulation

3.7.11 View results

Results can be viewed, and plotted, while the simulation is running.

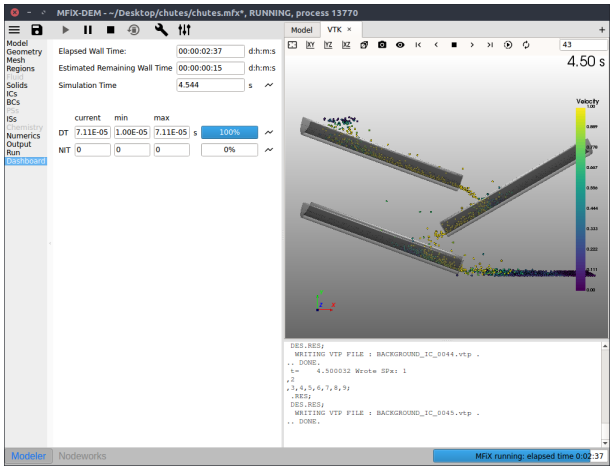
- Create a new visualization tab by pressing the in the upper right hand corner.
- Select an item to view, such as plotting the time step (dt) or click the `VTK` button to view the vtk output files.

3.8 SMS meshing workflow, cyclone, Discrete Element Model (DEM)

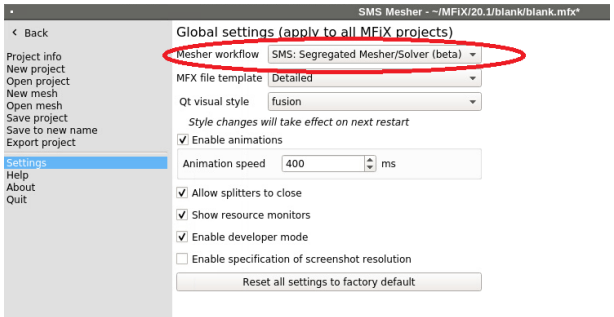
This tutorial shows how to use the new meshing workflow, called SMS (Segregated Mesher/Solver). The SMS workflow breaks the CFD workflow into two major steps: Mesher and Solver modes. Each mode of operation is performed in a separate tab in the GUI.

During the Meshing step, the user focuses on generating the mesh and verifies it is appropriate before moving on to the Solver setup. Only three panes are available during Meshing: Geometry, Regions, and Mesh. Once the mesh is generated it must be accepted (after inspection) to unlock the other panes, and move to the second step (Solver mode). The solver mode allows to set all other model settings, and run the MFiX solver to obtain and visualize the simulation results.

The SMS workflow is currently available as a beta testing feature, and can be turned on from the settings menu (select SMS: Segregated Mesher/Solver (beta) from the Mesher workflow drop down list). The `blank` template will automatically

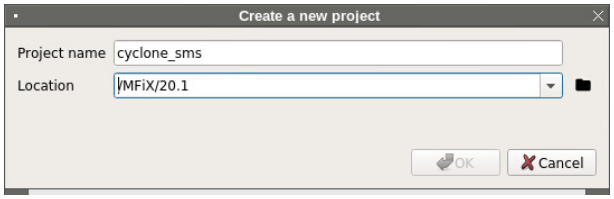


prompt the user to use the SMS workflow and users are encouraged to try it and provide feedback to the development team.



3.8.1 Create a new project

- On the main menu click on the New button.
- Create a new project by double-clicking on "Blank" template.
- Enter a project name (say 'cyclone_sms' and browse to a location for the new project.



- If you have not switched to SMS workflow, you will be prompted to switch. Accept SMS mode.

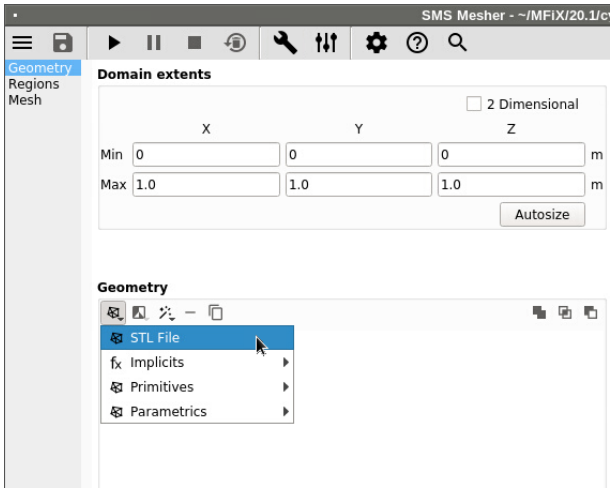
We are now starting a new project in SMS workflow. There are two tabs in the GUI to setup and run the simulation. The Mesher tab is where the mesh is setup and generated. The Modeler tab is where the rest of the settings and the simulation are performed. The Modeler tab is initially locked until the mesh has been generated and accepted. Running the MFiX solver is also not allowed until the mesh has been generated. At run time, the solver will read the mesh and proceed with the CFD solution.

3.8.2 Import the geometry

We will use an STL file as the geometry input. Download the file `here`

Go to the `Geometry` pane:

- Choose `STL file` as the geometry input.



- Navigate to where you downloaded `cyclone.stl`, select it and answer `Yes` when prompted to copy the file to the project directory.
- Verify the STL normals are pointing in the right direction. In the model window, show the Geometry, hide the Background Mesh, Hide the Regions (visibility is toggled by clicking on each icon). Set the Geometry style to `edges`, opacity to 1.0, check the `Show normals` box, set the `Scale` to 0.05, set `count` to 10,000 (to show all normals).

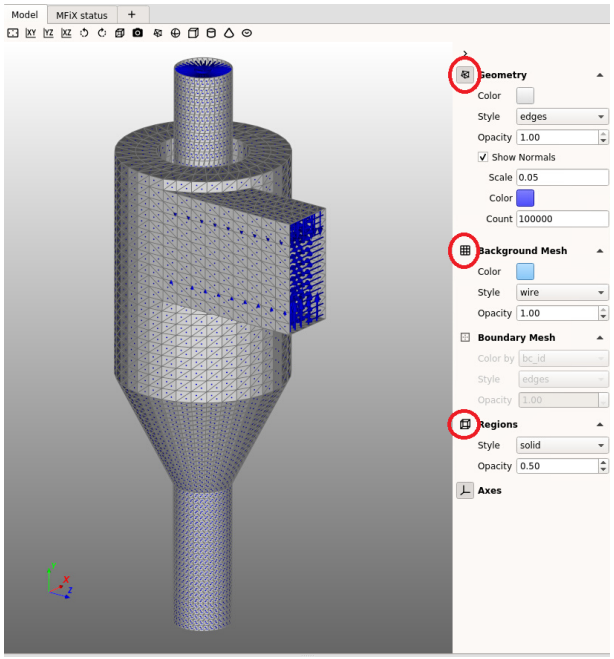
The normal vectors should point towards the fluid region (here towards the inside of the cyclone). For internal flows, the tips of the arrows are generally not visible, except through openings in the STL geometry. Openings in the STL geometry are allowed as long as they are outside of the domain extents.

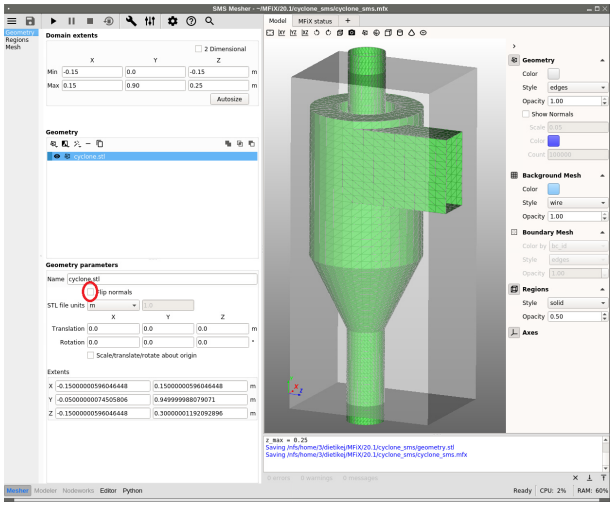
Note: When the normals are pointing in the wrong direction, they can be flipped in the Geometry parameters. Check the `Flip normals` box and notice the difference in the Model view. Uncheck the box since the original STL file was correctly oriented.

- Toggle the Region visibility back on.
- In `Domain extent`, click on Autosize, and adjust the following:
  - Enter `0.0` for the Min Y value.
  - Enter `0.9` for the Max Y value.
  - Enter `0.25` for the Max Z value.

Note that the cyclone inlet and outlets extend beyond the domain extents. This is intentional, as it usually provides a cleaner intersection and easier way to define boundary conditions regions when they are aligned with the domain box (planes along x=xmin, x=xmax etc.).

- Save the project by clicking the button.


Model
MFiX status
Geometry
Color
Style
edges
Opacity
1.00
Show Normals
Scale
0.05
Color
Count
100000
Background Mesh
Color
Style
wire
Opacity
1.00
Boundary Mesh
Color by
bc_id
Style
edges
Opacity
1.00
Regions
Style
solid
Opacity
0.50
Axes


Domain extents
Geometry
Geometry parameters
Extents
Geometry
Background Mesh
Boundary Mesh
Regions
Axes

3.8.3 Create boundary conditions regions

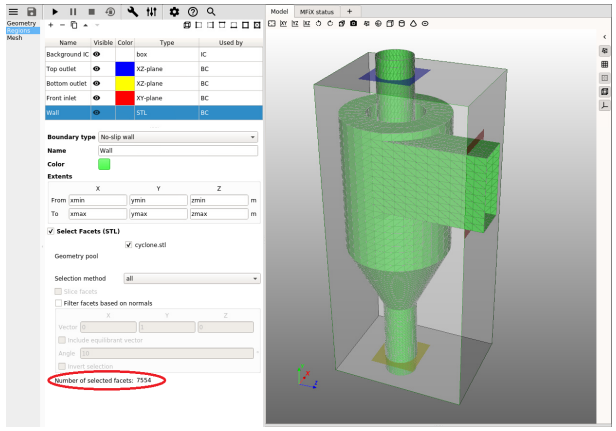
Here, we need to define the mesh boundaries. Boundaries include the cyclone wall (curved surface defined by the STL geometry), and the inlet and outlets, defined along the domain outer box (plane boundaries). A Boundary Condition type must be defined, but it will be possible to change it later (in Modeler Tab). The actual boundary conditions (like the inlet velocity, or outlet pressure) do not need to be set at this point.

Go to the `Regions` pane:

- There is already a `Background IC` that is predefined in the Blank template. We can ignore it for now.
- Click the button to create a new region to be used by pressure outflow boundary condition.
- Select `Pressure outflow` in the Boundary type drop down menu.
- Enter a name for the region in the Name field ("Top outlet").
- Change the region Color to Blue.
- Adjust the From/To coordinates in the x-direction: From -0.07 to 0.07.
- Adjust the From/To coordinates in the z-direction: From -0.07 to 0.07.
- Click the button to create a new region to be used by another pressure outflow boundary condition.
- Select `Pressure outflow` in the Boundary type drop down menu.
- Enter a name for the region in the Name field ("Bottom outlet").
- Change the region Color to Yellow.
- Adjust the From/To coordinates in the x-direction: From -0.07 to 0.07.
- Adjust the From/To coordinates in the z-direction: From -0.07 to 0.07.
- Click the button to create a new region to be used by mass inflow boundary condition.
- Select `Mass inflow` in the Boundary type drop down menu.
- Enter a name for the region in the Name field ("Front inlet").
- Change the region Color to Red.
- Adjust the From/To coordinates in the x-direction: From xmin to -0.025.
- Adjust the From/To coordinates in the y-direction: From 0.55 to 0.80.
- click the button to create a new region that covers the entire domain to be used for the wall boundary condition.
- Select `No-slip wall` in the Boundary type drop down menu.
- Enter a name for the region in the Name field ("Wall").
- Change the region Color to Green.
- Check `Select facets (STL)` and `cyclone.stl`, use `all` for the selection method. You should see that there are 7554 facets selected.

Note: The pressure outlet and mass inflow boundaries are located along the domain box. They are defined as rectangular 2D regions and must overlap the actual boundary areas (intersection of the STL file with the box planes). The exact

boundary area will be computed automatically when preprocessing is performed. Since there is only one Boundary condition on the Top, Bottom and Front planes, we could also have used the entire planes instead of adjusting the region coordinates.



- Save the project by clicking the button.

3.8.4 Setup the mesh

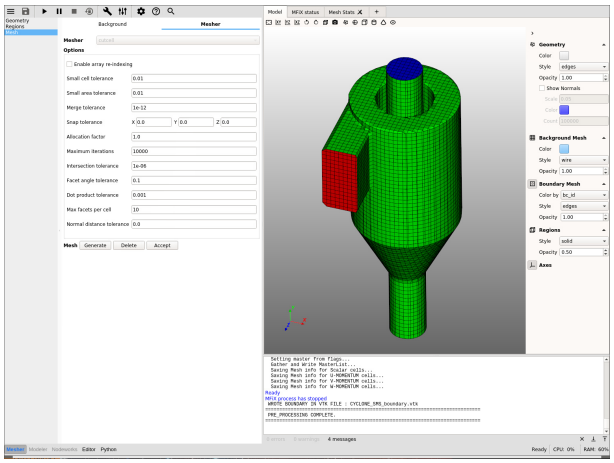
Go to the `Mesh` pane, `Background` sub-pane:

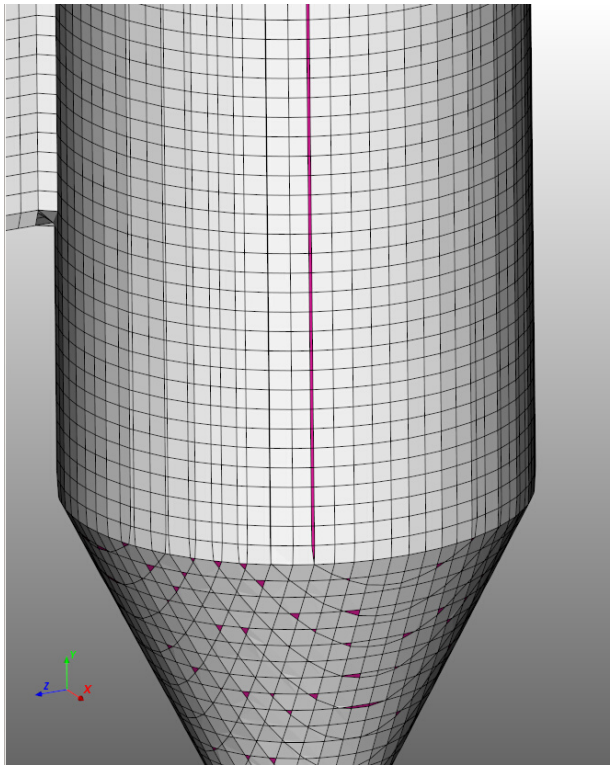
- Enter `25` for the x cell value.
- Enter `70` for the y cell value.
- Enter `33` for the z cell value.

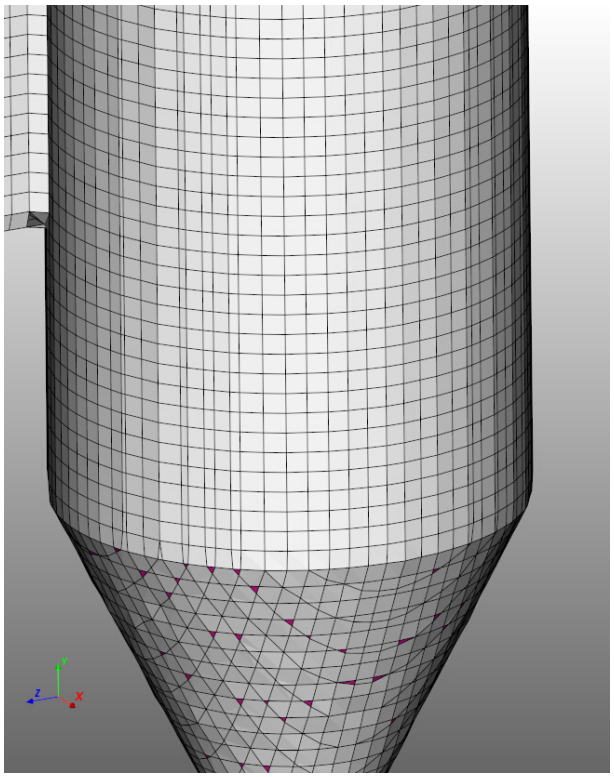
Go to the `Mesh` pane, `Mesher` sub-pane:

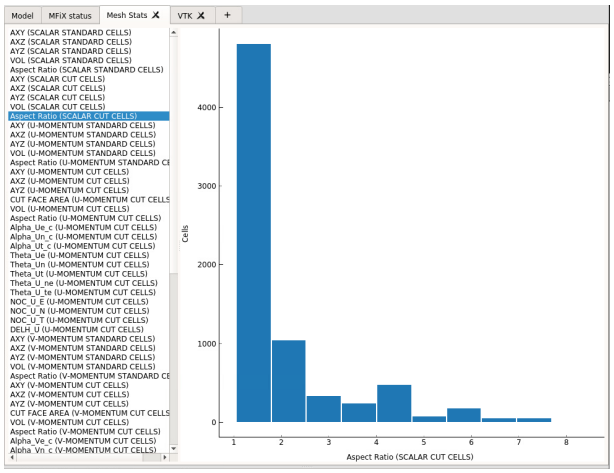
- Keep all default settings and click `Generate`. Select the default solver, and click `Run`
- Look at the console output to verify the mesh generation completed successfully.
- In the Model view, hide the Geometry, Background Mesh and Regions, show the Boundary_Mesh, set Color by to `bc_id`, Style to `edges` and opacity to `1.0`.
- The Boundary_Mesh should look like a closed surface with colors matching the Boundary Conditions regions.
- Color the Boundary_Mesh by `small_cell`. This will show small cells, as defined by the Small cell tolerance. The default value of 0.01 means a cut cell is considered small if its volume is below 0.01 times the volume of the corresponding standard, uncut cell. These small cells will be removed from the computation. Having a Small cell tolerance set to 0.0 will keep all small cells and this may lead to numerical stiffness and loss of convergence.
- Having a few small cells is usually acceptable. In some cases it is possible to reduce the number of small cells by adjusting some cut cell parameters. Increase the Snap tolerance to 0.1 in the x, y, and z-direction. Generate the mesh again, and visualize the small cells. The vertical column of small cells was eliminated.
- Mesh statistics (histogram) can be viewed in the Mesh Stats viewport. Various quantities are available. For example the aspect ratio of cut cells is between 1 and 8, with most aspect ratios between 1 and 2. This is acceptable.

After inspection, the mesh is deemed acceptable and we can move to the second step (Solver mode). Click `Accept`. This will unlock the Modeler tab.


Geometry
Regions
Axes
Background
Mesher
Mesher
cutcell
Options
Enable array re-indexing
Small cell tolerance 0.01
Small area tolerance 0.01
Merge tolerance 1e-12
Snap tolerance X 0.0 Y 0.0 Z 0.0
Allocation factor 1.0
Maximum iterations 10000
Intersection tolerance 1e-06
Facet angle tolerance 0.1
Dot product tolerance 0.001
Max facets per cell 10
Normal distance tolerance 0.0
Mesh Generate Delete Accept
Model MFiX status Mesh Stats
Geometry
Color
Style edges
Opacity 1.00
Show Normals
Scale 0.100
Color
Count 100000
Background Mesh
Color
Style wire
Opacity 1.00
Boundary Mesh
Color by bc_id
Style edges
Opacity 1.00
Regions
Style solid
Opacity 0.50
Axes
Setting master from Flags...
Gather and Write MasterList...
Saving Mesh info for P-CELLS...
Saving Mesh info for U-MOMENTUM cells...
Saving Mesh info for V-MOMENTUM cells...
Saving Mesh info for W-MOMENTUM cells...
Ready
MFiX-exec has stopped
WRITE GEOMBOUNDARY IN VTK FILE : CYCLONE_SMS_boundary.vtk
==========
PRE_PROCESSING COMPLETE
==========
4 messages
Mesher Modeler Nodeworks Editor Python
Ready CPU: 0% RAM: 60%

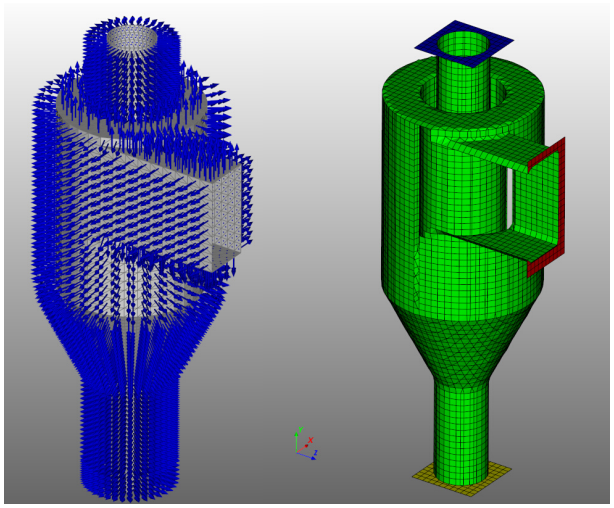



Z
X


Model
MFiX status
Mesh Stats
VTK
AXY (SCALAR STANDARD CELLS)
AXZ (SCALAR STANDARD CELLS)
AYZ (SCALAR STANDARD CELLS)
VOL (SCALAR STANDARD CELLS)
Aspect Ratio (SCALAR STANDARD CELLS)
AXY (SCALAR CUT CELLS)
AXZ (SCALAR CUT CELLS)
AYZ (SCALAR CUT CELLS)
VOL (SCALAR CUT CELLS)
Aspect Ratio (SCALAR CUT CELLS)
AXY (U-MOMENTUM STANDARD CELLS)
AXZ (U-MOMENTUM STANDARD CELLS)
AYZ (U-MOMENTUM STANDARD CELLS)
VOL (U-MOMENTUM STANDARD CELLS)
Aspect Ratio (U-MOMENTUM STANDARD CE
AXY (U-MOMENTUM CUT CELLS)
AXZ (U-MOMENTUM CUT CELLS)
AYZ (U-MOMENTUM CUT CELLS)
CUT FACE AREA (U-MOMENTUM CUT CELLS
VOL (U-MOMENTUM CUT CELLS)
Aspect Ratio (U-MOMENTUM CUT CELLS)
Alpha_Ue_c (U-MOMENTUM CUT CELLS)
Alpha_Un_c (U-MOMENTUM CUT CELLS)
Alpha_Ut_c (U-MOMENTUM CUT CELLS)
Theta_Ue (U-MOMENTUM CUT CELLS)
Theta_Un (U-MOMENTUM CUT CELLS)
Theta_Ut (U-MOMENTUM CUT CELLS)
Theta_U_ne (U-MOMENTUM CUT CELLS)
Theta_U_te (U-MOMENTUM CUT CELLS)
NOC_U_E (U-MOMENTUM CUT CELLS)
NOC_U_N (U-MOMENTUM CUT CELLS)
NOC_U_T (U-MOMENTUM CUT CELLS)
DELH_U (U-MOMENTUM CUT CELLS)
AXY (V-MOMENTUM STANDARD CELLS)
AXZ (V-MOMENTUM STANDARD CELLS)
AYZ (V-MOMENTUM STANDARD CELLS)
VOL (V-MOMENTUM STANDARD CELLS)
Aspect Ratio (V-MOMENTUM STANDARD CE
AXY (V-MOMENTUM CUT CELLS)
AXZ (V-MOMENTUM CUT CELLS)
AYZ (V-MOMENTUM CUT CELLS)
CUT FACE AREA (V-MOMENTUM CUT CELLS
VOL (V-MOMENTUM CUT CELLS)
Aspect Ratio (V-MOMENTUM CUT CELLS)
Alpha_Ve_c (V-MOMENTUM CUT CELLS)
Alpha_Vn_c (V-MOMENTUM CUT CELLS)
Cells
4000
3000
2000
1000
0
1
2
3
4
5
6
7
8
Aspect Ratio (SCALAR CUT CELLS)

Note: Optional but recommended: To see examples of unacceptable meshes, try the following:

- Go back to the Geometry pane, and flip the normals. This will incorrectly orient the STL file. This is a common error when importing STL files. Generate the mesh and visualize the Boundary_Mesh. Having an open surface and boundaries inside-out is an indication that the normals are not properly oriented. The mesh should not be accepted.



- Uncheck the `Flip normals` box in the Geometry pane to get correct orientation. Go to the Region pane, and select the Front inlet region in the region table and set the Boundary type to `None`. Save the project and generate the mesh. The Boundary_Mesh will look like an open surface, this is an indication a Boundary condition is missing. The mesh should not be accepted.

Remember to set the Front inlet Region boundary type to `Mass inflow` again before continuing. Generate the mesh, inspect it and accept it before moving to the Modeler tab.

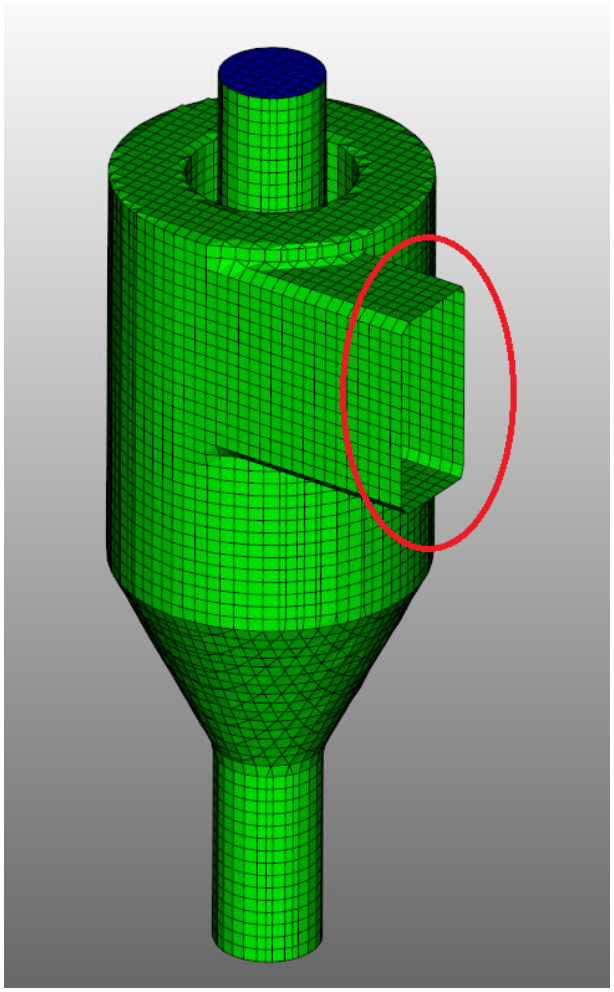
3.8.5 Model settings

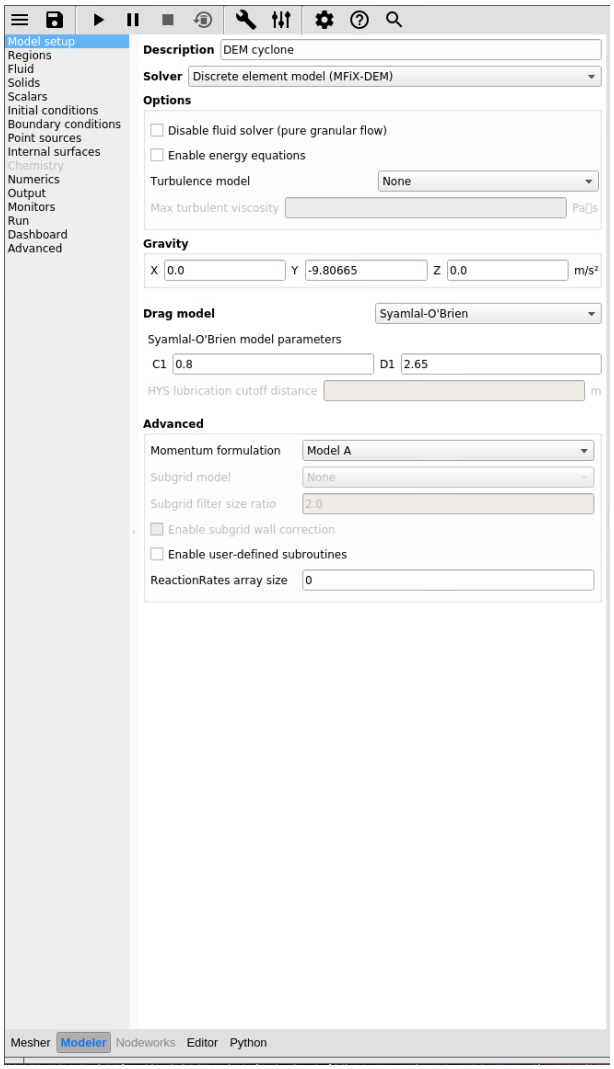
Switch to the Modeler tab (second tab at the bottom left corner of the GUI).

- On the `Model` pane, enter a descriptive text in the `Description` field.
- Select "Discrete Element Model (MFiX-DEM)" in the `Solver` drop-down menu.
- Keep all other settings at their default values.

3.8.6 Regions settings

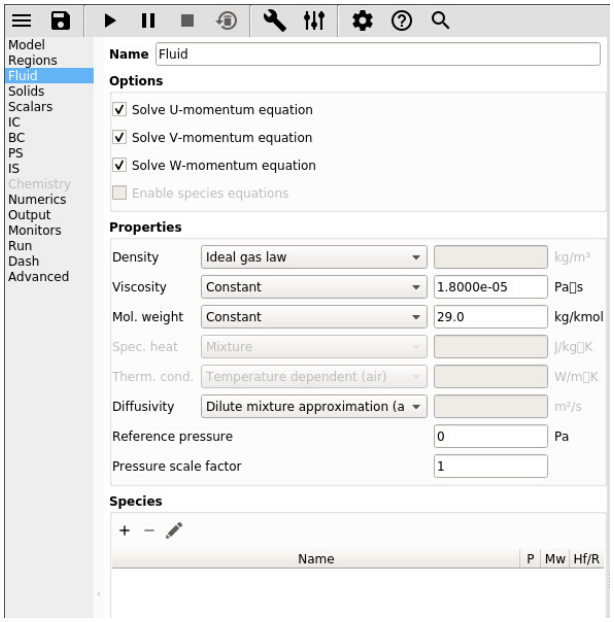
We already have the Background_IC region that is predefined. We will use it to initialize the flow field. We have defined all Boundary Condition regions during the meshing step. There are no other regions that need to be defined in this tutorial. We will re-use Background_IC region for the VTK output file.




Model setup
Regions
Fluid
Solids
Scalars
Initial conditions
Boundary conditions
Point sources
Internal surfaces
Chemistry
Numerics
Output
Monitors
Run
Dashboard
Advanced
Description
DEM cyclone
Solver
Discrete element model (MFiX-DEM)
Options
Disable fluid solver (pure granular flow)
Enable energy equations
Turbulence model
None
Max turbulent viscosity
Pa s
Gravity
X
0.0
Y
-9.80665
Z
0.0
m/s²
Drag model
Syamlal-O'Brien
Syamlal-O'Brien model parameters
C1
0.8
D1
2.65
HYS lubrication cutoff distance
m
Advanced
Momentum formulation
Model A
Subgrid model
None
Subgrid filter size ratio
2.0
Enable subgrid wall correction
Enable user-defined subroutines
ReactionRates array size
0
Mesher
Modeler
Nodeworks
Editor
Python

3.8.7 Fluid settings

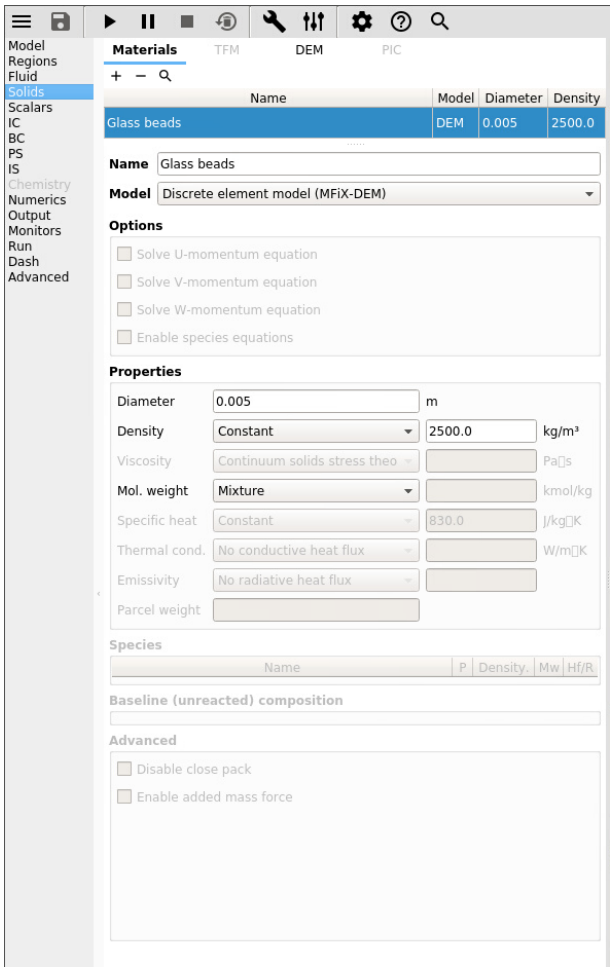
- Change Density to `Ideal gas law`.
- Keep all other settings at their default values.

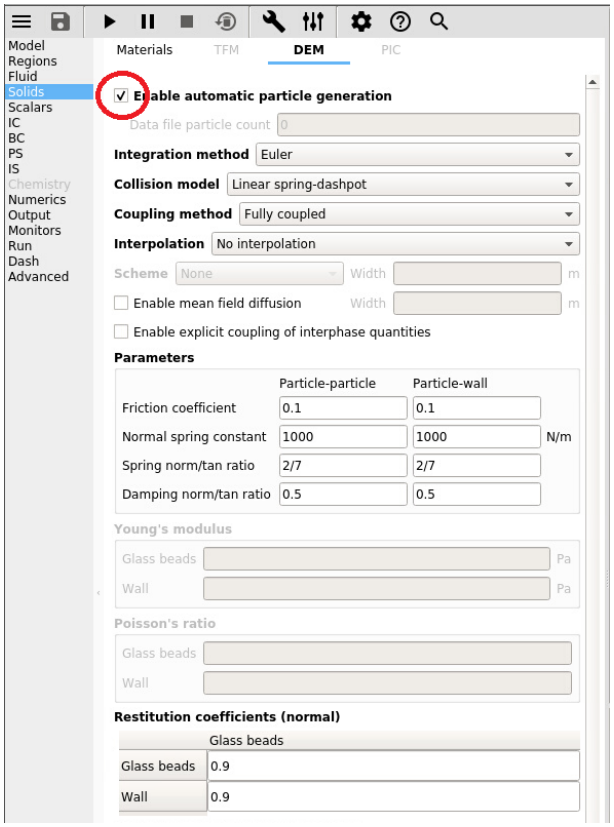


3.8.8 Solids settings

On the `Solids` pane, `Materials` sub-pane:

- Click the button to create a new solid.
- Enter a descriptive name in the `Name` field ("Glass beads").
- Verify the solids model is already set to "Discrete Element Model (MFiX-DEM)".
- Enter the particle diameter of `0.005` m in the `Diameter` field.
- Enter the particle density of `2500` kg/m^3 in the `Density` field.
- Select the `Solids` pane, `DEM` sub-pane.
- Check the `Enable automatic particle generation` checkbox. Although we will start with an empty cyclone, this setting is necessary so we don't attempt to read initial particle location from `particle_input.dat`.
- Keep all other settings at their default values.


Model
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Materials
TFM
DEM
PIC
Name
Model
Diameter
Density
Glass beads
DEM
0.005
2500.0
Name
Glass beads
Model
Discrete element model (MFiX-DEM)
Options
Solve U-momentum equation
Solve V-momentum equation
Solve W-momentum equation
Enable species equations
Properties
Diameter
0.005
m
Density
Constant
2500.0
kg/m³
Viscosity
Continuum solids stress theo
Pa□s
Mol. weight
Mixture
kmol/kg
Specific heat
Constant
830.0
J/kg□K
Thermal cond.
No conductive heat flux
W/m□K
Emissivity
No radiative heat flux
Parcel weight
Species
Name
P
Density.
Mw
Hf/R
Baseline (unreacted) composition
Advanced
Disable close pack
Enable added mass force



3.8.9 Initial Conditions settings

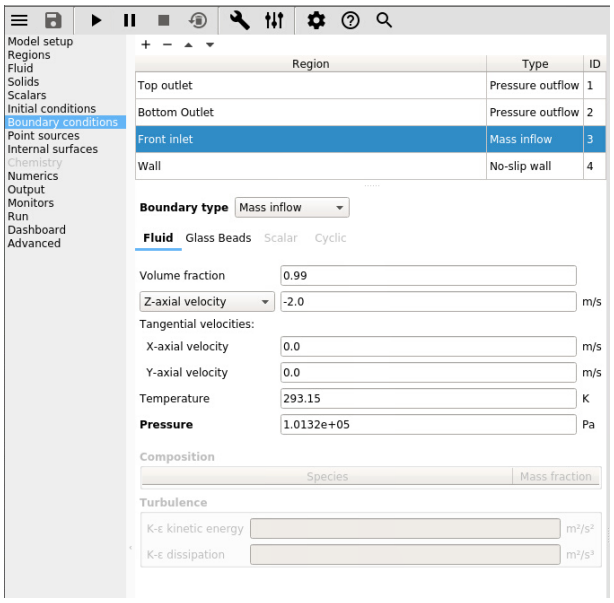
On the `Initial conditions` pane:

- There is already a pre-defined "Background IC" region. This will initialize the entire flow field with air at rest.
- Keep all settings at their default values.

3.8.10 Boundary Conditions settings

On the `Boundary conditions` pane:

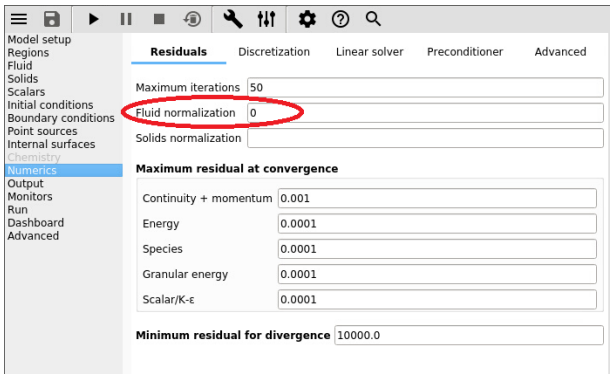
- The Top outlet and Bottom outlet Boundary Conditions are already set to atmospheric pressure. No changes are needed.
- The Wall Boundary Conditions is already set to No-slip wall. No changes are needed.
- Select the Front Inlet region from the list.
  - In the Glass beads tab: set the Volume fraction to 0.01, and Z-axial velocity to -2.0.
  - In the Fluid tab, verify the Gas volume fraction was automatically set to 0.99 and set the Z-axial velocity to -2.0.
- Keep all other settings at their default values.



3.8.11 Numerics settings

On the `Numerics` pane:

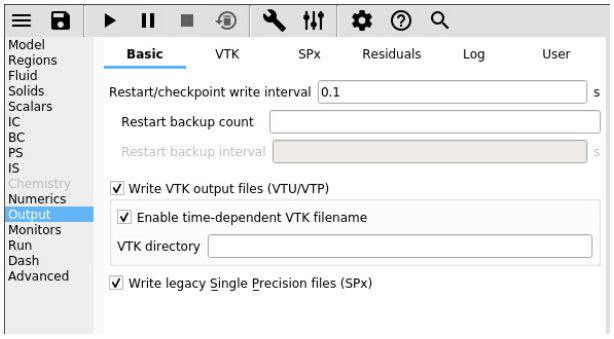
- Set the Fluid normalization to 0.0.
- Keep all other settings at their default values.

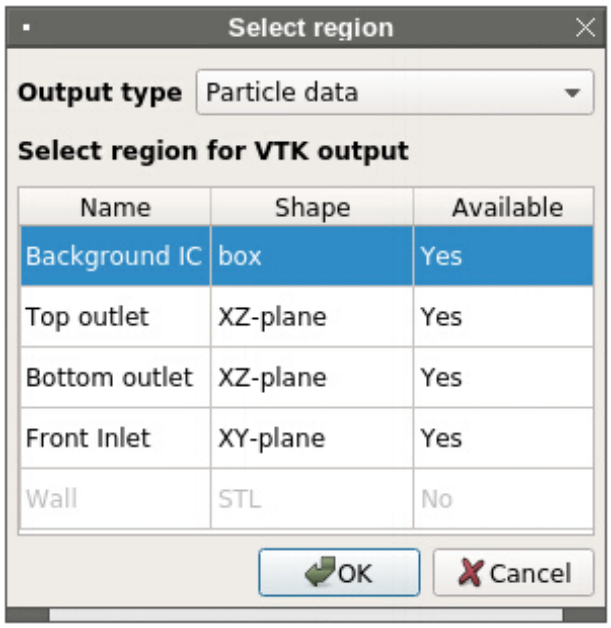


3.8.12 Output settings

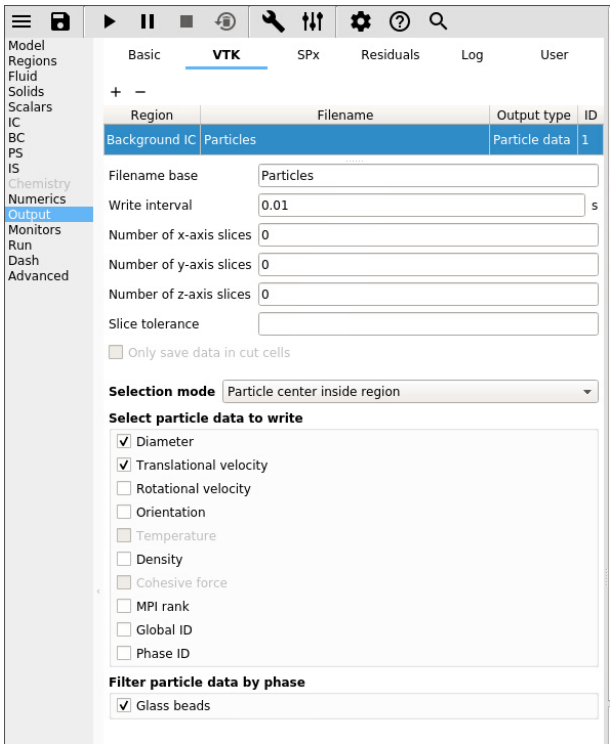
On the `Output` pane:

- On the `Basic` sub-pane, check the `Write VTK output files (VTU/VTP)` checkbox.
- Select the `VTK` sub-pane.
- Create a new output by clicking the button.
- Select "Particle Data" from the 'Output type' drop-down menu.
- Select the "Background IC" region from the list to save all the particle data.
- Click `OK` to create the output.


Model
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Basic
VTK
SPx
Residuals
Log
User
Restart/checkpoint write interval
0.1
s
Restart backup count
Restart backup interval
s
Write VTK output files (VTU/VTP)
Enable time-dependent VTK filename
VTK directory
Write legacy Single Precision files (SPx)


Select region
Output type
Particle data
Select region for VTK output
Name
Shape
Available
Background IC
box
Yes
Top outlet
XZ-plane
Yes
Bottom outlet
XZ-plane
Yes
Front Inlet
XY-plane
Yes
Wall
STL
No
OK
Cancel

- Enter a base name for the `*.vtu` files in the `Filename base` field ("Particles").
- Change the `Write interval` to `0.01` seconds.
- Select the `Diameter` and `Translational Velocity` check-boxes.



3.8.13 Run settings

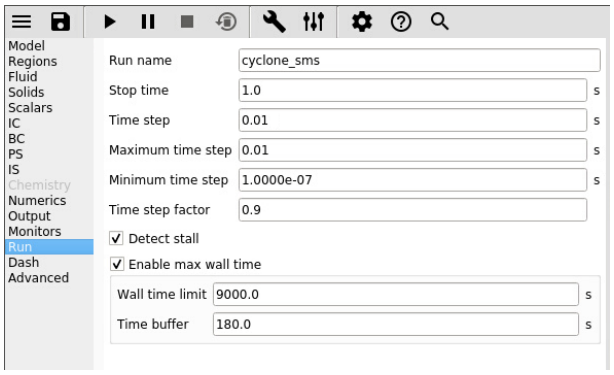
Save the project by clicking the button.

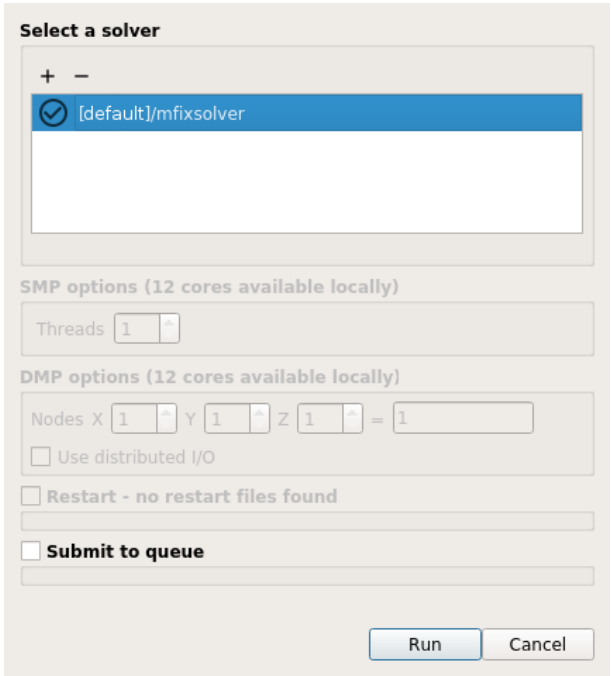
On the `Run` pane:

- Set the Stop time to `1.0`.
- Set the Time step to `0.01`.
- Set the Maximum time step to `0.01`.
- Keep all other settings at their default values.

3.8.14 Run the simulation

- Run the project by clicking the button.
- On the `Run` dialog, select the executable from the combo-box.
- Click the `Run` button to actually start the simulation.

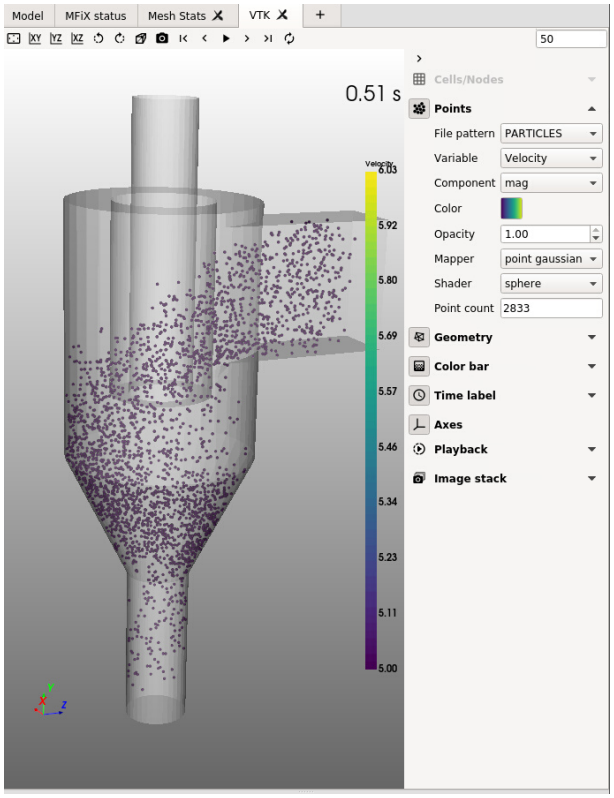

Model
Regions
Fluid
Solids
Scalars
IC
BC
PS
IS
Chemistry
Numerics
Output
Monitors
Run
Dash
Advanced
Run name
cyclone_sms
Stop time
1.0
s
Time step
0.01
s
Maximum time step
0.01
s
Minimum time step
1.0000e-07
s
Time step factor
0.9
Detect stall
Enable max wall time
Wall time limit
9000.0
s
Time buffer
180.0
s


Select a solver
[default]/mfixsolver
SMP options (12 cores available locally)
Threads
1
DMP options (12 cores available locally)
Nodes X
1
Y
1
Z
1
=
1
Use distributed I/O
Restart - no restart files found
Submit to queue
Run
Cancel

3.8.15 View results

Results can be viewed, and plotted, while the simulation is running.

- Create a new visualization tab by pressing the next to the *Model* tab.
- Click the `3D view` button to view the vtk output files.
- On the `VTK` results tab, the visibility and representation of the `*.vtk` files can be controlled with the menu on the side.
- Change frames with the , , , and buttons.
- Click the button to play the available vtk files.
- Change the playback speed under the section on the sidebar.



3.9 DEM Initial Conditions

This tutorial will illustrate various seeding options when initializing particles in an initial condition region. It is based on the 3d hopper tutorial. We will only change the initial conditions.

3.9.1 Open the 3d hopper tutorial

- On the main menu, select `New project`
- Double click on the "3D DEM granular flow hopper" tutorial and specify the project location.
- Run the project as is to get familiar with it.

3.9.2 Change the initial condition configuration

- On the `Initial conditions` pane, select the "initial solids" region.
- Change the Solid 1 volume fraction to 0.4. Note that we are using an hexagonal lattice.
- On the `Run` pane, change `Stop time` to 0.001. We are only interested to look at the initial conditions.
- Run the simulation. Visualize the particles and take a look at the output in the console and verify that seeding was successful:

```
Generating initial particle configuration:
Phase  1: Number of particles to seed =        18236
|-------------------------------------------------------------------|
| IC Region:   2                                                    |
|-------------------------------------------------------------------|
| Phase |   Input    |   Input    | Number of |    EPs     |   Solid    |
|   ID  | Specified  |   Value    | Particles |            |   mass     |
|-------|------------|------------|-----------|------------|------------|
|    1  |   EP_S     |  4.00E-01  |     18236 |  4.00E-01  |  1.91E+02  |
|-------|------------|------------|-----------|------------|------------|
Phase  1: Successful seeding
```

- On the `Initial conditions` pane, select the "initial solids" region.
- Change the lattice configuration to "Cubic".
- Run the simulation. Visualize the particles and take a look at the output in the console and notice that with the cubic lattice, we are not able to reach the specified volume fraction of 0.4.

```
Generating initial particle configuration:
Phase  1: Number of particles to seed =        18236
|-------------------------------------------------------------------|
| IC Region:   2                                                    |
|-------------------------------------------------------------------|
| Phase |   Input    |   Input    | Number of |    EPs     |   Solid    |
|   ID  | Specified  |   Value    | Particles |            |   mass     |
|-------|------------|------------|-----------|------------|------------|
|    1  |   EP_S     |  4.00E-01  |     15693 |  3.44E-01  |  1.64E+02  |
|-------|------------|------------|-----------|------------|------------|
Phase  1: Unsuccessful seeding
```

Instead of specifying a solids fraction, we can input a solids mass or a number of particles:

- On the `Initial conditions` pane, select the "initial solids" region.

- In the drop down menu, select "Inventory" and enter `100.0`.
- Run the simulation and verify seeding was successful.
- On the `Initial conditions` pane, select the "initial solids" region.
- In the drop down menu, select "Particle count" and enter `10000`.
- Run the simulation and verify seeding was successful.

The spacing between particles can also be increased:

- On the `Initial conditions` pane, select the "initial solids" region.
- Change the Solid 1 volume fraction to `0.1`.
- Change the spacing to `0.5`, run the simulation and observe how particles are more spaced out.
- Change the random factor to `1.0`, run the simulation and observe how particles have randomly shifted from their lattice positions.

Repeat the above (spacing option) with the hexagonal lattice.

3.9.3 Particle size distribution

- On the `Initial conditions` pane, select the "initial solids region.
- Make sure the Solid 1 volume fraction is `0.1`, and the lattice is `Hexa`.
- Change the spacing back to `0.05`, and the random factor back to `1.0`.
- In the Solids>DEM pane, Click the button in the Particle size distribution table (the table is initially empty if no psd have been defined yet):

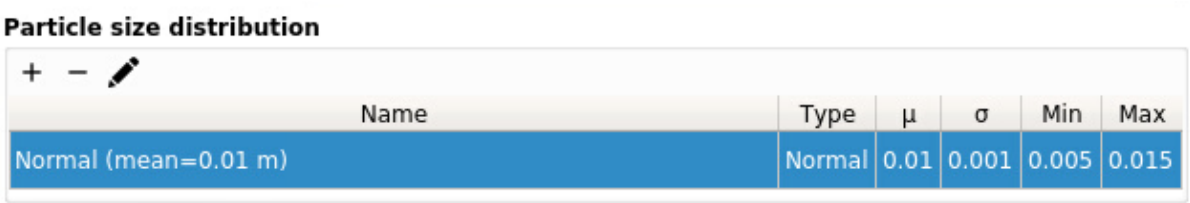


Fig. 3.1: PSD table.

- Define a Normal psd:
  - Name: `Normal (mean = 0.01 m)` (use a descriptive name).
  - Distribution type: `Normal`
  - Mean: `0.01`
  - Standard deviation: `0.001`
  - Minimum: `0.005`
  - Maximum: `0.015`
  - Plot from 0 to `0.020`
- Plot the psd by clicking on `Plot distribution`. Verify the plot corresponds to the parameters.
- Click `OK`.

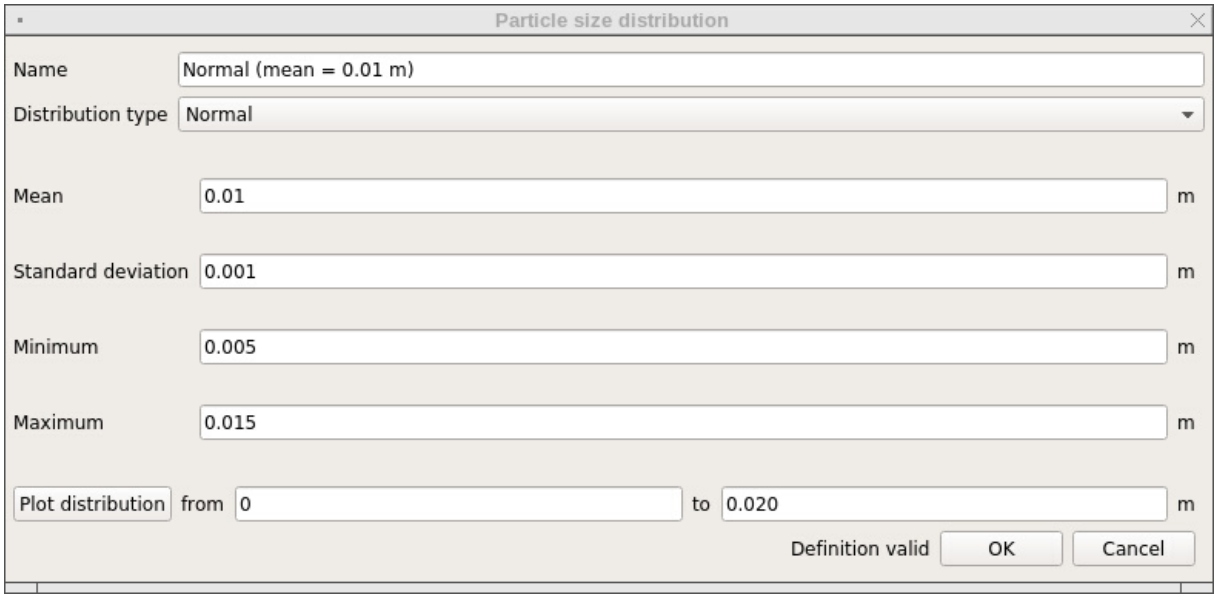


Fig. 3.2: Normal PSD definition.

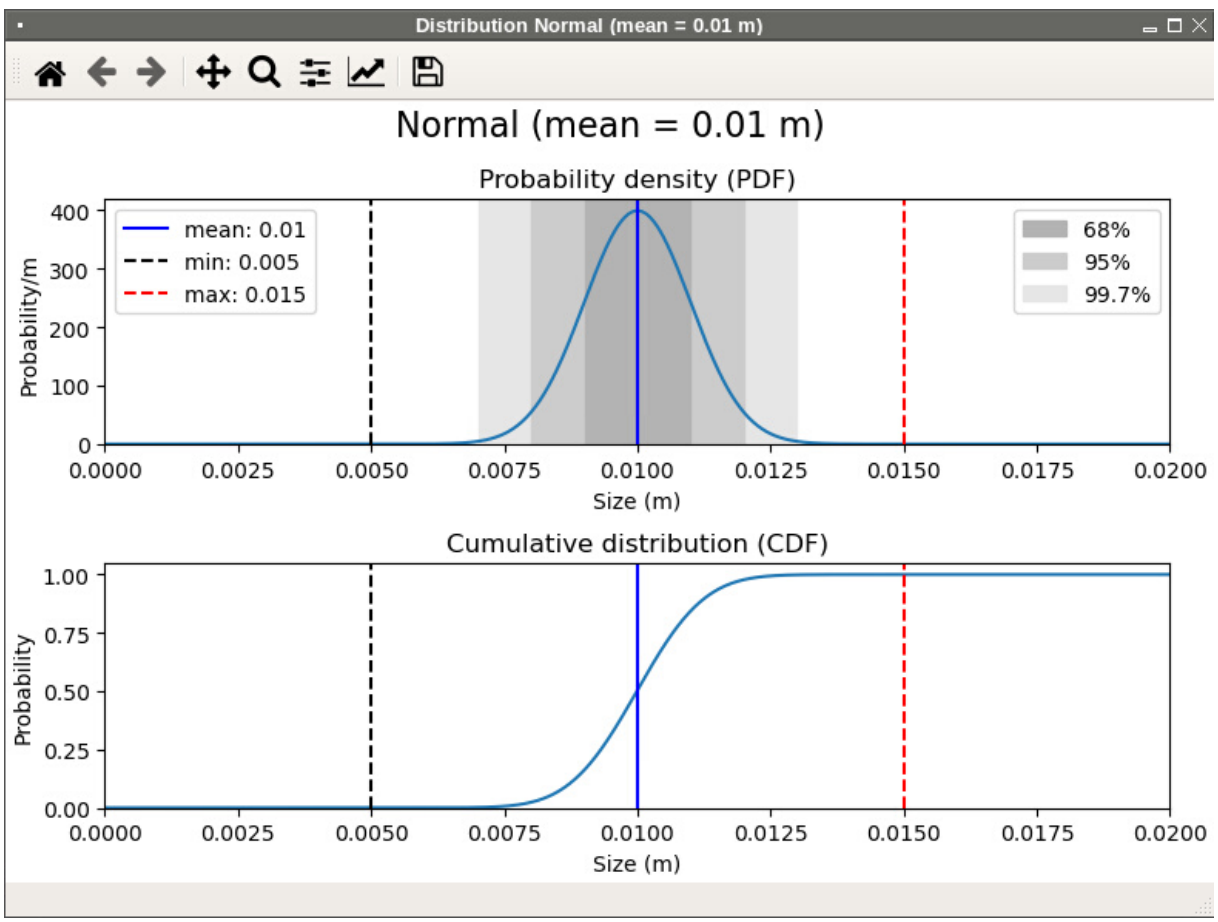


Fig. 3.3: Normal PSD plot.

We now have a psd that we can use in initial or mass inlet boundary conditions.

- Go to the `Initial condition` pane, go to the drop down menu under 'Particle size distribution' and select the psd we just defined, `Normal (mean = 0.01 m)`.
- Run the simulation and visualize the particles.

Besides the Normal and Log-normal psd's, we can also use custom distribution from a text file. It can be edited through any text editor including the GUI built-in editor. The PSD custom file is organized as follows:

1st line: Number of points, number of distribution (this must be set to one currently).
2nd line: PDF (probability density function) of CDF (Cumulative distribution function).
3rd line: Header.
4th line and below: The data itself. First column is the diameter, second column is either the PDF or CDF value.

The figure below shows an example of a custom PSD file. Only the first few lines are shown.

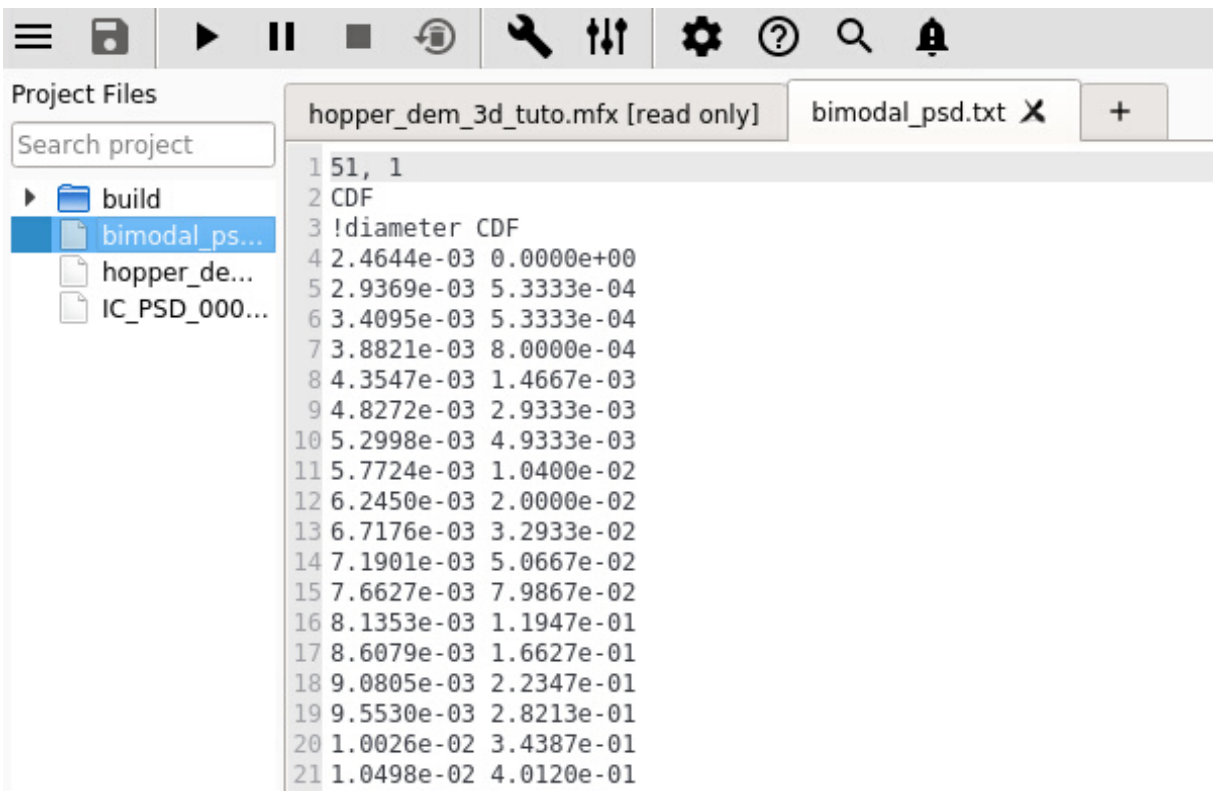


Fig. 3.4: Example of a Custom file used to define a Particle Size Distribution.

- In the Solids>DEM pane, Click the button in the Particle size distribution table, and define a custom psd:
- Name: `Bimodal psd` (use a descriptive name).
- Distribution type: `Custom`
- Select file: Choose `bimodal_psd.txt` in the project directory.
- Click `Plot distribution`.

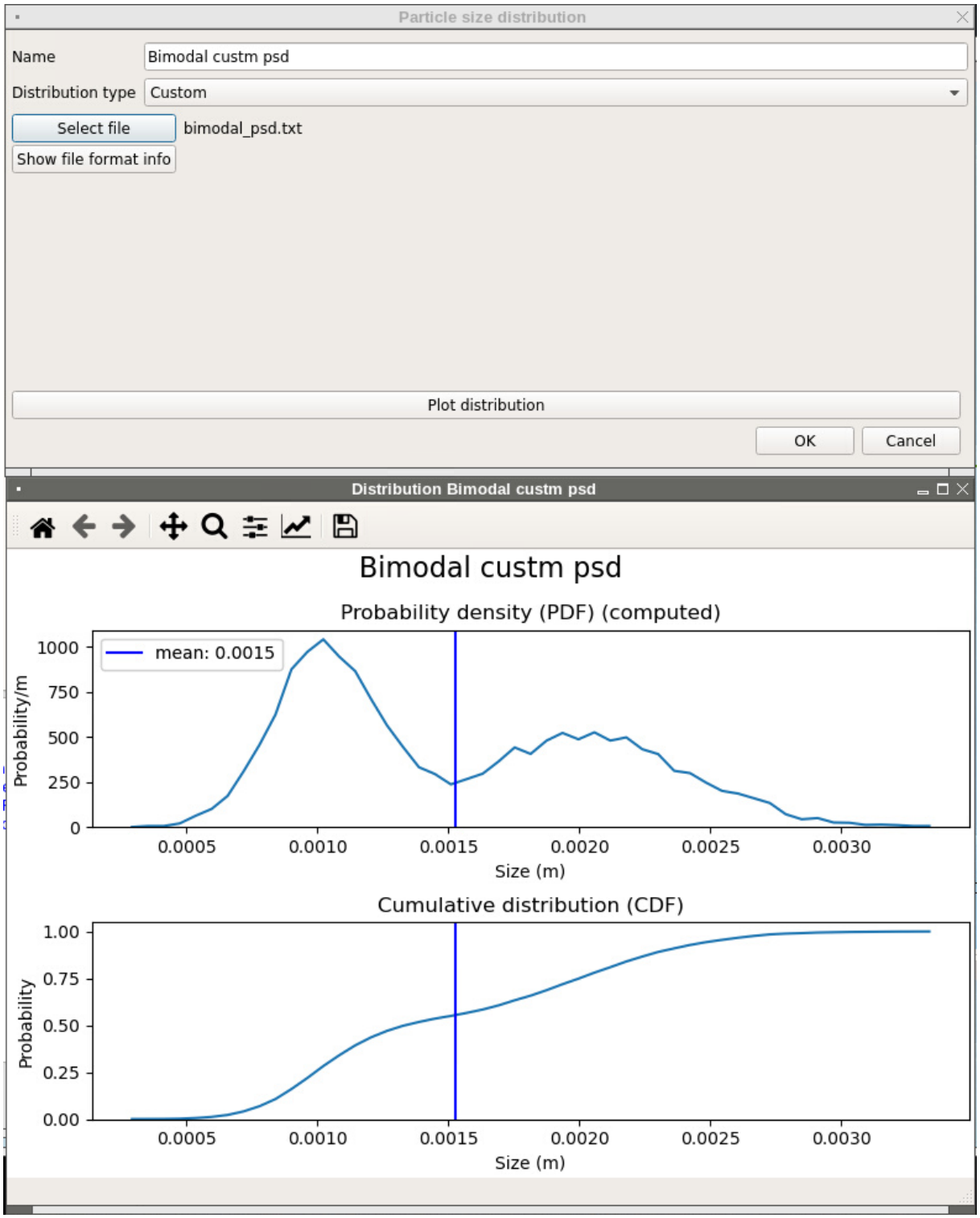


Fig. 3.5: Custom PSD.

- Run the simulation and visualize particles.
- Notice that we are not able to seed particles with the requested solids volume fraction of 0.1. This is because the lattice spacing is based on the largest diameter in the psd.
- Change the spacing to `0.02`, run the simulation and verify that we get a volume fraction of 0.1 (or near 0.1).
- While the simulation is running, we can visualize a histogram showing the particle size distribution. Open a new visualization viewport by clicking on the + sign on the right-most tab and select the histogram view. This will setup a histogram plot. Users can change the file to view, the variable, number of bins and style (lines or bars). To adjust the scale, right-click on the view and enter the axis limits or have them be automatically updated based on data.

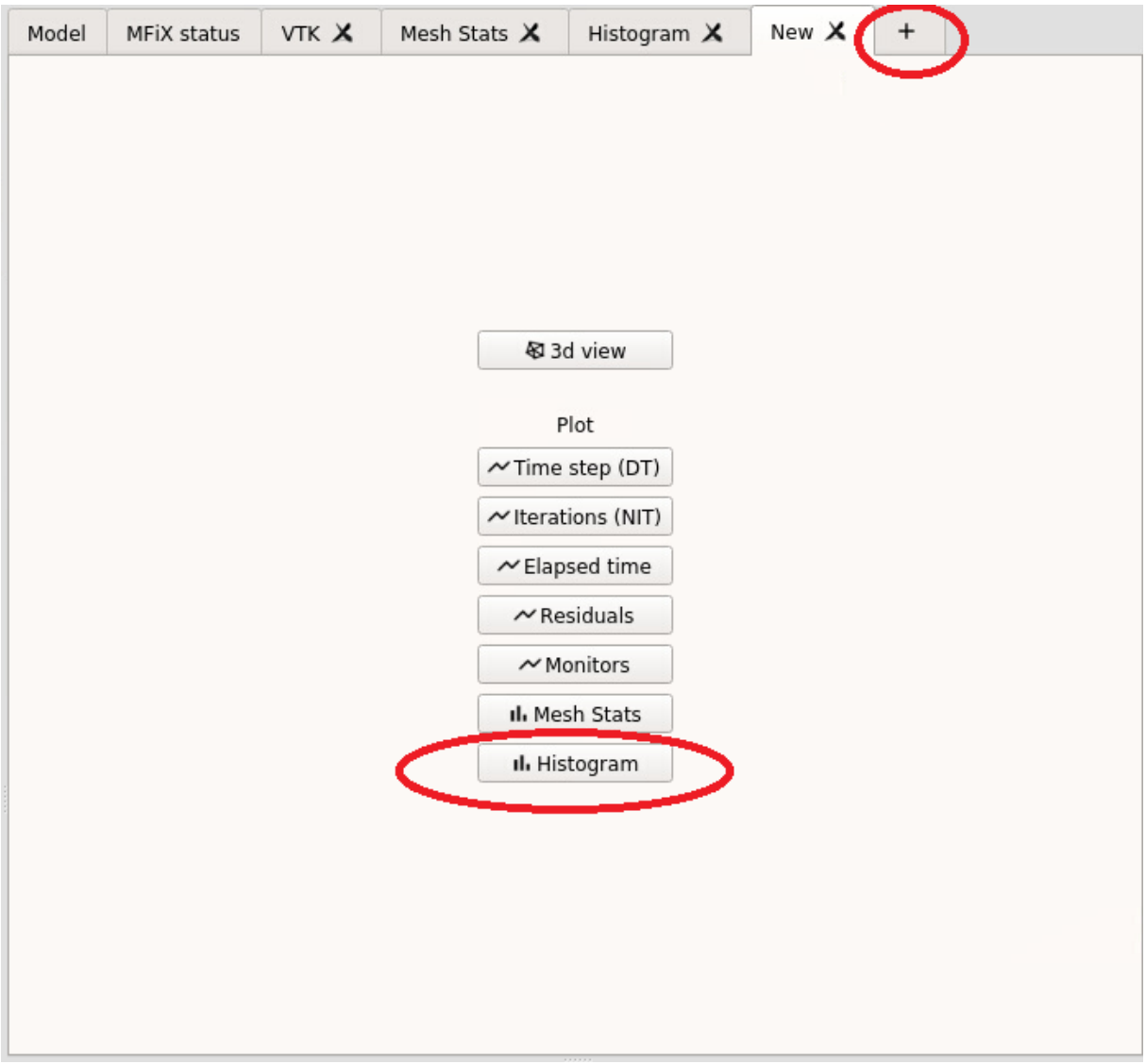


Fig. 3.6: Opening a histogram view.

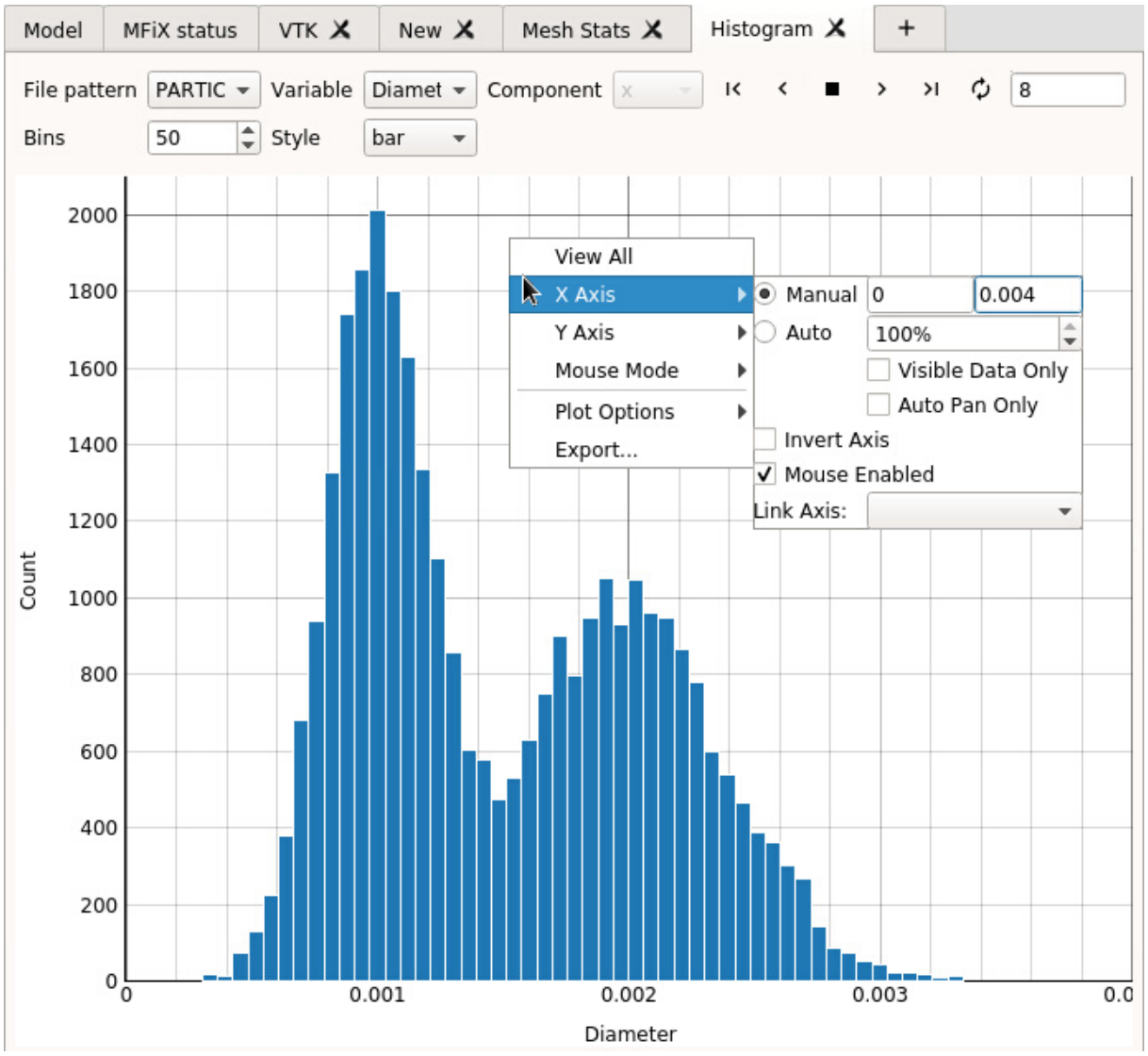


Fig. 3.7: Visualizing particle diameter histogram.

3.9.4 Reading initial particle data from a text file

We can read the initial particle data from a text file, called `particle_input.dat`. The format of this file has changed in the 20.3 release. The old file version can still be used but it is recommended to use the latest format.

The new format allows to either read variables or set a constant value for all particles. The file contains a long header where information about the data is specified. This controls how many columns and which variables are read from the file. Next, the data section contains the data for each particle. Each column corresponds to a variable. In 3D, the first 3 columns are always the x,y and z coordinates of the particle's center. In 2D, the first 2 columns are always the x and y coordinates of the particle's center. There are two options for the other variables: either read the data for each particle in the next column, or assign a constant value to all particles.

Below is an example of a `particle_input.dat` file (the line number is shown on the left and it not part of the data itself):

- Line 1 : File version. Do not change this line.
- Lines 2-18 : Instructions to use the file.
- Line 19 : This is a 3D geometry. The first 3 columns in the data section will be x,y,z.
- Line 20 : We will read data for 35361 particles.
- Line 24 : We always read coordinates in the first 2 or 3 columns (do not change)
- Line 25 : We do not read particle's phase ID for each particle. It is assigned a constant value: 1
- Line 26 : We will read the particle's diameter in the next column (column#4).
- Line 27 : We do not read particle's density. It is assigned a constant value: 2500.0 kg/m^3
- Line 25 : We do not read particle's velocity. It is assigned a constant value: (u,v,w) = (0.0,0.0,0.0) m/s
- Lines 29-30: This is a cold, non reacting case, we do not read Temperature or species.
- Line 31 : We do not have user scalars, so we do not read them.
- Lines 32-34: Data header

The data starts on Line 35. Based on the above settings, we need to read, x,y,z, and the diameter of each particles, i.e., 4 columns, for 35361 particles. Only the first few lines of the data is shown.

```
1 Version 2.0
2 ======================================================================
3 Instructions:
4 Dimension: Enter "2" for 2D, "3" for 3D (Integer)
5 Particles: Number of particles to read from this file (Integer)
6 For each variable, enter whether it will be read from the file
7 ("T" to be read (True), "F" to not be read (False)). When not read, enter the
8 default values assign to all particles.
9 Coordinates are always read (X,Y in 2D, X,Y,Z in 3D)
10 Phase_ID, Diameter, Density, Temperature are scalars
11 Velocity requires U,V in 2D, U,V,W in 3D
12 Temperature is only read/set if the energy equation is solved
13 Species are only read/set if the species equations are solved, and requires
14 all species to be set (if phase 1 has 2 species and phase 2 has 4 species,
15 all particles must have 4 entries, even phase 1 particles (pad list with zeros)
16 User scalars need DES_USR_VAR_SIZE values
17 Data starts at line 35. Each column correspond to a variable.
18 ======================================================================
19 Dimension:    3
20 Particles:    35361
```

(continues on next page)

(continued from previous page)

```
21 ========================================================================
22 Variable       Read(T/F)      Default value (when Read=F)
23 ========================================================================
24 Coordinates:     T            Must always be T
25 Phase_ID:        F            1
26 Diameter:        T
27 Density:         F            2500.0
28 Velocity:        F            0.0 0.0 0.0
29 Temperature:     T            (Ignored if energy eq. is not solved)
30 Species:         T            (Ignored if species eq. are not solved)
31 User_Scalar:     T            (Ignored if no user scalars are defined)
32 ========================================================================
33     X (m)            Y (m)            Z (m)        Diameter (m)
34 ========================================================================
35 -5.07749E-02    -7.98147E-01    -2.00612E-01    9.71052E-03
36 -4.86164E-02    -8.16202E-01    -2.25035E-01    8.34185E-03
37 -3.16460E-02    -8.13702E-01    -2.19164E-01    1.09747E-02
38 -3.32833E-02    -8.25178E-01    -2.19393E-01    9.67983E-03
39 -2.48844E-02    -8.30159E-01    -2.16423E-01    9.84182E-03
40 -8.57591E-03    -8.38121E-01    -2.08773E-01    9.92759E-03
41 -2.75312E-03    -8.32665E-01    -2.00770E-01    1.07560E-02
42 1.57276E-02     -8.37839E-01    -2.08196E-01    1.04387E-02
43 1.96605E-02     -8.31447E-01    -2.15105E-01    9.97808E-03
44 2.71083E-02     -8.26450E-01    -2.19492E-01    9.99619E-03
45 3.55440E-02     -8.28885E-01    -2.15687E-01    9.15771E-03
46 4.45244E-02     -8.28239E-01    -2.11771E-01    1.05089E-02
47 4.70488E-02     -8.22092E-01    -2.19129E-01    9.33350E-03
```

This data files contains particles that have settled in the top half of the hopper. To use it as initial conditions:

- In the Solids>DEM pane, uncheck `Enable automatic particle generation`, and enter a data file particle count of `35361`. An entry is required to maintain compatibility with the old format, but the value on line 20 of the version 2.0 file will actually be used.
- In the `Initial condition` pane, delete the `Initial solids` initial condition.
- In the `Run` pane, set the stop time to 1.0 seconds.
- Run the simulation. You should see the initial configuration shown below:

3.10 Advanced Tutorials

The following tutorials demonstrate advanced features that can be implemented with user defined files (UDFs). They require editing Fortran source files (`usr_*.f` or other `*.f` files) and building the custom solver.

- Reacting flows
  - Silane Pyrolysis (2D TFM): Setup TFM reaction rates.
  - Silane Pyrolysis (3D PIC): Setup PIC/DEM reaction rates.
  - Absorption column (2D TFM): Setup TFM reaction rates and user-defined properties.
- Transient keyframe data
  - Conveyor belts (Granular DEM): Impose tangential velocity to wall. `Download pdf file`
  - Rotating drum (Granular DEM): Impose tangential velocity to wall. `Download pdf file`

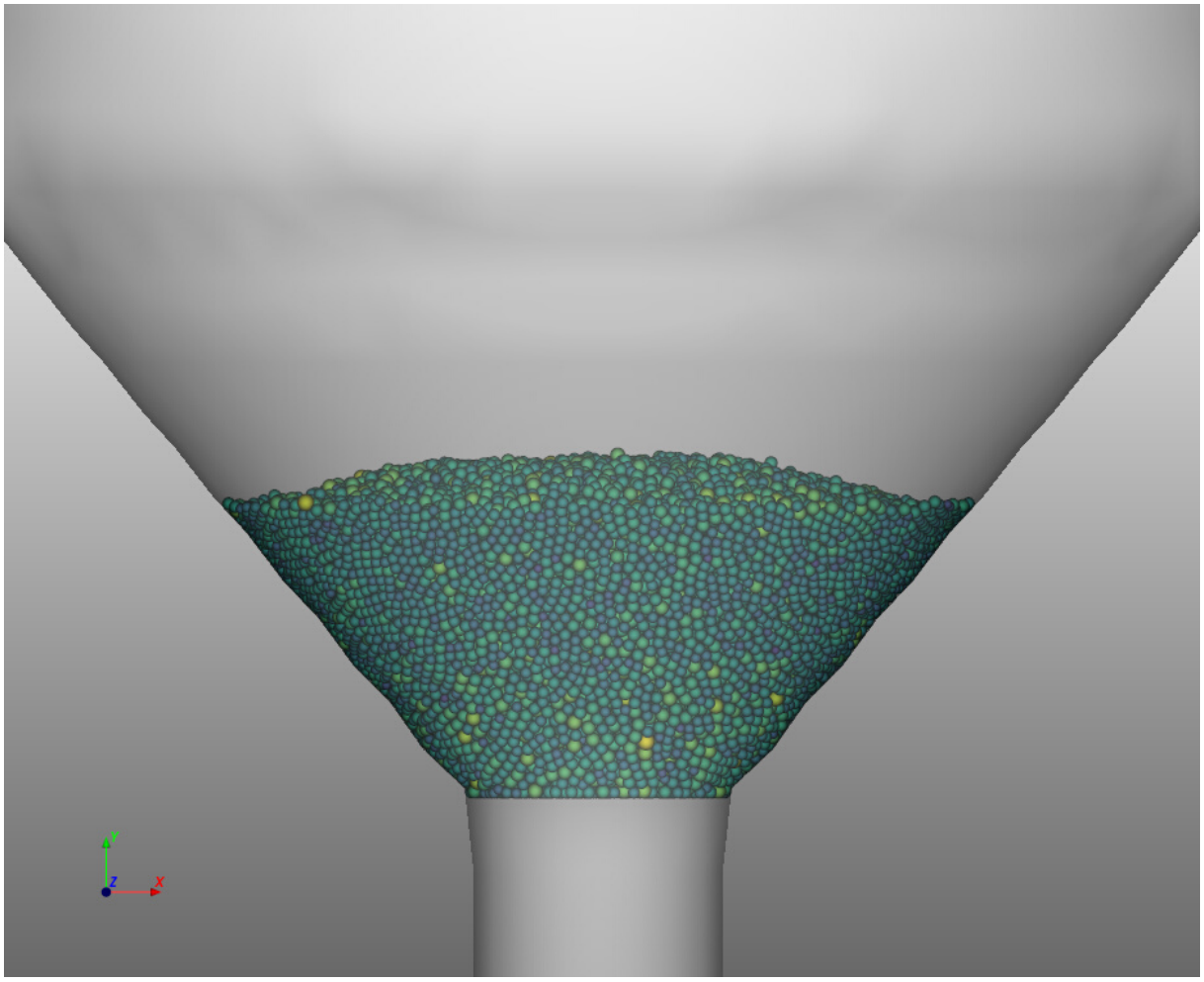


Fig. 3.8: Example of a DEM initial condition using particle_input.dat version 2.0.

- Screwfeeder (Granular DEM): Control a group of particles to represent moving geometry. `Download pdf file`
- Pulsating fluidized bed (TFM): Impose fluctuating inlet velocity `Download pdf file`

CHAPTER

FOUR

MODEL GUIDE

4.1 Model Setup

The Model pane is used to specify global project settings. Depending on what is selected, other panes are enabled or disabled.

- **Description** allows for a short model description to be provided. This is written in the .OUT file by the solver.
- **Solver** specifies the model solver.
  - *Single phase* is the MFiX fluid solver. This disables all solids model inputs.
  - *Two-Fluid Model (MFiX-TFM)* treats both the fluid and solids as interpenetrating continua.
  - *Discrete Element Model (MFiX-DEM)* treats the fluid as a continuum while modeling individual particles and collisions.
  - *Particle in Cell Model (MFiX-PIC)* treats the fluid as a continuum while using "parcels" to represent groups of real particles with similar physical characteristics.

- **Disable the fluid phase** turns off the fluid solver for MFiX-TFM and MFiX-DEM simulations for pure granular flows. The fluid solver cannot be disabled for single phase flows.
- **Enable thermal energy equations** solves thermal transport equations for all phases.
- **Turbulence Model** incorporates the selected turbulence model.
  - *None*
  - *L-Scale Mixing* - Do not include turbulence.
    - Requires a turbulent length scale definition for all initial condition regions.
  - *K-Epsilon* - Do not include turbulence.
    - Requires turbulent kinetic energy and turbulent dissipation definitions for all initial condition regions and all mass and pressure inflow boundary conditions.
- **Max turbulent viscosity** has units of $(Pa \cdot sec)$ and is used to bound turbulent viscosity.

- **Gravity** has units of $(\frac{m}{sec^2})$ and defines gravitational acceleration in the x, y, and z directions.

- **Drag Model** specifies the fluid-particle drag model. This option is only available with the MFiX-TFM and MFiX-DEM solvers.

  - *Syamlal-O'Brien* [SB1988]
    - Requires the specification of the C1 tuning parameter, 0.8 by default.
    - Requires the specification of the D1 tuning parameter, 2.65 by default.
  - *Beestra-van der Hoef-Kuipers* [BVK2007]
  - *Gidaspow* [DG1990]
  - *Gidaspow Blend* [LB2000]
  - *Holloway-Yin-Sundaresan* [HYS2010]
    - Requires the specification of the lubrication cutoff distance, 1e-6 meters by default.
  - *Koch-Hill* [HKL2001]
  - *Wen-Yu* [WY1966]
  - *User-Defined Function (UDF)*
    - A custom drag model must be provided in the usr_drag.f file
    - A custom solver must be built.

Note: The *polydisperse* tag following a specified drag model indicates that the polydisperse correction factor is available. For additional details see [HBK2005], [BVK2007a], and [BVK2007b].

Other advanced options that can be selected include:

- Momentum formulation (Model A, Model B, Jackson, or Ishii)
  - *Model A*
  - *Model B*
  - *Jackson*
  - *Ishii*
- Select sub-grid model
  - *None*
  - *Igci* [IPBS2012]
  - *Milioli* [MMHAS2013]
- Sub-grid filter size
- Sub-grid wall correction

Note: There are some restrictions to when using sub-grid models. They are only available with MFiX-TFM simulations using the Wen-Yu drag law, and without turbulence model. Additional restrictions apply.

4.2 Geometry

The Geometry pane is used to define the model geometry. This includes whether the model is 2D or 3D, and the overall domain extents (xmin, xmax, ymin, ymax, zmin, zmax). If there is complex geometry, the "Auto-size" button can automatically set the extents to surround the geometry.

The geometry section provides tools for adding geometry objects (from STL files, or primitive elements), filters to transform geometry objects, and wizards to automate creating common complex geometries. Geometry objects can be copied, removed, or combined using Boolean operations. All the geometry operations are done using the Visualization Toolkit (VTK)'s library.

Geometry toolbar icons:

Icon	Description
	Add geometry to model
	Modify selected geometry with filter
	Add geometry from wizard
	Remove the selected geometry
	Add duplicate of the selected geometry (copy/paste)
	Add union of two or more selected geometries
	Add intersection of two or more selected geometries
	Add difference of two or more selected geometries

In the geometry tree, the geometry object is displayed with the following icons:

Icon	Geometry Type
	polydata
	implicit
	filter
	union
	intersect
	difference

4.2.1 Adding Geometry

To add a Geometry object, click on the (Geometry) icon and select the geometry to add from the drop-down menu.

There are two types of geometric objects supported in the GUI: implicit functions (quadrics), and objects defined by polydata (everything else: STL files, primitives, parametrics, wizard geometry).

The following geometric objects can be added:

- STL files (select a *.stl file from a file dialog)
- Implicit (Quadric) functions (sphere, box, cylinder, cone, quadric, superquadric)
- Primitives (sphere, box, cylinder, cone)
- Parametrics (torus, boy, conic spiral, etc.)

4.2.2 Applying Filters

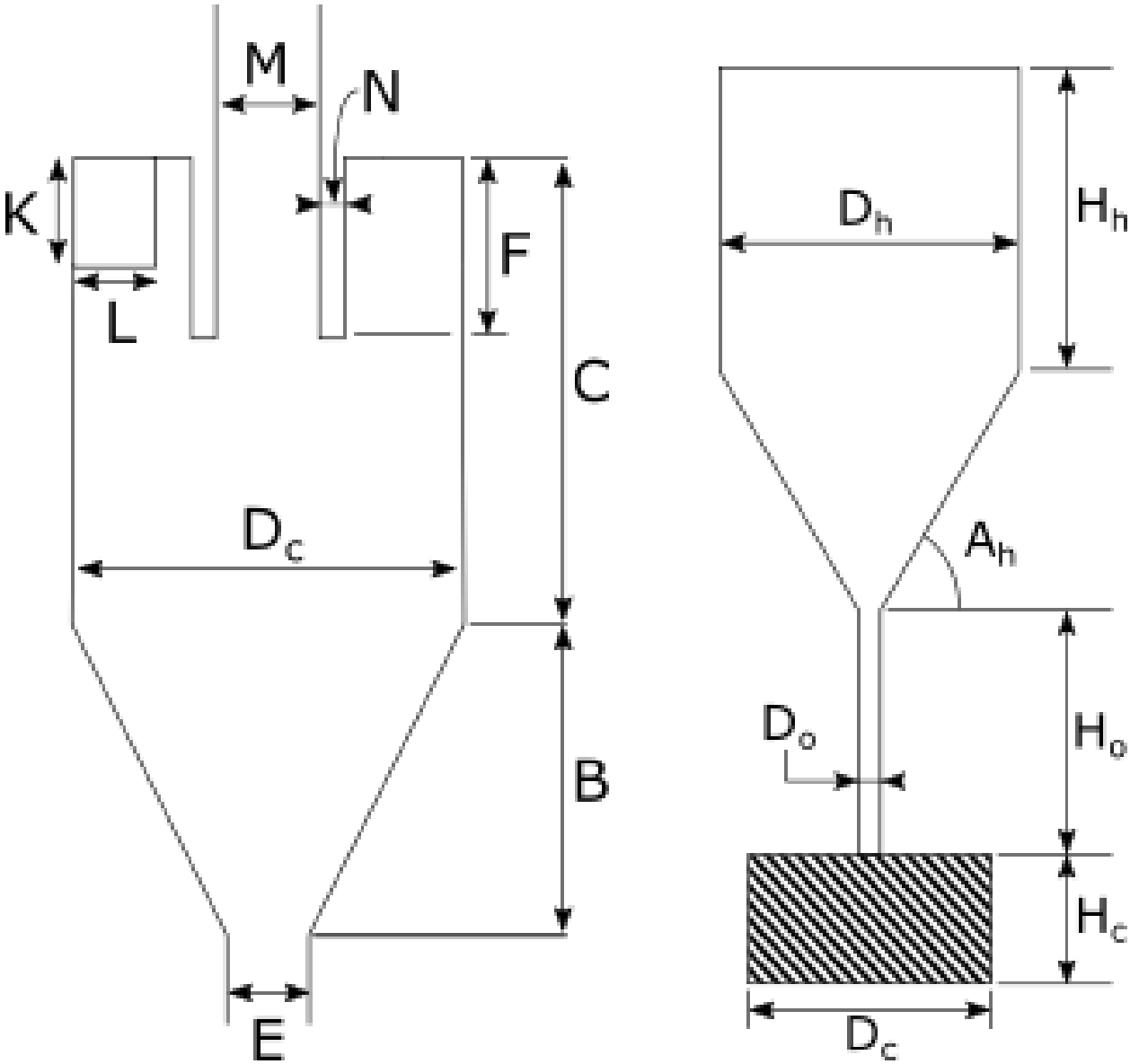
To apply a filter to the selected geometry, select a filter from the Filter menu. The filter options can be edited in the parameter section. The following filters are included:

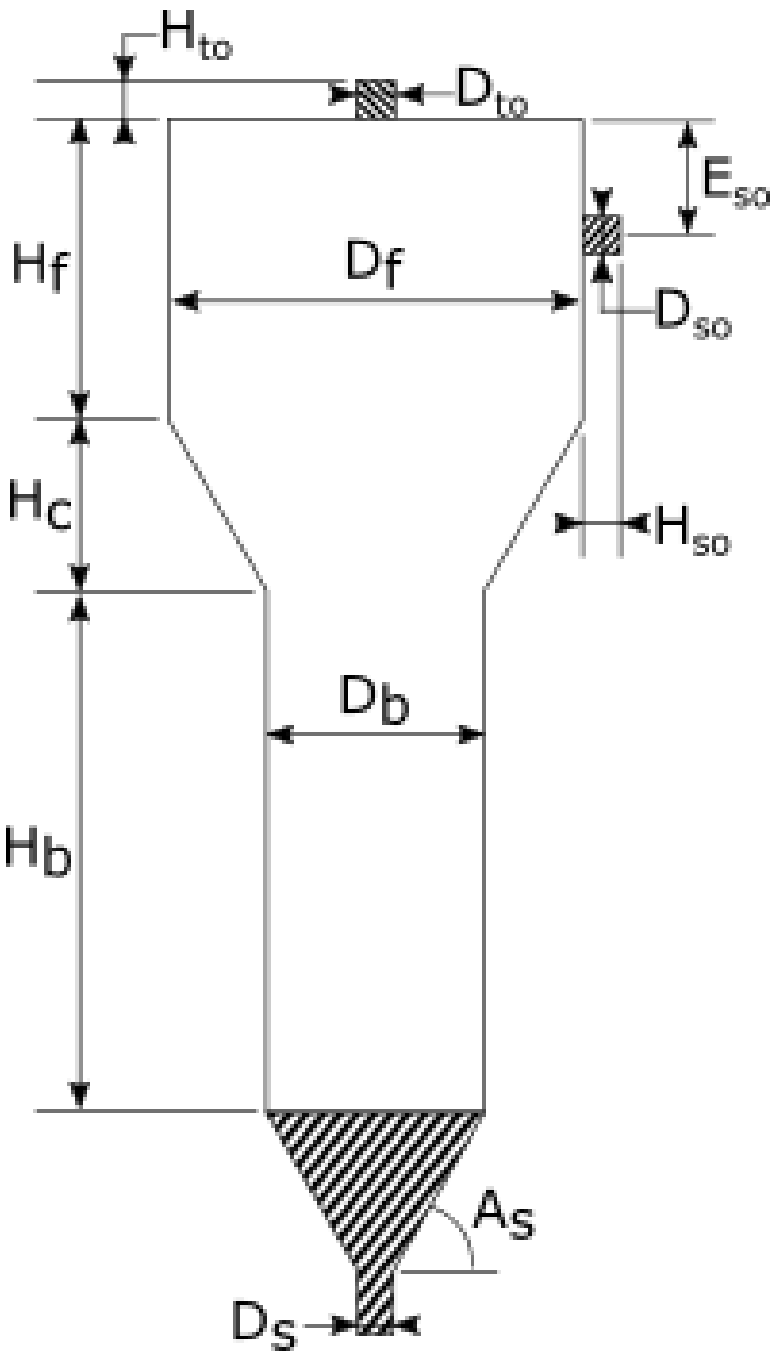
Filter	Description	vtk class
sample implicit	converts an implicit function to polydata	vtkSampleFunction
transform	rotate, scale, translate polydata	vtkTransformPolyDataFilter
clean	merge duplicate points and remove unused points and degenerate cells	vtkCleanPolyData
fill holes	fill holes	vtkFillHolesFilter
triangle	make sure all polys are triangles	vtkTriangleFilter
decimate	reduce the number of triangles	vtkDecimatePro
quadric decimation	reduce the number of triangles	vtkQuadricDecimation
quadric clustering	reduce the number of triangles	vtkQuadricClustering
linear subdivision	subdivide based on a linear scheme	vtkLinearSubdivisionFilter
loop subdivision	subdivide based on the Loop scheme	vtkLoopSubdivisionFilter
butterfly subdivision	subdivide based on 8-point butterfly scheme	vtkButterflySubdivisionFilter
smooth	move points based on Laplacian smoothing	vtkSmoothPolyDataFilter
windowed sinc	move points based on a windowed sinc function interpolation kernel	vtkWindowedSincPolyDataFilter
reverse sense	reverse order and/or normals of triangles	vtkReverseSense

Refer to the VTK website for details.

4.2.3 Wizards

Three wizards are available to more easily create common multiphase flow geometries: cyclones, reactors, and hoppers. A special "distributed" wizard is used to distribute one geometry inside another geometry with random, cubic, or body centered cubic positions. Random rotations can also be applied with the wizard.





4.2.4 Remove & Copy

To remove a selected geometry, click the (Remove) button. If a geometry is grouped as part of a *Boolean Operation*, removing it is disabled until that Boolean Operation is removed first.

To duplicate the selected geometry, click the (Duplicate) button. If multiple geometry objects are selected, duplication is disabled.

4.2.5 Boolean Operation

There are two types of geometric objects supported in the GUI: implicit functions (quadrics) and objects defined by polydata (everything else: STL files, primitives, parametrics, wizard geometry). Boolean operations cannot be performed between polydata and implicit geometry objects; the implicit function needs to be converted to polydata by using the `sample implicit` filter. Converting the implicit function also needs to be done in order for the GUI to export a STL file that the mfixsolver can use.

Boolean operations can only be performed with geometry objects of the same type (implicit, polydata). Boolean operations can not be performed between polydata and implicit geometry objects. If an implicit and a polydata must be managed together, the implicit object must be first converted to a polydata object using the sample implicit filter.

Note: Boolean operation between two polydata objects is supported by MFiX, but for complex objects the VTK library may crash.

4.3 Mesh

The Mesh pane has two tabs:

- *Background Mesh* for the specification of the background mesh
- *Mesher* for changing cut-cell tolerances

4.3.1 Background Mesh

The Background tab is for specifying the mesh used by the Eulerian solver. A uniform mesh can be specified by entering the number of cells in the x, y, and z directions. The resulting Cell Size is computed from the geometry and the number of cells (Cell Size is not editable directly). The resulting mesh will be visible in the model setup view. The visibility of the mesh can be toggled with the (Visibility) button at the top of the model setup view.

Control points can be added by pressing the button. Once a control point has been added, the position, number of cells, stretch, first and last parameters can be changed. To split a control point, `right-click` on the control point and select `split` from the resulting menu. This operation will create a new control point at the midpoint between the previous control point and the selected control point, dividing the cells evenly between the two. Control points can be removed by pressing the button.

The stretch parameter is a value that will apply a non-uniform grid spacing to the cells. The value is defined as $\frac{LastWidth}{FirstWidth}$. A value larger than 1 stretches the grid as x increases, while a value smaller than one compresses the grid as x increases. A value of exactly 1 will keep the spacing uniform.

The first and last values allow the specification of the first and last widths. The other cell widths adjust accordingly. If a negative value is specified in the column "First", the last width from the previous grid segment is copied. If a negative value is specified in the column "Last", the first width from the next segment is copied.

4.3.2 Mesher

Meshing of complex geometry is performed dynamically by the solver at runtime. The Mesher tab exposes options to adjust the cut-cell mesher. These options include:

Option	Description
External flow	select internal or external flow. Note: this depends on which way the normals are pointing on the STL file. If they are pointing out of the geometry, then the text will be correct.
Small cell tolerance	tolerance to detect, and remove small cells
Small area tolerance	tolerance to detect, and remove cells with small faces
Merge tolerance	tolerance used to merge duplicate nodes
Snap tolerance	tolerance to move an intersection point to an existing cell corner
Allocation Factor	factor used in allocation of cut-cell arrays
Maximum iterations	maximum number of iterations used to find intersection points
Intersection tolerance	tolerance used to find intersection of background mesh and STL triangles
Facet angle tolerance	ignore STL facets that have an angle less than this tolerance
Dot product tolerance	tolerance used to determine if a point lies in a facet
Max facets per cell	maximum number of facets allowed in a cell

4.4 Regions

The Regions pane defines spatial regions (points, lines, planes, boxes, or STLs) of the geometry that are used for:

- Initial Conditions
- Boundary Conditions
- Point Sources
- Internal Surfaces
- Monitors
- Outputs

The following buttons are at the top of the Regions pane:

Icon	Description
	create a new region
	delete the selected region
	duplicate the selected region
	create a region that encompasses the entire domain
	create a region on the left side of the domain
	create a region on the right side of the domain
	create a region on the top side of the domain
	create a region on the bottom side of the domain
	create a region on the front side of the domain
	create a region on the back side of the domain

The (Add) button creates a new region. The region will be created with a generic name such as `R_1`; it is strongly recommended to rename the region to something more descriptive, so that it is easier to refer to it later. The Color button will change the color of the region in the model setup view.

The table created with the "From" and "To" fields for the X, Y, and Z axes define the extents of the region. These widgets take special parameters, `min` and `max`, that reference the minimum and maximum values of the domain, as specified in the geometry section. These values will get automatically updated when the extents in the geometry section are updated. The region type will be inferred from the specified extents.

If the region needs to be a collection of triangles from the STL file, select the Select Facets (STL) check-box. The selection shape can be changed between a box and an ellipsoid. Triangles that fall on the edge of the shape can be sliced by selecting the Slice Facets check-box. The triangles can be further filtered by the normal direction by specifying a vector and a deviation angle around that vector.

Hint: If a region is referenced by an item in the `Used By` column, the region can not be deleted and its type (STL vs non-STL) cannot be changed.

4.5 Fluid

The fluid pane is used to select the models and parameters defining the fluid phase. The fluid pane is only accessible if the fluid phase is being solved.

Name The name used to refer to the continuous phase in the GUI. By default, the continuous phase is termed "fluid", However, it may be renamed for convenience.

4.5.1 Fluid Model Options

Solve Momentum Equation By default, all momentum equations are solved. Individual momentum equations may be disabled by toggling the check box.

Solve Species Equations By default, species transport equations are not solved for the fluid phase. If species equations are enabled, species will need to be added to the fluid phase using the fluid species tool.

Enable Scalar Equations By default, no additional scalar transport equations are solved with the fluid phase. Additional scalars may be added by toggling the scalar checkbox and specifying the number of additional scalars to track.

Energy Equations Energy equations cannot be enabled or disabled on a per-phase basis. As such, they are enabled or disabled for all phases in the Model Setup pane.

4.5.2 Fluid Properties

Density has units of $(\frac{kg}{m^3})$ and may be specified using one of the following approaches.

- *Constant:*
  - A positive (non-zero) number must be provided.
- *Ideal gas law*
  - Requires fluid temperature be specified for the whole domain and all flow boundary conditions.
  - Requires a fluid molecular weight.
- *User-Defined Function (UDF)*
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Viscosity has units of $(Pa \cdot sec)$ and may be specified using one of the following approaches.

- *Constant:*
  - A positive (non-zero) number must be provided.
- *Sutherland's law*
  - Requires fluid temperature be specified for the whole domain and all flow boundary conditions.
- *User-Defined Function (UDF)*
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Molecular Weight has units of $(\frac{kg}{kmol})$ and may be specified using one of the following approaches.

- *Constant:*
  - A positive (non-zero) number must be provided.

- *Mixture*
  - Requires fluid species definition.
  - Requires fluid species mass fractions specification for the whole domain and all flow boundary conditions

Specific Heat has units of $(\frac{J}{kg \cdot K})$ and may be specified using one of the following approaches.

- *Constant:*
  - A positive (non-zero) number must be provided.
- *Mixture*
  - Requires fluid species definition.
  - Requires fluid species mass fractions specification for the whole domain and all flow boundary conditions
- *User-Defined Function (UDF)*
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Thermal Conductivity has units of $(\frac{W}{m \cdot K})$ may be specified using one of the following approaches.

- *Constant:*
  - A non-negative number must be provided.
- *Dilute Mixture Approximation*
  - Requires fluid temperature be specified for the whole domain and all flow boundary conditions.
- *User-Defined Function (UDF)*
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Reference pressure has units of (Pa) and is zero by default. A constant value may be specified to shift the simulation pressure prior to scaling.

Pressure scale factor is dimensionless and is one by default. A constant value may be specified to scale the simulation pressure.

$$P_{scaled} = \frac{P_{simulation} - P_{reference}}{P_{scalefactor}}$$

4.5.3 Fluid Model Species

Specie that comprise the fluid phase are summarized in the species overview table. New species are added by clicking the add button, , at the top of the species table, and selecting one or more valid regions (see Fig. 4.6).

4.6 Solids

The solids pane is used to set solids phases material properties. The "Materials" tab sets material properties shared by most models (TFM, DEM and PIC). Specific settings for each model are accessed through their respective tabs (TFM, DEM, or PIC).

4.6.1 Creating a Solids phase

A new solids phase is created by clicking the add button, , at the top of the solids table. This will create a new entry in the table. The table summarizes all defined solids phases by listing the name, model, particle diameter and density. A blank entry in the density means the particle density is variable (it depends on its chemical composition).

Name The name used to refer to the solids phase in the GUI. By default, the solids phase is called "Solid" followed by its assigned phase number. The phase number is incremented every time the button is pressed. The solids phase name may be renamed for convenience (optional). This name will appear in other panes or tabs when referring to the solids phase (say while setting initial or boundary conditions).

Model (read only)

The modeling approach used to represent the solids phase. Available options include TFM (Two-Fluid Model), DEM (Discrete Element Model) and PIC (Particle in Cell). The setting is currently locked (the same solid model must be used for all solids phases), and mirrors the solver selection in the Model Setup pane.

4.6.2 Deleting a Solids phase

A solids phase can be deleted by selecting it (click on its name in the solids table) and clicking the remove button, , at the top of the solids table. If the solids phase is used in the model with a non-zero solids fraction (say in an initial or boundary condition), the deletion will need to be confirmed by the user. If confirmed, the deleted solids volume fraction will be assigned back to the fluid phase.

4.6.3 General Solid Model Options

Solve momentum equation (TFM only) By default, all momentum equations are solved. Individual momentum equations may be disabled by toggling the check box.

Solve species equations By default, species transport equations are not solved for the solids phase. If species equations are enabled, species will need to be added to the solids phase using the solids species tool. ADD REF

4.6.4 Solids phase Material Properties

Some material properties are only needed for a specific solid model or when the energy or species equation is solved.

Diameter (m) The initial particle diameter. It will remain constant for non-reactive flows, and for reactive flows with a variable density. If the density is constant, and chemical reactions take place, the particle diameter will vary to reflect particle mass gain/loss.

Density $(kg\ /\ m^3)$ The particle density can be set as:

- *Constant:*
  - A positive (non-zero) number must be provided.
  - The diameter may change due to chemical reactions.
- *Variable:*
  - The density is computed from particle's mass and volume.
  - The particle mass is function of the particle's chemical composition.
  - A variable density can only be used if species equations are solved.

Viscosity $(Pa \cdot s)$ **TFM only** It may be specified using one of the following approaches:

- *Constant:*
  - A positive (non-zero) number must be provided.
- *Continuum Solids stress Theory*
- *User-Defined Function (UDF)*
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Molecular weight $(\frac{kg}{kmol})$ It may only be specified using the following approach, and is only used when Energy and solids species equations are solved:

- *Mixture*
  - Requires species be used to defined the solid.
  - Requires species mass fractions be specified for the whole domain and all flow boundary conditions

Specific heat $(\frac{J}{kg \cdot K})$ It may be specified using one of the following approaches (only used when Energy and species equations are solved):

- *Constant:*
  - A positive (non-zero) number must be provided.
- *Mixture*
  - Requires species be used to defined the fluid.
  - Requires species mass fractions be specified for the whole domain and all flow boundary conditions
- *User-Defined Function (UDF)*
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Thermal conductivity $(\frac{W}{m \cdot K})$ It may be specified using one of the following approaches:

- *Constant:*

  - A non-negative number must be provided.
- *Temperature-Dependent*
  - Option available for TFM only.
  - Requires solids phase temperature be specified for the whole domain and all flow boundary conditions.
- *User-Defined Function (UDF)*
  - Option available for TFM only.
  - A custom equation of state must be provided in the usrproperties.f
  - A custom solver must be built.

Emissivity $(-)$ **DEM only** The emissivity of the DEM solids phase. Leaving it blank or setting a zero value turns radiation off.

Parcel weight $(-)$ **PIC only** A default parcel weight (number of particles per parcel) may be specified for convenience (default value of one). it is used to define a default statistical weight for parcels in initial conditions regions and mass inflows. Instead of defining the same parcel weight in every initial condition and mass inflow, this default value is automatically applied. The default value can be overwritten where needed.

4.6.5 Solids Model Species

Specie that comprise the solid phase are summarized in the species overview table. New species are added by clicking the add button, , at the top of the species table.

4.6.6 Advanced settings

Disable close pack (TFM only) Options to enable/disable a TFM phase from forming a packed bed. This is typically used to make the solids phase behave as a liquid phase.

Enable added mass force (TFM only) Options to enable/disable the added (or virtual) mass force in a TFM phase. This tends to stabilize bubbly gas/liquid flows.

4.6.7 TFM Settings

Specific Two-Fluid Model settings are accessed from the TFM tab.

Packed bed void fraction $(-)$ The void fraction at close pack.

Viscous stress model The solids phase stress model. Options include the algebraic formulation or various kinetic theories requiring solving a Partial Differential Equation for the granular energy:

- *Algebraic Formulation [DEFAULT]*
- *Lun et al, 1984*
- *Iddir & Arastoopour, 2005*
- *Simonin, 1996* (requires k-ε turbulence enabled)
- *Cao & Ahmadi, 1995* (requires k-ε turbulence enabled)
- *Garzo and Dufty, 1999* (monodisperse system only)
- *Garzo, Tenneti, Subramaniam, Hrenya, 2012* (monodisperse system only)

- *Garzo, Hrenya and Dufty, 2007* - Requires at most two solids phases - Requires Wen-Yu or HYS drag model - Selection not available with added mass force

Frictional stress model

- *Schaeffer model* [DEFAULT]
- *Srivastava and Sundaresan*
- *Only solids pressure*

Solids volume fraction at onset of friction $(-)$ The minimum solids fraction above which the Srivastava and Sundaresan model sets in.

Particle-particle restitution coefficient $(-)$ **:** The coefficient of restitution for particle-particle collision.

Interphase friction coefficient $(-)$ **:** The coefficient of friction between particles of two solids phases.

Angle of internal friction (deg) **:** The angle of internal friction. The plastic regime stress can be turned off by setting this value to zero.

Radial distribution function

- *Carnahan-Startling* (only option for monodisperse systems)
- *Lebowitz* (default for polydisperse system)
- *Mansoori* (polydisperse system)
- *Modified Lebowitz* (polydisperse system)
- *Modified Mansoori* (polydisperse system)

Stress blending The blending function used to smooth transition around the packed bed volume fraction. It requires the Schaeffer frictional stress model.

- *None* [DEFAULT]
- *Hyperbolic Tangent*
- *Sigmodial*

Segregation slope coefficient $(-)$ Coefficient used in calculating the initial slope in segregation for polydisperse systems.

Max packing correlation The correlation used to compute the maximum packing for polydisperse systems:

- *Constant* [DEFAULT]
- *Yu & Standish*
- *Fedors & Landel* (only with two solids phases)

Excluded volume in Boyle-Massoudi stress $(?)$ The excluded volume in Boyle-Massoudi stress. It is only used with the algebraic formulation of viscous stress model (optional).

4.6.8 DEM Settings

Specific Discrete Element Model settings are accessed from the DEM tab.

Enable automatic particle generation Initialize particle location and velocity based on information from Initial Condition regions. If this option is disabled and any initial condition uses a non-zero solids volume fraction, the user will be prompted to turn on this option. Disabling this option will read initial particle location and velocity from a user generated text file (particle_input.dat) that must be saved in the project directory.

Data file particle count $(-)$ The number of particles read from particle_input.dat file when the automatic particle generation is turned off. This number must be smaller or equal to the number of lines in the file.

Particle Size Distribution (PSD) A series of PSDs can be defined to be used in an initial condition region or a mass inlet boundary condition. A PSD represents the number of particles having a given size. It is currently a number-based (not volume-based) distribution. A new PSD is created by clicking the add button, , and entering the PSD parameters. The PSD can follow a Normal or Log-normal distribution. There is an option to load a text file containing a custom PSD.

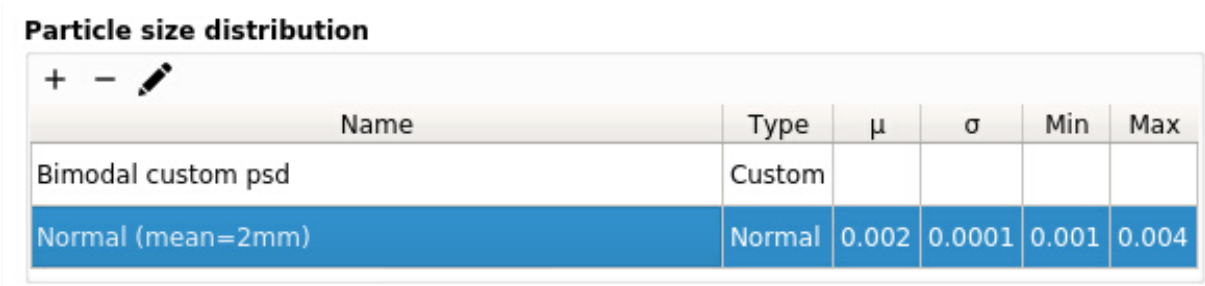


Fig. 4.1: Particle Size Distribution table.

Initially, the table will be empty. Double click on one row to edit a PSD, Click on to remove a PSD from the table.

First, enter a descriptive name for the PSD and choose the PSD type.

Parameters for the Normal and Log-normal distribution are:

- *Mean particle diameter*
- *Standard deviation*
- *Minimum diameter* , used to clip the distribution and avoid extremely small particles
- *Maximum diameter* , used to clip the distribution and avoid extremely large particles

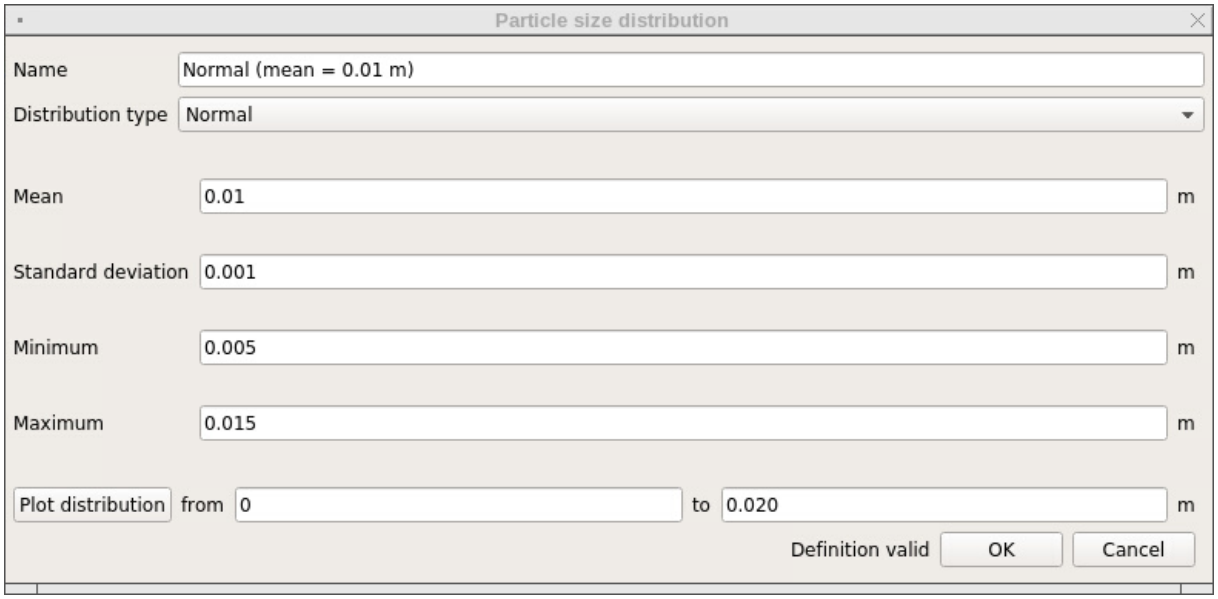


Fig. 4.2: Example of a Normal Particle Size Distribution definition.

Custom PSDs will read the distribution from a text file. It can be edited through any text editor including the GUI built-in editor. The PSD custom file is organized as follows:

1st line: Number of points, number of distribution (this must be set to one currently).
2nd line: PDF (probability density function) of CDF (Cumulative distribution function).
3rd line: Header.
4th line and below: The data itself. First column is the diameter, second column is either the PDF or CDF value.

The figure below shows an example of a custom PSD file. Only the first few lines are shown.

hopper_dem_custom_psd.mfx [read only] | bimodal_psd.txt

```
51, 1
CDF
!Diameter CDF
2.9156e-04 0.0000e+00
3.5244e-04 3.0000e-04
4.1332e-04 6.0000e-04
4.7420e-04 1.8000e-03
5.3508e-04 5.6000e-03
5.9596e-04 1.1700e-02
6.5684e-04 2.2200e-02
7.1773e-04 4.1000e-02
7.7861e-04 6.8800e-02
8.3949e-04 1.0680e-01
9.0037e-04 1.6010e-01
9.6125e-04 2.1930e-01
```

Fig. 4.3: Example of a Custom file used to define a Particle Size Distribution.

Once the parameters are defined, the PSD can be plotted by clicling the 'Plot distribution' button.

Integration method The DEM time stepping scheme when integrating the particle trajectories (acceleration to velocity and velocity to position):

- *Euler* [DEFAULT]
- *Adams-Bashforth*

Collision model The soft-sphere collision model:

- *Linear Spring Dashpot* (LSD)
- *Hertzian*

Coupling method The level of coupling between the gas phase and the solids phase:

- *One-way coupled*
- *Fully coupled*

Interpolation The direction of interpolation between field and particle data:

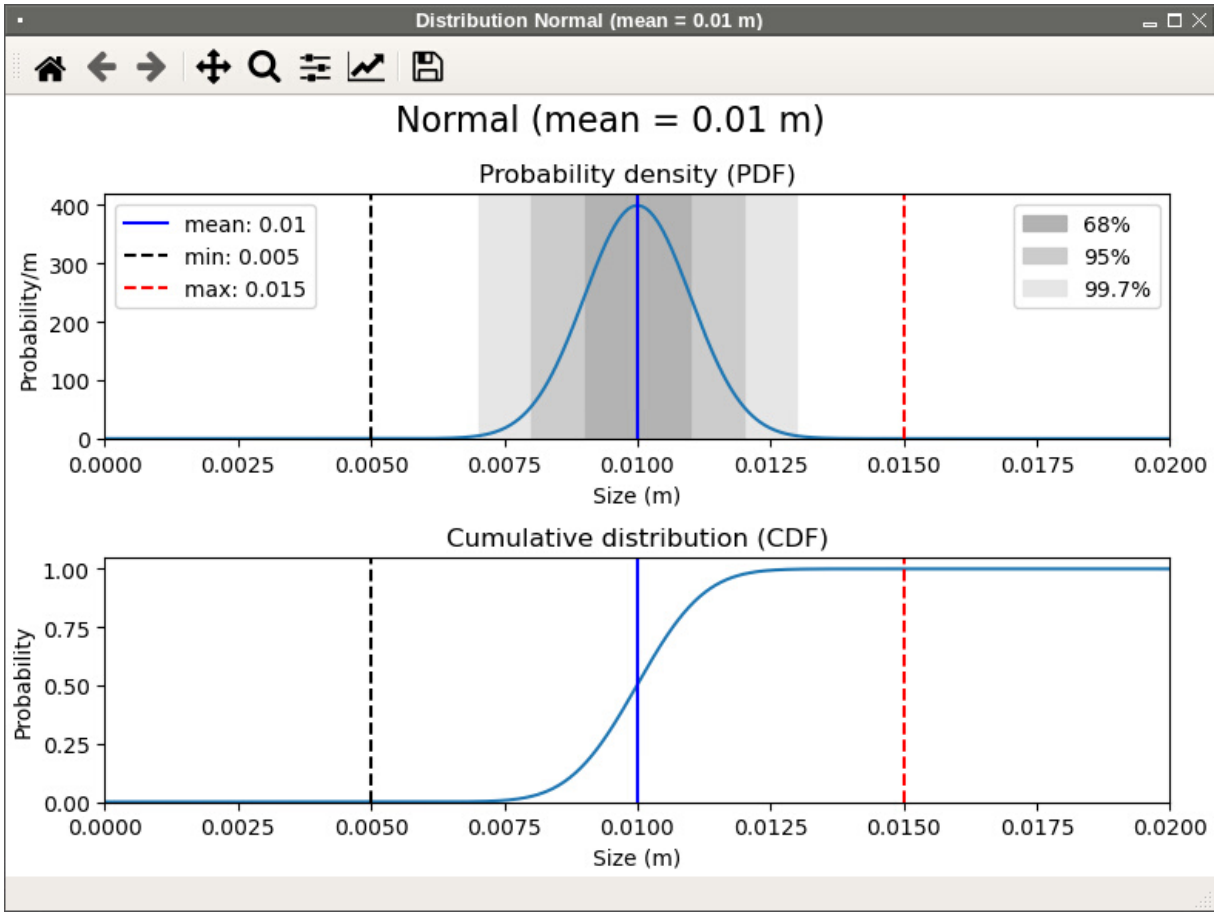


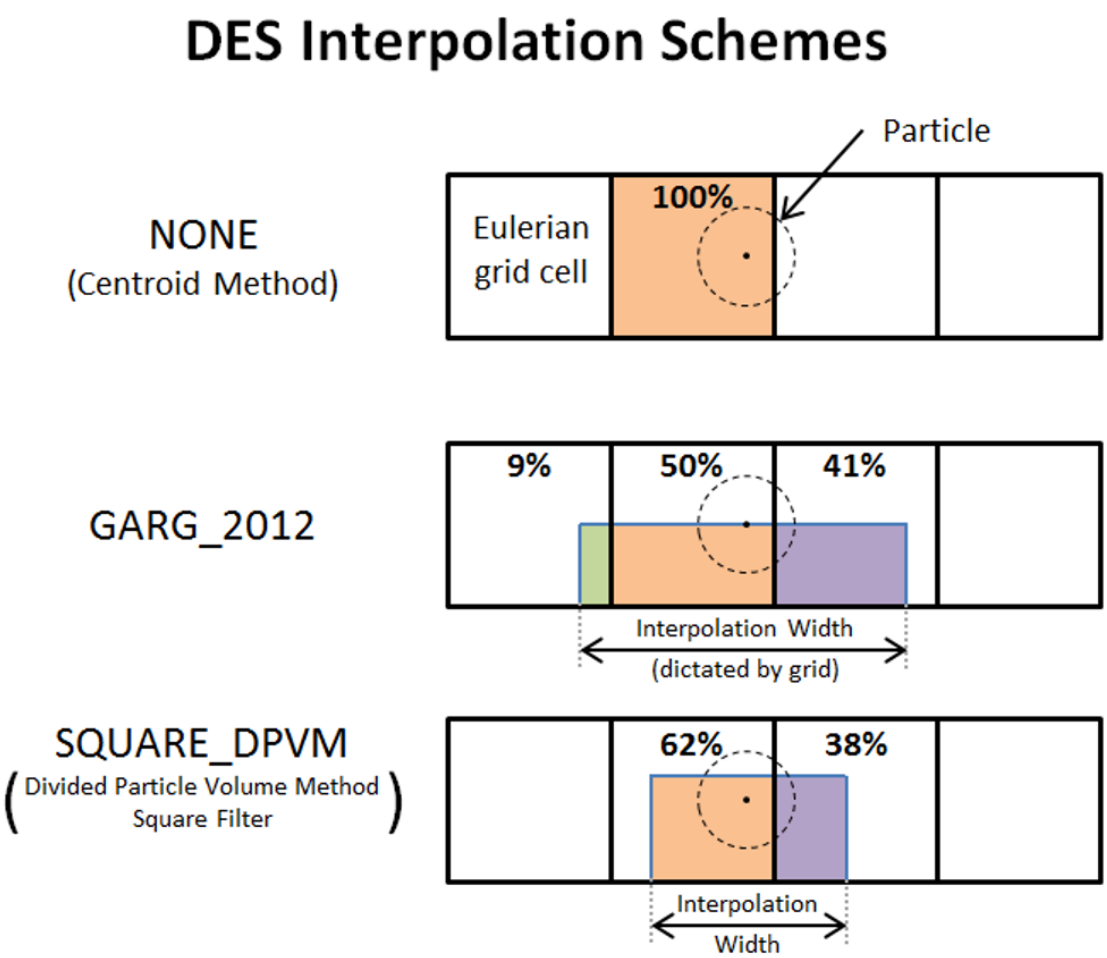
Fig. 4.4: Plotting a Particle Size Distribution.

- *No interpolation*
- *field-to-particle and particle-to-field*
- *field-to-particle only*
- *particle-to-field only*

Scheme The interpolation scheme:

- *None* (centroid method)
- *Garg, 2012* (interpolation width is dictated by grid spacing)
- *Square DPVM* (Divided Particle Volume Method), requires an interpolation width.

Width (m) The square DPVM interpolation width.



Enable mean field diffusion Smooths the disperse phase average fields by solving a diffusion equation.

Width (m) The diffusion length scale.

Enable explicit coupling of interphase quantities Option to explicitly couple the fluid and solids hydrodynamics.

Friction coefficient $(-)$ The particle-particle and particle-wall Coulomb friction coefficient. This is required for both the LSD and Hertzian collision models.

Normal spring constant $(\frac{N}{m})$ The particle-particle and particle-wall normal spring constant (LSD collision model).

Spring norm/tan ratio $(-)$ The ratio of normal to tangential spring constants for particle-particle and particle-wall (LSD collision model).

Damping norm/tan ratio $()$ The ratio of normal to tangential damping factors for particle-particle and particle-wall (LSD collision model).

Young's modulus (Pa) The wall and particles (one entry per phase) Young's modulus (Hertzian collision model).

Poisson's ratio $(-)$ The wall and particles (one entry per phase) Poisson's ratio (Hertzian collision model).

Restitution coefficients (normal) $(-)$ The list of normal restitution coefficients for particle-wall and particle-particle interaction. The particle-particle entries are arranged into a symmetrical matrix. The diagonal terms and the upper side of the matrix need to be filled. There is one entry per phase for the particle-wall coefficient. This settings is required for both the LSD and Hertzian collision models.

Restitution coefficients (tangential) $(-)$ The list of tangential restitution coefficients for particle-wall and particle-particle interaction. The particle-particle entries are arranged into a symmetrical matrix. The diagonal terms and the upper side of the matrix need to be filled. There is one entry per phase for the particle-wall coefficient. This settings is only required for the Hertzian collision model.

Cohesion model Toggles the van der Waals cohesion model. When turned on, additional parameters need to be set (see below):

Hamaker constant (J) The particle-particle and particle-wall Hamaker constant used in the van der Waals cohesion model.

Outer cutoff (m) The particle-particle and particle-wall maximum separation distances above which the van der Waals cohesion forces are set to zero.

Inner cutoff $(-)$ The particle-particle and particle-wall minimum separation distances below which the van der Waals cohesion forces are computed using a surface adhesion model.

Asperities $(-)$ The mean radius of surface asperities used in the cohesive force model.

Minimum fluid volume fraction $(-)$ Threshold used to clip the fluid phase volume fraction (to avoid non-physical values, typically when a particle size is near a small cut cell).

Neighbor search method The neighbor search algorithm:

- *Grid-based*
- *N-Square*

Max steps between neighbor search $(-)$ The maximum number of DEM iterations between two neighbor searches. The neighbor search may be called earlier if the particle moves a distance greater than a defined quantity (see below).

Factor defining particle neighborhood $(-)$ A multiplying factor applied to the particle's radius to define the region where particle neighbors are searched from.

Distance/diameter triggering search $(-)$ The distance a particle can travel before triggering an automatic neighbor search.

Search grid partition $(-)$ The number of DEM grid cells in each direction used for the neighbor search. This is an optional setting. If left undefined, the number of cells is computed so that the DEM grid size is three times the maximum particle diameter.

Enable user scalar tracking Turns on the tracking of user-defined scalars attached to each particle. When the tracking is enables, the number of scalars (positive integer) needs to be set.

Minimum conduction distance (m) The minimum separation distance between particles (used in the particle-fluid-particle conduction model to remove singularity).

Fluid lens proportionality constant $(-)$ Constant used to calculate the fluid lens radius that surrounds the particle (used in the particle-fluid-particle conduction model).

Young's modulus used to correct DEM conduction (Pa) The wall and particles (one entry per phase) Young's modulus used to correct the DEM conduction. These optional input are used with both LSD and Hertzian collision models. Particles are typically made softer to increase the DEM time step. This may lead to inaccurate conduction. If defined, this Young's modulus will be used to correct the DEM conduction model.

Poisson's ratio used to correct DEM conduction $(-)$ The wall and particles (one entry per phase) Poisson's ratio used to correct the DEM conduction. These optional input are used with both LSD and Hertzian collision models.

4.6.9 PIC Settings

Specific Particle in Cell settings are accessed from the PIC tab.

Many of the PIC settings refer to the PIC solids stress model, which influences parcel motion:

$\tau_p = \frac{P_p \epsilon_p^\gamma}{\max[\epsilon_{cp} - \epsilon_p, \delta(1-\epsilon_p)]}$

Void fraction at close pack $(-)$ The void fraction (ϵ_{cp}) at maximum packing.

Volume fraction exponential scale factor $(-)$ The empirical exponent (γ) on solids fraction in the solids stress model. Typical values are between 2.0 to 5.0.

Pressure linear scale factor (Pa) The empirical pressure (P_p) constant in the solids stress model. Typical values are 10 to 1000 Pa.

Empirical dampening factor $(-)$ A factor in a restitution coefficient for comparing solids stress with slip velocity. Typical value is 0.85.

Non-singularity constant $(-)$ The constant (δ) to prevent division by zero in solids stress model. Typical value is 1.0E-8.

Wall normal restitution coefficient $(-)$ The parcel-wall restitution coefficient for normal velocity component after wall collision.

Wall tangential restitution coefficient $(-)$ The parcel-wall restitution coefficient for tangential velocity component after wall collision.

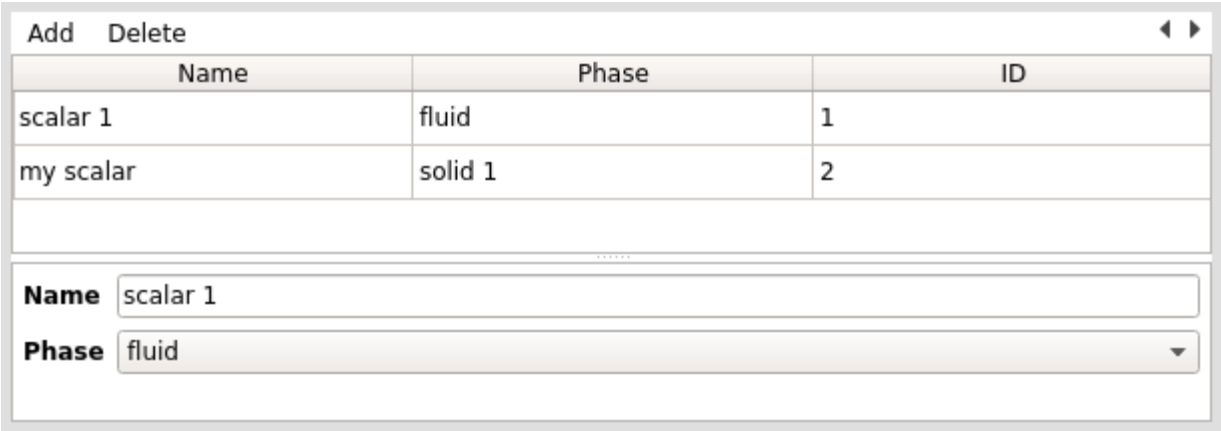
Solids slip velocity scale factor $(-)$ A damping coefficient on slip velocity. Typical value is 1.0.

4.7 Scalars

The MFiX GUI supports adding user-defined scalars.

This section defines how scalars are implemented into MFIX GUI. An example layout of the Scalar Pane is provided in `scalars_pane`. The layout is similar to the *Solids Model* pane: a table summarizes previously defined scalars and add/delete buttons allow scalars to be added and removed. Select a row of the table to edit the corresponding Scalar's Name and Phase.

A table lists the user-defined scalar name, the convecting phase, and the scalar index.



4.7.1 Add/Delete Scalars

To add a Scalar, navigate to the Scalars pane from the Modeler View, and click the (+) button.

To remove a Scalar, navigate to the Scalars pane from the Modeler View, select the scalar to remove, and click the (-) button.

4.7.2 Scalar Name: GUI-only description

By default, scalars are named "Scalar ID" where ID is the index of the scalar. It is recommended to edit the Name to something more descriptive. However, the scalar name is local to the GUI and not provided to the solver; in the solver and UDFs the Scalar is only referred to by Index.

4.7.3 Scalar Phase: Fluid or TFM Solid

There is a drop-down box to select the convecting phase from the phases defined in *Fluids* or *Solids* panes.

Selecting a phase with index `PHASE` sets keyword `PHASE4SCALAR(ID) = PHASE` where `ID` is the scalar index for the current scalar. `PHASE` is `0` for the fluid phase, or a positive integer for TFM solids.

The fluid phase is only an option when the fluid solver is enabled (Disable Fluid Solver is unchecked on *Model Pane*). Any defined TFM solids phases are listed as options for `PHASE`, but DEM and PIC solids phases are not listed as options.

4.7.4 Scalar ID (Index)

Each Scalar has an ID, indexed starting at zero.

Deleting Scalars and Contiguous indices

The MFiX solver assumes scalar array inputs are contiguous. As a result, if one or more scalar equations is removed or the order of the scalars is changed, then all scalar input arrays must be updated. For example, consider three scalars that are convected with a different phase.

```
NScalar = 3
Phase4Scalar(1) = 0
Phase4Scalar(2) = 1
Phase4Scalar(3) = 2
```

If a user deletes the second scalar, then array values assigned to the third scalar must be shifted down.

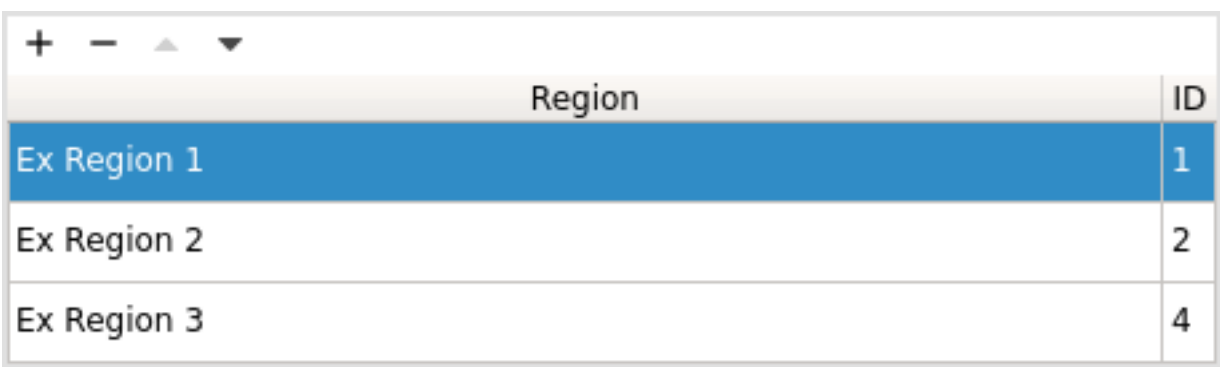
```
NScalar = 2
Phase4Scalar(1) = 0
Phase4Scalar(2) = 2
```

Array inputs for scalars include the following:

```
Phase4Scalar(1:NScalar)

IC_Scalar(:, 1:NScalar)

BC_HW_Scalar(:, 1:NScalar)
BC_ScalarW(:, 1:NScalar)
BC_C_Scalar(:, 1:NScalar)
BC_Scalar(:, 1:NScalar)

VTK_Scalar(:,1:NScalar)
MONITOR_Scalar(:,1:NScalar)
```

4.8 Initial Conditions

Initial Conditions (ICs) are specified in the Initial conditions task pane. ICs summarized in the *IC summary table* illustrated in Fig. 4.5. The MFiX solver will apply ICs in the order that they are defined in this pane. As a result, if two or more regions overlap, the IC with the larger index (ID) is used in the intersecting region. IC IDs are shown in the table, and can be changed using the up and down arrows.



Region	ID
Ex Region 1	1
Ex Region 2	2
Ex Region 3	4

Fig. 4.5: Initial condition (IC) summary table.

4.8.1 Creating an Initial Condition Region

To define an initial condition, there must be a region already defined in the *Regions Pane*.

Initial conditions are specified over rectangular regions corresponding to the scalar grid. A new IC region is created by clicking the add button, , at the top of the IC region table, and selecting one or more valid regions (see Fig. 4.6).

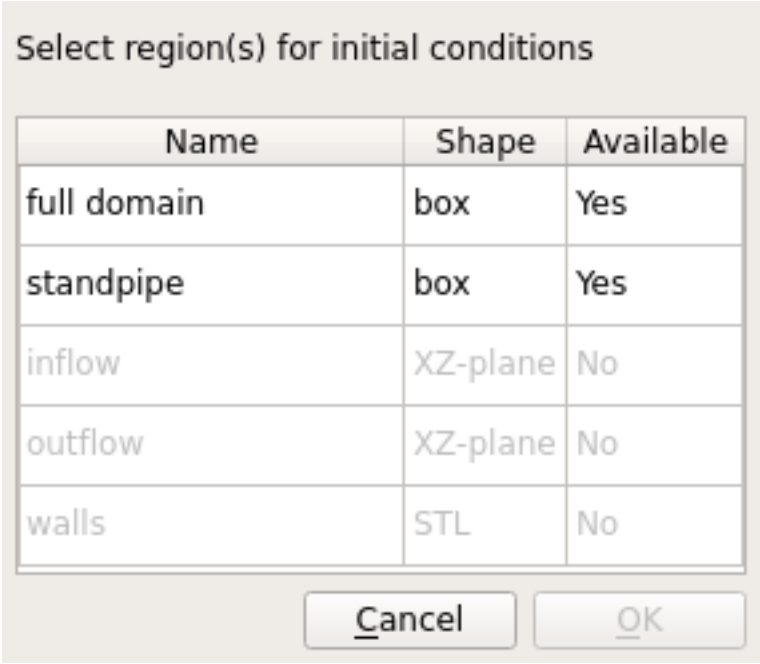


Fig. 4.6: Initial condition region selection popup window.

Valid IC Regions

- Box regions
- XY-planes (2D simulations only)

Invalid IC Regions

- Any region already used by another IC
- XZ- and YZ-planes
- Point regions
- STL regions

4.8.2 Setting Fluid Phase Initial Values

When the fluid phase is enabled, initial values for fluid field variables must be specified for the entire computational domain. The models being solved dictate which variables require an initial value.

Volume Fraction The fluid phase volume fraction has a default value of one. If solids are defined and given volume fractions greater than zero, the fluid phase volume fraction is automatically calculated by subtracting the sum of the solids phase volume fractions from one.

Temperature The fluid phase temperature has a default value of 293.15K. Specification of the fluid temperature is required when the following settings are used:

- The energy equation is solved
- Fluid density is computed by the Ideal Gas Law
- Fluid viscosity is computed by the Sutherland's Law

Pressure An initial pressure field may be specified. If none is provided, the fluid solver will attempt to impose a hydrostatic pressure drop across the domain.

Velocity The fluid phase velocity field has a default value of zero.

Species Mass Fractions Species mass fractions must be defined for all species when solving the species transport equations. By default the species mass fraction of the last defined fluid species is set to one and all others are set to zero. The total of all species mass fractions must equal one.

Turbulence If using the mixing length turbulence model, a mixing length scale must be defined for the whole domain.

If solving the k-ε turbulence model, initial values for turbulent kinetic energy and turbulent dissipation must be provided.

Radiation Coefficient A radiation coefficient has a default value of zero, and can only be specified when solving the energy equations. The default value is strongly recommended.

Radiation Temperature The radiation temperature has a default value of 293.15K, and can only be specified when solving the energy equations. The default value is strongly recommended.

4.8.3 Setting Solids Phase Initial Values

When one or more solids phases are enabled, initial values for solids field variables must be specified for the entire computational domain. The models being solved dictate which variables require an initial value.

Volume Fraction The solids phase volume fraction has a default value of zero. The sum of all solids phase volume fractions are used to calculate the volume (void) fraction of the fluid phase. If solids are defined and given volume fractions greater than zero, the fluid phase volume fraction is automatically calculated by subtracting the sum of the solids volume fractions from one.

Inventory *(DEM solids only)* Amount of solids (by mass) in the initial condition region.

Particle count *(DEM solids only)* Number of particles in the initial condition region.

Particle size distribution *(DEM solids only)* Particle size distribution in the initial condition region. This can be left as "Uniform" (default) and the particle diameter (d_p0) defined in the Solids>Material pane will be used. If a PSD was defined in the Solids>DEM pane, it can be chosen here to initialize particles in this region with a PSD.

Temperature The solids phase temperature has a default value of 293.15K. Specification of solids temperature is required when solving energy equations.

Pressure *(TFM solids only)* Solids pressure is an optional input for TFM solids models. There is only one value for all solids phases.

Granular Temperature The initial granular temperature has a default value of zero.

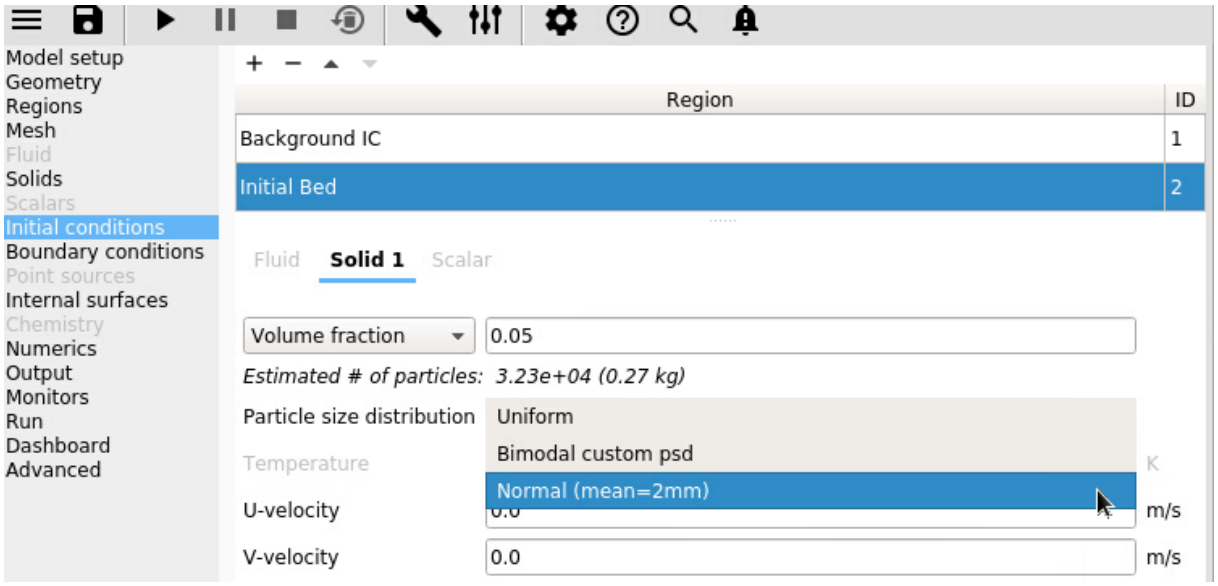


Fig. 4.7: PSD used in initial condition region.

- For TFM solids, granular temperature can only be specified when solving one of the non-algebraic viscous stress models.
- For DEM solids models, a value may be specified when using the automatic particle generator to modify the initial particle velocity distribution.

Species Mass Fractions Species mass fractions must be defined for all species when solving the species transport equations. By default the species mass fraction of the last defined fluid species is set to one and all others are set to zero. The total of all species mass fractions must equal one.

Fit particles to region *(DEM solids only)* This option is used with the automatic particle generator to expand the initial particle lattice to span the region.

DEM seeding options

Initial lattice distribution *(DEM solids only)* Particles are initially seeding on a lattice, which can be cubic or hexagonal arrangement. Hexagonal arrangement can typically achieve higher packing density than cubic arrangement.

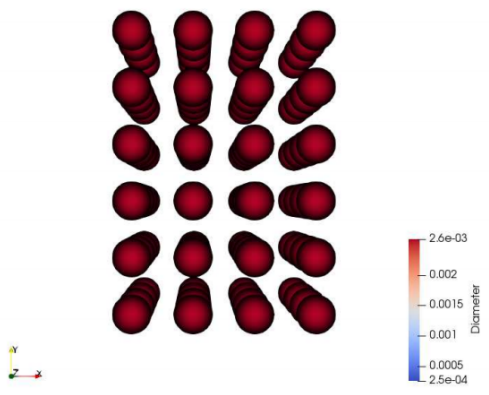
Spacing between particle *(DEM solids only)* Spacing (expressed as a fraction of the largest diameter, must be between zero and 1) between particles in the lattice. Default value is 0.05 (5% of the particle diameter). This spacing applies equally in all directions unless one of the spacing factors (x,y, or z direction) is less than one.

Spacing factor in x-direction *(DEM solids only)* Factor used to reduce the spacing in the x-direction (between zero and one). This factor is a multiplier to the overall spacing (if the spacing is 0.05 and the factor in x-direction is 0.5, then the effective spacing in the x-direction is 0.025 times the diameter).

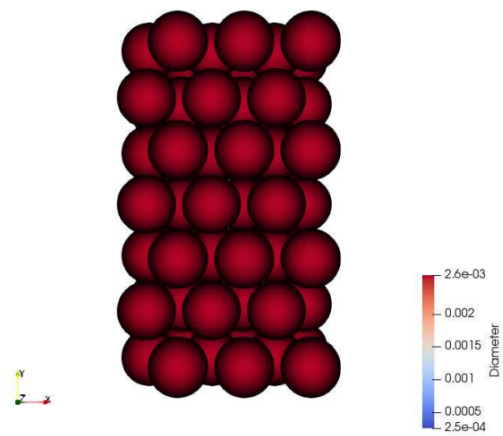
Spacing factor in y-direction *(DEM solids only)* Factor used to reduce the spacing in the y-direction (between zero and one). This factor is a multiplier to the overall spacing (if the spacing is 0.05 and the factor in y-direction is 0.5, then the effective spacing in the x-direction is 0.025 times the diameter).

Spacing factor in z-direction *(DEM solids only)* Factor used to reduce the spacing in the z-direction (between zero and one). This factor is a multiplier to the overall spacing (if the spacing is 0.05 and the factor in z-direction is 0.5, then the effective spacing in the x-direction is 0.025 times the diameter).

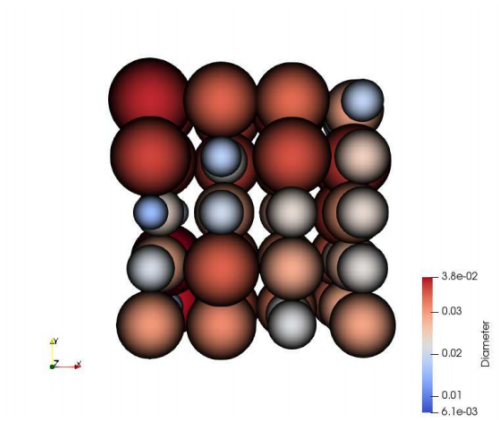
Random factor applied to particle positions *(DEM solids only)* Factor to randomize the particle position around the baseline lattice position (between zero and one). The same factor applies in all directions unless a random factor less than one is specified in the x,y or z-direction.



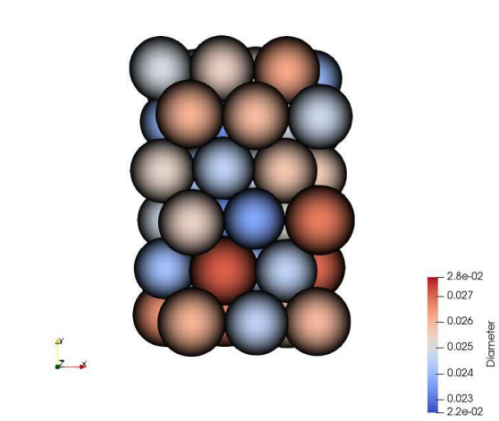
(a) Simple cubic packing for a monodisperse system



(b) Hexagonal packing for a monodisperse system



(c) Simple cubic packing for a polydisperse system



(d) Hexagonal packing for a polydisperse system

Fig. 4.8: Examples of cubic and hexagonal lattice

Random factor in x-direction *(DEM solids only)* Factor used to reduce the randomness in the x-direction (between zero and one).

Random factor in y-direction *(DEM solids only)* Factor used to reduce the randomness in the y-direction (between zero and one).

Random factor in z-direction *(DEM solids only)* Factor used to reduce the randomness in the z-direction (between zero and one).

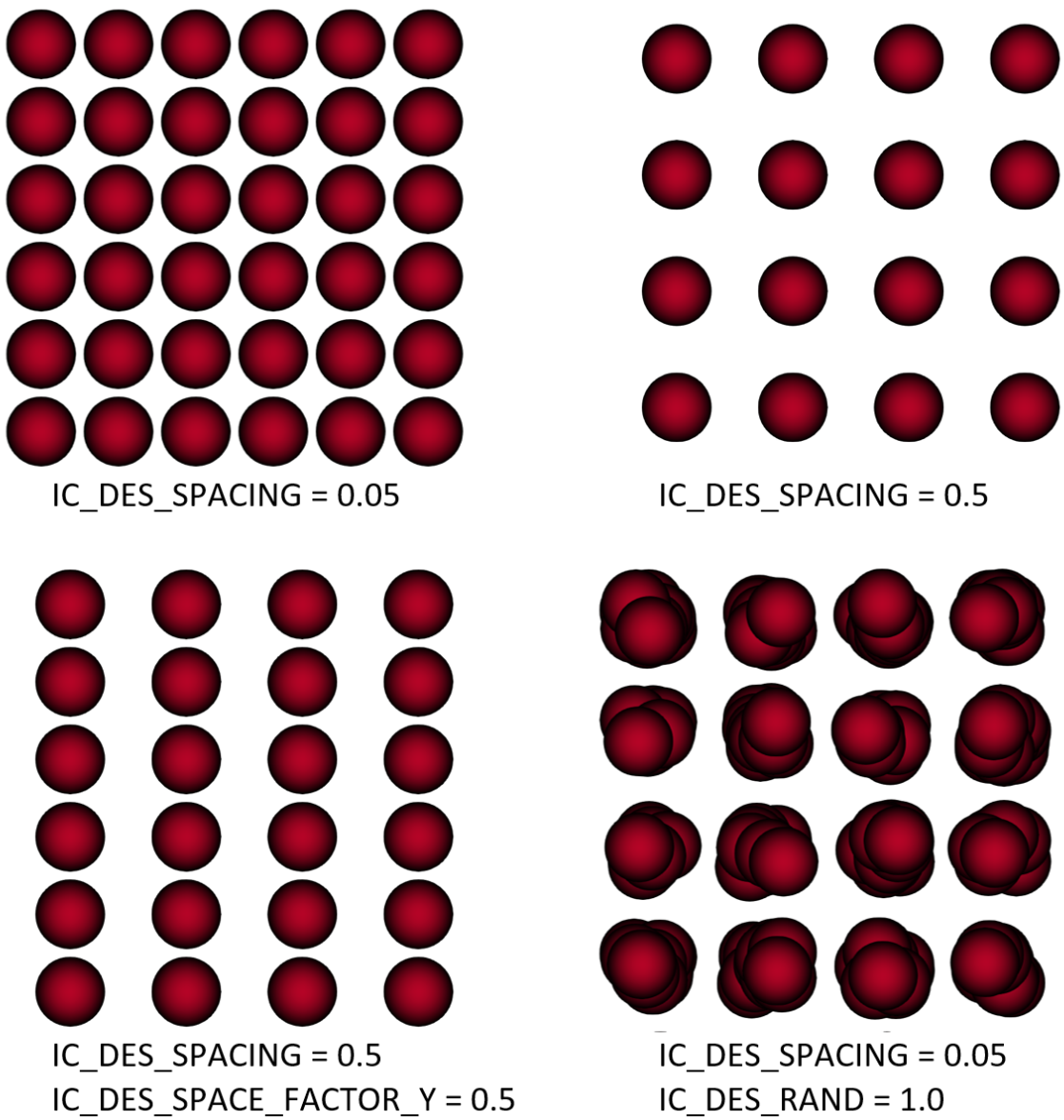


Fig. 4.9: Examples of spacing options with cubic lattice

Radiation Coefficient The radiation coefficient has a default value of zero, and can only be specified when solving the energy equations. The default value is strongly recommended.

Radiation Temperature The radiation temperature has a default value of 293.15K, and can only be specified when solving the energy equations. The default value is strongly recommended.

4.8.4 Setting User-Defined Scalar Initial Values

When one or more user-defined scalars are enabled, initial values for the scalars must be specified for the entire computational domain.

Scalar Value For each user-defined scalar, an initial field value must be provided. By default, all scalars are set to zero.

4.9 Boundary Conditions

Boundary Conditions (BCs) are specified in the Boundary conditions task pane.

4.9.1 Creating a Boundary Condition Region

To define a boundary condition, there must be a region already defined in the *Regions Pane*.

Boundary conditions are specified over rectangular regions.

Valid BC Regions

- Box regions
- XY-planes (2D simulations only)

Invalid BC Regions

- Any region already used by another IC
- Point regions
- STL regions

4.9.2 Setting Fluid Phase Boundary Values

Volume Fraction The fluid phase volume fraction has a default value of one. If solids are defined and given volume fractions greater than zero, the fluid phase volume fraction is automatically calculated by subtracting the sum of the solids phase volume fractions from one.

Temperature The fluid phase temperature has a default value of 293.15K. Specification of the fluid temperature is required when the following models are used:

- Solving energy equations
- Fluid density mode is the Ideal Gas Law
- Fluid viscosity model is Sutherland's Law

Pressure An initial pressure field may be specified. If none is provided, the fluid solver imposes a hydrostatic pressure drop across the domain if one or more pressure outflow boundary conditions is defined.

Velocity The fluid phase velocity field has a default value of zero.

Species Mass Fractions Species mass fractions must be defined for all components when solving the species transport equations. By default the species mass fraction of the last defined fluid species is set to one and all others are set to zero. The total of all species mass fractions must equal one.

Turbulence If solving the mixing length turbulence model, a mixing length scales must be defined for the whole domain.

If solving the k-ε turbulence model, initial values for the turbulent kinetic energy and turbulent dissipation must be provided.

Radiation Coefficient A radiation coefficient has a default value of zero, and can only be specified when solving the energy equations. The default value is strongly recommended.

Radiation Temperature The radiation temperature has a default value of 293.15K, and can only be specified when solving the energy equations. The default value is strongly recommended.

4.10 Point Sources

Point sources (PS) are used in place of mass inlets where either the geometry and/or grid resolution prohibit proper boundary condition specification. For example, a point source may be used to model an injector with dimensions smaller than the grid. Point sources may be defined within a single computational cell, along a plane, or as a volume of computational cells.

Point sources introduce mass directly into a computational cell unlike a boundary condition which specifies flow along a cell face. One consequence of this implementation is that point sources are subjected to convection/diffusion forces and may not travel parallel to the specified directional preference. Directional preference may be specified with a velocity vector (i.e., `PS_U_g`, `PS_V_g`, etc.), however, directional preference is not required.

Examples showing how to setup point sources can be found in: `legacy_tutorials/point_source_spiral`

4.10.1 Creating a Point Source Region

To define a point source, there must be a region already defined in the *Regions Pane*.

4.10.2 Add/Delete Point Sources

To create a new point source, press the button which will bring up the Region Selection dialog. Select a region to associate with the new point source and press OK.

Mass flow rate The mass flow rate is the rate of mass introduced for the point source. It has a default value of 0.0 kg/s

Temperature The temperature only applies to cases when energy equations are turned on under *Model Setup*. It has a default value of 293.15 K.

Velocity The velocity of the mass introduced by the point source is always available. The default velocity is zero.

4.11 Monitors

A Monitor is a tool for capturing data from the solver about the model.

Data (such as volume fraction, pressure, velocity, etc.) for a given *Region Selection* is written to a *CSV file* while the solver is running.

The number of monitors that can be defined for a project is limited to a maximum of 100.

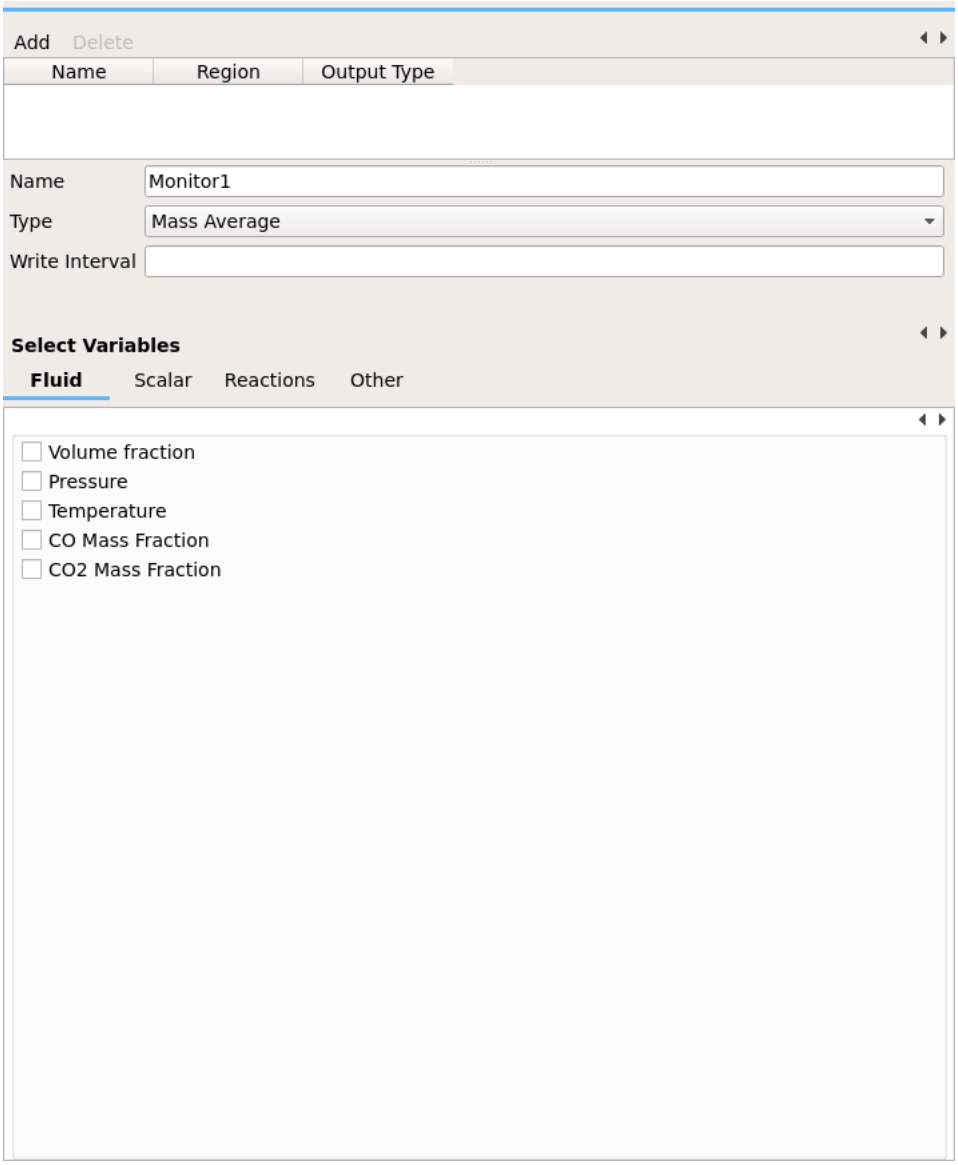


Fig. 4.10: Monitor pane

4.11.1 Region Selection

To define a monitor, there must be a region already defined in the *Regions Pane*. Click the button at the top, which will open a popup window for selecting the region. A Monitor region is a single point, plane, or volume. Multiple regions cannot be combined for a monitor, and STL regions cannot be used for monitors.

4.11.2 Monitor Output

Filename The monitor output file will have a default name based on the name of the monitor's region. You can edit the filename of a monitor by selecting the monitor from the list of monitors and changing "Filename base". The monitor data will be output to the Filename base with the extension `.csv`.

The monitor output file is in Comma Separated Value (CSV) format. The first line of the file provides header information. For example, running the Silane Pyrolysis tutorial (SP2D) will generate a file `PROBE_SPECIES.csv`:

```
#
# Run type: NEW
# "Time","x_g(1)","x_g(2)","x_g(3)","x_g(4)","x_g(5)","x_s(1,1)","x_s(1,2)"
    0.0000000    ,    0.0000000    ,    0.0000000    ,    0.0000000    ,    0.
↪0000000    ,    1.0000000    ,    0.0000000    ,    1.0000000
```

Write Interval The write interval defines how frequently the monitor data will be written to the output file. It has a default value of 0.05 seconds of simulation time.

After creating a monitor, use the menus and check boxes to select one or more variables.

4.11.3 Eulerian Monitors

The monitor variables available for the fluid phase are:

- Volume fraction (of all fluid species)
- Fluid Pressure
- Fluid Velocity
- Fluid Temperature
- Turbulent Kinetic Energy
- Turbulent Dissipation
- Volume fraction of each individual fluid species

The variables available for TFM solids include:

- Velocity of this solid phase
- Bulk Density of this solid phase
- Temperature of this solid phase
- Granular Temperature of this solid phase
- Pressure (total for all solid species for this solid phase)
- Mass fraction of each individual species of this solid phase

There is a monitor variable available for each scalar defined on the *Scalars* tab.

There is a monitor variable available for each reaction defined on the *Chemical Reactions* tab.

There are different types of monitors available. A monitor type applies an operator (for example a sum, an area integral or a volume integral) to the variable. The dimensionality of the region determines which operators can be applied.

The table below summarizes the nomenclature used to describe the monitor operators:

Symbol	Description
ϕ_{ijk}	Variable value at indexed cell
ε_{ijk}	Phase **volume fraction** at indexed cell
ρ_{jk}	Phase **density** at indexed cell
$\vec{v}_{jk}$	Phase **velocity** at indexed cell
A_{ijk}	Cross-sectional area of cell
V_{ijk}	Volume of indexed cell

Point Region

For a point region, the monitor data value is simply the value of the variable at that point:

Value Returns the value of the field quantity in the selected region.

$$\phi_{ijk}$$

Area or Volume Region

The following monitor types are valid for area and volume regions:

Sum The sum is computed by summing all values of the field quantity in the selected region.

$$\sum_{ijk} \phi_{ijk}$$

Min Minimum value of the field quantity in the selected region.

$$\min_{ijk} \phi_{ijk}$$

Max Maximum value of the field quantity in the selected region.

$$\max_{ijk} \phi_{ijk}$$

Average Average value of the field quantity in the selected region where N is the total number of observations (cells) in the selected region.

$$\phi_0 = \frac{\sum_{ijk} \phi_{ijk}}{N}$$

Standard Deviation The standard deviation of the field quantity in the selected region where ϕ_0 is the average of the variable in the selected region.

$$\sigma_\phi = \sqrt{\frac{\sum_{ijk} (\phi_{ijk} - \phi_0)^2}{N}}$$

Surface Integrals

The following types are only valid for area regions:

Area Area of selected region is computed by summing the areas of the facets that define the surface.

$$\int dA = \sum_{ijk} |A_{ijk}|$$

Area-Weighted Average The area-weighted average is computed by dividing the summation of the product of the selected variable and facet area by the total area of the region.

$$\frac{\int \phi dA}{A} = \frac{\sum_{ijk} \phi_{ijk} |A_{ijk}|}{\sum_{ijk} |A_{ijk}|}$$

Flow Rate The flow rate of a field variable through a surface is computed by summing the product of the phase volume fraction, density, the selected field variable, phase velocity normal to the facet v_n, and the facet area.

$$\int \varepsilon \rho \phi v_n dA = \sum_{ijk} \varepsilon_{ijk} \rho_{ijk} \phi_{ijk} v_{n,ijk} |A_{ijk}|$$

Mass Flow Rate The mass flow rate through a surface is computed by summing the product of the phase volume fraction, density, phase velocity normal to the facet v_n, and the facet area.

$$\int \varepsilon \rho v_n dA = \sum_{ijk} \varepsilon_{ijk} \rho_{ijk} v_{n,ijk} |A_{ijk}|$$

Mass-Weighted Average **FIXME** The mass flow rate through a surface is computed by summing the product of the phase volume fraction, density, phase velocity normal to the facet, and the facet area.

$$\frac{\int \varepsilon \rho \phi |v_n dA|}{\int \varepsilon \rho |v_n dA|} = \frac{\sum_{ijk} \varepsilon_{ijk} \rho_{ijk} \phi_{ijk} |v_{n,ijk} A_{ijk}|}{\sum_{ijk} \varepsilon_{ijk} \rho_{ijk} |v_{n,ijk} A_{ijk}|}$$

Volume Flow Rate The volume flow rate through a surface is computed by summing the product of the phase volume fraction, phase velocity normal to the facet v_n, and the facet area.

$$\int \varepsilon v_n dA = \sum_{ijk} \varepsilon_{ijk} v_{n,ijk} |A_{ijk}|$$

Volume Integrals

The following types are only valid for volume regions:

Volume The volume is computed by summing all of the cell volumes in the selected region.

$$\int dV = \sum_{ijk} |V_{ijk}|$$

Volume Integral The volume integral is computed by summing the product of the selected field variable and the cell volume.

$$\int \phi dV = \sum_{ijk} \phi_{ijk} |V_{ijk}|$$

Volume-Weighted Average The volume-weighted average is computed by dividing the summation of the product of the selected field variable and cell volume by the sum of the cell volumes.

$$\frac{\int \phi dV}{V} = \frac{\sum_{ijk} \phi_{ijk} |V_{ijk}|}{\sum_{ijk} |V_{ijk}|}$$

Mass-Weighted Integral The mass-weighted integral is computed by summing the product of phase volume fraction, density, selected field variable, and cell volume.

$$\int \varepsilon \rho \phi dV = \sum_{ijk} \varepsilon_{ijk} \rho_{ijk} \phi_{ijk} |V_{ijk}|$$

Mass-Weighted Average The mass-weighted average is computed by dividing the sum of the product of phase volume fraction, density, selected field variable, and cell volume by the summation of the product of the phase volume fraction, density, and cell volume.

$$\frac{\int \phi \rho \varepsilon dV}{\int \rho \varepsilon dV} = \frac{\sum_{ijk} \varepsilon_{ijk} \rho_{ijk} \phi_{ijk} |V_{ijk}|}{\sum_{ijk} \varepsilon_{ijk} \rho_{ijk} |V_{ijk}|}$$

4.11.4 Lagrangian Monitors

The variables available for DEM and PIC solids are:

- Radius
- Mass
- Volume
- Density
- Translational velocity components
- Rotational velocity components (DEM only)
- Temperature
- Mass fraction of each individual species
- DES user variable (DES_USR_VAR)

There are different types of monitors available. A monitor type applies an operator (for example a sum, an area integral or a volume integral) to the variable. The dimensionality of the region determines which operators can be applied.

The table below summarizes the nomenclature used to describe the monitor operators:

Symbol	Description
ϕ_p	Variable value of the indexed particle
m_p	**Mass** of the indexed particle
V_p	**Volume** of the indexed particle
$\sqsupseteq_p$	**Statistical weight** of the indexed particle[1]

[1] *The statistical weight is one for DEM simulations.*

General particle properties

General particle properties can be obtained from area (plane) and volume regions. For area regions, all particles in Eulerian cells that intersect the plane are used in evaluating the average.

Sum The sum of particle property, ϕ_p in the selected region is calculated using the following expression.

$$\sum_p w_p \phi_p$$

Min The minimum value of particle property phi_p is the selected region is obtained using the following expression.

$$\min_p \phi_p$$

Max The maximum value of particle property phi_p is the selected region is obtained using the following expression.

$$\max_p \phi_p$$

Averaged particle properties

Particle properties can be averaged over area (plane) and volume regions. For area regions, all particles in Eulerian cells that intersect the plane are used in evaluating the average.

Average The average value of particle property, ϕ_p in the selected region is calculated using the following expression. For DEM simulations, the statistical weight of a particle, w_p, is one such that the sum of the weights is the total number of observations in the selected region.

$$\bar{\phi} = \frac{\sum_p w_p \phi_p}{\sum_p w_p}$$

Standard Deviation The standard deviation of particle property, phi_p in the selected region is calculated using the following expression. $\bar{\phi}$ is the averaged variable in the selected region.

$$\sigma_\phi = \sqrt{\frac{\sum_p w_p (\phi_p - \bar{\phi})^2}{\sum_p w_p}}$$

Mass-weighted average Mass-weighted average value of particle property, ϕ_p in the selected region is calculated using the following expression.

$$\bar{\phi}_m = \frac{\sum_p w_p m_p \phi_p}{\sum_p w_p m_p}$$

Volume-weighted average Volume-weighted average value of particle property, ϕ_p in the selected region is calculated using the following expression.

$$\bar{\phi}_v = \frac{\sum_p w_p V_p \phi_p}{\sum_p w_p V_p}$$

Flow rates

Flow rate monitors for Lagrangian particles (DEM/PIC) are only valid for area (plane) regions. The set of particles crossing the flow plane, Γ is approximated using the height of the plane, h, the position of the particle, x_p, and the particle velocity normal to the plane, v_p such that

$$(v_p)(\frac{x_p - h}{\Delta t}) > 0$$

and

$$|v_p| \geq \left|\frac{x_p - h}{\Delta t}\right|$$

Flow rate The net flow rate of a general particle property ϕ_p is computed by summing the properties of the set of particles projected to have crossed the flow plane, Γ.

$$\sum_{p \in \Gamma} w_p \phi_p \frac{v_p}{|v_p|}$$

Mass-weighted flow rate The net mass-weighted flow rate is the sum of the general particle property ϕ_p multiplied by the particle mass, m_p of the set of particles projected to have crossed the flow plane, Γ.

$$\sum_{p \in \Gamma} w_p m_p \phi_p \frac{v_p}{|v_p|}$$

Volume-weighted flow rate The net volume-weighted flow rate is the sum of the general particle property ϕ_p multiplied by the particle volume, V_p of the set of particles projected to have crossed the flow plane, Γ.

$$\sum_{p \in \Gamma} \phi_p w_p V_p \frac{v_p}{|v_p|}$$

4.12 Internal Surfaces

Internal surfaces are surfaces defined within the overall boundaries of the model.

To define an internal surface, there must be a region already defined in the *Regions Pane*.

4.12.1 Add/Delete Internal Surfaces

To add an internal surface, navigate to the `ISs` pane from the Modeler View, and click the (+) button. A popup dialog will appear to select *region(s)* that will define the internal surface. Each region can only be used in a single internal surface.

All regions defined in the *regions Pane* are listed in the popup, but only the regions of the selected Surface Type are selectable.

- Plane Regions can be Impermeable internal surfaces.
- Box Regions can be X-Axis, Y-Axis, or Z-Axis Impermeable internal surfaces.
- Plane Regions can be Semi-permeable internal surfaces.
- Box Regions can be X-Axis, Y-Axis, or Z-Axis Semi-permeable internal surfaces.

To remove an internal surface, navigate to the `ISs` pane from the Modeler View, select the internal surface to remove, and click the (-) button.

4.12.2 Internal Surface Type

Planar internal surfaces can be of type:

- Impermeable
- Semi-Impermeable

Volumetric internal surfaces can be of type:

- X-Axis Impermeable
- Y-Axis Impermeable
- Z-Axis Impermeable
- X-Axis semi-permeable
- Y-Axis semi-permeable
- Z-Axis semi-permeable

4.12.3 Fluid permeability

The permeability is only definable for semipermeable regions. The default value is 1.0d32 m^2.

4.12.4 Internal resistance coefficient

The internal resistance is only definable for semipermeable regions. The default value is 0.0 m^{-1}.

4.12.5 Internal Surface Velocities

For each *solid phase*, specify the velocity of that solid through the surface (in meters/second).

Disabled for Impermeable surface (since by definition the velocity would be zero).

4.13 Chemical Reactions

This section describes how to setup reacting cases.

4.13.1 Chemical Reactions Setup

Add Reaction

To add a reaction, click on the button. **Initially the new reaction will be invalid, and disabled**. This is because it is empty, with no species defined. Add species for Reactants and Products; make sure the reaction is balanced, and then the Ok button will be enabled. Click Ok in the bottom right to finish adding the reaction.

To cancel adding an unfinished reaction, either click the button in the upper left, or the Cancel button in the lower right.

Delete Reaction

To remove an existing reaction, select the row in the table corresponding to that reaction, and click on the button.

Disable/Enable Reaction

To disable an existing reaction (without deleting it), uncheck the checkbox in first column of the reaction table.

To re-enable a disabled reaction, re-check the checkbox.

Stiff Chemistry Solver

This checkbox enables the stiff chemistry solver. This selection is always available.

Chemical reaction input is handled different from keyword pair inputs. All homogeneous gas phase chemical reactions and all heterogeneous gas-tfm solids reactions are specified between @(RXNS) and @(END) in the reaction block. All heterogeneous gas-dem and gas-pic reactions are specified between @(DES_RXNS) and @(END) in the reaction block.

Users use the globally unique species aliases to specify the chemical reaction equation. Each reaction is specified with a unique reaction identify.

Reaction Name

This is the name of the currently selected reaction in the reaction table. The reaction name is used by the solver to identify the reaction. The name of a reaction must start with a letter, contain alphanumeric characters or underscores, contain no spaces, and is limited to 32 characters.

Reactants

The Reactants are the reactant species for the currently selected reaction. Use the and buttons to add and remove species.

Products

The Products are the product species for the currently selected reaction. Use the and buttons to add and remove species.

Coefficients

The values in the Coefficients column for Reactants and Products must be set so that the reaction balances stoichiometrically, otherwise the Ok button to save changes is disabled.

The coefficient must be a positive integer or floating point number.

Species

Use the drop-down menu to select which species to use for the Reactant or Product. Each species can only be used once.

Phase

Each species for Reactants and Products can belong to any of the Phases for the model, *Fluid* or *Solid*.

Specify Heat of Reaction

Check this checkbox to optionally specify the user-defined heat of reaction for the currently selected reaction (Joules/kmol).

The "Fraction assigned to (Fluid, Solid 1, …)" is fraction of that heat of the reaction assigned to each phase. Editing the Fraction for a phase will automatically adjust the fraction(s) of the other phase(s) to add up to one.

Reaction Restrictions

For heterogeneous reactions referencing three or more phases, either only one phase has a net mass loss or one phase has a net mass gain. Specifically, two or more phases cannot have a net mass loss while two or more phases have a net mass gain.

The following examples illustrate valid and invalid reactions. For these examples, FC1, FC2, and sSoot, represent solid carbon in solids phases 1, 2 and 3, respectively. All other species belong to the gas.

- Example 1: **VALID**
  - 2*FC1 + O2 –> 2CO
  - One phase has a net mass loss, one phase has a net mass gain
- Example 2: **VALID**
  - FC1 + FC2 + O2 –> 2CO
  - Two phases have a net mass loss, one phase has a net mass gain
- Example 3: **VALID**
  - 2.2*FC1 + O2 –> 2CO + 0.2*sSoot
  - One phase has a net mass loss, two phases have a net mass gain
- Example 4: **INVALID**
  - 1.1*FC1 + 1.1*FC2 + O2 –> 2CO + 0.2*sSoot
  - Two phases have a net mass loss, two phases have a net mass gain
  - The heat of reaction, DH, and how it is distributed between phases, fracDH, must be specified if there are two or more solids phases listed as either reactants or products.

- Example 5: DH and fracDH **NOT REQUIRED**
  - 2.2*FC1 + O2 –> 2CO + 0.2*sSoot
  - Two solids phases referenced
  - Solid phase 1, FC1, is a reactant
  - Solid phase 2, FC2, is a product
- Example 6: DH and fracDH **REQUIRED**
  - FC1 + FC2 + 2O2 –> 2CO2
  - Two solids phases referenced as reactants
- Example 6: DH and fracDH **REQUIRED**
  - sSoot –> FC1 + FC2
  - Two solids phases referenced as products

4.13.2 Setting up Chemical Reaction Cases

Chemical reactions are specified in the data file (mfix.dat) by providing species aliases and chemical equations. Rate expressions are specified in one or two user defined subroutines, `usr_rates.f` and `usr_rates_des.f`, respectively. Heats of reaction are automatically calculated. Optionally, users may specify constant heats of reaction in the data file.

Tip: An overview of using legacy `rrates.f` files is given at the end of this section. However, this input method is no longer supported.

Chemical Reactions Specification

There are five general steps to incorporating chemical reactions into a simulation:

1. Provide species names in the data file (SPECIES_G, SPECIES_S).
2. Assign a unique identifier (alias) to each species in the data file. (SPECIES_ALIAS_G and SPECIES_ALIAS_S)
3. Define chemical reaction parameters in the data file.
4. Define chemical reaction rates in `usr_rates.f` and/or `usr_rates_des.f`.
5. Use build_mfixsolver to rebuild the MFiX executable.

Note: Species names must appear exactly as given in the materials database (see the Thermochemical Properties section). Species names must be exactly characters, and for most species, trailing spaces are needed.

Note: Reactions are limited to homogeneous or two-phase heterogeneous reactions (e.g., species from three separate phases cannot be referenced by any single chemical reaction).

The following explains each step in more detail.

Species Names

Provide species names in the data file (SPECIES_G, SPECIES_S).

Species Identifiers

Each species must be assigned a unique identifier (alias).

Alias formatting restrictions:

- Aliases must be unique.
- Aliases are limited to 32 characters and must be a valid Fortran variable name (i.e., alphanumeric characters or underscore, and starting with a letter).
- Aliases are not case sensitive.
- Aliases cannot conflict with existing MFiX variable names (e.g., a species alias of MU_g will cause an error when compiling MFiX).

Reaction Parameters

Define chemical reactions in the data file using species aliases.

Each reaction is identified by a reaction construct, and a reaction block is used to group reaction constructs in the data file. A reaction construct has the format, rxn_name{…}, where rxn_name is a unique identifier for the reaction. Reaction identifiers are limited to 32 characters and must follow Fortran variable naming convention.

Reaction input format:

MFiX processes chemical reaction data differently than other input in the data file. A reaction block indicates the start and end of the reaction input. A reaction construct groups a single reaction's input parameters. There are two reaction block types:

- @(RXNS)…@(END) – indicates continuum phase chemical reactions (all TFM gas and solids phase reactions and DEM homogeneous gas phase reactions).
- @(DES_RXNS)…@(DES_END) – indicates heterogeneous DEM chemical reactions (particle/gas).

Note: A data file can only contain one reaction block of each type, whereas a reaction block must contain one or more reaction constructs.

The following keywords are available within a reaction construct.

- **CHEM_EQ** The chemical equation, contained in quotes
- **DH** Heat of reaction
- **fracDH(Phase)** Fractional heat of reaction to assign to the indicated phase

Reaction construct formatting notes:

- Chemical reactions are always specified as irreversible with reactants on the left and products on the right. (CHEM_EQ = "Reactants –> Products")
- An arrow or equals sign can be used to distinguish reactants from products. (Reactants –> Products or Reactants = Products)
- Reversible reactions are specified as two irreversible reactions. (see below example, Athermal, gas phase, reversible reaction)

- Chemical equations may span several lines by including an ampersand (&) at the end of the line. As the example below illustrates, each line of the chemical equation is contained in quotation marks and the ampersand is located to the right of the second quotation mark.

```
@(RXNS)                          !  Begin reaction block
CH4_Combustion {                 !  Reaction 1 construct
chem_eq = "CH4 + 2O2 --> " &     !  Chemical Reaction Line 1
"CO2 + 2H2O"                     !  Chemical Reaction Line 2
}                                !  End reaction 1 construct
@(END)
```

- Chemical equations are limited to 512 characters, including spaces.
- Chemical equations can be bound within single or double quotes. CHEM_EQ = 'Reactants = Products' or "Reactants = Products")
- Catalytic reactions should contain a species from the catalyst phase in the chemical equation with a coefficient of zero. This insures the proper assignment of the heat of reaction. (CHEM_EQ = 'A + 0.Cat –>3.0*R' where Cat is a catalyst phase species)
- Catalyst phase species can be listed as a product, reactant, or both.

Several examples illustrating the data file input (steps 2 and 3) are provided below. Within the data input file, comments are preceded with an exclamation mark (!).

Build Custom Solver

Use `build_mfixsolver` to rebuild the MFiX executable.

See *Building Custom Interactive Solver* for detailed instructions on building the custom MFiX solver.

Rebuilding mfixsolver is required after making any of the following modifications:

- Changing the number, order, or alias of any species in the data file.
- Changing the number, order, or name of any chemical reaction in the data file.
- Changing the chemical reaction rates in either `usr_rates.f` or `usr_rates_des.f`.

Note: `build_mfixsolver` preprocesses the data file to generate the `species.inc` file which is included within the `usr_rates.f` and `usr_rates_des.f` files as code. Therefore changes in the data file may result in the executable being out of date.

Extra Notes

Below is additional reaction information.

To write out reaction rates to SPx file:

1. In the data file, mfix.dat, set NRR to the desired number of reaction rates to be written out to the file `*.SPA`. This number is typically less than or equal to the total number of reactions. 2. In a reaction UDF (`usr_rates.f` or `usr_rates_des.f`) assign the desired reaction information to the variable ReactionRates. ReactionRates is a two-dimensional array. The first index references the fluid cell, IJK, while the second index ranges from 1 to NRR.

Note: If the second index exceeds NRR, a run time error can result from over indexing the array. Using logical checks can eliminate potential errors!

Two of the above examples illustrate using the ReactionRates variable:

1. Methane Combustion: The calculated reaction rate is directly stored and a logical check is used to prevent over indexing of the ReactionRates array.
2. DES droplet evaporation: The calculated reaction rate is added to the storage array. Adding the calculated data to the storage variable is needed in DES since several discrete particles may exist in a single fluid cell. Again, a logical check is performed to prevent over indexing the array.

Use an existing (legacy) `rrates.f` file:

The legacy rrates.f file should be copied to the run directory. Additionally, the following keywords should be specified in the data file:

- *USE_RRATES*
- *SPECIES_NAME(PHASE)*
- *NMAX(PHASE)*

Note: Legacy species keywords, NMAX(m) and SPECIES_NAME(n), are required when using a legacy rrates.f file. Current species keywords NMAX_g, NMAX_s, SPECIES_g, and SPECIES_s cannot be used.

Note: The only modification needed for a legacy mfix.dat and rrates.f file combination is the inclusion of USE_RRATES=.TRUE. in the data file. An example of legacy file usage can be found in: legacy_tutorials/reactor1b

Additional remarks:

- Building with chemical reaction support requires that the data file, mfix.dat, be present in the run directory as the species aliases and reaction identifiers are needed to construct a species.inc file.
- Species aliases and reaction identifiers must be unique. The build performs a cursory check on the supplied data and exits if non-unique entries are identified.
- If any species alias or reaction identifier conflicts with an existing global variable in MFiX, an error will be reported and the code will fail to compile.

Stiff Chemistry Solver

A stiff chemistry solver has been fully integrated into MFiX. This approach first solves the convection/diffusion equations without chemical reaction source terms. A coupled set of ODEs is then directly integrated to impose chemical reactions effects. This approach may decrease simulation time by permitting larger time steps within the convection/diffusion model. However, the stiff chemistry solver may increase simulation time, especially if reactions are not stiff. Reactions are specified using the same approach outlined in the chemical reactions section.

The stiff chemistry solver is invoked by specifying the keywords *STIFF_CHEMISTRY* and *STIFF_CHEM_MAX_STEPS*

Note: The stiff chemistry solver does not support legacy rrates.f files

Note: The stiff chemistry solver is not available with DES simulations.

Additional remarks:

- Variables governing ODE convergence criteria are specified as parameters in `stiff_chem_mod.f` found in the `model/chem` directory. Additional information on these parameters and their usage is available in `model/ODEPACK.F`.
- Running your simulation in debug mode is recommended for the first time. This will catch some common programmatic errors in the `usr_rates.f` file. Additionally, the stiff chemistry solver checks for NaNs calculated in the `usr_rates.f` file.
- The tutorial located in `tutorials/tfm/silane_pyrolysis_2d` shows how to use the stiff chemistry solver.

4.13.3 Chemical Reaction Examples

Example: Methane Combustion

$$\mathrm{CH_4}(g) + 2\mathrm{O_2} \rightarrow \mathrm{CO_2}(g) + 2\mathrm{H_2O}(g)$$

Note: Heat of reaction is automatically calculated (default).

```
NMAX_g = 4                        ! No. of gas phase species
Species_g(1) = "CH4 ANHARMONIC " ! Methane
Species_g(2) = "O2"               ! Oxygen
Species_g(3) = "CO2"              ! Carbon dioxide
Species_g(4) = "H2O"              ! Water Vapor

Species_Alias_g(1)  = "CH4"  ! Methane
Species_Alias_g(2)  = "O2"   ! Oxygen
Species_Alias_g(3)  = "CO2"  ! Carbon dioxide
Species_Alias_g(4)  = "H2O"  ! Water Vapor

@(RXNS)                               ! Begin reaction block
CH4_Combustion {                      ! Reaction 1 construct
chem_eq = "CH4 + 2O2 --> CO2 + 2H2O" ! Chemical Reaction Eq
}                                     ! End reaction 1 construct
@(END)                                ! End reaction block
```

Example: Athermal, gas phase, reversible reaction

$$\mathrm{A}(g) \leftrightarrow \mathrm{R}(g)$$

Notes:

- Species database names and aliases are defined on single lines.
- The forward and backward reactions are defined separately.
- The heats of reaction are defined as zero (athermal) and explicitly assigned to the gas phase.

```
NMAX_g = 2
Species_g(1) = "A" "R"
Species_Alias_g(1) = "A" "R"

! No. of gas phase species
```

(continues on next page)

(continued from previous page)

```
! Database names
! Species aliases

@(RXNS)              !  Begin reaction block
fwd_AtoR {           !  Reaction 1 construct
chem_eq = "A --> R" !  Chemical Reaction Eq
DH = 0.0             !  (cal/moles-reacted)
fracDH(0) = 1.0      !  Gas phase HoR
}                    !  End reaction 1 construct
rvs_AtoR {           !  Reaction 2 construct
chem_eq = "R --> A" !  Chemical Reaction Eq
DH = 0.0             !  (cal/moles-reacted)
fracDH(0) = 1.0      !  Gas phase HoR
}                    !  End reaction 2 construct
@(END)               !  End reaction block
```

Example: Char combustion

$$\mathrm{C}(s) + 0.5\mathrm{O}_2(g) \rightarrow \mathrm{CO}(g)$$

Notes:

- Species database names and aliases are defined on single lines.
- The heat of reaction is defined.
- The gas phase receives 20% of the heat of reaction.
- Solids phase 1 receives 80% of the heat of reaction.

```
NMAX_g = 2

! No. gas phase species

Species_g(1) = "O2" "CO"
Species_Alias_g(1) = "O2" "CO"

! Database names
! Species aliases

NMAX_s(1) = 2
Species_s(1,1) = "C(GR) REF ELEMENT"
Species_s(1,2) = "Coal Ash"

! No. solids phase species
! Fixed Carbon (graphite)
! Coal Ash

Species_Alias_s(1,1) = "C" "Ash"

! Fixed Carbon and Coal Ash

@(RXNS)                        ! Begin reaction block
Char_Combustion {              ! Reaction 1 construct
chem_eq = "C + 0.5O2 --> CO"   ! Chemical Reaction Eq
DH = -52834.0                  ! (cal/moles-reacted)
fracDH(0) = 0.2                ! HoR assigned to gas phase
```

(continues on next page)

(continued from previous page)

```
fracDH(1) = 0.8                ! HoR assigned to s. phase 1
}                              ! End reaction 1 construct
@(END)                         ! End reaction block
```

Example: Compound DEM reaction

CO combustion:

$$\mathrm{CO}(g) + 0.5\mathrm{O_2}(g) \rightarrow \mathrm{CO_2}(g)$$

CO2 gasification:

$$\mathrm{C}(s) + \mathrm{CO_2}(g) \rightarrow 2\mathrm{CO}(g)$$

Char combustion:

$$\mathrm{C}(s) + 0.5\mathrm{O_2}(g) \rightarrow \mathrm{CO}(g)$$

Notes:

- Gas phase species names and aliases are defined on the same line.
- Heats of reaction for all reactions are calculated automatically.
- A TFM reaction block is used for the gas phase homogeneous reaction.
- A DEM reaction block is used for gas/solids reactions.
- Reaction constructs are given in one line.

```
! Gas phase species data
NMAX_g = 3
Species_g(1) = "O2"
Species_Alias_g(1) = "O2"
Species_g(2) = "CO"
Species_Alias_g(2) = "CO"
Species_g(3) = "CO2"
Species_Alias_g(3) = "CO2"
! DES solids phase species data
NMAX_s(1) = 2
Species_s(1,1) = "C(GR) REF ELEMENT"
Species_s(1,2) = "Coal Ash"
Species_Alias_s(1,1) = "C"
Species_Alias_s(1,2) = "Ash"
! Homogeneous gas phase reactions
@(RXNS)
CO_Combustion { chem_eq = "CO + 0.5O2 --> CO2" }
@(END)
! DES Reaction block
@(DES_RXNS)
CO2_Gasification { chem_eq = "2.0C + O2 --> 2CO" }
Char_Combustion { chem_eq = "C + CO2 --> 2CO" }
@(DES_END)
```

Additional comments:

- Coal Ash is not a species included in the thermochemical database and would require that its properties be given in the data file (see Section 8.14 Thermochemical properties).

- One-line reaction constructs are only possible when the heat of reaction is automatically calculated (i.e., the chemical equation is the only input parameter).

Reaction Rates

Define chemical reaction rates in user defined function (UDF) files.

- A reaction rate should be given in either `usr_rates.f` or `usr_rates_des.f` for each reaction listed in the data file.
- All TFM gas and solids phase reactions as well as homogeneous gas phase reactions for DEM simulations are to be included in `usr_rates.f`. Reaction rates defined in `usr_rates.f` must have units of reacted moles per time per volume (i.e., moles/sec/cm3 for CGS units and kmoles/sec/m3 for SI units).
- All discrete phase heterogeneous (particle/gas) reactions are to be included in `usr_rates_des.f` located in the des subfolder. Reaction rates defined in `usr_rates_des.f` must have units of reacted moles per time (i.e., moles/sec for CGS units and kmoles/sec for SI units).

Note: Formation and consumption rates are automatically calculated for each species from the reaction rate and chemical equation.

The rate in terms of reacted moles is related to the rates of formation and consumption through the stoichiometric coefficients. For example, consider the homogeneous gas phase reaction of methane combustion:

$$\mathrm{CH_4} + 2\mathrm{O_2} \rightarrow \mathrm{CO_2} + 2\mathrm{H_2}O$$

The rate in terms of reacted moles is related to the rates of formation and consumption as

$$\begin{aligned} rate &= \frac{-r_{CH_4}}{1}\left(\frac{kmol_{CH_4}/(s\cdot m^3)}{mol_{CH_4}}\right) = \frac{-r_{O_2}}{2}\left(\frac{kmol_{O_2}/(s\cdot m^3)}{mol_{O_2}}\right) \\ &= \frac{r_{CO_2}}{1}\left(\frac{kmol_{CO_2}/(s\cdot m^3)}{mol_{CO_2}}\right) = \frac{r_{H_2O}}{2}\left(\frac{kmol_{H_2O}/(s\cdot m^3)}{mol_{H_2O}}\right) \end{aligned}$$

where $-r_{CH_4}$ and $-r_{O_2}$ are the rates of consumption of methane and oxygen, and r_{CO_2} and r_{H_2O} are the rates of formation of carbon dioxide and water vapor, respectively.

Each reaction rate is assigned to the variable `RATES(rxn_name)`, where `rxn_name` is the reaction identifier used in the reaction construct. To minimize input errors when specifying reaction rates, species aliases (`SPECIES_ALIAS`) defined in the data file should be used in lieu of the associated species index.

For example, if oxygen is defined as gas phase species 2 with an alias of `O2`, (e.g., `SPECIES_ALIAS_g(2)="O2"`), when accessing gas phase species data for oxygen (e.g., molecular weight; MW_g), "O2" should be used and not the integer index 2, (e.g, `MW_g(O2)`).

Example: Methane Combustion with UDF

$$\mathrm{CH_4}(g) + 2\mathrm{O_2} \rightarrow \mathrm{CO_2}(g) + 2\mathrm{H_2O}(g)$$

Notes:

- Species database names and alias are defined on the same line.
- The fluid cell index (`IJK`) is passed as a dummy argument.
- Global field variables are referenced (RO_g, X_g, T_g, and EP_g)

- Species aliases (O2 and CH4) are used instead of the species indices.
- Reaction identifier (CH4_Combustion) is used in the rates array.
- Reaction rate is stored for post processing (see below).

```
###  mfix.dat:
NMAX_g = 4
Species_g(1) = "CH4 ANHARMONIC"
Species_g(2) = "O2"
Species_g(3) = "CO2"
Species_g(4) = "H2O"

Species_Alias_g(1) = "CH4"
Species_Alias_g(2) = "O2"
Species_Alias_g(3) = "CO2"
Species_Alias_g(4) = "H2O"

@(RXNS)
CH4_Combustion { chem_eq = "CH4 + 2O2 --> CO2 + 2H2O" }
@(END)
```

```
### usr_rates.f:
SUBROUTINE USR_RATES(IJK, RATES)

DOUBLE PRECISION, INTENT(IN) :: IJK ! Fluid Cell Index
DOUBLE PRECISION, INTENT(OUT) :: RATES(:) ! Reaction Rates
DOUBLE PRECISION c_O2 ! Oxygen concentration (mol/cm^3)
DOUBLE PRECISION c_CH4 ! Methane concentration (mol/cm^3)

! Calculate species concentrations:
c_O2 = (RO_g(IJK) * X_g(IJK,O2))/MW_g(O2)
c_CH4 = (RO_g(IJK) * X_g(IJK,CH4))/MW_g(CH4)

! Methane Combustion
!-----------------------------------------------------------------//
RATES(CH4_Combustion) = 6.7d12 * exp(-2.4358d4/T_g(IJK)) * &
EP_g(IJK) * (c_O2**1.3) * (c_CH4**0.2)

! Store the reaction rate for output/post processing.
IF(CH4_Combustion <= NRR) &
ReactionRates(IJK, CH4_Combustion) = RATES(CH4_Combustion)

END SUBROUTINE USR_RATES
```

Example: Athermal, gas phase, reversible reaction with UDF

$$A(g) \leftrightarrow R(g)$$

Notes:

- Species database names and alias are defined on the same line.
- The fluid cell index (`IJK`) is passed as a dummy argument.
- Global field variables are referenced (RO_g, X_g, T_g, and EP_g)

```
### mfix.dat:
NMAX_g = 2                      ! No. of gas phase species
Species_g(1) = "A" "R"          ! Database names
Species_Alias_g(1) = "A" "R"    ! Species Aliases

@(RXNS)              !  Begin reaction block
fwd_AtoR {           !  Reaction 1 construct
chem_eq = "A --> R" !  Chemical Reaction Eq
DH = 0.0             !  (cal/moles-reacted)
fracDH(0) = 1.0      !  Gas phase HoR
}                    !  End reaction 1 construct
rvs_AtoR {           !  Reaction 2 construct
chem_eq = "R --> A" !  Chemical Reaction Eq
DH = 0.0             !  (cal/moles-reacted)
fracDH(0) = 1.0      !  Gas phase HoR
}                    !  End reaction 2 construct
@(END)               !  End reaction block

usr_rates.f:
SUBROUTINE USR_RATES(IJK, RATES)
DOUBLE PRECISION, INTENT(IN) :: IJK

! Fluid Cell Index
DOUBLE PRECISION, INTENT(OUT) :: RATES(:) ! Reaction Rates
DOUBLE PRECISION c_A ! species A concentration (mol/cm^3)
DOUBLE PRECISION c_R ! species R concentration (mol/cm^3)

! Calculate species concentrations:
c_A = (RO_g(IJK) * X_g(IJK,A))/MW_g(A)
c_R = (RO_g(IJK) * X_g(IJK,R))/MW_g(R)

! Forward Reaction: A --> R (reacted moles/sec.cm^3)
!-------------------------------------------------------//
RATES(fwd_AtoR) = 1.2d17 * exp(-5.837d3/T_g(IJK)) * &
EP_g(IJK) * c_A

! Reverse Reaction: R --> A (reacted moles/sec.cm^3)
!-------------------------------------------------------//
RATES(rvs_AtoR) = 2.5d41 * exp(-1.4897d4/T_g(IJK)) * &
EP_g(IJK) * c_R

END SUBROUTINE USR_RATES
```

Example: Char combustion with UDF

$$\mathrm{C}(s) + 0.5\mathrm{O}_2(g) \rightarrow \mathrm{CO}(g)$$

Notes:

- The fluid cell index (`IJK`) is passed as a dummy argument.
- Algebraic expressions for the rate limiting steps are omitted for brevity.

```
###mfix.dat: see step 3.
###usr_rates.f:
SUBROUTINE USR_RATES(IJK, RATES)
DOUBLE PRECISION, INTENT(IN) :: IJK
```

(continues on next page)

(continued from previous page)

```
! Fluid Cell Index
DOUBLE PRECISION, INTENT(OUT) :: RATES(:) ! Reaction Rates
⋮
! Rate limiting steps:
DOUBLE PRECISION k_f ! film diffusion (cm/sec)
DOUBLE PRECISION k_a ! ash layer diffusions (cm/sec)
DOUBLE PRECISION k_s ! chemical kinetics (cm/sec)
DOUBLE PRECISION k_eff ! effective rate (cm/sec)

! Total surface area of solids phase 1 in IJK
DOUBLE PRECISION Sa ! (cm^2/cm^3)

! C + 0.5O2 --> CO (reacted moles/sec.cm^3)
!-------------------------------------------------------//
! Verify that solids are present
IF(.NOT.COMPARE(EP_g(IJK),ONE)) THEN
  ! Calculate film diffusion rate
  k_f = < film diffusion rate expression >
  ! (cm/sec)
  ! Calculate ash diffusion rate
  k_a = < ash diffusion rate expression >
  ! (cm/sec)
  ! Calculate kinetic rate
  k_s = < kinetic rate expression >
  ! (cm/sec)
  ! Effective rate (cm/sec)
  k_eff = ONE/(ONE/k_a + ONE/k_f + ONE/k_s)
  ! Calculate total surface area of solids phase 1
  Sa = 6.0 * EP_s(IJK,1) / D_p0(1)
  ! Calculate the reaction rate.
  RATES(Char_Combustion) = 2.0 *(Sa * k_eff * Conc(O2))
ELSE
  ! No solids --> No reaction
  RATES(Char_Combustion) = ZERO
ENDIF

END SUBROUTINE USR_RATES
```

See legacy_tutorials/SpoutedBedCombustor for details on a similar simulation setup.

Example: DES droplet evaporation with UDF

$$H_2O(l) \rightarrow H_2O(g)$$

Notes:

- Various algebraic expressions in the sample UDF are omitted for brevity.
- The global particle index (`NP`), phase index (`pM`), and fluid cell index (`IJK`) are passed as dummy arguments.

```
### mfix.dat:
NMAX_g = 2                          ! No. of gas phase species
Species_g(1) = "Air" "H2O"          ! Database names
Species_Alias_g(1) = "Air" "Vapor"  ! Species Aliases

NMAX_s(1) = 1                     ! No. of solids phase species
```

(continues on next page)

(continued from previous page)

```
Species_s(1,1) = "H2O(L)"        ! Database names
Species_Alias_s(1,1) = "Liquid"  ! Species Aliases

@(DES_RXNS)
Evap { Liquid --> Vapor }
@(DES_END)

###usr_rates_des.f:
SUBROUTINE USR_RATES_DES(NP, pM, IJK, DES_RATES)
DOUBLE PRECISION, INTENT(IN) :: NP  ! Global particle index
DOUBLE PRECISION, INTENT(IN) :: pM  ! Particle solid phase
DOUBLE PRECISION, INTENT(IN) :: IJK ! Fluid Cell Index
DOUBLE PRECISION, INTENT(OUT) :: DES_RATES(:) ! Reaction Rates
!-------------------------------------------------------//
! Calculate the concentration gradient (mole/cm^3)
Cmg_H2O = < expression for calculating gradient >

IF(Cmg_H2O > ZERO) THEN
  ! Calculate mass transfer coefficient (cm/sec)
  H2O_xfr = < mass transfer coeff calculation >
  ! Calculate droplet surface area (cm^3)
  Sa = Pi * 4.0d0 * (DES_RADIUS(NP)**2)
  ! Calculate the mass transfer rate (moles/sec)
  DES_RATES(Evap) = Sa * H2O_xfr * Cmg_H2O
ENDIF

! Store the reaction rate for post processing.
IF(Evap <= NRR) ReactionRates(Evap) = &
  ReactionRates(IJK, Evap) + DES_RATES(Evap)

END SUBROUTINE USR_RATES_DES
```

See legacy_tests/dem-tests/evaporation for additional details.

There are two workflows available to setup and run simulations:

1) The standard workflow lets users set up the entire simulation parameters (including the mesh parameters) before running the simulation. The mesh is generated at run time and therefore cannot be inspected before the simulation starts. This is the default workflow (available since the first GUI release in MFiX 17.1). This can sometimes lead to the utilization of a poor mesh and loss of productivity.

   The entire standard workflow is performed in the Modeler tab, which contains the following panes (panes are ordered in a logical progression):

- Modeler Tab
  - *Model Setup* defines the overall solver type (Single phase, TFM, DEM).
  - *Geometry* defines the overall physical space for the model.
  - *Regions* defines geometrical surfaces and volumes within the overall domain that are used in specifying boundary conditions, initial conditions, point sources, internal surfaces, outputs and monitors.
  - *Mesh* defines the spatial discretization.
  - *Fluid* defines fluid model options and physical properties.
  - *Solids* defines solids model options and physical properties.
  - *Scalars* defines scalar arrays different phases in the model.

- *Initial Conditions* defines the model initial conditions.
- *Boundary Conditions* defines the model boundary conditions.
- *Point Sources* defines point sources for the model.
- *Internal Surfaces* defines internal surfaces for the model.
- *Chemical Reactions* defines chemical reactions for the model.

2) The SMS (Segregated Mesh/Solver) workflow allows to generate and inspect the mesh before other model settings are specified. This can help identify poor meshes before run time. The mesh is saved into a .msh file and it will be read at run time by the solver. Other output files ([RUN_NAME]_boundary.vtk, [RUN_NAME]_MESH.vtu, MESH_STATS.txt) are generated to visualize the mesh and gather mesh statistics. The SMS workflow is currently (20.1 release) available as a beta feature and can be activated in the Settings menu.

   Please see *SMS meshing workflow, cyclone, Discrete Element Model (DEM)* for a tutorial showing how to use the SMS workflow.

- Mesher Tab

  The mesh generation is performed in the Mesher tab, which contains the following panes (panes are ordered in a logical progression):

  - *Geometry* defines the overall physical space for the model.
  - *Regions* defines regions that are used in boundary conditions.
  - *Mesh* defines the spatial discretization. The mesh is generated by clicking the `Generate` button. After inspection, a valid mesh must be accepted by clicking on the `Accept` button. This will unlock the Modeler tab.

- Modeler Tab

  - *Model Setup* defines the overall solver type (Single phase, TFM, DEM).
  - *Regions* defines geometrical surfaces and volumes within the overall domain that are used in specifying initial conditions, point sources, internal surfaces, outputs and monitors.
  - *Fluid* defines fluid model options and physical properties.
  - *Solids* defines solids model options and physical properties.
  - *Scalars* defines scalar arrays different phases in the model.
  - *Initial Conditions* defines the model initial conditions.
  - *Boundary Conditions* defines the model boundary conditions.
  - *Point Sources* defines point sources for the model.
  - *Internal Surfaces* defines internal surfaces for the model.
  - *Chemical Reactions* defines chemical reactions for the model.

Note: Selections on panes may enable or disable widgets based on what input parameters are required or available.

CHAPTER

FIVE

BUILDING THE SOLVER

5.1 Building Custom Interactive Solver

If you are just starting with MFiX, you could use the *default solver* installed with MFiX. It is possible to run most of the *Tutorials* with the default solver.

Note: Among the tutorials listed on the New Menu, only the tfm/silane_pyrolysis_2d case has UDFs and requires running with a custom solver to run correctly.

However, for some cases you may want to use a custom mfixsolver. For instance, when running cases with UDFs (User Defined Functions, which are defined in Fortran source files in the project directory), it is necessary to build a custom solver. A custom solver is an `mfixsolver` executable built in the project directory, specific to that project. This includes projects with *chemical reactions*, since those have UDFs.

5.1.1 Custom Solver From GUI

Press the (Build) button to display the Build Dialog box. Select appropriate compilation options, then press the "Build Solver" button. It may take a few minutes to compile. See *Build Dialog* for further details about compilation options. The "Build Command" of the build dialog will show the equivalent command for *Building Custom Interactive Solver*.

After building the custom solver, run a simulation by pressing (Run) to display the *Run Dialog*. Select the mfixsolver file you have just built.

You can pause, unpause, stop, or get info from the solver.

5.1.2 Custom Solver From Command Line

For example, the `tfm/silane_pyrolysis_2d` tutorial has UDFs and will not run with the default solver.

```
> cd tutorials/tfm/silane_pyrolysis_2d/
> ls
SP2D.mfx  usr0.f  usr1.f  usr_mod.f  usr_rates.f
```

To build a custom solver for `tfm/silane_pyrolysis_2d`:

```
> build_mfixsolver
...build output...
> ls  *
build/ lib/ mfixsolver SP2D.mfx  usr0.f  usr0.o usr1.f  usr_mod.f  usr_rates.f
```

The `build_mfixsolver` command creates a wrapper script `mfixsolver`, that runs the case-specific MFiX solver (which is installed in the `lib` directory).

The `build` directory can be deleted, and the custom `mfixsolver` will still run. However, leaving the `build` directory will greatly speed up rebuilds if you want to edit the UDFs and run `build_mfixsolver` again.

5.2 Building a Batch Solver

The "MFiX Batch Solver" refers to the command line version of MFiX as in MFiX 2016-1 and earlier versions, to contrast it with the "MFiX Interactive Solver". The batch solver binary is not installed in the MFiX conda package; it needs to be compiled before being used (as in previous versions of MFiX).

Using the Batch Solver is appropriate when you do not want to use the interactive features with the GUI. You cannot pause, unpause, stop, or monitor status from the batch solver.

To build the solver without interactive support, run:

```
> build_mfixsolver --batch
> ls
build mfixsolver
```

This command will build an executable `mfixsolver` in the current directory.

The `build` directory can be deleted, and the custom `mfixsolver` will still run. However, leaving the `build` directory will greatly speed up rebuilds if you want to edit the UDFs and run `build_mfixsolver` again.

5.2.1 Configuration options

Arguments may be passed to `build_mfixsolver` to specify various options: compiler, compiler flags, SMP/DMP support, and other options. All configuration options can be displayed with:

```
> build_mfixsolver --help

usage: build_mfixsolver [--batch] [--dmp] [--smp] [-k] [-j]
                        [FC=<Fortran compiler>]
                        [FCFLAGS=<Fortran flags>]
```

The most common arguments are given in the following table.

Argument	Description
`FC=<compiler>`	Specify the Fortran compiler command
`FCFLAGS=<flags>`	Specify Fortran compiler flags
`--dmp`	Enable distributed memory support (MPI)
`--smp`	Enable shared memory parallel support (OpenMP)

The Fortran compiler is specified by passing the `FC` argument to `build_mfixsolver`. If the `FC` argument is not specified, the script will search the environment for a Fortran compiler (usually `gfortran`).

Table 5.1: compiler value for FC

Compiler	Description
gfortran	GNU Fortran compiler (serial/smp)
ifort	Intel Fortran compiler (serial/smp)
mpif90	Generic MPI Fortran wrapper (dmp/combined smp-dmp)
mpifort	Generic MPI Fortran wrapper (dmp/combined smp-dmp)
mpiifort	Intel MPI Fortran wrapper (dmp/combined smp-dmp). Older versions commonly use the mpif90 command.

Compiler flags can be specified by passing the `FCFLAGS` argument to `build_mfixsolver`. If the `FCFLAGS` argument is not specified, the compiler defaults are used.

Table 5.2: Compiler Flags for gfortran and Intel Fortran

GNU Fortran	Intel Fortran	Description
`-g`	`-g`	Produce debugging information
`-O0`, `-O1`, `-O2`, `-O3`,	`-O0`, `-O1`, `-O2`, `-O3`,	Optimization level (refer to the compiler documentation for details on level) The following options are commonly used: • `-O0` for debugging • `-O2` for production
`-fcheck=all`	`-check all`	Generates runtime checks useful for debugging
`-fbacktrace`	`-backtrace`	Print a full stack trace when a runtime error occurs
`-g`	`-g`	Generates complete debugging information

Running `build_mfixsolver` will display the flags specified with `FCFLAGS`, and other compiler flags that are always defined by the build scripts:

```
> build_mfixsolver --batch FCFLAGS="-Wall"
...
-- MFIX build settings summary:
--    Build type        = RelWithDebInfo
--    Fortran compiler  = /usr/bin/gfortran
--    Fortran flags     = -Wall -cpp -ffree-line-length-0 -fconvert=big-endian -fPIC
```

The `build_mfixsolver` script will test specified Fortran compiler with any specified flags (covered next). If the compiler is not found or if the compiler doesn't work with the specified flags, no Makefile will be generated and an error message will be printed. When posting a question on the forum about build issues, include any error messages.

5.2.2 Make options

`build_mfixsolver` will pass the options `-j` and `-k` on to `make`. Building with `-j` on multicore systems will speed up the build process.

```
> build_mfixsolver --batch -j
```

5.2.3 Building

To build the MFiX batch solver, run `build_mfixsolver` with the `--batch` option.

```
> cd tutorials/tfm/silane_pyrolysis_2d/
> ls
SP2D.mfx  usr0.f  usr1.f  usr_mod.f  usr_rates.f
> build_mfixsolver --batch
...build output...
> ls  *
build mfixsolver SP2D.mfx  usr0.f  usr0.o usr1.f  usr1.o  usr_mod.f  usr_mod.o usr_
↪rates.f usr_rates.o
```

Note the absence of a `lib` directory. The `lib` directory has the Python module for the interactive solver, and therefore is not present for a batch solver. Also, `mfixsolver` is a binary executable (not a wrapper script).

5.3 Build from Source (for Developers)

5.3.1 Obtaining MFiX Source

This section describes how developers build MFiX packages for distribution. Most users will not need to do this.

Building from source requires the MFiX source tarball, which can be obtained on the MFiX download page. Extract the tarball and go into the top level source directory:

```
> tar xf mfix-20.4-dev.tar.gz
> cd mfix-20.4-dev
> pwd
/home/user/mfix-20.4-dev
```

5.3.2 Building Solver from Source

You can build the *Batch Solver* from source, without the MFiX conda package installed. This is how MFiX was built in version 2016 and earlier.

For prerequisites, see *Linux Build Dependencies* or *Mac Build Dependencies*.

Run `cmake` to configure MFiX and generate a Makefile, then run `make` to build the MFiX solver.

Running `cmake` will generate a `CMakeCache.txt` file in the directory it is invoked from. Please include `CMakeCache.txt` when posting a question about build issues to the forum.

Configuring

The default configuration for MFiX is serial with optimizations and debug symbols (standard optimization; for GFortran use flags `-g -O2`):

```
> pwd
/home/user/mfix-20.4-dev   # actual working directory will vary
> mkdir release-build
> cd release-build
> cmake .. -DCMAKE_BUILD_TYPE=RelWithDebInfo   # .. is the relative path to the top-
↪level MFiX source directory
> cmake .. # same thing (RelWithDebInfo is the default CMAKE_BUILD_TYPE)
```

Note: Using an "out of source" build is recommended. Create a new directory, `cd` to that directory, and run `cmake <path>` where `<path>` is the top-level MFiX source directory (containing `CMakeLists.txt`). Out of source builds are convenient for having different solvers built with different configurations. The build directory can be whatever you want; `release-build` and `debug-build` used here are just examples.

To configure the MFiX solver in serial with debug mode (no optimization; for GFortran use flags `-g -O0`):

```
> mkdir debug-build
> cd debug-build
> cmake .. -DCMAKE_BUILD_TYPE=Debug
```

If `mpifort` is your MPI compiler wrapper command, then to configure the MFiX solver with DMP with optimizations and debug symbols:

```
> mkdir release-build
> cd release-build
> cmake .. -DCMAKE_BUILD_TYPE=RelWithDebInfo -DENABLE_MPI=1
> cmake .. # same thing (RelWithDebInfo is the default CMAKE_BUILD_TYPE)
```

If `mpifort` is your MPI compiler wrapper command, then to configure the MFiX solver with DMP with debug mode:

```
> mkdir debug-build
> cd debug-build
> cmake .. -DCMAKE_BUILD_TYPE=Debug -DENABLE_MPI=1
```

To specify the Fortran compiler, define CMAKE_Fortran_COMPILER:

```
> cmake .. -DCMAKE_Fortran_COMPILER=ifort  # for example
```

To add specific Fortran compiler flags, define CMAKE_Fortran_FLAGS:

```
> cmake .. -DCMAKE_Fortran_FLAGS="-O0 -g -Wall -fcheck=all"  # for example
```

To configure with SMP support:

```
> cmake .. -ENABLE_OpenMP=1
```

Options can be combined in a single command:

```
> cmake .. -ENABLE_OpenMP=1 -DCMAKE_Fortran_COMPILER=ifort -DCMAKE_Fortran_FLAGS="-O0␣
↪-g -Wall -fcheck=all"
```

Example with gfortran, serial optimized solver configuration:

```
> cmake .. -DENABLE_MPI=0 -DCMAKE_Fortran_COMPILER=gfortran -DCMAKE_Fortran_FLAGS="-
↪O2 -g"
```

Example with gfortran, DMP optimized solver configuration:

```
> cmake .. -DENABLE_MPI=1 -DMPI_Fortran_COMPILER=mpifort -DCMAKE_Fortran_FLAGS="-O2 -g
↪"
```

Example with gfortran, serial debug mode solver configuration:

```
> cmake .. -DENABLE_MPI=0 -DCMAKE_Fortran_COMPILER=gfortran -DCMAKE_Fortran_FLAGS="-
↪O0 -fcheck=all -ffpe-trap=invalid,zero,overflow -g"
```

Example with gfortran, DMP debug mode solver configuration:

```
> cmake .. -DENABLE_MPI=1 -DMPI_Fortran_COMPILER=mpifort -DCMAKE_Fortran_FLAGS="-O0 -
↪fcheck=all -ffpe-trap=invalid,zero,overflow -g"
```

Note: For DMP support, defining `-DENABLE_MPI=1` is required. Defining `CMAKE_Fortran_COMPILER` as your MPI wrapper (`mpifort`) is recommended, but not strictly required. CMake should automatically detect MPI include files and libraries for your default compiler, but specifying `CMAKE_Fortran_COMPILER` is better to ensure you are using the exact compiler you intend to use.

Note: CMake uses configurations values from both `CMakeCache.txt` and from command line arguments. Command line arguments take precedence over `CMakeCache.txt`, but `CMakeCache.txt` is used if nothing is specified on the command line. For instance, if you run `cmake .. -DCMAKE_Fortran_FLAGS="-fcheck=all"`, and then run `cmake ..` again (in the same build directory), `-fcheck=all` will still be used (because it is still in `CMakeCache.txt`). If this is not what you want, either edit `CMakeCache.txt`, or just delete `CMakeCache.txt` and run `cmake` again with different options.

Building

After configuring, build with make. For further details about the make utility, type `man make` at the prompt.

```
> make
> make -j         # for parallel build jobs
> make VERBOSE=1 # to view the full command lines used for compiling and linking
```

At the end of a successful build, the MFiX solver will be located in the current directory. The solver is called `mfix-solver` on Linux.

5.3.3 Building Conda package

To build the conda package of MFiX, you will need to install the build dependencies for your platform. Please see *Linux Build Dependencies* or *Mac Build Dependencies*.

Install `conda-build`:

```
> conda install conda-build
```

The conda package recipes are under `conda/`. Refer to the conda-build documentation for details.

CHAPTER

SIX

RUNNING THE SOLVER

6.1 Default Interactive Solver

The default solver does not need to be compiled, because the binary executable is installed with MFiX. The default `mfixsolver` is automatically installed when following the steps in *Getting Started*.

6.1.1 Running in GUI

Start the MFiX GUI from the Anaconda Prompt (Windows) or `bash` (Mac or Linux):

```
# Windows
(mfix-20.4-dev) C:\> mfix

# Linux
(mfix-20.4-dev) > mfix

# Mac
(mfix-20.4-dev) > mfix
```

From the main menu browse to create a new project, or open an existing one.

When running a simulation, press (run) and select the default mfixsolver.

You can pause, unpause, and stop the solver from the toolbar.

6.1.2 Running From Command Line

If you have a project file `<RUN_NAME>.mfx` saved in a project directory named `<RUN_NAME>`, navigate to that project directory to launch the default solver locally. One can accomplish this with the following commands:

```
> cd <RUN_NAME>
> mfixsolver -f <RUN_NAME>.mfx
```

For example, to run the 3D fluidized bed tutorial, with the project file `FB3D.mfx` created in the GUI and saved in a directory `FB3D`:

```
> cd FB3D
> mfixsolver -f FB3D.mfx
```

To show additional command line options for the solver, type:

```
> mfixsolver --help
```

6.2 Running Custom Interactive Solver

First *build a custom solver*.

6.2.1 Running Custom Solver From GUI

After building the custom solver, run a simulation by pressing (Run) to display the *Run Dialog*. Select the mfixsolver file you have just built.

You can pause, unpause, stop, or get info from the solver.

6.2.2 Running Custom Solver From Command Line

As an example, the `tfm/silane_pyrolysis_2d` tutorial has UDFs and will not run with the default solver.

After *building the solver*, on a Linux or Mac system, run the custom solver with the command:

```
~/projectdir> ./mfixsolver -f SP2D.mfx
```

Use `./` in front of mfixsolver to use the solver in the project directory, not the default solver in your `PATH`.

On Windows, run the custom solver in the project directory with:

```
C:\projectdir > mfixsolver -f SP2D.mfx
```

6.3 Running a Batch Solver

The "MFiX Batch Solver" refers to the command line version of MFiX including MFiX 2016-1 and earlier versions. The "MFiX Interactive Solver" refers to all later releases of MFiX. The batch solver binary is not installed in the MFiX conda package; it needs to be compiled before being used (as in previous versions of MFiX).

Using the Batch Solver is appropriate when you do not want to use the interactive features with the GUI. You cannot pause, unpause, stop, or monitor status from the batch solver.

To build the batch solver see *Building a Batch Solver*.

6.3.1 Running (Serial)

To run a serial instance of the MFiX solver, type: .. code-block:: shell

> ./mfixsolver -f DES_FB1.mfx

6.3.2 Running with DMP

If you built the solver *configured with DMP support*, then you can run MPI jobs with the standard MPI wrapper command `mpirun`. For instance, to run using 4 cores with the domain divided along the X and Y axes:

```
> mpirun -np 4 ./mfixsolver -f DES_FB1.mfx NODESI=2 NODESJ=2
```

The number of MPI processes specified with `-np` must be equal to the product of the keywords `NODESI*NODESJ*NODESK`.

For further details on using `mpirun`, see the documentation for the MPI implementation for your system.

6.3.3 Running with SMP

If you built the solver *configured with SMP support*, then you can set the number of OpenMP threads using the `OMP_NUM_THREADS` environment variable. For instance, to run with four threads:

```
> env OMP_NUM_THREADS=4 ./mfixsolver -f DES_FB1.mfx
```

CHAPTER

SEVEN

VISUALIZATION

This section explains how to visualize results in the GUI, with paraview, and with postmfix.

7.1 Visualization in GUI

As a means to demonstrate visualization techniques with the MFiX GUI, references within this section are associated with the `dem/hopper_3d` case. We advise loading and running this project to get familiar with the results visualization techniques.

The `dem/hopper_3d` case will take several minutes to run. On the *Run* Pane, you can see that the Stop time is 5.0 seconds.

- Click on the (new) button in the upper right corner of the window.
- Click on the (VTK) button to create a new *VTK Tab(s)*.

You should see the particles for the 3D Hopper case.

- Click the (play) button in the middle of the top of the VTK results window.

If you don't see any change, wait a few minutes for more output files to be written.

- Click the (first) button to play the visualization from the beginning.

7.2 Post Processing

MFiX can generate output data in several formats for visualization and analysis. The command line tool `postmfix` is distributed with MFiX. In addition, the MFiX output file format is supported by the open source GUI tools ParaView (version 4.1 or later) and VisIt.

The following table lists the files typically associated with an MFiX simulation.

Table 7.1: Output Format Table

Filename	File Type	Fluid Data	Particle Data	Paraview	Visit
`<RUN_NAME>.RES`	MFiX Binary	Yes	No	Yes	Yes
`<RUN_NAME>_###_DES.RES`	MFiX Binary	No	Yes	No	No
`<RUN_NAME>.pvd`	VTK	Yes	No	Yes	No
`<RUN_NAME>_###.vtu`	VTK	Yes	No	Yes	Yes
`<VTK_REGION>.pvd`	VTK	Yes	No	Yes	No
`<VTK_REGION>_###.vtu`	VTK	Yes	No	Yes	Yes
`<RUN_NAME>_DES.pvd`	VTK	No	Yes	Yes	No
`<RUN_NAME>_DES_###.vtp`	VTK	No	Yes	Yes	Yes
`<VTK_REGION>.pvd`	VTK	No	Yes	Yes	No
`<VTK_REGION>_###.vtp`	VTK	No	Yes	Yes	Yes

Notes:

1. pvd files only contain information linking the respective .vtu and .vtp files to time- step information.
2. <VTK_REGION >_###.vtp files are binary files. The <RUN_NAME >_DES_###.vtp are ASCII files.

7.2.1 Running ParaView

This walk through demonstrates how to use ParaView to visualize the results of the *dem/fluid_bed_3d* tutorial. It is assumed that the *dem/fluid_bed_3d* tutorial was successfully run with default settings and that ParaView is installed on your system. First, the tutorial demonstrates how to visualize the fluid field; then, the DEM particles are added using spherical glyphs.

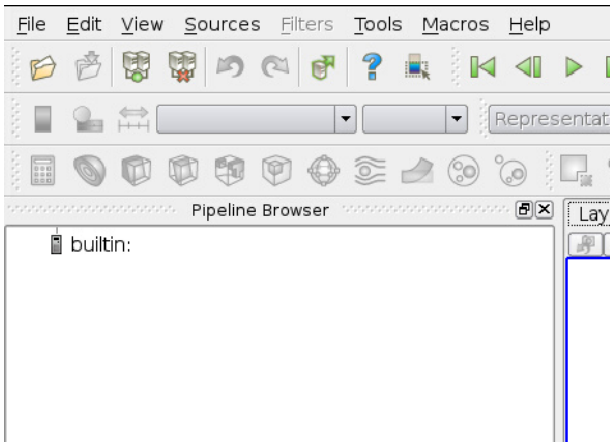


Fig. 7.1: Click on File icon , or File > Open

Fluid field results for a standard structured mesh are displayed in ParaView by loading the .RES file. To open the RES file, click on File/Open (Fig. 7.1), and navigate to your run directory.

Select the `DES_FB1.RES` file (Fig. 7.2). Do NOT load the DES restart file (`DES_FB1_DES.RES`). This binary file is not supported by any visualization software. It only contains restart data and will likely cause ParaView to crash if loaded.

Once the file is loaded, you need to click on the green "Apply" icon to load all of the variables.

Newer versions of ParaView require that you select the appropriate reader. Choose to open the data with "MFiX Unstructured Grid Files". (Fig. 7.3)

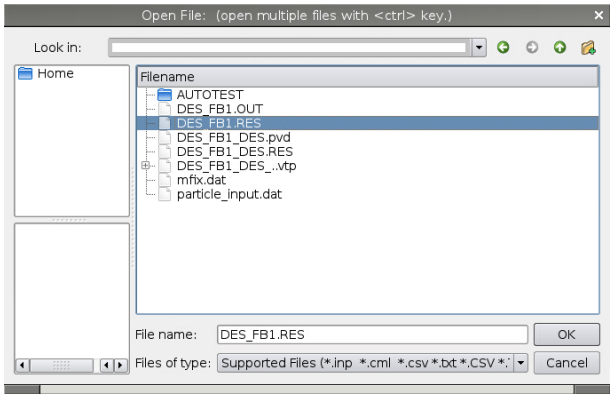


Fig. 7.2: Load `DES_FB1.RES` (do NOT load `DES_FB1_DES.RES`)

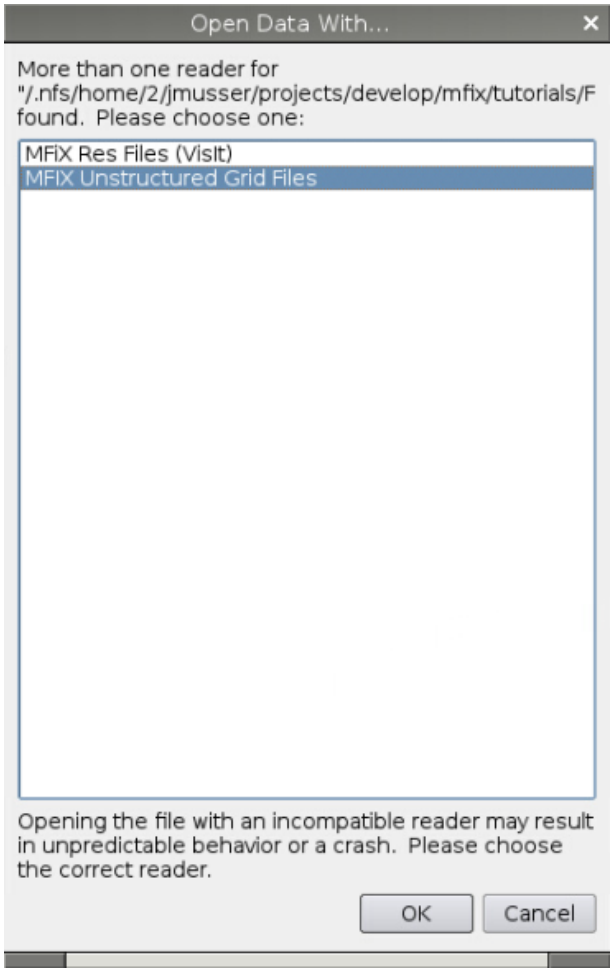


Fig. 7.3: Select "MFiX Unstructured Grid Files"

ParaView typically displays the gas phase volume fraction once the data is loaded. It may be necessary to rescale the data range as the initial range may only be suitable for displaying the initial conditions. This can be done by clicking the icon.

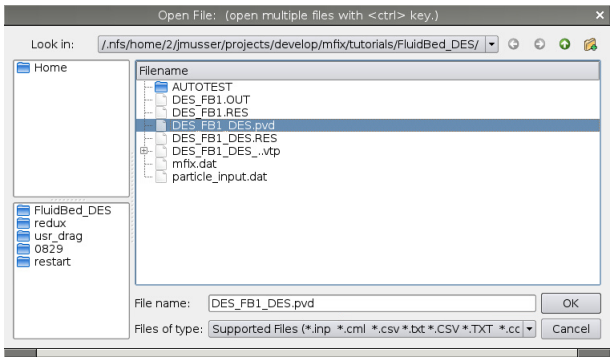


Fig. 7.4: Open DES_FB1_DES.pvd

DEM and PIC particle simulation data is loaded into ParaView by opening the `.pvd` file. (Fig. 7.4)

Again, click on the green "Apply" icon to load all of the variables. (Fig. 7.5)

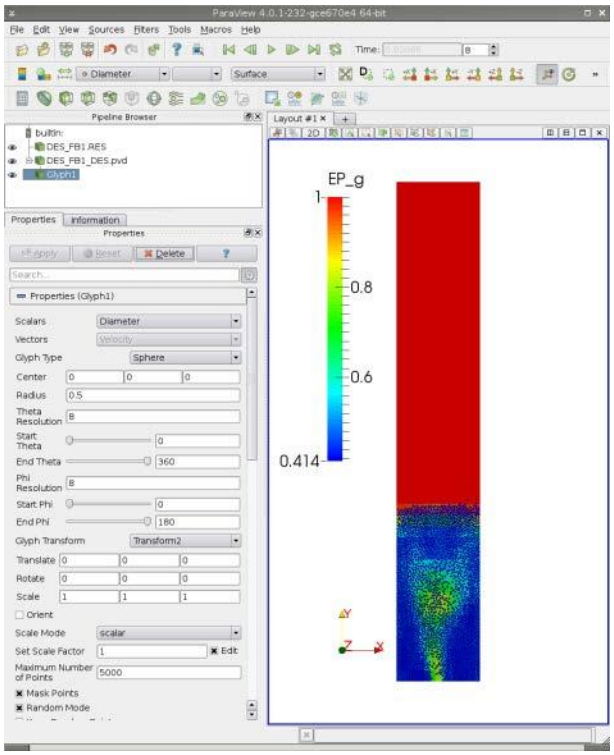


Fig. 7.5: Load all variables

Particles are shown by applying a glyph to the dataset. To apply a glyph filter, either:

- Filters > Alphabetical > Glyph
- or click the icon.

Commonly, particle data is represented using:

1. *Sphere* glyph type and
2. Scalar scale mode with a scale factor of one.

Note that these screenshots may appear differently depending on your version of ParaView.

7.2.2 Building postmfix

The `postmfix` command is used for reading the binary MFiX `.SPx` output files and outputting data to text files. The following walk through demonstrates how to run `postmfix` on the results from the *dem/fluid_bed_3d* tutorial with the default settings.

The `postmfix` executable is built by running `build_mfixsolver` with the argument `postmfix`:

From the Anaconda package:

```
> build_mfixsolver postmfix
```

From source:

```
> cd mfix-20.4
> mkdir build
> cd build/
> cmake .. -DENABLE_POSTMFIX=1
> make postmfix
```

7.2.3 Running postmfix

After building, run the executable and enter the run name:

```
> ./postmfix
Enter the RUN_NAME to post_process > DES_FB1 ? DES_FB1<Enter>
```

A simple menu of options is presented. Type *1* to examine/print data and press Enter.

```
**************************************************
0 - Exit POST_MFIX
1 - Examine/print data
2 - Write .RES from data in .SPx files
3 - Write .RES for a new grid, using old data
4 - Calculate miscellaneous quantities
5 - Print out variables
6 - Call user defined subroutine USR_POST
7 - Write a new SPx file with selected records
8 - Write new SPx files with time averaged data
9 - Perform ORNL calculations
10 - run scavenger code
********************************** **************
Enter menu selection > 1

Interactive data retrieval program. Type ? any time for help,
or press RETURN to select default values shown in parenthesis.
```

Type F to use the default data precision and press Enter.

```
Write output using user - supplied precision? (T/F) F ? F<Enter>
```

Enter a time range for data extraction and analysis. The default simulation has a simulation length of one second, so enter a range from 0.1 seconds to 0.9 seconds. The next prompt asks if the data should be time averaged. Press Enter to skip the averaging.

```
Time: (0.000, 0.000) > 0.1, 0.9
Time average ? (N) > ? < Enter >
```

Enter the variable of interest. The default is the gas phase volume fraction, EP_G. A complete list of possible entries is given by typing *?* and Enter. Press Enter to select the gas phase volume fraction.

```
Variable: (EP_g) > ? < Enter >
```

Next, enter the spatial range to extract the data. This requires an understanding of the I/J/K values for your simulation. Basic geometric information for the simulation is provided in the .OUT file. For this example, we will take a vertical slice from the approximate center of the 2D domain.

```
I range: (1, 1) > 8 , 8 ? 8,8< Enter >
J range: (1, 1) >2,40 ? 2,40< Enter >
Average or sum over J? (N) > ? < Enter >
K range: (1, 1) > ? < Enter >
```

Specify where to output the data. Press Enter to select the default `*`, which will send the data to the terminal. Alternatively, specify a file name to save the data in a data file.

```
File: (*) > ? <Enter> prints to screen
(filename<Enter> saves to filename)
X = 6.5000
Z = - 0.20000
Time = 0.10075
Y      EP_g
1.0000 0.43971
3.0000 0.45994
....etc....
```

After the output is generated, you will be presented with a prompt to continue data extraction or analysis. Type *q* and Enter to return to the main menu. From the main menu, type *0* and Enter to exit .

```
Time: (1.000, 1.000) > q ? q< Enter > to quit
< main menu is displayed again >
Enter menu selection > 0 ? 0< Enter > to exit
```

CHAPTER

EIGHT

REFERENCE

8.1 GUI Reference

8.1.1 Main Toolbar

The Main toolbar is located at the top of the GUI window. The following table illustrates the icons in the toolbar.

Icon	Description
	Menu
	Saves the current project file
	Start/resume MFIX simulation
	Pause simulation
	Stop simulation
	Reset - clear output files
	Build Dialog
	Parameter Dialog

The interactive solver sends its status during the simulation, and the solver controls will be enabled or disabled, depending on the solver state.

Solver Controls:

	Solver State is....			
	...Running and....		...Stopped with...	
	...Unpaused.	...Paused.	no RES.	RES.
Start	Disabled	Unpause	Start	Resume
Pause	Pause	Disabled		
Stop	Stops the solver		Disabled	
Reset	Disabled			Clear RES

If no MFiX job is currently running, the Start button will open the run dialog where settings for a job can be changed. The simulation can be started from this dialog. If an MFiX job is running, you can pause it with the pause button and un-pause it with the start button. You can stop a running MFiX job with the stop button. A stopped job leaves restart (*.RES) and output files to resume the simulation. Starting a job by pressing the Start button with *.RES files present will resume the job where it stopped. The Reset button will allow the option to delete the *.RES files and output files from the project directory. This will allow the simulation to start from the beginning the next time the Start button is pressed.

Run Dialog

The run dialog is used for entering options on starting the solver.

Different solvers can be selected.

If a custom solver is compiled with SMP or DMP support, the following options may be available:

- Threads (number of threads used for OpenMP)
- NODESI (number divisions in X direction)
- NODESJ (number divisions in Y direction)
- NODESK (number divisions in Z direction)

$NODESI \times NODESJ \times NODESK = n$ where n is the number of MPI processes running. If not using MPI, $NODESI = NODESJ = NODESK = 1$.

The GUI supports running both locally as well as submitting to a queue.

Note: SMP and DMP options are disabled for the default solver. You will need to build a custom solver with DMP or SMP flags to enable these options.

Run Locally

To run locally, select a solver from the drop-down list or click the browse button to specify a solver that is not in the list. Usually the default installed solver should be sufficient. If running a case with UDFs, you need to first build a case-specific MFiX as described in the *Getting Started*. You may want to build your own solver for other reasons, such as specifying various compiler flags to optimize the executable for your specific hardware.

Make sure the "Submit to queue" check-box is unchecked, and click "Run" in the Run dialog to start the solver.

Submit to Queue

To submit a job to a queue (submit to queuing system, e.g. Grid Engine, PBS, SLURM), select the "Submit to Queue" tab. Select an executable from the drop-down list or click the browse button to specify an executable that is not in the list. Next, select a template from the drop-down list or click the browse button to choose a template that is not in the drop-down list. Based on the selected template, the widgets in the "queue options" section will update automatically. Once the options are specified, click "submit" in the run dialog to submit the job to the queue.

Custom queue scripts are supported. The format for this script is described in the *Queue Templates* section.

Build Dialog

The build dialog is used for building custom solvers in project run directories. This is used to build *Building Custom Interactive Solver*.

There is a combo box to select the Fortran compiler command. It will be populated with any *known Fortran compilers* in `PATH`. Alternatively, type in the command for the compiler.

There is a field to type in the flags for the Fortran compiler. See your compiler manual for details, such as the GNU Fortran Manual.

There are checkboxes for building with SMP or DMP support. Note that DMP support requires both checking the DMP box, and selecting an MPI compiler (such as *mpifort*). The "Build solver in parallel" checkbox will run `make` with multiple jobs. DMP and SMP are not available on Windows.

If "Enable developer mode" is checked in *Settings*, then there will be a checkbox for building either the interactive solver or the batch solver.

"Build Command" displays the command line used for building the solver with the selected options.

Reset Dialog

The Reset dialog allows for optional deleting of output files from the run directory. These files include:

File Type	Wild-card match
Restart	*.RES
SPx	*.SP?
VTK	*.vtp, *.vtu, *.pid
Other	*.OUT, *.pid, *.error, *.e[0-9]*, *.pe[0-9]*, *.po[0-9]*

*.RES and *.SPx files have to be removed from the run directory before a simulation can be played from the beginning. It is recommended to remove VTK files from the run directory as well because they will be over-written.

Note: If there are restart files present in the run directory, some widgets will be disabled because certain model parameters can not be edited during a resume, or restart state.

Parameter Dialog

The parameter dialog allows users to create parameters, or variables, that can then be referenced in widgets that support the use of these parameters. This functionality allows user to create relationships among various inputs and change the values of multiple items by changing the value of a single parameter. In many respects, this is similar to features present in most commercial CAD packages.

There are six special parameters; `xmin`, `xmax`, `ymin`, `ymax`, `zmin`, and `zmax` that reference the simulation domain extents, entered on the geometry pane. These parameters make it easy to setup regions that automatically adjust to the simulation domain if the extents change.

8.1.2 Menu

The main menu allows for opening, creating, copying, and exporting projects as well as viewing project meta data, changing settings, and viewing available help documentation, including this document.

Project Info

Selecting File > Project Info displays metadata about the current project file. The following table details the content of the metadata.

Label	Description
File name	full path to MFiX project file
Project version	version number, incremented each time project is saved
Created with MFiX version	version of MFiX that project was created with
Author	user name who created the project
Modified By	list of user names that edited the project
Last modified	date and time the project was last modified
Created	date and time the project was created
Notes	area where the user can record notes about the project

New

Selecting File > New creates a new project file from a template file. When a template is selected, the GUI will ask for a project name and parent directory to save the project to. A new directory with the project name will be created in the parent directory. A .mfx file will be created in that directory by copying the contents of the selected template file. Any other files, such as *.stl files, will also be copied if present. The list of templates can be filtered by selecting one or more of the following model types:

Icon	Description
	Single phase Model
	Two Fluid Model (TFM)
	Particle in Cell Model (PIC)
	Discrete Element Model (DEM)
	Hybrid Model (TFM + DEM)
	Cartesian cut-cell (complex) geometry
	Chemistry

Open

Selecting File > Open accesses a file dialog where the user can navigate to and select an existing project file both in .mfx format of the older mfix.dat format. Note that mfix.dat files used with older released versions of MFiX will be saved in the current .mfx format. The GUI also performs a number of updates, including keyword amendments and CGS to SI unit conversion. Any user files are not converted. It is strongly suggested that the project be checked after this update.

Save

Selecting File > Save saves the current project files in the current project directory.

Save as

Selecting File > Save as opens a file dialog where the user can navigate to and select a different file location or change the current file name. Subsequently, the project is opened in that location and with that filename.

Export project

Export the current project to a new directory and/or as a new filename, but keeps the original project opened.

Submit bug report

When reporting issues, errors, or questions to the *Support Forum*, it is helpful to include your project file, UDFs, and any other files associated with your project, along with the current versions of MFiX and its dependencies, so that we may reproduce the reported issue.

We offer a convenient way of sending this information in a single file.

Clicking on 'Submit bug report' will create a .ZIP file the current directory, containing the relevant project files, and some version information about your system.

The bug report is not automatically transmitted to the MFiX developers. You are encouraged to open the zipfile and examine the files, if you have any concern about what information is being shared. Then post the zipfile as an attachment to https://mfix.netl.doe.gov/forum.

The bug report dialog will also pop up automatically if an exception occurs while using MFiX. This kind of report is especially helpful to MFiX developers, because it includes the stack trace.

Settings

Choosing File > Settings opens a dialog where the user can change settings that affect how the GUI behaves. Available options are listed in the following table.

Option	Description
Style	change the application style
Enable animations	enable or disable animated widgets
Animation Speed	set the speed at which animations occur
Enable Developer Tools	hide/show widgets that are mainly for developers of the GUI

Help

Selecting File > Help opens a dialog linking the user to available documentation (including this document) and tutorials, both text based and videos (if connected to the Internet) demonstrating features of the GUI.

About

Choosing File > About displays the version of MFiX and the versions of the various dependencies that the GUI is using. If issues are discovered while using the GUI, it is recommended that any reports include this version information which can be easily copied by pressing the "copy version info" button.

Quit

Selecting File > Quit exits MFiX. The GUI will ask for confirmation if project is unsaved or if a job is running.

8.1.3 Model Panes

Each pane in the main window allows editing of different options for an MFiX project. The Panes are ordered in a logical progression for traditional CFD problem setups: Model defines the overall solver type (TFM, DEM, PIC). Geometry and Mesh define the overall domain space of the solver. Regions are geometrical surfaces and volumes within the overall domain. Fluid and Solids specify details on the phases the solver deals with. Initial Conditions (ICs), Boundary Conditions (BCs), Point Sources (PSs), and Internal Surfaces (ISs) depend on Regions, Fluid, and Solids, so they come next. Chemistry provides options for reacting cases, and Numerics specifies details on the Eulerian solver. Output specifies how output data is written. Monitors provides options to probe the flow field at select locations and compute basic quantities of interest. Run specifies how long to run the solver for. (Post is not yet implemented).

Note: Selections on panes may enable or disable widgets based on what input parameters are required or available.

Model Setup

The Model pane is used to specify overall options for the project. Depending on what is selected, other panes may be enabled or disabled. Options that are specified on this pane include:

- Solver (Single Phase, Two-Fluid Model, DEM, PIC, Hybrid)
- Option to disable the fluid phase
- Option to enable thermal energy equations
- Option to enable turbulence, if the fluid phase is enabled
- Gravity in the x, y, and z directions
- Drag Model including parameters for the selected drag model

Other advanced options that can be selected include:

- Momentum formulation (Model a, Model B, Jackson, or Ishii)
- Sub-grid model (only available with TFM, Wen-Yu drag model, etc…)
- Sub-gird filter size
- Sub-grid wall correction

Geometry

Mesh

Regions

Fluid

The fluid pane is used to define the physical properties of the fluid phase. This includes adding fluid phase species which can either be imported from the BURCAT thermodynamic database or created by the user. The GUI will generate the correct data format to use thermodynamic data in MFiX.

Solids

The solids pane is used to define the physical properties of each solid(s) phase. Names can be given to each solid. This name will be used throughout the GUI to reference that solid phase (i.e. when defining initial conditions, boundary conditions, etc.). The solid phase model (TFM, DEM, or PIC) can also be specified here, however the selectable solid phase model depends on the solver selected on the model pane.

The selected solver will also enable/disable the TFM, DEM, and PIC sub-panes where various options for those solids models can be accessed.

Initial Conditions (ICs)

The initial conditions pane is used to define initial conditions for a selected Region (defined on each Region pane) for each phase (defined on Fluid or Solids panes). Initial conditions are required to cover the entire domain and provide an initial value for certain quantities that the solver will use to start the simulation. Experienced CFD users realize that providing reasonable initial values for volume fraction, temperature, pressure, and velocity allow simulations to reach/meet convergence criteria more quickly.

To create a new initial condition, press the button which will bring up the Region Selection dialog. Select a region to associate with the new initial condition and press OK. Regions that are not volumes or are already used by another initial condition will not be selectable.

Once the region has been created, values can be edited. Sub-panes will be created dynamically based on model parameters as well as the number of solid phases.

Note: Initial condition regions may overlap. When an overlap occurs, the initial condition with the higher IC number will be used at that location.

Boundary Conditions (BCs)

The boundary conditions pane is used to define boundary conditions for a selected Region (defined on each Region pane) for each phase (defined on Fluid or Solids panes). This is where inflow, outflow, pressure, walls, and cyclic boundary conditions are created.

To create a new boundary condition, press the button which will bring up the Region Selection dialog. Next, select a boundary type from the combo box. Regions will be enabled/disabled based on the selected boundary type and whether or not the region is already used by a boundary condition. Boundary conditions must be surfaces, either planes or collections of facets (STL). Pressing OK will create the boundary condition.

The sub-panes are created dynamically for each boundary condition based on the boundary condition type as well as other model parameters including the number of solid species.

Note: Two boundary surfaces must not intersect.

Point Sources (PSs)

Point sources (PSs) are used in place of mass inlets where either the geometry and/or grid resolution prohibit traditional boundary condition specification. For example, a point source may be used to model an injector with dimensions smaller than the grid. Point sources may be defined within a single computational cell, along a plane, or as a volume of computational cells.

Point sources introduce mass directly into a computational cell unlike a boundary condition which specifies flow along a cell face. One consequence of this implementation is that point sources are subjected to convection/diffusion forces and may not travel parallel to the specified directional preference. Directional preference is specified with a velocity vector (i.e., PS_U_g, PS_V_g, etc.), however, it is not required.

To create a new point source, press the button which will bring up the Region Selection dialog. Select a region to associate with the new point source and press OK.

The sub-panes are created dynamically for each point source based on the model parameters including the number of solid species.

Internal Surfaces (ISs)

Internal surfaces allow the user to create a zero-thickness walls that are normal to one of the coordinate directions and coincide with one of the faces of the scalar control volume.

Two types of internal surfaces are available: Impermeable, which acts as a free-slip wall for both gas and solids phases, and Semi-Impermeable, which allows only the gas phase to pass through the internal surface.

To create a new internal surface, press the button which will bring up the Region Selection dialog. Select a region to associate with the new Internal Surface. Select the type of Internal surface and press OK. For Semi-Impermeable surfaces, the fluid permeability and Internal Resistance Coefficient must be provided as well.

Typically, the region associated with an Internal Surface will be a plane. To specify a large number of internal surfaces in a region, a 3D region may be selected. In this case, a prefix (X_, Y_, or Z_) is added to the Internal Surface type to indicate the direction of the internal surfaces; e.g., X_IMPERMEABLE specifies impermeable internal surfaces parallel to the X coordinate.

Internal surfaces act as free-slip walls in stress computations. This default condition cannot be changed.

Chemistry

Setting up a project with chemical reactions is a multi-step process that requires coding the reactions rates in `usr_rates.f` for TFM or `usr_rates_des.f` for DEM. The solver needs to be built to take the reaction rates into account. A brief overview is outlined below. The Silane pyrolysis tutorial is an good way for the user to familiarize him/herself with setting up chemical reactions in MFiX.

- Check the "Enable user-defined subroutine" check-box in the "Model" pane.
- Check the "Enable Species equations" check-box and define species in the "Fluid" and "Solids" panes.
- In the "Chemistry" pane, define chemical equations by pressing the button. Define the reaction equation name, and add reactants and products with the button. Select the phase and species from the dropdown list, and enter the stoichiometric coefficient. Repeat until all products and reactants are defined, and press OK to validate. Specific heats of reactions are computed automatically, but this setting can be manually overwritten if needed by checking the "Specify Heat of Reaction" check-box, entering the heat of reaction and assigning the gas and solids phases fractions.
- (Optional) turn on the stiff chemistry solver in the "Chemistry" pane by checking the "Enable stiff chemistry solver" check-box.
- Edit and save the `usr_rates.f` or `usr_rates_des.f` UDF, and any other UDF needed for the reaction rates calculation.
- Build the custom solver by pressing the button.

Numerics

The Numerics panes defines various numerical parameters, which are grouped in several sub panes:

- Residuals: Defines convergence criteria for each type of equation, as well as the maximum number of iterations and residual normalization options.
- Discretization: Defines temporal, spatial discretization schemes and relaxation factors for each equation.
- Linear Solver: Defines the Linear Equation solver, tolerance and maximum number of iterations for each equation.
- Preconditioner: Defines the preconditioner options for each equation.
- Advanced: Defines less common parameters, such as the maximum inlet velocity factor, drag and IA theory under relaxation factors and fourth order interpolation scheme.

Outputs

A variety of output formats can be written by the solver including restart files (`*.RES`), VTK files (`*.pvd`, `*.vtp`, `*.vtu`), and legacy Single Precision files (`*.SP?`). The VTK files can be read by and displayed in the GUI. Most of these files can be viewed with the post-processing programs ParaView and VisIt.

The Basic sub-pane is where the restart file write frequency as well as the VTK and SPx outputs can be enabled. Once these are enabled, the associated sub-pane will be enabled, allowing for specific control over these outputs.

The VTK output has the most flexibility. Selected particle data or cell data variables can be exported at a specific region and write frequency. To create a new VTK output, press the button which will bring up the Region Selection dialog. Next, select an output type (particle or cell), select a region, and press OK. The file pattern name, write frequency, and variables can be selected for the newly created VTK output.

The following table lists the files typically associated with an MFiX run. In addition, their associated file type, data content and readability by ParaView and VisIt is indicated.

Table 8.1: Output Format Table

Filename	File Type	Fluid Data	Particle Data	Paraview	Visit
`<RUN_NAME>.RES`	MFiX Binary	Yes	No	Yes	Yes
`<RUN_NAME>_###_DES.RES`	MFiX Binary	No	Yes	No	No
`<RUN_NAME>.pvd`	VTK	Yes	No	Yes	No
`<RUN_NAME>_###.vtu`	VTK	Yes	No	Yes	Yes
`<VTK_REGION>.pvd`	VTK	Yes	No	Yes	No
`<VTK_REGION>_###.vtu`	VTK	Yes	No	Yes	Yes
`<RUN_NAME>_DES.pvd`	VTK	No	Yes	Yes	No
`<RUN_NAME>_DES_###.vtp`	VTK	No	Yes	Yes	Yes
`<VTK_REGION>.pvd`	VTK	No	Yes	Yes	No
`<VTK_REGION>_###.vtp`	VTK	No	Yes	Yes	Yes

Monitors

Planned.

Run

The Run pane is used to define parameters related to how long the solver runs, time-step controls, as well as options to cleanly terminate a solver after a certain amount of time. These options are particularly useful when running in a queue environment because they allow the solver to cleanly exit before the job gets killed by the queuing system. This minimizes the chance of corrupting output files if the process is killed while writing a file.

8.1.4 Visualization window

The visualization window provides a collection of 3D views and 2D plots for visualizing the model setup and model outputs. New windows, or tabs, can be created by pressing the button. Once the tab has been added, the type of view can be selected. A Tab can be closed by pressing the button located in its upper right hand corner.

Model

The Model tab is always present and cannot be closed. This 3D view shows the setup of the project including the background mesh, geometry, and regions.

The scene can be manipulated with the mouse and the toolbar buttons defined in the following table:

Icon	Description
	Reset view, make all items visible
	Change to XY view
	Change to XZ view
	Change to YZ View
	Toggle between perspective and parallel projections
	Save an image of the current view
	Change the visibility and properties of actors

The visibility menu allows the user to manipulate objects in the scene by:

- changing the visibility with the button
- changing the representation (wire, solids, edges, and points)
- changing the color

- changing the transparency of the objects

Plot Tab(s)

The plot tab(s) can be used to graph live statistics from the solver while it is running. Values such as the time step (dt), number of iterations (nit), and the simulated time (time) can be plotted. The plot can be manipulated with the mouse, and a `right-click` menu provides access to options that can be changed. In addition, export options for saving an image of the plot or exporting the data to a text file are available in this menu.

Note: Plotting only available when solver is running

VTK Tab(s)

The VTK tab provides a way to quickly view results from the solver. For more complex visualization of solver data, ParaView or VisIt is recommended.

The tab automatically sets-up VTK pipelines for reading and displaying the content of *.vtu and *.vtp files.

Note: The directory is automatically searched every second for *.pvd files. If a *.pvd file is found, the GUI will read and show the new VTK file if the button is pressed.

Results display can be "played" as well as manipulated using the following toolbar:

Icon	Description
	Reset view, make all items visible
	Change to XY view
	Change to XZ view
	Change to YZ View
	Toggle between perspective and parallel projections
	Save an image of the current view
	Change the visibility and properties of actors
	Go to the first frame
	Go back one frame
	Play available frames, starting at the current frame
	Go to the next frame
	Go to the last frame
	Change the playback speed, or the amount of time in between frames

The visibility menu allows for the manipulation of how the objects in the scene represented including:

- visibility
- the data array used to color
- color bars
- transparency

Further options for the points can be adjusted by clicking the button next to the label including:

- Maximum number of particles to be displayed
- The mapper (sprites requires VTK version 7.0+)
- The glyph object

8.1.5 Terminal window

The terminal window displays the output of the solver job that would be displayed when running the solver on the command line. Error messages and warnings, from both the GUI and a running solver, are displayed and colored in red.

Informational messages from the GUI unrelated to the solver are colored in blue.

8.1.6 Mode bar

The Mode bar allows switching the GUI between various modes including:

- Modeler, used to setup a project
- Nodeworks, future feature to support creation, management, post processing, and optimization of projects.

A status bar is also present, showing the current status of the GUI or a running solver, including a progress bar showing the current progress of the solver and elapsed time.

8.1.7 Running Interactive Solver Job in Queue

MFiX solver jobs can also be submitted to a queue through the *Run Dialog*. This functionality must be added to the GUI by the user by creating a *Queue Template*. This template allows users to customize the functionality of the GUI for their specific system.

Currently only a queue template that supports queue submissions on Joule (NETL's supercomputer) is included. Joule uses gridengine as the queueing system.

If you do not want to use the GUI or interactive features with your batch job, you can build a *Batch Solver* and run as in MFiX 2016 and earlier.

Queue Templates

Queue templates are included with the MFiX source code and can be found in the `MFIX_HOME\queue_templates` directory. Queue templates are files that contain two sections. The first section is the configuration section that tells the GUI what widgets to display as well as various options for those widgets. The second section is the actual script that is executed.

Variables can be used throughout the template, including with the widgets, and are reference by `${my_variable}`. Each template include some built in variables including:

Variable	Description
SCRIPT	the path of this script, in the run directory
PROJECT_NAME	the name of the project
JOB_ID	the job id extracted from the submit command
COMMAND	the command that executes mfix

The configuration section starts with `## CONFIG` and ends with `## END CONFIG`. This section uses the Microsoft Windows INI file format. Sections are defined with a `[section]` header, followed by a collection of `key=value` or `key: value` entries for defining parameters. For example:

```
## CONFIG
[My Section]
foodir: %(dir)s/whatever
dir=frob
```

(continues on next page)

(continued from previous page)

```
long: this value continues
   in the next line
## END CONFIG
```

The configuration section has a special section called `[options]` where the following options can be specified:

Key	Description
name	name of the template, this is displayed in the template drop-down box in the GUI
job_id_regex	regular expression to extract the job id from the output of the submit command
status_regex	regular expression to extract the job status from the status command
submit	the submission command
delete	the job cancel or delete command
status	the job status command

An example of values that work with Grid Engine:

```
[options]
name: Joule
job_id_regex: (\d+)
status_regex: ([rqw])
submit: qsub ${SCRIPT}
delete: qdel ${JOB_ID}
status: qstat -j ${JOB_ID}
```

To define a new variable and widget to edit that variable in the GUI, create a new section:

```
[my_value]
```

The widget and options for that widget can then be selected by specifying various parameters including:

Parameter	Description	Values
widget	the widget to be used	`lineedit`, `combobox`, `checkbox`, `spinbox`, `doublespinbox`, `listwidget`
label	text to be placed beside the widget	`any string`
value	default value	a value such as `1`, `10.3`, `True`, `some text`
items	list of items for the combobox or listwidget	items delimited by | character
help	text to be displayed in the tooltip for that widget	`this widget does this`
true	value to be returned if a checkbox is checked	a value such as `1`, `10.3`, `True`, `some text`
false	value to be returned if a checkbox is un-checked	a value such as `1`, `10.3`, `True`, `some text`

An example defining a combo box:

```
[my_email]
widget: combobox
label: email
value: you@mail.com
items: you@mail.com|
```

(continues on next page)

(continued from previous page)

```
       me@mail.net|
       hi@mail.org
help: email to send notifications to
```

The value of this widget can now be referenced throughout the template with `${my_email}`

The rest of the configuration file, outside of the `## CONFIG` to `## END CONFIG` section is the script that needs to be executed to submit a job to your specific queue system. For example, with Grid Engine on Joule, the following script specifies the job name, job type, cores, and queue. In addition, it loads the required modules needed for the MFiX run, and finally runs mfix with `${COMMAND}`.

```
## Change into the current working directory
#$ -cwd
##
## The name for the job. It will be displayed this way on qstat
#$ -N ${JOB_NAME}
##
## Number of cores to request
#$ -pe ${JOB_TYPE} ${CORES}
##
## Queue Name
#$ -q ${QUEUE}
##

##Load Modules
module load ${MODULES}
##Run the job
${COMMAND}
```

8.2 User-Defined Functions

User-Defined Functions (UDFs) are source files in the same directory as the project file that define code that is called by specific locations in the main MFiX code. UDFs are used to customize the behavior of MFiX. One common usage is for creating user-specific output files. In many cases, creating a personalized UDF is preferable to using post_mfix.

Because MFiX is open source, users may modify any `*.f` or `*.inc` file under the model directory. To modify a file, first copy it from the model directory into the run directory. All modifications should be made to the file in the run directory. For example, list the contents of the `adiabaticFlame` test case located in the `legacy_tests` directory:

```
> cd tutorials/tfm/silane_pyrolysis_2d
> ls
SP2D.mfx    usr0.f      usr1.f      usr_mod.f   usr_rates.f
```

The `SP2D.mfx` file is the project file, and the `usr*.f` files are the UDFs used by this project. simulation. After running `build_mfixsolver`, a re-examination of the same folder reveals:

```
> build_mfixsolver
...long output...
> ls
Makefile    build       mfixsolver  usr.mod     usr0.f      usr1.d      usr1.o      ␣
↪usr_mod.f   usr_rates.d usr_rates.o
SP2D.mfx    lib         species.inc usr0.d      usr0.o      usr1.f      usr_mod.d   ␣
↪usr_mod.o   usr_rates.f
```

Each UDF has a corresponding dependency (`.d`) file and object (`.o`) file.

If a project has a `usr_rates.f` UDF, an additional `species.inc` file is generated from the project file during the reaction preprocessing step of the build.

The `.d` and `.o` files are intermediate dependency and object files that are created and linked with the executable, mfix.

The following is a list of MFIX files that are usually modified to include user defined scalars, user defined drag models, and chemical reactions, physical properties, and source terms for the governing equations:

File Name:	Usage Description
`scalar_prop.f`	Properties and source terms in scalar transport equations
`usr_drag.f`	User-defined drag function
`usr_rates.f`	Chemical reaction rates
`usr_rates_des.f`	DES chemical reaction rates
`usr_properties.f`	Physical properties (density, specific heat, etc.)
`usr_sources.f`	Governing equation source terms

The following routines are used for writing user-defined output:

File Name:	Usage Description
`write_usr0.f`	**Called once at the start of a run. This file is** used for opening user-defined output files.
`write_usr1.f`	Called at intervals defined by USR_DT

To activate the calls to the following routines, make sure the "Enable user-defined subroutines" checkbox is checked in the Model pane.

File Name:	Usage Description
`usr0.f`	Subroutine that is called once every run, just before the time-loop begins.
`usr1.f`	Subroutine that is called once every timestep.
`usr2.f`	Subroutine that is called once every iteration.
`usr3.f`	Subroutine that is called once every run, after the time-loop ends.
`usrnlst.inc`	List of user-defined keywords. These may be used to enter data through the input data file mfix.dat.
`usr_init_namelist.f`	Initialize user-defined keywords.
`usr_mod.f`	User-defined module. Include `USE usr` to use userdefined variables in this module. If allocatable arrays are defined in this module, allocate them in usr0.f. [1]
`usr0_des.f`	Subroutine called before entering the DES time loop. [1]
`usr1_des.f`	Subroutine called every DEM timestep after calculating DES source terms but before source terms are applied to the particles.
`usr2_des.f`	Subroutine called every DES timestep after source terms are applied to the particles.
`usr3_des.f`	Subroutine that is called after completing the DES time loop.

The following figures illustrate the local call structure for UDFs, for typical TFM/PIC and DEM runs:

DES User-defined subroutines call structure:

Note that the `USR_` keywords do not have any explicit behavior. They are only basic input mechanisms to interact with user-defined functions. For example, specifying `USR_X_w` does not result in the associated `I` index, `USR_I_W` being calculated. It is upon the user to ensure that all user-hooks are fully specified.

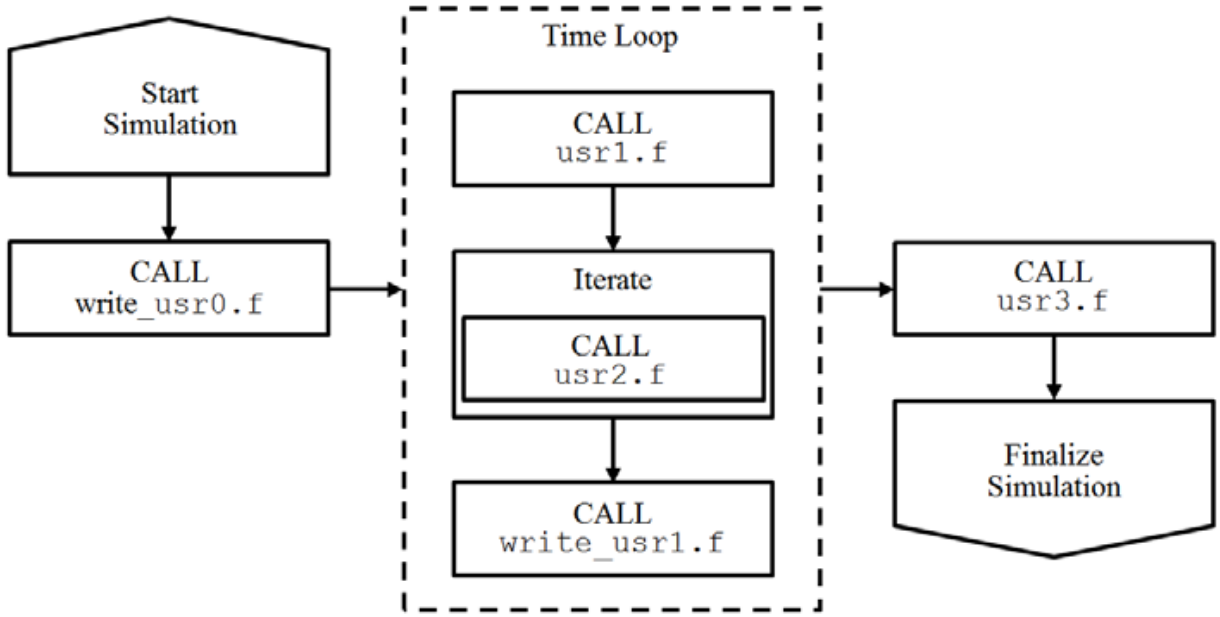


Fig. 8.1: Call Graph for UDF Subroutines

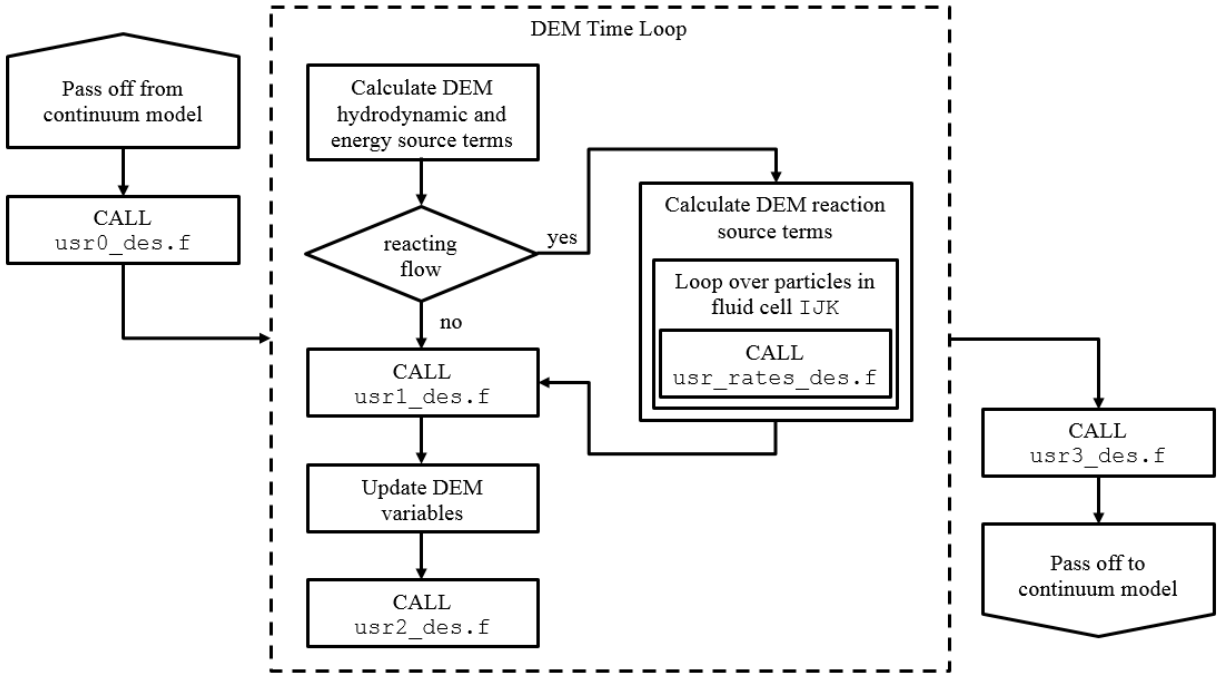


Fig. 8.2: Call Graph for UDF DES Subroutines

8.3 Keywords

8.3.1 Run Control

RUN_NAME

Data Type: CHARACTER

required

Name used to create output files. The name should generate legal file names after appending extensions. Ex: Given the input, RUN_NAME = "bub01", MFIX will generate the output files: BUB01.LOG, BUB01.OUT, BUB01.RES, etc.

DESCRIPTION

Data Type: CHARACTER

Problem description. Limited to 60 characters.

PROJECT_VERSION

Data Type: CHARACTER

Project version. Limited to 80 characters.

UNITS

Data Type: CHARACTER

required

Simulation input/output units.

Table 8.2: Valid Values

Name	Default?	Description
`cgs`		All input and output in CGS units (g, cm, s, cal).
`SI`		All input and output in SI units (kg, m, s, J).

RUN_TYPE

Data Type: CHARACTER

required

Type of run.

Table 8.3: Valid Values

Name	Default?	Description
`new`		A new run. There should be no .RES, .SPx, .OUT, or .LOG files in the run directory.
`RESTART_1`		Traditional restart. The run continues from the last time the .RES file was updated and new data is added to the SPx files.
`RESTART_2`		Start a new run with initial conditions from a .RES file created from another run. No other data files (SPx) should be in the run directory.

TIME

Data Type: DOUBLE PRECISION

Simulation start time. This is typically zero.

TSTOP

Data Type: DOUBLE PRECISION

Simulation stop time.

OPTFLAG1

Data Type: DOUBLE PRECISION

Use nrel-cu optimization routines?.

DT

Data Type: DOUBLE PRECISION

Initial time step size. If left undefined, a steady-state calculation is performed.

DT_MAX

Data Type: DOUBLE PRECISION

Maximum time step size.

DT_MIN

Data Type: DOUBLE PRECISION

Minimum time step size.

DT_FAC

Data Type: DOUBLE PRECISION

Factor for adjusting time step.

- The value must be less than or equal to 1.0.
- A value of 1.0 keeps the time step constant which may help overcome initial non-convergence.

PERSISTENT_MODE

Data Type: LOGICAL

Force a forward time-step if the maximum number of iterations, MAX_NIT, is reached. The forward time-step is only forced after reaching the minimum time-step, DT_MIN, for adjustable time-step simulations (DT_FAC /= 1). This option should be used with caution as unconverged time-steps may lead to poor simulation results and/or additional convergence issues.

Table 8.4: Valid Values

Name	Default?	Description
`.TRUE.`		Force forward time-step when DT=DT_MIN and the maximum number of iterations are reached.
`.FALSE.`	◉	Abort run when DT < DT_MIN.

AUTO_RESTART

Data Type: LOGICAL

Flag to restart the code when DT < DT_MIN.

MOMENTUM_X_EQ(PHASE)

Data Type: LOGICAL

- $0 \leq Phase \leq 10$

Flag to enable/disable solving the X-momentum equations.

Table 8.5: Valid Values

Name	Default?	Description
`.TRUE.`	◉	Solve X-momentum equations.
`.FALSE.`		The X velocity initial conditions persist throughout the simulation.

MOMENTUM_Y_EQ(PHASE)

Data Type: LOGICAL

- $0 \leq Phase \leq 10$

Flag to enable/disable solving the Y-momentum equations.

Table 8.6: Valid Values

Name	Default?	Description
`.TRUE.`	◉	Solve Y-momentum equations.
`.FALSE.`		The Y velocity initial conditions persist throughout the simulation.

MOMENTUM_Z_EQ(PHASE)

Data Type: LOGICAL

- $0 \leq Phase \leq 10$

Flag to enable/disable solving the Z-momentum equations.

Table 8.7: Valid Values

Name	Default?	Description
`.TRUE.`	◉	Solve Z-momentum equations.
`.FALSE.`		The Z velocity initial conditions persist throughout the simulation.

JACKSON

Data Type: LOGICAL

Flag to enable Jackson form of momentum equations. See Anderson and Jackson, (1967), IECF, 6(4), p.527.

Table 8.8: Valid Values

Name	Default?	Description
`.TRUE.`		Solve Jackson form of momentum equations.
`.FALSE.`	◉	Default form.

ISHII

Data Type: LOGICAL

Flag to enable Ishii form of momentum equations. See Ishii, (1975), Thermo-fluid dynamic theory of two-phase flow.

Table 8.9: Valid Values

Name	Default?	Description
`.TRUE.`		Solve Ishii form of momentum equations.
`.FALSE.`	◉	Default form.

ENERGY_EQ

Data Type: LOGICAL

Solve energy equations.

Table 8.10: Valid Values

Name	Default?	Description
`.TRUE.`	◉	Solve energy equations.
`.FALSE.`		Do not solve energy equations.

SPECIES_EQ(PHASE)

Data Type: LOGICAL

- $0 \leq Phase \leq 10$

Solve species transport equations.

Table 8.11: Valid Values

Name	Default?	Description
`.TRUE.`	◉	Solve species equations.
`.FALSE.`		Do not solve species equations.

TURBULENCE_MODEL

Data Type: CHARACTER

Gas phase turbulence model. ["NONE"]

For K_EPSILON (K-epsilon turbulence model for single-phase flow):

- Numerical parameters (like under-relaxation) are the same as the ones for SCALAR (index = 9).
- All walls must be defined (NSW, FSW or PSW) in order to use standard wall functions. If a user does not specify a wall type, the simulation will not contain the typical turbulent profile in wall-bounded flows.

Table 8.12: Valid Values

Name	Default?	Description
`NONE`	◉	No turbulence model.
`MIXING_LENGTH`		Turbulent length scale must be specified for the full domain using keyword IC_L_SCALE.
`K_EPSILON`		K-epsilon turbulence model (for single-phase flow) using standard wall functions.

MU_GMAX

Data Type: DOUBLE PRECISION

Maximum value of the turbulent viscosity of the fluid, which must be defined if any turbulence model is used. A value MU_GMAX =1.E+03 is recommended. (see calc_mu_g.f)

DRAG_TYPE

Data Type: CHARACTER

Available gas-solids drag models. Note: The extension _PCF following the specified drag model indicates that the polydisperse correction factor is available. This option is available for TFM solids only. For PCF details see:

- Van der Hoef MA, Beetstra R, Kuipers JAM. (2005) Journal of Fluid Mechanics.528:233-254.
- Beetstra, R., van der Hoef, M. A., Kuipers, J.A.M. (2007). AIChE Journal, 53:489-501.
- Erratum (2007), AIChE Journal, Volume 53:3020

Table 8.13: Valid Values

Name	Default?	Description
SYAM_OBRIEN	◉	Syamlal M, OBrien TJ (1988). International Journal of Multiphase Flow 14:473-481. Two additional parameters may be specified: DRAG_C1, DRAG_D1
GIDASPOW		Ding J, Gidaspow D (1990). AIChE Journal 36:523-538
GIDASPOW_BLEND		Lathouwers D, Bellan J (2000). Proceedings of the 2000 U.S. DOE Hydrogen Program Review NREL/CP-570-28890.
WEN_YU		Wen CY, Yu YH (1966). Chemical Engineering Progress Symposium Series 62:100-111.
KOCH_HILL		Hill RJ, Koch DL, Ladd JC (2001). Journal of Fluid Mechanics, 448: 213-241. and 448:243-278.
BVK		Beetstra, van der Hoef, Kuipers (2007). Chemical Engineering Science 62:246-255
TPKKV		Tang, Peters, Kuipers, Kreibitzsch, & van der Hoef. AIChE J., 61(2), pp.688-698 (2015).
HYS		Yin, X, Sundaresan, S. (2009). AIChE Journal 55:1352-1368 This model has a lubrication cutoff distance, LAM_HYS, that can be specified.
USER_DRAG		Invoke user-defined drag law. (usr_drag.f)
GIDASPOW_PCF		(TFM only). see GIDASPOW
GIDASPOW_BLEND_PCF		(TFM only). see GIDASPOW_BLEND
WEN_YU_PCF		(TFM only). see WEN_YU
KOCH_HILL_PCF		(TFM only). see KOCH_HILL

DRAG_C1

Data Type: DOUBLE PRECISION

Quantity for calibrating Syamlal-O'Brien drag correlation using Umf data. This is determined using the Umf spreadsheet.

DRAG_D1

Data Type: DOUBLE PRECISION

Quantity for calibrating Syamlal-O'Brien drag correlation using Umf data. This is determined using the Umf spreadsheet.

LAM_HYS

Data Type: DOUBLE PRECISION

The lubrication cutoff distance for HYS drag model. In practice this number should be on the order of the mean free path of the gas for smooth particles, or the RMS roughness of a particle if they are rough (if particle roughness is larger than the mean free path).

SUBGRID_TYPE

Data Type: CHARACTER

Applies to Solids Model(s): **TFM**

Subgrid models.

Table 8.14: Valid Values

Name	Default?	Description
`Igci`		Igci, Y., Pannala, S., Benyahia, S., and Sundaresan S. (2012). Industrial & Engineering Chemistry Research, 2012, 51(4):2094-2103
`Milioli`		Milioli, C.C., Milioli, F. E., Holloway, W., Agrawal, K. and Sundaresan, S. (2013). AIChE Journal, 59:3265-3275.

FILTER_SIZE_RATIO

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Ratio of filter size to computational cell size.

SUBGRID_WALL

Data Type: LOGICAL

Applies to Solids Model(s): **TFM**

Flag for subgrid wall correction.

Table 8.15: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Do not include wall correction.
`.TRUE.`		Include subgrid wall correction.

MODEL_B

Data Type: LOGICAL

Shared gas-pressure formulation. See Syamlal, M. and Pannala, S. "Multiphase continuum formulation for gas-solids reacting flows," chapter in Computational Gas-Solids Flows and Reacting Systems: Theory, Methods and Practice, S. Pannala, M. Syamlal and T.J. O'Brien (editors), IGI Global, Hershey, PA, 2011.

Table 8.16: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Use Model A. See J.X. Bouillard and R.W. Lyczkowski (1991), Powder Tech, 68:31-51.
`.TRUE.`		Use Model B. See Bouillard, J.X., Lyczkowski, R.W., Folga, S., Gidaspow, D., Berry, G.F. (1989). Canadian Journal of Chemical Engineering 67:218-229.

NSCALAR

Data Type: INTEGER

The number of user-defined scalar transport equations to solve.

PHASE4SCALAR(SCALAR EQUATION)

Data Type: INTEGER

- $1 \leq ScalarEquation \leq 100$

The phase convecting the indexed scalar transport equation.

PPO

Data Type: LOGICAL

Flag to run the pre-processing (mesh generation) only and exit.

8.3.2 Physical Parameters

P_REF

Data Type: DOUBLE PRECISION

Reference pressure. [0.0]

P_SCALE

Data Type: DOUBLE PRECISION

Scale factor for pressure. [1.0]

GRAVITY

Data Type: DOUBLE PRECISION

Gravitational acceleration. [9.807 m/s^2 in SI]

GRAVITY_X

Data Type: DOUBLE PRECISION

X-component of gravitational acceleration vector.

GRAVITY_Y

Data Type: DOUBLE PRECISION

Y-component of gravitational acceleration vector.

GRAVITY_Z

Data Type: DOUBLE PRECISION

Z-component of gravitational acceleration vector.

8.3.3 Numerical Parameters

This section contains keywords for defining numerical parameters. Keywords related to solving the governing equations (e.g., `LEQ_IT`, `DISCRETIZE`, `UR_FAC`, etc.) are dimensioned for the ten types of equations:

Index	Equation Type
1	Gas Pressure
2	Solids Volume Fraction
3	Gas and Solids U Momentum Equation
4	Gas and Solids V Momentum Equation
5	Gas and Solids W Momentum Equation
6	Gas and Solids Energy Equations (Temperature)
7	Gas and Solids Species Mass Fractions
8	Granular Temperature
9	User-Defined Scalar and K-Epsilon Equation
10	DES Diffusion Equation

For example, `LEQ_IT(3) = 10`, specifies to use 10 iterations within the linear equation solver for the U Momentum Equations (of type 3).

MAX_NIT

Data Type: INTEGER

Maximum number of iterations [500].

NORM_G

Data Type: DOUBLE PRECISION

Factor to normalize the gas continuity equation residual. The residual from the first iteration is used if NORM_G is left undefined. NORM_G=0 invokes a normalization method based on the dominant term in the continuity equation. This setting may speed up calculations, especially near a steady state and for incompressible fluids. But, the number of iterations for the gas phase pressure, LEQ_IT(1), should be increased to ensure mass balance.

NORM_S

Data Type: DOUBLE PRECISION

Factor to normalize the solids continuity equation residual. The residual from the first iteration is used if NORM_S is left undefined. NORM_S = 0 invokes a normalization method based on the dominant term in the continuity equation. This setting may speed up calculations, especially near a steady state and incompressible fluids. But, the number of iterations for the solids volume fraction, LEQ_IT(2), should be increased to ensure mass balance.

TOL_RESID

Data Type: DOUBLE PRECISION

Maximum residual at convergence (Continuity + Momentum) [1.0d-3].

TOL_RESID_T

Data Type: DOUBLE PRECISION

Maximum residual at convergence (Energy) [1.0d-4].

TOL_RESID_X

Data Type: DOUBLE PRECISION

Maximum residual at convergence (Species Balance) [1.0d-4].

TOL_RESID_TH

Data Type: DOUBLE PRECISION

Maximum residual at convergence (Granular Energy) [1.0d-4].

TOL_RESID_SCALAR

Data Type: DOUBLE PRECISION

Maximum residual at convergence (Scalar Equations) [1.0d-4].

TOL_RESID_K_EPSILON

Data Type: DOUBLE PRECISION

Maximum residual at convergence (K_Epsilon Model) [1.0d-4].

TOL_DIVERGE

Data Type: DOUBLE PRECISION

Minimum residual for declaring divergence [1.0d+4]. This parameter is useful for incompressible fluid simulations because velocity residuals can take large values for the second iteration (e.g., 1e+8) before dropping down to smaller values for the third iteration.

DETECT_STALL

Data Type: LOGICAL

Reduce the time step if the residuals stop decreasing. Disabling this feature may help overcome initial non-convergence.

Table 8.17: Valid Values

Name	Default?	Description
`.FALSE.`		Continue iterating if residuals stall.
`.TRUE.`	◉	Reduce time step if residuals stall.

LEQ_METHOD(EQUATION ID NUMBER)

Data Type: INTEGER

- $1 \leq EquationIDNumber \leq 10$

LEQ Solver selection. BiCGSTAB is the default method for all equation types.

Table 8.18: Valid Values

Name	Default?	Description
`1`		SOR - Successive over-relaxation
`2`	◉	BiCGSTAB - Biconjugate gradient stabilized.
`3`		GMRES - Generalized minimal residual method
`5`		CG - Conjugate gradient

LEQ_TOL(EQUATION ID NUMBER)

Data Type: DOUBLE PRECISION

- $1 \leq EquationIDNumber \leq 10$

Linear Equation tolerance [1.0d-4].

LEQ_IT(EQUATION ID NUMBER)

Data Type: INTEGER

- $1 \leq EquationIDNumber \leq 10$

Number of iterations in the linear equation solver.

Default value: - 20 iterations for equation types 1-2 - 5 iterations for equation types 3-5,10 - 15 iterations for equation types 6-9

LEQ_SWEEP(EQUATION ID NUMBER)

Data Type: CHARACTER

- $1 \leq EquationIDNumber \leq 10$

Linear equation sweep direction. This applies when using GMRES or when using the LINE preconditioner with BiCGSTAB or CG methods. 'RSRS' is the default for all equation types.

Table 8.19: Valid Values

Name	Default?	Description
RSRS	◉	(Red/Black Sweep, Send Receive) repeated twice
ISIS		(Sweep in I, Send Receive) repeated twice
JSJS		(Sweep in J, Send Receive) repeated twice
KSKS		(Sweep in K, Send Receive) repeated twice
ASAS		(All Sweep, Send Receive) repeated twice

LEQ_PC(EQUATION ID NUMBER)

Data Type: CHARACTER

- $1 \leq EquationIDNumber \leq 10$

Linear preconditioner used by the BiCGSTAB and CG LEQ solvers. 'LINE' is the default for all equation types.

Table 8.20: Valid Values

Name	Default?	Description
NONE		No preconditioner
LINE	◉	Line relaxation
DIAG		Diagonal Scaling

UR_FAC(EQUATION ID NUMBER)

Data Type: DOUBLE PRECISION

- $1 \leq EquationIDNumber \leq 10$

Under relaxation factors. Default value: - 0.8 for equation types 1,9 - 0.5 for equation types 2,3,4,5,8 - 1.0 for equation types 6,7,10

UR_F_GS

Data Type: DOUBLE PRECISION

The implicitness calculation of the gas-solids drag coefficient may be underrelaxed by changing ur_f_gs, which takes values between 0 to 1.

- 0 updates F_GS every time step
- 1 updates F_GS every iteration

UR_KTH_SML

Data Type: DOUBLE PRECISION

Under relaxation factor for conductivity coefficient associated with other solids phases for IA Theory [1.0].

DISCRETIZE(EQUATION ID NUMBER)

Data Type: INTEGER

- $1 \leq EquationIDNumber \leq 10$

Discretization scheme used to solve equations.

Table 8.21: Valid Values

Name	Default?	Description
0	◉	First-order upwinding.
1		First-order upwinding (using down-wind factors).
3		Smart.
2		Superbee (recommended method).
5		QUICKEST (does not work).
4		ULTRA-QUICK.
7		van Leer.
6		MUSCL.
8		minmod.
9		Central (often unstable; useful for testing).

DEF_COR

Data Type: LOGICAL

Use deferred correction method for implementing higher order discretization.

Table 8.22: Valid Values

Name	Default?	Description
.FALSE.	◉	Use down-wind factor method (default).
.TRUE.		Use deferred correction method.

CHI_SCHEME

Data Type: LOGICAL

This scheme guarantees that the set of differenced species mass balance equations maintain the property that the sum of species mass fractions is one. This property is not guaranteed when a flux limiter is used with higher order spatial discretization schemes. Note: The chi-scheme is implemented for SMART and MUSCL discretization schemes. Darwish, M.S., Moukalled, F. (2003). Computer Methods in Applied Mech. Eng., 192(13):1711-1730.

Table 8.23: Valid Values

Name	Default?	Description
.FALSE.	◉	Do not use the chi-scheme.
.TRUE.		Use the chi-scheme correction.

CN_ON

Data Type: LOGICAL

Temporal discretization scheme.

Table 8.24: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Implicit Euler based temporal discretization scheme employed (first-order accurate in time).
`.TRUE.`		Two-step implicit Runge-Kutta method based temporal discretization scheme employed. This method should be second- order accurate in time excluding pressure terms and restart time step which are first-order accurate. However, as of 2015, testing shows that second-order accuracy is not observed.

MAX_INLET_VEL_FAC

Data Type: DOUBLE PRECISION

The code declares divergence if the velocity anywhere in the domain exceeds a maximum value. This maximum value is automatically determined from the boundary values. The user may scale the maximum value by adjusting this scale factor [1.0d0].

DO_TRANSPOSE

Data Type: LOGICAL

Solve transpose of linear system. (BICGSTAB ONLY).

ICHECK_BICGS

Data Type: INTEGER

Frequency to check for convergence. (BICGSTAB ONLY)

OPT_PARALLEL

Data Type: LOGICAL

Sets optimal LEQ flags for parallel runs.

USE_DOLOOP

Data Type: LOGICAL

Use do-loop assignment over direct vector assignment.

IS_SERIAL

Data Type: LOGICAL

Calculate dot-products more efficiently (Serial runs only.)

DIL_EP_S

Data Type: DOUBLE PRECISION

Dilute flow threshold. When the volume fraction of a certain phase in a cell is smaller than this value the momentum equation for that phase is not solved in the cell. It is strongly encouraged to keep the default value of 1E-4.

ZERO_EP_S

Data Type: DOUBLE PRECISION

Minimum value of solids volume fraction tracked It is strongly encouraged to keep the default value of 1E-8.

ZERO_X_GS

Data Type: DOUBLE PRECISION

Small value for species mass fraction for disregarding residual calculation It is strongly encouraged to keep the default value of 1E-7.

8.3.4 Geometry and Discretization

COORDINATES

Data Type: CHARACTER

Coordinate system used in the simulation.

Table 8.25: Valid Values

Name	Default?	Description
`Cartesian`		Cartesian coordinates.
`cylindrical`		Cylindrical coordinates.

IMAX

Data Type: INTEGER

Number of cells in the x (r) direction.

DX(CELL)

Data Type: DOUBLE PRECISION

- $0 \leq Cell \leq 5000$

Cell sizes in the x (r) direction. Enter values from DX(0) to DX(IMAX-1).

- Use uniform mesh size with higher-order discretization methods.
- DX should be kept uniform in cylindrical coordinates for strict momentum conservation.

XMIN

Data Type: DOUBLE PRECISION

The inner radius in the simulation of an annular cylindrical region.

XLENGTH

Data Type: DOUBLE PRECISION

Simulation domain length in the x (r) direction.

X_MIN

Data Type: DOUBLE PRECISION

Simulation domain lower bound in the x-direction.

X_MAX

Data Type: DOUBLE PRECISION

Simulation domain upper bound in the x-direction.

JMAX

Data Type: INTEGER

Number of cells in the y-direction.

DY(CELL)

Data Type: DOUBLE PRECISION

- $0 \leq Cell \leq 5000$

Cell sizes in the y-direction. Enter values from DY(0) to DY(JMAX-1). Use uniform mesh size with second-order discretization methods.

YLENGTH

Data Type: DOUBLE PRECISION

Simulation domain length in the y-direction.

Y_MIN

Data Type: DOUBLE PRECISION

Simulation domain lower bound in the y-direction.

Y_MAX

Data Type: DOUBLE PRECISION

Simulation domain upper bound in the y-direction.

NO_K

Data Type: LOGICAL

Flag to disable the third dimension (i.e., 2D simulation).

- Z axis in Cartesian coordinate system
- Theta in Cylindrical coordinate system

Table 8.26: Valid Values

Name	Default?	Description
`.FALSE.`	◉	3D simulation.
`.TRUE.`		2D simulation.

KMAX

Data Type: INTEGER

Number of cells in the z-direction.

DZ(CELL)

Data Type: DOUBLE PRECISION

- $0 \leq Cell \leq 5000$

Cell sizes in the z (theta) direction. Enter values from DZ(0) to DZ(KMAX-1). Use uniform mesh size with second-order discretization methods.

Z_MIN

Data Type: DOUBLE PRECISION

Simulation domain lower bound in the z-direction.

Z_MAX

Data Type: DOUBLE PRECISION

Simulation domain upper bound in the z-direction.

ZLENGTH

Data Type: DOUBLE PRECISION

Simulation domain length in the z (theta) direction.

CYCLIC_X

Data Type: LOGICAL

Flag for making the x-direction cyclic without pressure drop. No other boundary conditions for the x-direction should be specified.

Table 8.27: Valid Values

Name	Default?	Description
`.FALSE.`	◉	No cyclic condition at x-boundary.
`.TRUE.`		Cyclic condition at x-boundary.

CYCLIC_X_PD

Data Type: LOGICAL

Flag for making the x-direction cyclic with pressure drop. If the keyword FLUX_G is given a value, this becomes a cyclic boundary condition with specified mass flux. No other boundary conditions for the x-direction should be specified.

Table 8.28: Valid Values

Name	Default?	Description
`.FALSE.`	◉	No cyclic condition at x-boundary.
`.TRUE.`		Cyclic condition with pressure drop at x-boundary.

DELP_X

Data Type: DOUBLE PRECISION

Fluid pressure drop across XLENGTH when a cyclic boundary condition with pressure drop is imposed in the x-direction.

CYCLIC_Y

Data Type: LOGICAL

Flag for making the y-direction cyclic without pressure drop. No other boundary conditions for the y-direction should be specified.

Table 8.29: Valid Values

Name	Default?	Description
`.FALSE.`	◉	No cyclic condition at y-boundary.
`.TRUE.`		Cyclic condition at x-boundary.

CYCLIC_Y_PD

Data Type: LOGICAL

Flag for making the y-direction cyclic with pressure drop. If the keyword FLUX_G is given a value this becomes a cyclic boundary condition with specified mass flux. No other boundary conditions for the y-direction should be specified.

Table 8.30: Valid Values

Name	Default?	Description
`.FALSE.`	◉	No cyclic condition at y-boundary.
`.TRUE.`		Cyclic condition with pressure drop at y-boundary.

DELP_Y

Data Type: DOUBLE PRECISION

Fluid pressure drop across YLENGTH when a cyclic boundary condition with pressure drop is imposed in the y-direction.

CYCLIC_Z

Data Type: LOGICAL

Flag for making the z-direction cyclic without pressure drop. No other boundary conditions for the z-direction should be specified.

Table 8.31: Valid Values

Name	Default?	Description
`.FALSE.`	◉	No cyclic condition at z-boundary.
`.TRUE.`		Cyclic condition at z-boundary.

CYCLIC_Z_PD

Data Type: LOGICAL

Flag for making the z-direction cyclic with pressure drop. If the keyword FLUX_G is given a value this becomes a cyclic boundary condition with specified mass flux. No other boundary conditions for the z-direction should be specified.

Table 8.32: Valid Values

Name	Default?	Description
`.FALSE.`	◉	No cyclic condition at z-boundary.
`.TRUE.`		Cyclic condition with pressure drop at z-boundary.

DELP_Z

Data Type: DOUBLE PRECISION

Fluid pressure drop across ZLENGTH when a cyclic boundary condition with pressure drop is imposed in the z-direction.

SHEAR

Data Type: LOGICAL

Imposes a mean shear on the flow field as a linear function of the x coordinate. This feature should only be used when CYCLIC_X is .TRUE. and the keyword V_SH is set.

V_SH

Data Type: DOUBLE PRECISION

Specifies the mean y velocity component at the eastern boundary of the domain (V_SH), and the mean Y velocity (-V_SH) at the western boundary of the domain.

FLUX_G

Data Type: DOUBLE PRECISION

If a value is specified, then the domain-averaged gas flux is held constant at that value in simulations over a periodic domain. A pair of boundaries specified as periodic with fixed pressure drop is then treated as periodic with fixed mass flux. Even for this case a pressure drop must also be specified, which is used as the initial guess in the simulations.

CYLINDRICAL_2D

Data Type: LOGICAL

Applies the 2.5D model for cylindrical column by combining 2D assumption and axi-symmetric assumption. Li et al. (2015). A 2.5D computational method to simulate cylindrical fluidized beds, Chemical Engineering Science, 123:236-246.

I_CYL_NUM

Data Type: INTEGER

Parameter to control the plate half width and the wedge radius in the 2.5D cylindrical model. This value should be less than half the grid cells in the radial direction (IMAX/2). [1]

I_CYL_TRANSITION

Data Type: INTEGER

Parameter to smooth the transition from cylindrical to 2D in the 2.5D cylindrical model. [2]

Table 8.33: Valid Values

Name	Default?	Description
2	◉	Two cell smoothing transition.
1		One cell smoothing transition.
0		No smoothing.

CPX(CTRL)

Data Type: DOUBLE PRECISION

- $0 \leq CTRL \leq MAX_CP$

Location of control points in x-direction.

NCX(CTRL)

Data Type: INTEGER

- $1 \leq CTRL \leq MAX_CP$

Number of cells within a segment (x-direction).

ERX(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Expansion ratio (last DX/first DX) in a segment (x-direction).

FIRST_DX(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Value of first DX in a segment (x-direction). A negative value will copy DX from previous segment (if available).

LAST_DX(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Value of last DX in a segment (x-direction). A negative value will copy DX from next segment (if available).

CPY(CTRL)

Data Type: DOUBLE PRECISION

- $0 \leq CTRL \leq MAX_CP$

Location of control points in y-direction.

NCY(CTRL)

Data Type: INTEGER

- $1 \leq CTRL \leq MAX_CP$

Number of cells within a segment (y-direction).

ERY(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Expansion ratio (last DY/first DY) in a segment (y-direction).

FIRST_DY(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Value of first DY in a segment (y-direction). A negative value will copy DY from previous segment (if available).

LAST_DY(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Value of last DY in a segment (y-direction). A negative value will copy DY from next segment (if available).

CPZ(CTRL)

Data Type: DOUBLE PRECISION

- $0 \leq CTRL \leq MAX_CP$

Location of control points in z-direction.

NCZ(CTRL)

Data Type: INTEGER

- $1 \leq CTRL \leq MAX_CP$

Number of cells within a segment (z-direction).

ERZ(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Expansion ratio (last DZ/first DZ) in a segment (z-direction).

FIRST_DZ(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Value of first DZ in a segment (z-direction). A negative value will copy DZ from previous segment (if available).

LAST_DZ(CTRL)

Data Type: DOUBLE PRECISION

- $1 \leq CTRL \leq MAX_CP$

Value of last DZ in a segment (z-direction). A negative value will copy DZ from next segment (if available).

8.3.5 Two-Fluid Model

This section contains keywords relating to the Two-Fluid Model.

KT_TYPE

Data Type: CHARACTER

Applies to Solids Model(s): **TFM**

Solids phase stress model [Algebraic].

Table 8.34: Valid Values

Name	Default?	Description
ALGEBRAIC	◉	Granular energy algebraic formulation.
AHMADI		Cao and Ahmadi (1995). Int. J. Multiphase Flow 21(6), 1203.
GD_99		Garzo and Dufty (1999). Phys. Rev. E 59(5), 5895.
GHD		Garzo, Hrenya and Dufty (2007). Phys. Rev. E 76(3), 31304
GTSH		Garzo, Tenneti, Subramaniam, Hrenya (2012). J.Fluid Mech. 712, 129.
IA_NONEP		Iddir & Arastoopour (2005). AIChE J. 51(6), 1620
LUN_1984		Lun et al (1984). J. Fluid Mech., 140, 223.
SIMONIN		Simonin (1996). VKI Lecture Series, 1996-2

FRICTION_MODEL

Data Type: CHARACTER

Applies to Solids Model(s): **TFM**

Solids stress friction model selection.

Table 8.35: Valid Values

Name	Default?	Description
NONE		Only solids pressure
SCHAEFFER	◉	Schaeffer friction model
SRIVASTAVA		Srivastava friction model

BLENDING_FUNCTION

Data Type: CHARACTER

Applies to Solids Model(s): **TFM**

Blend the Schaeffer stresses with the stresses resulting from algebraic kinetic theory around the value of EP_STAR. [NONE]

Table 8.36: Valid Values

Name	Default?	Description
NONE	◉	No blending
TANH_BLEND		Hyperbolic tangent function
SIGM_BLEND		Scaled sigmodial function

YU_STANDISH

Data Type: LOGICAL

Applies to Solids Model(s): **TFM**

Use Yu-Standish correlation to compute maximum packing for polydisperse systems.

A.B. Yu and N. Standish. Powder Tech, 52 (1987) 233-241

Table 8.37: Valid Values

Name	Default?	Description
`.TRUE.`		Use the Yu-Standish correlation.
`.FALSE.`	◉	Do not use the Yu-Standish correlation.

FEDORS_LANDEL

Data Type: LOGICAL

Applies to Solids Model(s): **TFM**

Use Fedors-Landel correlation to compute maximum packing for binary (only) mixtures of powders.

R.F. Fedors and R.F. Landel. Powder Tech, 23 (1979) 225-231

Table 8.38: Valid Values

Name	Default?	Description
`.TRUE.`		Use the Fedors-Landel correlation.
`.FALSE.`	◉	Do not use the Fedors-Landel correlation.

RDF_TYPE

Data Type: CHARACTER

Applies to Solids Model(s): **TFM**

Radial distribution function (RDF) at contact.

Table 8.39: Valid Values

Name	Default?	Description
`CARNAHAN_STARLING`		Carnahan, N.F. and Starling K.E., (1969). The Journal of Chemical Physics, Vol. 51(2):635-636. Only applies to monodisperse cases.
`MA_AHMADI`		Ma, D. and Ahmadi, G., (1986). The Journal of Chemical Physics, 84(6):3449. Only applies to monodisperse cases.
`LEBOWITZ`		Lebowitz, J.L. (1964) The Physical Review, A133, 895-899. Only applies to polydisperse cases.
`MODIFIED_LEBOWITZ`		Iddir, H. Y., Modeling of the multiphase mixture of particles using the kinetic theory approach. Doctoral Dissertation, Illinois Institute of Technology, Chicago, Illinois, 2004, (chapter 2, equations 2-49 through 2-52.) Only applies to polydisperse cases.
`MANSOORI`		Mansoori, GA, Carnahan N.F., Starling, K.E. Leland, T.W. (1971). The Journal of Chemical Physics, Vol. 54:1523-1525. Only applies to polydisperse cases.
`MODIFIED_MANSOORI`		van Wachem, B.G.M., Schouten, J.C., van den Bleek, C.M., Krishna, R. and Sinclair, J. L. (2001) AIChE Journal 47:1035–1051. Only applies to polydisperse cases.

ADDED_MASS

Data Type: LOGICAL

Applies to Solids Model(s): **TFM**

Flag to include the added (or virtual) mass force. This force acts to increase the inertia of the dispersed phase, which tends to stabilize simulations of bubbly gas-liquid flows.

M_AM

Data Type: INTEGER

Applies to Solids Model(s): **TFM**

The disperse phase number to which the added mass is applied.

C_E

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Coefficient of restitution for particle-particle collisions.

R_P(PHASE, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 2$
- $1 \leq Phase \leq 2$

Coefficient of restitution for particle-particle collisions specific to GHD theory implementation.

E_W

Data Type: DOUBLE PRECISION

Coefficient of restitution for particle-wall collisions when using Johnson and Jackson partial slip BC (BC_JJ_PS).

PHIP

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Specularity coefficient associated with particle-wall collisions when using Johnson and Jackson partial slip BC (BC_JJ_PS). If Jenkins small frictional BC are invoked (JENKINS) then PHIP is not used.

PHIP0

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Specify the value of specularity coefficient when the normalized slip velocity goes to zero when BC_JJ_M is .TRUE.. This variable is calculated internally in MFiX. Do not modify unless an accurate number is known.

C_F

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Coefficient of friction between the particles of two solids phases.

PHI

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Angle of internal friction (in degrees). Set this value to zero to turn off plastic regime stress calculations.

PHI_W

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Angle of internal friction (in degrees) at walls. Set this value to non-zero (PHI_W = 11.31 means TAN_PHI_W = MU = 0.2) when using Johnson and Jackson partial slip BC (BC_JJ_PS) with Friction model or Jenkins small frictional boundary condition.

EPS_F_MIN

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Minimum solids fraction above which friction sets in. [0.5]

EP_S_MAX(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Maximum solids volume fraction at packing for polydisperse systems (more than one solids phase used). The value of EP_STAR may change during the computation if solids phases with different particle diameters are specified and Yu_Standish or Fedors_Landel correlations are used.

SEGREGATION_SLOPE_COEFFICIENT

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Used in calculating the initial slope of segregation: see Gera et al. (2004) - recommended value 0.3. Increasing this coefficient results in decrease in segregation of particles in binary mixtures.

V_EX

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Excluded volume in Boyle-Massoudi stress.

Table 8.40: Valid Values

Name	Default?	Description
`0.0`		Boyle-Massoudi stress is turned off.

MU_S0(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Specified constant viscosity. If any value is specified then:

- **kinetic theory calculations (granular_energy) are off, which** means zero granular pressure contribution (P_S = 0)
- **frictional/plastic calculations are off, which means zero** frictional viscosity contributions, however, a plastic pressure term is still invoked (P_STAR)
- LAMBDA_S = -2/3 MU_S0

DIF_S0(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Specified constant solids diffusivity [m^2/s in SI].

EP_STAR

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**

Packed bed void fraction. Used to calculate plastic stresses (for contribution to viscosity) and when to implement plastic pressure, P_STAR. Specifically, if EP_G < EP_STAR, then plastic pressure is employed in the momentum equations.

CLOSE_PACKED(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to enable/disable a phase from forming a packed bed. Effectively removes plastic pressure term from the solids phase momentum equation.

Table 8.41: Valid Values

Name	Default?	Description
`.TRUE.`	◉	The phase forms a packed bed with void fraction EP_STAR.
`.FALSE.`		The phase can exceed close pack conditions so that it maybe behave like a liquid.

JENKINS

Data Type: LOGICAL

This flag affects how the momentum and granular energy boundary conditions are implemented when using BC_JJ_PS BC.

Table 8.42: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Use standard boundary conditions.
`.TRUE.`		Use Jenkins small frictional boundary condition.

8.3.6 Discrete Element

Discrete Element - Common

These are keywords common to Discrete Element Model (DEM) and Particles In Cell (PIC).

PARTICLES

Data Type: INTEGER

Applies to Solids Model(s): **DEM**

Number of particles to be read in from the particle_input.dat file. This value is overwritten when using automatic particle generation. A simulation with a mass inflow BC can start without solids by setting PARTICLES = 0.

GENER_PART_CONFIG

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Automatically generate the initial particle position and velocity data based on the parameters specified for each initial condition (IC) region.

Table 8.43: Valid Values

Name	Default?	Description
`.TRUE.`		Generate particle configuration based on the initial condition parameters. Data provided in the particle_input.dat file, if present, is ignored.
`.FALSE.`	◉	Particle position and velocity data are provided in the particle_input.dat file. A run-time error occurs if this file is not provided.

DES_ONEWAY_COUPLED

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Run one-way coupled simulations [default .FALSE.]. If set, the fluid does not see the particles in terms of drag force. The effect of particle volume is still felt by the fluid through non-unity voidage values.

Table 8.44: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Two-way particle-fluid coupling.
`.TRUE.`		One-way particle-fluid coupling (fluid does not see particle drag).

DES_INTG_METHOD

Data Type: CHARACTER

Applies to Solids Model(s): **DEM**

Time stepping scheme.

Table 8.45: Valid Values

Name	Default?	Description
`EULER`	◉	First-order Euler scheme.
`ADAMS_BASHFORTH`		Second-order ADAMS_BASHFORTH scheme (DEM only)

DESGRIDSEARCH_IMAX

Data Type: INTEGER

Applies to Solids Model(s): **DEM**

Number of des grid cells in the I-direction. If left undefined, then it is set by MFiX such that its size equals three times the maximum particle diameter with a minimum of 1 cell.

DESGRIDSEARCH_JMAX

Data Type: INTEGER

Applies to Solids Model(s): **DEM**

Number of des grid cells in the J-direction. If left undefined, then it is set by MFiX such that its size equals three times the maximum particle diameter with a minimum of 1 cell.

DESGRIDSEARCH_KMAX

Data Type: INTEGER

Applies to Solids Model(s): **DEM**

Number of des grid cells in the K-direction. If left undefined, then it is set by MFiX such that its size equals three times the maximum particle diameter with a minimum of 1 cell.

DES_INTERP_SCHEME

Data Type: CHARACTER

Applies to Solids Model(s): **DEM**

Specify the scheme used to map data to/from a particle's position and the Eulerian grid. This keyword is required when DES_INTERP_MEAN_FIELDS and/or DES_INTERP_ON are specified.

Table 8.46: Valid Values

Name	Default?	Description
`NONE`	◉	Do not use interpolation.
`GARG_2012`		Interpolate to/from a particle's position using the corners (nodes) of the fluid cells. This was the default behavior prior to the 2015-1 Release. See Garg et al. (2012) Documentation of the open-source MFiX-DEM software for gas-solids flows.
`SQUARE_DPVM`		Divided Particle Volume Method: Information is interpolated to/from a particles position using a square filter of size DES_INTERP_WIDTH.
`LINEAR_HAT`		Linear interpolation: Hat functions are used to distribute particle information.

DES_INTERP_WIDTH

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Length used in interpolating data to/from a particle's position and the Eulerian grid. The interpolation width is only applicable to the DPVM_SQUARE and DPVM_GAUSS interpolation schemes as the GARG_2012 scheme's interpolation width is determined by the Eulerian grid dimensions.

- The interpolation half-width cannot exceed the minimum cell dimension because interpolation is restricted to the 27-cell neighborhood surrounding a particle (9-cell neighborhood in 2D).
- It is recommended that the DES_INTERP_WIDTH be set equal to the maximum particle diameter when using STL defined boundaries. Field data can be smoothed by specifying DES_DIFFUSE_WIDTH.

DES_INTERP_ON

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Enable/disable interpolation of field quantities to a particle's position. This is used in calculating gas-particle interactions, such as the drag force.

Table 8.47: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Use fluid values from the cell containing the particle's center.
`.TRUE.`		Interpolate fluid values from the 27-cell neighborhood to a particle's position.

DES_INTERP_MEAN_FIELDS

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Enable/disable interpolation of particle data (e.g., solids volume and drag force) from a particle's position to the Eulerian grid.

Table 8.48: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Assign particle data to the fluid grid cell containing the particle's center.
`.TRUE.`		Interpolate particle data from the particle's position to the 27-cell neighborhood surrounding the particle.

DES_DIFFUSE_WIDTH

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

The length scale used to smooth dispersed phase averaged fields by solving a diffusion equation. This approach is typically used when particle sizes near or exceed the size of the Eulerian grid cell sizes.

- Mean field diffusion is disabled if DES_DIFFUSE_WIDTH is not specified.
- Mean field diffusion cannot be used with the GARG_2012 interpolation scheme.
- It is recommended that mean field diffusion be used in conjunction with DES_EXPLICITLY_COUPLED to minimize the computational cost of diffusing field data.
- The DES diffusion equation is listed as equation type 10 in the Numerical Parameters section.

DES_EXPLICITLY_COUPLED

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Enable/Disable explicit coupling of DEM solids and the fluid. This algorithm is presently limited to hydrodynamic simulations.

Table 8.49: Valid Values

Name	Default?	Description
`.FALSE.`	◉	The fluid and particles calculate interphase forces at their respective time scales. The fluid phase calculates the interphase coupling forces once per fluid time step. Similarly, DEM particles calculate the interface coupling forces at each solids time-step. The DEM must also bin particles to the fluid grid and recalculate the fluid volume fraction every time-step.
`.TRUE.`		Interphase forces are calculated during the fluid time step and stored for each particle. The interphase forces are then distributed among the solids time-steps. This approach can substantially reduce the computational overhead for coupled simulations.

Discrete Element Model

These keywords relate to the Discrete Element Model (DEM).

NFACTOR

Data Type: INTEGER

The number of iterations of a pure granular simulation to let the initial particle configuration settle before a coupled gas-solid calculation is started.

NEIGHBOR_SEARCH_N

Data Type: INTEGER

Maximum number of steps through a DEM loop before a neighbor search will be performed. The search may be called earlier based on other logic.

DES_NEIGHBOR_SEARCH

Data Type: INTEGER

Flag to set the neighbor search algorithm.

Table 8.50: Valid Values

Name	Default?	Description
1		N-Square search algorithm (most expensive)
4	◉	Grid-Based Neighbor Search (Recommended)

NEIGHBOR_SEARCH_RAD_RATIO

Data Type: DOUBLE PRECISION

Ratio of the distance (imaginary sphere radius) to particle radius that is allowed before a neighbor search is performed. This works in conjunction with the logic imposed by NEIGHBOR_SEARCH_N in deciding calls to the neighbor search algorithm.

FACTOR_RLM

Data Type: DOUBLE PRECISION

Effectively increases the radius of a particle (multiple of the sum of particle radii) during the building of particle neighbor list.

USE_VDH_DEM_MODEL

Data Type: LOGICAL

Flag to use van der Hoef et al. (2006) model for adjusting the rotation of the contact plane. See the MFiX-DEM documentation.

DES_COLL_MODEL

Data Type: CHARACTER

Collision model for the soft-sphere approach used in DEM model. All models require specifying the following parameters: DES_EN_INPUT, DES_EN_WALL_INPUT, MEW, and MEW_W.

Table 8.51: Valid Values

Name	Default?	Description
`LSD`	◉	The linear spring-dashpot model. Requires: KN, KN_W, KT_FAC, KT_W_FAC, DES_ETAT_FAC, DES_ETAT_W_FAC.
`HERTZIAN`		The Hertzian model. Requires: DES_ET_INPUT, DES_ET_WALL_INPUT, E_YOUNG, EW_YOUNG V_POISSON, VW_POISSON.

KN

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Normal spring constant [N/m in SI] for inter-particle collisions. Required when using the linear spring-dashpot collision model.

KT_FAC

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Ratio of the tangential spring constant to normal spring constant for inter-particle collisions. Use it to specify the tangential spring constant for particle-particle collisions as KT_FAC*KN. Required when using the linear spring-dashpot collision model.

KN_W

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Normal spring constant [N/m in SI] for particle-wall collisions. Required when using the linear spring-dashpot collision model.

KT_W_FAC

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Ratio of the tangential spring constant to normal spring constant for particle-wall collisions. Use it to specify the tangential spring constant for particle-wall collisions as KT_W_FAC*KN_W. Required when using the linear spring-dashpot collision model.

MEW

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Inter-particle Coulomb friction coefficient.

MEW_W

Data Type: DOUBLE PRECISION

Particle-wall Coulomb friction coefficient.

DES_EN_INPUT(INDEX)

Data Type: DOUBLE PRECISION

- $1 \leq Index \leq MMAX * (MMAX - 1)/2$

Applies to Solids Model(s): **DEM**

Normal restitution coefficient for inter-particle collisions used to determine the inter-particle normal damping factor.

Values should be defined for a single dimensional array. For example, a simulation with three solids phases (MMAX=3) needs six values: en11, en12, en13; en22, en23; en33.

DES_EN_WALL_INPUT(INDEX)

Data Type: DOUBLE PRECISION

- $1 \leq Index \leq MMAX$

Applies to Solids Model(s): **DEM**

Normal restitution coefficient for particle-wall collisions used to determine the particle-wall normal damping factor.

Values should be defined in a single dimensional array. For example, a simulation with three solids phases (MMAX=3) needs three values: enw1, enw2, enw3.

DES_ET_INPUT(INDEX)

Data Type: DOUBLE PRECISION

- $1 \leq Index \leq MMAX * (MMAX - 1)/2$

Applies to Solids Model(s): **DEM**

Tangential restitution coefficient for inter-particle collisions. Values are defined in a one dimensional array. This is required input when using the Hertzian collision model.

DES_ET_WALL_INPUT(INDEX)

Data Type: DOUBLE PRECISION

- $1 \leq Index \leq MMAX$

Applies to Solids Model(s): **DEM**

Tangential restitution coefficient for particle wall collisions. Values are defined in a one dimensional array. This is required input when using the Hertzian collision model.

DES_ETAT_FAC

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **DEM**

Ratio of the tangential damping factor to the normal damping factor for inter-particle collisions. Required for the linear spring-dashpot collision model.

Table 8.52: Valid Values

Name	Default?	Description
`UNDEFINED`	◉	For LSD model, if left undefined, MFiX reverts to default value of 0.5.

DES_ETAT_W_FAC

Data Type: DOUBLE PRECISION

Ratio of the tangential damping factor to the normal damping factor for particle-wall collisions. Required for the linear spring-dashpot model for soft-spring collision modelling under DEM. For the Hertzian model, the tangential damping coefficients have to be explicitly specified and specification of this variable is not required.

Table 8.53: Valid Values

Name	Default?	Description
`UNDEFINED`	◉	For LSD model, if left undefined, MFiX will revert to default value of 0.5

EW_YOUNG

Data Type: DOUBLE PRECISION

Young's modulus for the wall [Pa in SI]. Required when using the Hertzian collision model.

VW_POISSON

Data Type: DOUBLE PRECISION

Poisson ratio for the wall. Required when using the Hertzian collision model.

E_YOUNG(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq DES_MMAX$

Young's modulus for the particle [Pa in SI]. Required when using the Hertzian collision model.

V_POISSON(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq DES_MMAX$

Poisson's ratio for the particle. Required when using the Hertzian collision model.

USE_COHESION

Data Type: LOGICAL

Flag to enable/disable cohesion model.

VAN_DER_WAALS

Data Type: LOGICAL

Flag to turn on the Hamaker van der Waals forces.

HAMAKER_CONSTANT

Data Type: DOUBLE PRECISION

Hamaker constant used in particle-particle cohesive interactions.

WALL_HAMAKER_CONSTANT

Data Type: DOUBLE PRECISION

Hamaker constant used in particle-wall cohesive interactions.

VDW_OUTER_CUTOFF

Data Type: DOUBLE PRECISION

Maximum separation distance above which van der Waals forces are not implemented.

VDW_INNER_CUTOFF

Data Type: DOUBLE PRECISION

Minimum separation distance below which van der Waals forces are calculated using a surface adhesion model.

WALL_VDW_OUTER_CUTOFF

Data Type: DOUBLE PRECISION

Maximum separation distance above which van der Waals forces are not implemented (particle-wall interactions).

WALL_VDW_INNER_CUTOFF

Data Type: DOUBLE PRECISION

Minimum separation distance below which van der Waals forces are calculated using a surface adhesion model (particle-wall interactions).

ASPERITIES

Data Type: DOUBLE PRECISION

Mean radius of surface asperities that influence the cohesive force. See H. Rumpf, Particle Technology, Chapman & Hall, London/New York, 1990.

DES_CONV_CORR

Data Type: CHARACTER

Specify the Nusselt number correlation used for particle-gas convection.

Table 8.54: Valid Values

Name	Default?	Description
`RANZ_1952`	◉	Ranz, W.E. and Marshall, W.R. (1952). Chemical Engineering Progress, 48: 141-146 and 173-180

DES_MIN_COND_DIST

Data Type: DOUBLE PRECISION

Minimum separation distance between the surfaces of two contacting particles.

FLPC

Data Type: DOUBLE PRECISION

Fluid lens proportionality constant used to calculate the radius of the fluid lens that surrounds a particle. This parameter is used in the particle-fluid-particle conduction model.

DES_EM(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq DES_MMAX$

Emissivity of solids phase.

E_YOUNG_ACTUAL(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq DES_MMAX$

Actual Young's modulus for the particle [Pa in SI]. Used for computing correction terms for DEM conduction.

EW_YOUNG_ACTUAL

Data Type: DOUBLE PRECISION

Actual Young's modulus for the walls [Pa in SI]. Used for computing correction terms for DEM conduction.

V_POISSON_ACTUAL(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq DES_MMAX$

Poisson's ratio for the particle. Used for computing correction terms for DEM conduction.

VW_POISSON_ACTUAL

Data Type: DOUBLE PRECISION

Poisson's ratio for the wall. Used for computing correction terms for DEM conduction.

MINIMIZE_DES_FACET_LIST

Data Type: LOGICAL

Flag to turn on/off optimizing the list of facets at each des grid cell.

REMOVE_ROGUE_PARTICLES

Data Type: LOGICAL

Flag to remove rogue particles. A rogue particle is defined as a particle that goes out of the MFiX box outside of a defined pressure or mass outlet BC.

DTSOLID_FACTOR

Data Type: DOUBLE PRECISION

Ratio between the collision time and the DEM time step (default value is 50). Minimum value is 10.0. It is highly recommended to keep the default value.

DTSOLID_UPDATE_DT

Data Type: DOUBLE PRECISION

Time interval at which the DEM solids time step is updated. Useful when particle's size or mass change over time. Only applicable to coupled gas/solids flows (this setting is ignored for pure granular flows).

DLB_DT

Data Type: DOUBLE PRECISION

Time interval at which Dynamic Load Balance (DLB) is performed (sec)

Table 8.55: Valid Values

Name	Default?	Description
`UNDEFINED`	◉	: UNDEFINED value turns Dynamic Load Balance off
`>0`		: Any positive value turns Dynamic Load Balance on

DLB_EGW

Data Type: DOUBLE PRECISION

Eulerian Grid Weight use in Dynamic Load Balance. A value of zero means the balancing only considers particles. A very large value means the partition will balance only the fluid mesh. Finding the optimal value will require trial and error.

Table 8.56: Valid Values

Name	Default?	Description
>=0		

Particles in Cell

These keywords relate to the Particle in Cell model. Please note that the PIC model is currently not supported by the GUI, because PIC support is undergoing a rewrite in the solver.

FRIC_EXP_PIC

Data Type: DOUBLE PRECISION

Volume fraction exponential scale factor in frictional stress model.

PSFAC_FRIC_PIC

Data Type: DOUBLE PRECISION

Pressure linear scale factor in frictional stress model.

MPPIC_COEFF_EN1

Data Type: DOUBLE PRECISION

An empirical dampening factor for the frictional stress model.

FRIC_NON_SING_FAC

Data Type: DOUBLE PRECISION

Non-singularity term in frictional stress model.

MPPIC_COEFF_EN_WALL

Data Type: DOUBLE PRECISION

Normal coefficient of restitution for parcel-wall collisions.

MPPIC_COEFF_ET_WALL

Data Type: DOUBLE PRECISION

Tangential coefficient of restitution for parcel-wall collisions.

MPPIC_VELFAC_COEFF

Data Type: DOUBLE PRECISION

Solids slip velocity scale factor. This term can be used to scale the bulk solids velocity when calculating parcel/bulk solids slip velocity. Scaling is uniform in all three directions.

PIC_CFL

Data Type: DOUBLE PRECISION

Solids CFL (Courant-Friedrichs-Lewy) value. This term can be used to arrest solids time step through a local examination of parcel velocity. Leave undefined (blank) to disable time step control.

PIC_CFL_PARCEL_FRACTION

Data Type: DOUBLE PRECISION

Parcel fraction threshold to invoke a change in solids time step based on solids CFL value.

PIC_CFL_CONTROL

Data Type: CHARACTER

Control variable for managing solids CFL value. Solids time step will be based on MAX or AVG calculated solids CFL.

PIC_COLLISION_DAMPING

Data Type: LOGICAL

Flag to turn on/off PIC collision damping model.

8.3.7 Gas Phase

RO_G0

Data Type: DOUBLE PRECISION

Specified constant gas density [kg/m^3 in SI]. An equation of state, the ideal gas law by default, is used to calculate the gas density if this parameter is undefined. The value may be set to zero to make the drag zero and to simulate granular flow in a vacuum. For this case, users may turn off solving for gas momentum equations to accelerate convergence.

MU_G0

Data Type: DOUBLE PRECISION

Specified constant gas viscosity [kg/(m.s) in SI].

K_G0

Data Type: DOUBLE PRECISION

Specified constant gas conductivity [J/(s.m.K) in SI].

C_PG0

Data Type: DOUBLE PRECISION

Specified constant gas specific heat [J/(kg.K) in SI].

DIF_G0

Data Type: DOUBLE PRECISION

Specified constant gas diffusivity [m^2/s in SI].

MW_AVG

Data Type: DOUBLE PRECISION

Average molecular weight of gas [kg/kmol in SI]. Used in calculating the gas density for non-reacting flows when the gas composition is not defined.

MW_G(SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq Species \leq 100$

Molecular weight of gas species [kg/kmol in SI].

NMAX_G

Data Type: INTEGER

Number of species comprising the gas phase.

SPECIES_G(SPECIES)

Data Type: CHARACTER

- $1 \leq Species \leq 100$

Name of gas phase species as it appears in the materials database.

SPECIES_ALIAS_G(SPECIES)

Data Type: CHARACTER

- $1 \leq Species \leq 100$

User defined name for gas phase species. Aliases are used in specifying chemical equations and must be unique.

8.3.8 Solids Phase

SOLIDS_MODEL(PHASE)

Data Type: CHARACTER

- $1 \leq Phase \leq 10$

Defines the model used for the solids phase.

Table 8.57: Valid Values

Name	Default?	Description
`TFM`	◉	Two-fluid model (continuum)
`DEM`		Discrete element model
`CGP`		Coarse-grained particle
`PIC`		Multiphase particle-in-cell

MMAX

Data Type: INTEGER

Applies to Solids Model(s): **TFM**, **DEM**

Number of solids phases.

D_P0(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Initial particle diameters [m in SI].

RO_S0(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Specified constant solids density [kg/m^3 in SI]. Reacting flows may use variable solids density by leaving this parameter undefined and specifying X_S0 and RO_XS0 as well as the index of the inert species.

X_S0(PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$
- $1 \leq Species \leq DIM_N_s$

Applies to Solids Model(s): **TFM**, **DEM**

Baseline species mass fraction. Specifically, the mass fraction of an unreacted sample (e.g., proximate analysis).

RO_XS0(PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$
- $1 \leq Species \leq DIM_N_s$

Applies to Solids Model(s): **TFM**, **DEM**

Specified constant solids species density [kg/m^3 in SI].

INERT_SPECIES(PHASE)

Data Type: INTEGER

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Index of inert solids phase species. This species should not be a product or reactant of any chemical reaction.

DIL_INERT_X_VSD(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Mass fraction of inert solids phase species in the dilute region. In dilute region (see DIL_FACTOR_VSD), the solids density is computed based on this inert species mass fraction, rather than the current inert species mass fraction. This may help convergence when the Variable Solids Density model is invoked.

DIL_FACTOR_VSD

Data Type: DOUBLE PRECISION

Applies to Solids Model(s): **TFM**, **DEM**

Factor to define the dilute region where the solids density is set using DIL_INERT_X_VSD. Cells where the solids volume fraction is between DIL_EP_S and DIL_EP_S x DIL_FACTOR_VSD will automatically set the solids density using DIL_INERT_X_VSD instead of the current inerts species mass fraction. Set this factor to zero to always use the current inert species mass fraction.

K_S0(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Specified solids or material conductivity [J/(s.m.K) in SI].

KS_MODEL(PHASE)

Data Type: CHARACTER

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Solids (TFM) or particle (DEM) thermal conductivity model for each phase.

Table 8.58: Valid Values

Name	Default?	Description
`NONE`		(TFM or DEM). K_s0 is set to 0 (no conduction). In DEM, calculations are skipped.
`CONST_EFF`		(TFM only). K_s0 specifies a constant effective solids phase conductivity.
`USR`		(TFM only). K_s0 is set based on user defined function.
`Bauer`		(TFM only). K_s0 specifies the material conductivity. See Bauer, R. and Schlünder, E.U. (1978). Int. Chem. Eng. 18(2), 181.
`Musser`		(DEM only). K_s0 specifies the material conductivity. See Musser, J. (2011), PhD Dissertation, WVU. Chapter 3 (3.6).

C_PS0(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Specified constant solids specific heat [J/(kg.K) in SI].

MW_S(PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$
- $1 \leq Species \leq DIM_N_s$

Applies to Solids Model(s): **TFM**, **DEM**

Molecular weight of solids phase species [kg/kmol in SI].

NMAX_S(PHASE)

Data Type: INTEGER

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**, **DEM**

Number of species comprising the solids phase.

SPECIES_S(PHASE, SPECIES)

Data Type: CHARACTER

- $1 \leq Phase \leq 10$
- $1 \leq Species \leq DIM_N_s$

Applies to Solids Model(s): **TFM**, **DEM**

Name of solids phase M, species N as it appears in the materials database.

SPECIES_ALIAS_S(PHASE, SPECIES)

Data Type: CHARACTER

- $1 \leq Phase \leq 10$
- $1 \leq Species \leq DIM_N_s$

Applies to Solids Model(s): **TFM**, **DEM**

User defined name for solids phase species. Aliases are used in specifying chemical equations and must be unique.

CGP_STAT_WT(PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **DEM**

Coarse grain particle statistical weight.

8.3.9 Initial Condition

Each initial condition (IC) is specified over a rectangular region (or pie-shaped for cylindrical coordinates) that corresponds to the scalar numerical grid. These are 3D regions: `X_w X_e`, `Y_s Y_n`, and `Z_t Z_b`. The region is defined by the constant coordinates of each of the six faces, which may be specified as the physical coordinates or the cell indices. The physical coordinates are easier to specify than the cell indices. If cell sizes are not small enough to resolve a region specified using physical coordinates, MFIX will indicate this problem with an error message.

In cylindrical coordinates, when the theta direction crosses the 0 value, split that region into two regions: e.g., Split a region spanning 1.9 pi to 0.1 pi as 1.9 pi to 2 pi and 0 to 0.1 pi.

Initial condition regions may overlap. When an overlap occurs, MFIX uses the conditions specified for the higher IC number.

IC_X_W(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

X coordinate of the west face.

IC_X_E(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

X coordinate of the east face.

IC_Y_S(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Y coordinate of the south face.

IC_Y_N(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Y coordinate of the north face.

IC_Z_B(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Z coordinate of the bottom face.

IC_Z_T(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Z coordinate of the top face.

IC_I_W(IC)

Data Type: INTEGER

- $1 \leq IC \leq 500$

I index of the west-most wall.

IC_I_E(IC)

Data Type: INTEGER

- $1 \leq IC \leq 500$

I index of the east-most wall.

IC_J_S(IC)

Data Type: INTEGER

- $1 \leq IC \leq 500$

J index of the south-most wall.

IC_J_N(IC)

Data Type: INTEGER

- $1 \leq IC \leq 500$

J index of the north-most wall.

IC_K_B(IC)

Data Type: INTEGER

- $1 \leq IC \leq 500$

K index of the bottom-most wall.

IC_K_T(IC)

Data Type: INTEGER

- $1 \leq IC \leq 500$

K index of the top-most wall.

IC_TYPE(IC)

Data Type: CHARACTER

- $1 \leq IC \leq 500$

Type of initial condition. Mainly used in restart runs to overwrite values read from the .RES file by specifying it as _PATCH_. The user needs to be careful when using the _PATCH_ option, since the values from the .RES file are overwritten and no error checking is done for the patched values.

IC_EP_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial void fraction in the IC region.

IC_P_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial gas pressure in the IC region. If this quantity is not specified, MFiX will set up a hydrostatic pressure profile, which varies only in the y-direction.

IC_P_STAR(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial solids pressure in the IC region. Usually, this value is specified as zero.

IC_L_SCALE(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Turbulence length scale in the IC region.

IC_ROP_S(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial bulk density (rop_s = ro_s x ep_s) of solids phase in the IC region. Users need to specify this IC only for polydisperse flow (MMAX > 1). Users must make sure that summation of (IC_ROP_s(ic,m) / RO_s(m)) over all solids phases is equal to (1.0 - IC_EP_g(ic)).

IC_EP_S(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial solids volume fraction of solids phase in the IC region. This may be specified in place of IC_ROP_s.

IC_PSD_TYPE(IC, PHASE)

Data Type: CHARACTER

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial particle size distribution type of solids phase in the IC region.

Table 8.59: Valid Values

Name	Default?	Description
`MONO`	◉	Uniform particle size.
`NORMAL`		Normal (Gaussian) particle size distribution.
`LOG_NORMAL`		Log-normal particle size distribution.
`CUSTOM`		Particle size distribution specified in external file.

IC_PSD_MEAN_DP(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial mean value of particle size distribution of solids phase in the IC region. For a log-normal psd, The mean diameter (μ) and standard deviation (σ) will be used to compute the log-normal parameters μ_ln and σ_ln : μ_ln = ln($\mu^2/\sqrt{(\mu^2 + \sigma^2)}$) and σ_ln = $\sqrt{(\ln(1 + (\sigma/\mu)^2))}$

IC_PSD_STDEV(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial standard deviation of particle size distribution of solids phase in the IC region. For a log-normal psd, The mean diameter (μ) and standard deviation (σ) will be used to compute the log-normal parameters μ_ln and σ_ln : μ_ln = ln($\mu^2/\sqrt{(\mu^2 + \sigma^2)}$) and σ_ln = $\sqrt{(\ln(1 + (\sigma/\mu)^2))}$

IC_PSD_MAX_DP(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial maximum particle size of particle size distribution of solids phase in the IC region.

IC_PSD_MIN_DP(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial minimum particle size of particle size distribution of solids phase in the IC region.

IC_T_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial gas phase temperature in the IC region.

IC_T_S(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial solids phase temperature in the IC region.

IC_THETA_M(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial solids phase granular temperature in the IC region.

IC_GAMA_RG(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Gas phase radiation coefficient in the IC region. Modify file rdtn2.inc to change the source term.

IC_T_RG(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Gas phase radiation temperature in the IC region.

IC_GAMA_RS(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase radiation coefficient in the IC region. Modify file energy_mod.f to change the source term.

IC_T_RS(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase radiation temperature in the IC region.

IC_U_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial x-component of gas velocity in the IC region.

IC_U_S(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial x-component of solids-phase velocity in the IC region.

IC_V_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial y-component of gas velocity in the IC region.

IC_V_S(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial y-component of solids-phase velocity in the IC region.

IC_W_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial z-component of gas velocity in the IC region.

IC_W_S(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial z-component of solids-phase velocity in the IC region.

IC_X_G(IC, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Species \leq 100$

Initial mass fraction of gas species.

IC_X_S(IC, PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Initial mass fraction of solids species.

IC_SCALAR(IC, SCALAR EQ.)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq ScalarEq. \leq 100$

Initial value of Scalar n, assigned to scalar equation.

IC_K_TURB_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial value of K in K-Epsilon equation.

IC_E_TURB_G(IC)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$

Initial value of Epsilon in K-Epsilon equation.

IC_DES_FIT_TO_REGION(IC)

Data Type: LOGICAL

- $1 \leq IC \leq 500$

Flag for inflating initial lattice distribution to the entire IC region.

IC_DES_LATTICE(IC, PHASE)

Data Type: CHARACTER

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Type of initial lattice distribution (simple cubic or hexagonal) to the entire IC region.

Table 8.60: Valid Values

Name	Default?	Description
`HEXA`	◉	Particles are distributed on a hexagonal close-packed lattice.
`CUBIC`		Particles are distributed on a cubical lattice.

IC_DES_SPACING(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Spacing between particle within initial lattice distribution (expressed as a fraction of particle diameter) to the entire IC region.

IC_DES_SPACE_FACTOR_X(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Spacing factor in x-direction (multiplier of IC_DES_SPACING).

IC_DES_SPACE_FACTOR_Y(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Spacing factor in y-direction (multiplier of IC_DES_SPACING).

IC_DES_SPACE_FACTOR_Z(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Spacing factor in z-direction (multiplier of IC_DES_SPACING).

IC_DES_RAND(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Random factor applied to particle positions within the lattice distribution.

IC_DES_RAND_FACTOR_X(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Random factor in x-direction (multiplier of IC_DES_RAND)

IC_DES_RAND_FACTOR_Y(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Random factor in y-direction (multiplier of IC_DES_RAND)

IC_DES_RAND_FACTOR_Z(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Random factor in z-direction (multiplier of IC_DES_RAND)

IC_DES_SM(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial solids mass within the entire IC region.

IC_DES_NP(IC, PHASE)

Data Type: INTEGER

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Initial particle count within the entire IC region.

IC_DES_CHECK_STL_OVERLAP(IC, PHASE)

Data Type: LOGICAL

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Check for particle overlap with SL geometry in the entire IC region.

IC_PIC_CONST_STATWT(IC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IC \leq 500$
- $1 \leq Phase \leq 10$

Flag to specify the initial constant statistical weight for computational particles/parcels. Actual number of parcels will be automatically computed.

8.3.10 Boundary Conditions

Boundary conditions (BC) are specified over flow planes or 2D surfaces that are normal to one of the coordinate directions and coincide with a face of the scalar control-volume. The values for one of the three pairs of coordinates are equal. The surface is defined by the constant coordinates of each of the four edges, which can be specified with physical coordinates or cell indices, and the two equal values for the direction normal to the face, which can only be specified with physical coordinates. If cell sizes are not small enough to resolve a surface specified using physical coordinates, MFIX will indicate this problem with an error message.

A flow plane must have a wall cell (or an outside boundary) on one side and a flow cell on the other side. The BC section is also used to specify obstacles in the flow domain. Obstacles are 3D regions, just as for the IC regions: `X_w X_e, Y_s Y_n, and Z_t Z_b`. By default the outside boundary is initialized as no-slip walls. For cylindrical coordinates the axis is initialized as a free-slip wall.

Two boundary surfaces must not intersect. Two obstacle regions may intersect.

BC_X_W(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

X coordinate of the west face or edge.

BC_X_E(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

X coordinate of the east face or edge.

BC_Y_S(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Y coordinate of the south face or edge.

BC_Y_N(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Y coordinate of the north face or edge.

BC_Z_B(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Z coordinate of the bottom face or edge.

BC_Z_T(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Z coordinate of the top face or edge.

BC_I_W(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

I index of the west-most cell.

BC_I_E(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

I index of the east-most cell.

BC_J_S(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

J index of the south-most cell.

BC_J_N(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

J index of the north-most cell.

BC_K_B(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

K index of the bottom-most cell.

BC_K_T(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

K index of the top-most cell.

BC_TYPE(BC)

Data Type: CHARACTER

- $1 \leq BC \leq 500$

Type of boundary.

Table 8.61: Valid Values

Name	Default?	Description
DUMMY		The specified boundary condition is ignored. This is useful for turning off some boundary conditions without having to delete them from the file.
MASS_INFLOW		Mass inflow rates for gas and solids phases are specified at the boundary.
MASS_OUTFLOW		The specified values of gas and solids mass outflow rates at the boundary are maintained, approximately. This condition should be used sparingly for minor outflows, when the bulk of the outflow is occurring through other constant pressure outflow boundaries.
P_INFLOW		Inflow from a boundary at a specified constant pressure. To specify as the west, south, or bottom end of the computational region, add a layer of wall cells to the west, south, or bottom of the PI cells. Users need to specify all scalar quantities and velocity components. The specified values of fluid and solids velocities are only used initially as MFIX computes these values at this inlet boundary.
P_OUTFLOW		Outflow to a boundary at a specified constant pressure. To specify as the west, south, or bottom end of the computational region, add a layer of wall cells to the west, south, or bottom of the PO cells.
FREE_SLIP_WALL		Velocity gradients at the wall vanish. If BC_JJ_PS is equal to 1, the Johnson-Jackson boundary condition is used for solids. A FSW is equivalent to using a PSW with Hw=0.
NO_SLIP_WALL		All components of the velocity vanish at the wall. If BC_JJ_PS is equal to 1, the Johnson-Jackson boundary condition is used for solids. A NSW is equivalent to using a PSW with vw=0 and Hw undefined.
PAR_SLIP_WALL		Partial slip at the wall implemented as the boundary condition: dv/dn + Hw (v - vw) = 0, where n is the normal pointing from the fluid into the wall. The coefficients Hw and vw should be specified. For free-slip set Hw = 0. For no-slip leave Hw undefined (Hw=+inf) and set vw = 0. To set Hw = +inf, leave it unspecified. If BC_JJ_PS is equal to 1, the Johnson-Jackson boundary condition is used for solids.

BC_PSD_TYPE(BC, PHASE)

Data Type: CHARACTER

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Particle size distribution type of solids phase at BC region.

Table 8.62: Valid Values

Name	Default?	Description
MONO	◉	Uniform particle size.
NORMAL		Normal (Gaussian) particle size distribution.
LOG_NORMAL		Log-normal particle size distribution.
CUSTOM		Particle size distribution specified in external file.

BC_PSD_MEAN_DP(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Mean value of particle size distribution of solids phase in the BC region. For a log-normal psd, The mean diameter (μ) and standard deviation (σ) will be used to compute the log-normal parameters μ_ln and σ_ln : μ_ln = ln(μ^2/√(μ^2 + σ^2)) and σ_ln = √(ln(1 + (σ/μ)^2)))

BC_PSD_STDEV(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Standard deviation of particle size distribution of solids phase in the BC region. For a log-normal psd, The mean diameter (μ) and standard deviation (σ) will be used to compute the log-normal parameters μ_ln and σ_ln : μ_ln = ln(μ^2/√(μ^2 + σ^2)) and σ_ln = √(ln(1 + (σ/μ)^2)))

BC_PSD_MAX_DP(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Maximum particle size of particle size distribution of solids phase in the BC region.

BC_PSD_MIN_DP(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Minimum particle size of particle size distribution of solids phase in the BC region.

BC_MI_START_TIME(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Mass inlet starting time (sec.).

BC_MI_END_TIME(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Mass inlet ending time (sec.).

BC_HW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas phase Hw for partial slip boundary.

BC_HW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase Hw for partial slip boundary.

BC_UW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas phase Uw for partial slip boundary.

BC_UW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase Uw for partial slip boundary.

BC_VW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas phase vw for partial slip boundary.

BC_VW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase vw for partial slip boundary.

BC_WW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas phase Ww for partial slip boundary.

BC_WW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase Ww for partial slip boundary.

BC_JJ_PS(BC)

Data Type: INTEGER

- $1 \leq BC \leq 500$

Johnson and Jackson partial slip BC.

Table 8.63: Valid Values

Name	Default?	Description
0		Do not use Johnson and Jackson partial slip bc. Default if granular energy transport equation is not solved.
1		Use Johnson and Jackson partial slip bc. Default if granular energy transport equation is solved.

BC_JJ_M

Data Type: LOGICAL

Use a modified version of Johnson and Jackson partial slip BC (BC_JJ_PS BC) with a variable specularity coefficient.

BC_THETAW_M(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Specified wall value, THETAw_M, in diffusion boundary condition: d(THETA_M)/dn + Hw (THETA_M - THETAw_M) = C, where n is the fluid-to-wall normal.

BC_HW_THETA_M(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Transfer coefficient, Hw, in diffusion boundary condition: d(THETA_M)/dn + Hw (THETA_M - THETAw_M) = C, where n is the fluid-to-wall normal.

BC_C_THETA_M(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Specified constant flux, C, in diffusion boundary condition: d(THETA_M)/dn + Hw (THETA_M - THETAw_M) = C, where n is the fluid-to-wall normal.

BC_HW_T_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas phase heat transfer coefficient, Hw, in diffusion boundary condition: d(T_g)/dn + Hw (T_g - Tw_g) = C, where n is the fluid-to-wall normal.

BC_TW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Specified gas phase wall temperature, Tw_g, in diffusion boundary condition: d(T_g)/dn + Hw (T_g - Tw_g) = C, where n is the fluid-to-wall normal.

BC_C_T_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Specified constant gas phase heat flux, C, in diffusion boundary condition: d(T_g)/dn + Hw (T_g - Tw_g) = C, where n is the fluid-to-wall normal.

BC_HW_T_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase heat transfer coefficient, Hw, in diffusion boundary condition: d(T_s)/dn + Hw (T_s - Tw_s) = C, where n is the fluid-to-wall normal.

BC_TW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Specified solids phase wall temperature, Tw_s, in diffusion boundary condition: d(T_s)/dn + Hw (T_s - Tw_s) = C, where n is the fluid-to-wall normal.

BC_C_T_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Specified constant solids phase heat flux, C, in diffusion boundary condition: d(T_s)/dn + Hw (T_s - Tw_s) = C, where n is the fluid-to-wall normal.

BC_HW_X_G(BC, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Species \leq 100$

Gas phase species mass transfer coefficient, Hw, in diffusion boundary condition: d(X_g)/dn + Hw (X_g - Xw_g) = C, where n is the fluid-to-wall normal.

BC_XW_G(BC, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Species \leq 100$

Specified wall gas species mass fraction, Xw, in diffusion boundary condition: d(X_g)/dn + Hw (X_g - Xw_g) = C, where n is the fluid-to-wall normal.

BC_C_X_G(BC, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Species \leq 100$

Specified constant gas species mass flux, C, in diffusion boundary condition: d(X_g)/dn + Hw (X_g - Xw_g) = C, where n is the fluid-to-wall normal.

BC_HW_X_S(BC, PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Solid phase species mass transfer coefficient, Hw, in diffusion boundary condition: d(X_s)/dn + Hw (X_s - Xw_s) = C, where n is the fluid-to-wall normal.

BC_XW_S(BC, PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Specified solids species mass fraction at the wall, Xw_s, in diffusion boundary condition: d(X_s)/dn + Hw (X_s - Xw_s) = C, where n is the fluid-to-wall normal.

BC_C_X_S(BC, PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Specified constant solids species mass flux, C, in diffusion boundary condition: d(X_s)/dn + Hw (X_s - Xw_s) = C, where n is the fluid-to-wall normal.

BC_HW_SCALAR(BC, SCALAR EQ.)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq ScalarEq. \leq 100$

Scalar transfer coefficient, Hw, in diffusion boundary condition: d(Scalar)/dn + Hw (Scalar - ScalarW) = C, where n is the fluid-to-wall normal.

BC_SCALARW(BC, SCALAR EQ.)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq ScalarEq. \leq 100$

Specified scalar value at the wall, ScalarW, in diffusion boundary condition: d(Scalar)/dn + Hw (Scalar - ScalarW) = C, where n is the fluid-to-wall normal.

BC_C_SCALAR(BC, SCALAR EQ.)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq ScalarEq. \leq 100$

Specified constant scalar flux, C, in diffusion boundary condition: d(Scalar)/dn + Hw (Scalar - ScalarW) = C, where n is the fluid-to-wall normal.

BC_EP_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Void fraction at the BC plane.

BC_P_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas pressure at the BC plane.

BC_ROP_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Bulk density of solids phase at the BC plane.

BC_EP_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids volume fraction at the BC plane.

BC_T_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas phase temperature at the BC plane.

BC_T_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase temperature at the BC plane.

BC_THETA_M(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids phase granular temperature at the BC plane.

BC_X_G(BC, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Species \leq 100$

Mass fraction of gas species at the BC plane.

BC_X_S(BC, PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Mass fraction of solids species at the BC plane.

BC_U_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

X-component of gas velocity at the BC plane.

BC_U_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

X-component of solids-phase velocity at the BC plane.

BC_V_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Y-component of gas velocity at the BC plane.

BC_V_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Y-component of solids-phase velocity at the BC plane.

BC_W_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Z-component of gas velocity at the BC plane.

BC_W_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Z-component of solids-phase velocity at the BC plane.

BC_VOLFLOW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas volumetric flow rate through the boundary.

BC_VOLFLOW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids volumetric flow rate through the boundary.

BC_MASSFLOW_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Gas mass flow rate through the boundary.

BC_MASSFLOW_S(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Solids mass flow rate through the boundary.

BC_DT_0(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

The interval at the beginning when the normal velocity at the boundary is equal to BC_Jet_g0. When restarting run, this value and BC_Jet_g0 should be specified such that the transient jet continues correctly. MFIX does not store the jet conditions. For MASS_OUTFLOW boundary conditions, BC_DT_0 is the time period to average and print the outflow rates. The adjustment of velocities to get a specified mass or volumetric flow rate is based on the average outflow rate.

BC_JET_G0(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Value of normal velocity during the initial time interval BC_DT_0.

BC_DT_H(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

The time interval when normal velocity is equal to BC_Jet_gh.

BC_JET_GH(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Value of normal velocity during the interval BC_DT_h.

BC_DT_L(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

The interval when normal velocity is equal to BC_JET_gL.

BC_JET_GL(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Value of normal velocity during the interval BC_DT_L.

BC_SCALAR(BC, SCALAR EQ.)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq ScalarEq. \leq 100$

Boundary value for user-defined scalar equation.

BC_K_TURB_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Boundary value of K for K-Epsilon Equation.

BC_E_TURB_G(BC)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$

Boundary value of Epsilon for K-Epsilon Equation.

BC_PIC_MI_CONST_STATWT(BC, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq BC \leq 500$
- $1 \leq Phase \leq 10$

Flag to specify the constant statistical weight for inflowing computational particles/parcels. Actual number of parcels will be automatically computed.

BC_PO_APPLY_TO_DES(BC)

Data Type: LOGICAL

- $1 \leq BC \leq 500$

Flag to make the PO BC invisible to discrete solids. Set this flag to .FALSE. to remove this BC for discrete solids.

BC_MI_APPLY_TO_DES(BC)

Data Type: LOGICAL

- $1 \leq BC \leq 500$

Flag to make the PO BC invisible to discrete solids. Set this flag to .FALSE. to remove this BC for discrete solids.

8.3.11 Internal Surface

IS_X_W(IS)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$

X coordinate of the west face or edge.

IS_X_E(IS)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$

X coordinate of the east face or edge.

IS_Y_S(IS)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$

Y coordinate of the south face or edge.

IS_Y_N(IS)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$

Y coordinate of the north face or edge.

IS_Z_B(IS)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$

Z coordinate of the bottom face or edge.

IS_Z_T(IS)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$

Z coordinate of the top face or edge.

IS_I_W(IS)

Data Type: INTEGER

- $1 \leq IS \leq 500$

I index of the west-most cell.

IS_I_E(IS)

Data Type: INTEGER

- $1 \leq IS \leq 500$

I index of the east-most cell

IS_J_S(IS)

Data Type: INTEGER

- $1 \leq IS \leq 500$

J index of the south-most cell

IS_J_N(IS)

Data Type: INTEGER

- $1 \leq IS \leq 500$

J index of the north-most cell

IS_K_B(IS)

Data Type: INTEGER

- $1 \leq IS \leq 500$

K index of the bottom-most cell

IS_K_T(IS)

Data Type: INTEGER

- $1 \leq IS \leq 500$

K index of the top-most cell

IS_TYPE(IS)

Data Type: CHARACTER

- $1 \leq IS \leq 500$

Type of internal surface

Table 8.64: Valid Values

Name	Default?	Description
IMPERMEABLE		No gas or solids flow through the surface.
SEMIPERMEABLE		Gas flows through the surface with an additional resistance. Solids velocity through the surface is set to zero or to a user- specified fixed value (i.e., solids momentum equation for this direction is not solved).
X_IMPERMEABLE		No gas or solids flow in the X direction.
X_SEMIPERMEABLE		Gas flows in the Z direction with additional resistance. Solids velocity in the Z direction is set to zero or to a user- specified fixed value (i.e., solids momentum equation for this direction is not solved).
Y_IMPERMEABLE		No gas or solids flow in the Y direction.
Y_SEMIPERMEABLE		Gas flows in the Y direction with additional resistance. Solids velocity in the Y direction is set to zero or to a user- specified fixed value (i.e., solids momentum equation for this direction is not solved).
Z_IMPERMEABLE		No gas or solids flow in the Z direction.
STL		Arbitrary shaped internal surface defined by an STL file. Only visible to DEM particles or PIC parcels. The fluid phase doesn't feel this stl internal surface.

IS_PC(IS, IDX)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$
- $1 \leq IDX \leq 2$

Parameters defining the internal surface. These values need to be specified for semipermeable surfaces only. The thickness used for pressure drop computation is that of the momentum cell (DX_e, DY_n, or DZ_t). To turn off the resistance, use a large value for permeability.

- IDX=1: Permeability [1.0E32]

- IDX=2: Inertial resistance coefficient [0.0]

IS_VEL_S(IS, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq IS \leq 500$
- $1 \leq Phase \leq 10$

Value of fixed solids velocity through semipermeable surfaces.

8.3.12 Point Source

Point sources (PS) are used in place of mass inlets where either the geometry and/or grid resolution prohibit proper boundary condition specification. For example, a point source may be used to model an injector with dimensions smaller than the grid. Point sources may be defined within a single computational cell, along a plane, or as a volume of computational cells.

Point sources introduce mass directly into a computational cell unlike a boundary condition which specifies flow along a cell face. One consequence of this implementation is that point sources are subjected to convection/diffusion forces and may not travel parallel to the specified directional preference. Directional preference is specified with a velocity vector (i.e., `PS_U_g`, `PS_V_g`, etc.), however, directional preference is not required.

Examples showing how to setup point sources can be found in: `legacy_tutorials/point_source_spiral`

PS_X_W(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

X coordinate of the west face or edge.

PS_X_E(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

X coordinate of the east face or edge.

PS_Y_S(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Y coordinate of the south face or edge.

PS_Y_N(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Y coordinate of the north face or edge.

PS_Z_B(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Z coordinate of the bottom face or edge.

PS_Z_T(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Z coordinate of the top face or edge.

PS_I_W(PS)

Data Type: INTEGER

- $1 \leq PS \leq 5000$

I index of the west-most cell.

PS_I_E(PS)

Data Type: INTEGER

- $1 \leq PS \leq 5000$

I index of the east-most cell.

PS_J_S(PS)

Data Type: INTEGER

- $1 \leq PS \leq 5000$

J index of the south-most cell.

PS_J_N(PS)

Data Type: INTEGER

- $1 \leq PS \leq 5000$

J index of the north-most cell.

PS_K_B(PS)

Data Type: INTEGER

- $1 \leq PS \leq 5000$

K index of the bottom-most cell.

PS_K_T(PS)

Data Type: INTEGER

- $1 \leq PS \leq 5000$

K index of the top-most cell.

PS_U_G(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

X-component of incoming gas velocity.

PS_V_G(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Y-component of incoming gas velocity.

PS_W_G(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Z-component of incoming gas velocity.

PS_MASSFLOW_G(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Gas mass flow rate through the point source.

PS_T_G(PS)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$

Temperature of incoming gas.

PS_X_G(PS, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Species \leq 100$

Gas phase incoming species n mass fraction.

PS_U_S(PS, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Phase \leq 10$

X-component of incoming solids velocity.

PS_V_S(PS, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Phase \leq 10$

Y-component of incoming solids velocity.

PS_W_S(PS, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Phase \leq 10$

Z-component of incoming solids velocity.

PS_MASSFLOW_S(PS, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Phase \leq 10$

Solids mass flow rate through the point source.

PS_T_S(PS, PHASE)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Phase \leq 10$

Temperature of incoming solids.

PS_X_S(PS, PHASE, SPECIES)

Data Type: DOUBLE PRECISION

- $1 \leq PS \leq 5000$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Solids phase incoming species n mass fraction.

8.3.13 Output Control

RES_DT

Data Type: DOUBLE PRECISION

required

Time interval at which restart (.res) file is updated.

RES_BACKUP_DT

Data Type: DOUBLE PRECISION

Time interval at which a backup copy of the restart file is created.

RES_BACKUPS

Data Type: INTEGER

The number of backup restart files to retain.

SPX_DT(SP VALUE)

Data Type: DOUBLE PRECISION

- $1 \leq SPValue \leq 11$

Time interval at which .SPX files are updated.

- SP1: void fraction (EP_G)
- SP2: Gas pressure (P_G) and Solids pressure (P_star)
- SP3: Gas velocity (U_G, V_G, W_G)
- SP4: Solids velocity (U_S, V_S, W_S)
- SP5: Solids bulk density (ROP_s)
- SP6: Gas and solids temperature (T_G, T_S)
- SP7: Gas and solids mass fractions (X_G, X_S)
- SP8: Granular temperature (THETA_M)
- SP9: User defined scalars (SCALAR)
- SPA: Reaction Rates (ReactionRates)
- SPB: Turbulence quantities (K_TURB_G, E_TURB_G)

NRR

Data Type: INTEGER

The number of user defined chemical reactions stored in the *.SPA and monitor files.

OUT_DT

Data Type: DOUBLE PRECISION

Time interval at which standard output (.OUT) file is updated. Only run configuration information is written if left undefined. Otherwise all field variables for the entire domain are written in ASCII format to the .OUT file at OUT_DT intervals.

REPORT_SOLID_INVENTORY

Data Type: LOGICAL

required

Option to report solid inventory.

REPORT_SOLID_INVENTORY_DT

Data Type: DOUBLE PRECISION

required

Interval at which solid inventory is reported (seconds).

BREAKDOWN_SOLID_INVENTORY_BY_PHASE

Data Type: LOGICAL

required

Option to breakdown solid inventory by phase (if more than one solids phase).

NLOG

Data Type: INTEGER

Number of time steps between .LOG file updates.

FULL_LOG

Data Type: LOGICAL

Display the residuals on the screen and provide messages about convergence on the screen and in the .LOG file.

RESID_STRING(RESIDUAL INDEX)

Data Type: CHARACTER

- $1 \leq ResidualIndex \leq 8$

Specifies the residuals to display.

Table 8.65: Valid Values

Name	Default?	Description
P0		Gas pressure
PM		Solids phase M pressure
R0		Gas density
RM		Solids phase M density
U0		Gas phase U-velocity
V0		Gas phase V-velocity
W0		Gas phase W-velocity
UM		Solids phase M U-velocity
VM		Solids phase M V-velocity
WM		Solids phase M W-velocity
T0		Gas temperature
TM		Solids phase M temperature
X0NN		Gas phase species NN mass fraction
XMNN		Solids phase M species NN mass fraction
K0		K-Epsilon model residuals

GROUP_RESID

Data Type: LOGICAL

Display residuals by equation.

REPORT_NEG_DENSITY

Data Type: LOGICAL

Provide detailed logging of negative density errors.

Table 8.66: Valid Values

Name	Default?	Description
.FALSE.	◉	Do not log negative density errors.
.TRUE.		Log negative density errors.

REPORT_NEG_SPECIFICHEAT

Data Type: LOGICAL

Provide detailed logging of zero or negative specific heat errors.

Table 8.67: Valid Values

Name	Default?	Description
.FALSE.	◉	Do not log zero or negative specific heat errors.
.TRUE.		Log zero or negative specific heat errors.

REPORT_MASS_BALANCE_DT

Data Type: DOUBLE PRECISION

Frequency to perform an overall species mass balance. Leaving undefined suppresses the mass balance calculations which can slightly extend run time.

PHIP_OUT_JJ

Data Type: LOGICAL

Output the variable specularity coefficient when BC_JJ_M is .TRUE.. The specularity coefficient will be stored in ReactionRates array for post-processing by post_mfix. User needs to set NRR to 1 for this purpose. Be careful with this setting when reacting flow is simulated.

BDIST_IO

Data Type: LOGICAL

Use distributed IO: Each MPI process generates RES/SPx/VTU/VTP files.

BSTART_WITH_ONE_RES

Data Type: LOGICAL

Restart a serial IO run (only one RES file was created) with distributed IO.

MONITOR_X_W(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

X coordinate of the monitor region west face.

MONITOR_X_E(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

X coordinate of the monitor region east face.

MONITOR_Y_S(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Y coordinate of the monitor region south face.

MONITOR_Y_N(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Y coordinate of the monitor region north face.

MONITOR_Z_B(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Z coordinate of the monitor region bottom face.

MONITOR_Z_T(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Z coordinate of the monitor region top face.

MONITOR_NAME(MONITOR)

Data Type: CHARACTER

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Monitor region output file name base.

MONITOR_DT(MONITOR)

Data Type: DOUBLE PRECISION

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Interval to collect monitor data.

MONITOR_TYPE(MONITOR)

Data Type: INTEGER

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Data monitor type.

Table 8.68: Valid Values

Name	Default?	Description
0		Point value
1		Sum over region
2		Minimum over region
3		Maximum over region
4		Arithmetic average over region
5		Standard deviation over region
6		Area-weighted average over surface
7		Flow rate across surface
8		Mass flow rate across surface
9		Mass-weighted average over surface
10		Volumetric flow rate over surface
11		Volume integral
12		Volume-weighted average
13		Mass-weighted volume integral
14		Mass-weighted volume average
101		Sum over region
102		Minimum over region
103		Maximum over region
104		Arithmetic average over region
105		Standard deviation over region
106		Mass-weighted average over region
107		Volume-weighted average over region
108		Flow rate across surface
109		Mass-weighted flow rate across surface
110		Volume-weighted flow rate across surface

MONITOR_EP_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write the void fraction in region.

MONITOR_RO_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write the gas density in region.

MONITOR_P_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write the gas pressure in region.

MONITOR_U_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write x-component of gas velocity vector in region.

MONITOR_V_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write y-component of gas velocity vector in region.

MONITOR_W_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write z-component of gas velocity vector in region.

MONITOR_T_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write gas temperature in region.

MONITOR_MW_MIX_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write gas mixture molecular weight in region.

MONITOR_X_G(MONITOR, SPECIES)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Species \leq 100$

Write gas phase species mass fraction in region.

MONITOR_Y_G(MONITOR, SPECIES)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Species \leq 100$

Write gas phase species molar fraction in region.

MONITOR_K_TURB_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write turbulent kinetic energy in region.

MONITOR_E_TURB_G(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write turbulent dissipation rate in region.

MONITOR_U_S(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write x-component of solids velocity vector in region.

MONITOR_V_S(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write y-component of solids velocity vector in region.

MONITOR_W_S(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write z-component of solids velocity vector in region.

MONITOR_ROP_S(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write solids bulk density in region.

MONITOR_RO_S(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write solids material density in region.

MONITOR_P_STAR(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write the solids pressure in region.

MONITOR_T_S(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write solids temperature in region.

MONITOR_X_S(MONITOR, PHASE, SPECIES)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$
- $1 \leq Species \leq 100$

Write solids phase species mass fraction in region.

MONITOR_THETA_M(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Write granular temperature in region.

MONITOR_SCALAR(MONITOR, SCALAR EQ.)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq ScalarEq. \leq 100$

Write scalar in region.

MONITOR_RRATE(MONITOR, RATE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq RATE \leq MONITOR_NRRMAX$

Write reaction rate in region.

MONITOR_PART_PHASE(MONITOR, PHASE)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Phase \leq 10$

Option to save or not save particles belonging to solids phases in region.

MONITOR_RADIUS(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle radius in region.

MONITOR_PMASS(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle mass in region.

MONITOR_PVOL(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle volume in region.

MONITOR_RO_P(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle density in region.

MONITOR_VEL_X(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle x-axis translational velocity in region.

MONITOR_VEL_Y(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle y-axis translational velocity in region.

MONITOR_VEL_Z(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle z-axis translational velocity in region.

MONITOR_ROT_X(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle x-axis rotational velocity in region.

MONITOR_ROT_Y(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle y-axis rotational velocity in region.

MONITOR_ROT_Z(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle z-axis rotational velocity in region.

MONITOR_T_P(MONITOR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$

Write particle temperature in region.

MONITOR_X_P(MONITOR, SPECIES)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq Species \leq 100$

Write particle species mass fraction in region.

MONITOR_DES_USR_VAR(MONITOR, USR_VAR)

Data Type: LOGICAL

- $1 \leq Monitor \leq DIMENSION_MONITOR$
- $1 \leq USR_VAR \leq 100$

Write des_usr_var in region.

WRITE_VTK_FILES

Data Type: LOGICAL

Write VTK files at regular intervals.

Table 8.69: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Do not write VTK files. If there are cut cells, they will not be displayed from the usual .res file
`.TRUE.`		Valid only if Cartesian_grid = .TRUE.

TIME_DEPENDENT_FILENAME

Data Type: LOGICAL

Use time-dependent VTK file names

Table 8.70: Valid Values

Name	Default?	Description
`.FALSE.`		The VTK file overwrites the previous file (recommended for steady-state computation).
`.TRUE.`	◉	A sequential integer is appended to the VTK filenames as they are written to create a series of files (recommended for transient computation).

VTK_DT(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

Time interval (expressed in seconds of simulation time) at which VTK files are written.

VTK_DBG_FILE(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

When VTK_DBG_FILE is .TRUE., the VTK region file is only written when the subroutine WRITE_DBG_VTU_AND_VTP_FILES is called, typically in a UDF.

VTK_VAR(IDX)

Data Type: INTEGER

- $1 \leq IDX \leq 20$

List of variables written in the VTK files.

Table 8.71: Valid Values

Name	Default?	Description
1		Void fraction (EP_g)
2		Gas pressure, solids pressure (P_g, P_star)
3		Gas velocity (U_g, V_g, W_g)
4		Solids velocity (U_s, V_s, W_s)
5		Solids density (ROP_s)
6		Gas and solids temperature (T_g, T_s)
7		Gas and solids mass fractions (X_g, X_s)
8		Granular temperature (Theta_m)
9		Scalar
10		Reaction rates
11		K and Epsilon
12		Vorticity magnitude and lambda_2
100		Grid Partition
101		Boundary Condition ID
102		Distance to wall
103		DEM facet count
104		DEM Neighboring facets
999		Cell IJK index
1000		Cut face normal vector

VTK_X_W(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

X coordinate of the VTK region west face.

VTK_X_E(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

X coordinate of the VTK region east face.

VTK_Y_S(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

Y coordinate of the VTK region south face.

VTK_Y_N(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

Y coordinate of the VTK region north face.

VTK_Z_B(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

Z coordinate of the VTK region bottom face.

VTK_Z_T(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

Z coordinate of the VTK region top face.

VTK_FILEBASE(VTK)

Data Type: CHARACTER

- $1 \leq VTK \leq 100$

VTK region output file name base.

VTK_DATA(VTK)

Data Type: CHARACTER

- $1 \leq VTK \leq 100$

Type of data to write in the VTK file.

Table 8.72: Valid Values

Name	Default?	Description
C	◉	Cell data (VTU file).
P		Particle data (VTP file).
G		Geometry data (STL file).

VTK_GEO(VTK, GEO)

Data Type: CHARACTER

- $1 \leq VTK \leq 100$
- $1 \leq GEO \leq VTK_GEO_MAX$

STL file to convert into VTU file.

VTK_SELECT_MODE(VTK)

Data Type: CHARACTER

- $1 \leq VTK \leq 100$

Particle selection mode in a VTK region.

Table 8.73: Valid Values

Name	Default?	Description
C	◉	Select particles with centers inside the VTK region.
P		Select particles that are entirely inside the VTK region.
I		Select particles that are inside or intersect the edges of the VTK region.

VTK_NXS(VTK)

Data Type: INTEGER

- $1 \leq VTK \leq 100$

Specifies the number of subdivisions in the x-axial direction to decompose a VTK region. Leave undefined to export the full region. (Slice a volume into planes; cut a plane into lines; or break a line into points.)

VTK_NYS(VTK)

Data Type: INTEGER

- $1 \leq VTK \leq 100$

Specifies the number of subdivisions in the y-axial direction to decompose a VTK region. Leave undefined to export the full region. (Slice a volume into planes; cut a plane into lines; or break a line into points.)

VTK_NZS(VTK)

Data Type: INTEGER

- $1 \leq VTK \leq 100$

Specifies the number of subdivisions in the z-axial direction to decompose a VTK region. Leave undefined to export the full region. (Slice a volume into planes; cut a plane into lines; or break a line into points.)

VTK_SLICE_TOL(VTK)

Data Type: DOUBLE PRECISION

- $1 \leq VTK \leq 100$

Tolerance to detect particles in a VTK region's slice.

VTK_EP_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write the void fraction in region.

VTK_P_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write the gas pressure in region.

VTK_P_STAR(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write the solids pressure in region (all phases).

VTK_VEL_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write the gas velocity vector in region.

VTK_U_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write x-component of gas velocity vector in region.

VTK_V_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write y-component of gas velocity vector in region.

VTK_W_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write z-component of gas velocity vector in region.

VTK_VEL_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write solids velocity vector in region.

VTK_U_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write x-component of solids velocity vector in region.

VTK_V_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write y-component of solids velocity vector in region.

VTK_W_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write z-component of solids velocity vector in region.

VTK_ROP_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write solids bulk density in region.

VTK_RO_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write solids density in region.

VTK_EP_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write solids volume fraction in region.

VTK_T_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write gas temperature in region.

VTK_T_S(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write solids temperature in region.

VTK_MW_MIX_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write gas phase mixture molecular weight in region.

VTK_X_G(VTK, GAS SPECIES)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq GasSpecies \leq DIM_N_g$

Write gas phase species mass fraction in region.

VTK_Y_G(VTK, GAS SPECIES)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq GasSpecies \leq DIM_N_g$

Write gas phase species molar fraction in region.

VTK_X_S(VTK, PHASE, SOLIDS SPECIES)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$
- $1 \leq SolidsSpecies \leq DIM_N_s$

Write solids phase species mass fraction in region.

VTK_THETA_M(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Write granular temperature in region.

VTK_SCALAR(VTK, SCALAR)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Scalar \leq DIM_scalar$

Write scalar in region.

VTK_RRATE(VTK, RATE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq rate \leq VTK_nRRmax$

Write reaction rate in region.

VTK_RRATE_LABEL(VTK, RATE)

Data Type: CHARACTER

- $1 \leq VTK \leq 100$
- $1 \leq rate \leq VTK_nRRmax$

Reaction rate custom label in region.

VTK_K_TURB_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write turbulent kinetic energy in region.

VTK_E_TURB_G(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write turbulent dissipation rate in region.

VTK_VORTICITY(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write vorticity magnitude in region.

VTK_LAMBDA_2(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write lambda_2 in region.

VTK_PARTITION(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write void grid partition in region.

VTK_BC_ID(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write boundary condition ID in region.

VTK_DWALL(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write wall distance in region.

VTK_FACET_COUNT_DES(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write STL facet count for DES in region.

VTK_NB_FACET_DES(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write neighboring facets in region.

VTK_IJK(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write cell IJK index in region.

VTK_NORMAL(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write cut face normal vector in region.

VTK_DEBUG(VTK, IDX)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq IDX \leq 15$

Write debug variable in region.

VTK_PART_DIAMETER(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle diameter in region.

VTK_PART_PHYSICAL_DIAMETER(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write Coarse Grain physical particle diameter in region.

VTK_PART_CGP_STAT_WT(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write Coarse Grain particle statistical weight in region.

VTK_PART_VEL(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle velocity in region.

VTK_PART_ANGULAR_VEL(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle angular velocity in region.

VTK_PART_ORIENTATION(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle orientation in region.

VTK_PART_USR_VAR(VTK, USR_VAR)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq USR_VAR \leq VTK_PART_USRmax$

Write particle user-defined variable in region.

VTK_PART_TEMP(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle temperature in region.

VTK_PART_X_S(VTK, SPECIES)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Species \leq 100$

Write particle species mass fraction in region.

VTK_PART_DENSITY(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle density in region.

VTK_PART_COHESION(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle cohesion in region.

VTK_PART_RANK(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle rank in region.

VTK_PART_ID(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle ID in region.

VTK_PART_PHASE_ID(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write particle phase ID in region.

VTK_PART_PHASE(VTK, PHASE)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$
- $1 \leq Phase \leq 10$

Option to save or not save particles belonging to solids phases in region.

VTK_CUTCELL_ONLY(VTK)

Data Type: LOGICAL

- $1 \leq VTK \leq 100$

Write cut-cell data only in region.

FRAME(VTK)

Data Type: INTEGER

- $1 \leq VTK \leq 100$

Starting Index appended to VTU regions.

VTU_DIR

Data Type: CHARACTER

Directory where VTK files are stored. By default, VTK files are written in the run directory.

DES_REPORT_MASS_INTERP

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Reports mass based on Lagrangian particles and continuum representation. Useful to ensure mass conservation between Lagrangian and continuum representations. Recommended use for debugging purposes.

PRINT_DES_DATA

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Allows writing of discrete particle data to output files. Relevant to both granular and coupled simulations.

VTP_DIR

Data Type: CHARACTER

Directory where particle vtp files are stored. The files are written in the run directory by default.

DES_OUTPUT_TYPE

Data Type: CHARACTER

Applies to Solids Model(s): **DEM**

Output file format for DES data.

Table 8.74: Valid Values

Name	Default?	Description
`PARAVIEW`	◉	ParaView formatted files (.vtp)
`TECPLOT`		Tecplot formatted files (.dat)

DEBUG_DES

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Runtime flag to generate debugging information. Additional data for FOCUS_PARTICLE is saved.

FOCUS_PARTICLE

Data Type: INTEGER

Applies to Solids Model(s): **DEM**

Specify particle number for particle level debugging details.

WRITE_PART_OUT

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Option to write a particle_output.dat file at the end of a simulation. This file has the same format as the particle_input.dat file (version 2.0 and up). After the simulation, the file can be renamed as particle_input.dat and used as initial condition for another simulation. For example, a first granular DEM simulation can be run fairly quickly to settle a bed of particles and the data saved in this file can be used to start a new coupled simulation. Default value is .False. (do not write the file).

PART_OUT_ZERO_VEL

Data Type: LOGICAL

Applies to Solids Model(s): **DEM**

Option to reset the velocity to zero in the particle_output.dat file. Default value is .False. (do not reset the velocity to zero).

8.3.14 UDF Control

CALL_USR

Data Type: LOGICAL

Flag to enable user-defined subroutines: USR0, USR1, USR2, USR3, USR0_DES, USR1_DES, USR2_DES, USR3_DES, USR4_DES.

Table 8.75: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined subroutines.
`.FALSE.`	◉	Do NOT call user-defined subroutines.

CALL_USR_SOURCE(EQUATION ID NUMBER)

Data Type: LOGICAL

- $1 \leq EquationIDNumber \leq 10$

Flag to enable user_defined subroutine, usr_source, for calculating source terms in the indicated equation.

Table 8.76: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined source.
`.FALSE.`	◉	MFIX default: No additional source.

USR_ROG

Data Type: LOGICAL

Flag to use the User Defined Function, USR_PROP_ROg, in model/usr_prop.f for calculating the gas phase density, RO_g.

Table 8.77: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_CPG

Data Type: LOGICAL

Flag to use the User Defined Function, USR_PROP_CPg, in model/usr_prop.f for calculating the gas phase constant pressure specific heat, C_pg.

Table 8.78: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_KG

Data Type: LOGICAL

Flag to use the User Defined Function, USR_PROP_Kg, in model/usr_prop.f for calculating the gas phase conductivity, K_g.

Table 8.79: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_DIFG

Data Type: LOGICAL

Flag to use the User Defined Function, USR_PROP_Difg, in model/usr_prop.f for calculating the gas phase diffusivity.

Table 8.80: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_MUG

Data Type: LOGICAL

Flag to use the User Defined Function, USR_PROP_Mug, in model/usr_prop.f for calculating the gas phase viscosity, Mu_g.

Table 8.81: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_ROS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_ROs, in model/usr_prop.f for calculating the solids phase density, RO_s.

Table 8.82: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_CPS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_CPs, in model/usr_prop.f for calculating the solids phase constant pressure specific heat, C_ps.

Table 8.83: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_KS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_Ks, in model/usr_prop.f for calculating the solids phase conductivity, K_s.

Table 8.84: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_DIFS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_Difs, in model/usr_prop.f for calculating the solids phase diffusivity, Dif_s.

Table 8.85: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_MUS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_Mus, in model/usr_prop.f for calculating the solids phase viscosity, Mu_s; second viscosity, lambda_s; and pressure, P_s.

Table 8.86: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_GAMA(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_Gama, in model/usr_prop.f for calculating the gas-solids phase heat transfer coefficient, Gama_gs.

Table 8.87: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_FGS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq 10$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_Fgs, in model/usr_prop.f for calculating the gas-solids phase drag coefficient due to relative velocity differences, F_gs. Currently unavailable.

Table 8.88: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

USR_FSS(PHASE)

Data Type: LOGICAL

- $1 \leq Phase \leq DIM_LM$

Applies to Solids Model(s): **TFM**

Flag to use the User Defined Function, USR_PROP_Fss, in model/usr_prop.f for calculating the solids-solids phase drag coefficient due to relative velocity differences, F_ss. Currently unavailable.

Table 8.89: Valid Values

Name	Default?	Description
`.TRUE.`		Call user-defined function.
`.FALSE.`	◉	Use MFIX default calculation.

C(USER-DEFINED ID)

Data Type: DOUBLE PRECISION

- $1 \leq User - definedID \leq DIMENSION_C$

User defined constants.

C_NAME(USER-DEFINED ID)

Data Type: CHARACTER

- $1 \leq User - definedID \leq DIMENSION_C$

Name of user-defined constant. (20 character max)

USR_DT(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

Interval at which subroutine write_usr1 is called.

USR_X_W(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

UDF hook: x coordinate of the west face or edge.

USR_X_E(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

UDF hook: x coordinate of the east face or edge.

USR_Y_S(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

UDF hook: y coordinate of the south face or edge.

USR_Y_N(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

UDF hook: y coordinate of the north face or edge.

USR_Z_B(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

UDF hook: z coordinate of the bottom face or edge.

USR_Z_T(USR)

Data Type: DOUBLE PRECISION

- $1 \leq USR \leq 5$

UDF hook: z coordinate of the top face or edge.

USR_I_W(USR)

Data Type: INTEGER

- $1 \leq USR \leq 5$

UDF hook: i index of the west-most cell.

USR_I_E(USR)

Data Type: INTEGER

- $1 \leq USR \leq 5$

UDF hook: i index of the east-most cell.

USR_J_S(USR)

Data Type: INTEGER

- $1 \leq USR \leq 5$

UDF hook: j index of the south-most cell.

USR_J_N(USR)

Data Type: INTEGER

- $1 \leq USR \leq 5$

UDF hook: j index of the north-most cell.

USR_K_B(USR)

Data Type: INTEGER

- $1 \leq USR \leq 5$

UDF hook: k index of the bottom-most cell.

USR_K_T(USR)

Data Type: INTEGER

- $1 \leq USR \leq 5$

UDF hook: k index of the top-most cell.

USR_TYPE(USR)

Data Type: CHARACTER

- $1 \leq USR \leq 5$

UDF hook: Type of user-defined output: Binary or ASCII.

USR_VAR(USR)

Data Type: CHARACTER

- $1 \leq USR \leq 5$

UDF hook: Variable to be written in the user-defined output files.

USR_FORMAT(USR)

Data Type: CHARACTER

- $1 \leq USR \leq 5$

UDF hook: Format for writing user-defined (ASCII) output file.

USR_EXT(USR)

Data Type: CHARACTER

- $1 \leq USR \leq 5$

UDF hook: File extension for the user-defined output.

DES_USR_VAR_SIZE

Data Type: INTEGER

Applies to Solids Model(s): **DEM**

Defines the size of the particle-based user variable: DES_USR_VAR(SIZE, PARTICLES). Information in this array follows the particle throughout a simulation.

8.3.15 Chemical Reactions

See *Setting up Chemical Reaction Cases* for information on using these keywords.

STIFF_CHEMISTRY

Data Type: LOGICAL

Flag to use stiff chemistry solver (Direct Integration).

STIFF_CHEM_MAX_STEPS

Data Type: INTEGER

Maximum number of internal steps ODEPACK may use to integrate over the time interval. Leaving this value unspecified permits an unlimited number of steps. The stiff solver reports the number of cells that exceed the number of steps as 'incomplete'.

USE_RRATES

Data Type: LOGICAL

Flag to use legacy chemical reaction UDFs.

SPECIES_NAME(PHASE)

Data Type: CHARACTER

- $0 \leq Phase \leq 10$

Names of gas and solids phase species as it appears in the materials database. The first NMAX(0) are the names of gas species. The next NMAX(1) are the names of solids phase-1 species, etc.

NMAX(PHASE)

Data Type: INTEGER

- $0 \leq Phase \leq 10$

Number of species in phase m. Note that the gas phase is indicated as m=0.

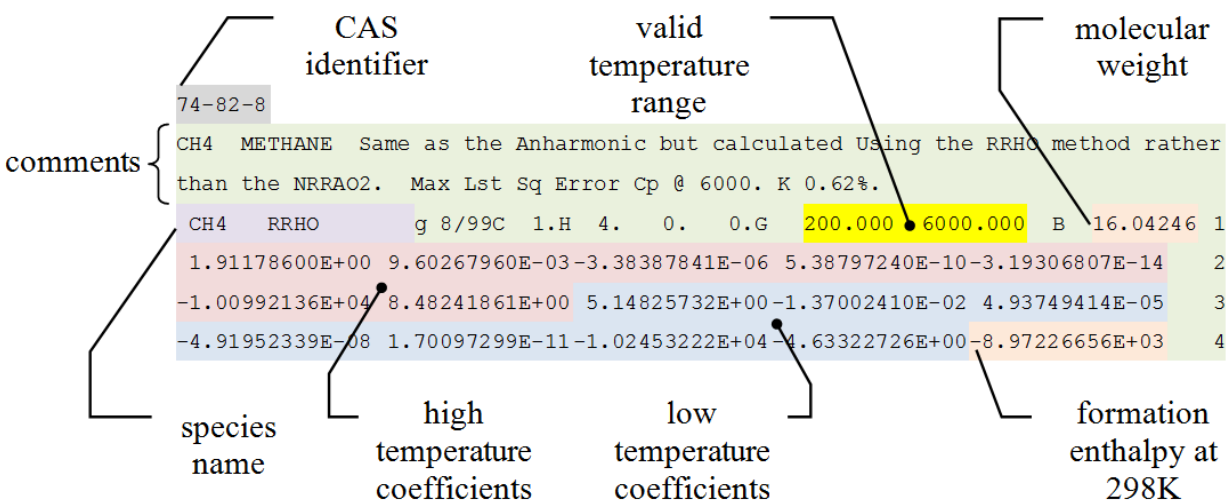
8.3.16 Thermochemical Properties

The directory `model/thermochemical` contains the database of Burcat and Ruscic (2005) and routines for reading the database. With linkage to this database the users need not manually enter data for molecular weight, specific heat, and heats of reactions. Instead the users only need to enter the names of the species (keyword SPECIES_g and SPECIES_s) in the data file. If such information is already provided in either the data file or a BURCAT.THR file in the run directory then MFiX will not reference the database. That is, MFiX reads the necessary thermo-chemical data from files in the following order:

1. `mfix.dat`
2. `BURCAT.THR` file in the run directory
3. `model/thermochemical/BURCAT.THR`

The species names are case sensitive and should match the names in `BURCAT.THR` exactly; alternatively aliases can be defined for common species, such as O2, in read_therm.f. See `tests/thermo` for a sample case that accesses the database. The format of `BURCAT.THR` file resembles CHEMKIN format, but with several notable differences. To include thermochemical data in the mfix.dat file then this information must start below a line that starts with THERMO DATA.

Example dataset from BURCAT.THR with notations:



Each entry in the database starts with a unique CAS identifier (74-82-8) for the species, followed by several lines of comments highlighted in green. The data section starts with the species name in columns 1-18 (CH4 RRHO). Common species names may be followed by strings (RRHO) that identify the method used to determine the coefficients. Additional information follows the species name. The numbers toward the end of the line are the temperature limits (200.000 6000.000) in degrees Kelvin where the property calculation is valid and the molecular weight (16.04246). Unlike CHEMKIN the common temperature for the high and low temperature branches are not recorded; it is always 1000 K. The next three lines give the fourteen coefficients (seven coefficients each for the high and low temperature branches) and the formation enthalpy at 298 K (which is also not included in CHEMKIN format). All the coefficients and the enthalpy of formation

are normalized with the gas constant R (cal/mol/K). The low temperature coefficients (a_L) should be used for temperatures in the range Tlow to 1000K and the high temperature coefficients (a_H) should be used for temperatures in the range 1000K to Thigh. The coefficients are stored in a fixed format (E15.0) as follows:

Table 8.90: Coefficients

a_H^1	a_H^2	a_H^3	a_H^4	a_H^4
a_H^6	a_H^7	a_L^1	a_L^2	a_L^3
a_L^4	a_L^4	a_L^4	a_L^4	ΔH_f°

where ΔH_f° is the formation enthalpy at 298K.

The normalized specific heat is given by

$$\frac{C_p}{R} = a_1 + a_2 T + a_3 T^2 + a_4 T^3 + a_5 T^4.$$

Additional database comments:

- A number of species in the database have a lower temperature limit of 300K which is 2 degrees above the reference temperature (298 K) used for formation enthalpy calculation. For those species MFiX relaxes the lower limit for Cp calculations to 298 K to enable heat of reaction calculation (see read_database.f).
- The database reader is set up such that the database is read only if necessary.
- For additional details see the Burcat and Ruscic (2005) report located in the thermo-chemical subdirectory, `model/thermochemical/intro.pdf`. If you include thermochemical properties in mfix.dat all keywords defined below the line that starts with THERMO DATA will be ignored!

8.3.17 Parallelization Control

Parallel performance depends on several things, and one has to evaluate different options before choosing the right strategy for any problem. For example if the J-direction is the strongest coupled direction, the preconditioning for the linear solver will be poor if there is decomposition in that direction. However, since decomposing in all the directions reduces the processor grid surface area, the communication cost will be less for the same computational grid. The preconditioners are chosen with the keyword `LEQ_PC`. In addition to LINE relaxation, one can choose the "DIAG" or "NONE" preconditioner that reduces inter-processor communications but this choice will increase the number of linear equation solver iterations. The DIAG and NONE choices for preconditioners may be appropriate for all equations except the continuity (or pressure and volume fraction correction) equations. The parallel performance is greatly dependent on the choices stated here, and some trial and error may be required to determine the right combination of decomposition direction with the choice of preconditioners to get the best performance in production runs.

`NODESI * NODESJ * NODESK` must be the same as the number of processors specified using `mpirun` (or equivalent command). Otherwise the code will return with an error.

NODESI

Data Type: INTEGER

Number of grid blocks in x-direction.

NODESJ

Data Type: INTEGER

Number of grid blocks in y-direction.

NODESK

Data Type: INTEGER

Number of grid blocks in z-direction.

DLB_NODESI(LAYOUT)

Data Type: INTEGER

- $1 \leq Layout \leq 100$

List of grid blocks in x-direction used in Dynamic Load balance (DLB). The DLB will test each partition layout defined by DLB_NODESI, DLB_NODESJ, DLB_NODESK, and choose the one providing the best load balance. For each layout, the product DLB_NODESIxDLB_NODESJxDLB_NODESK must match the number of cores used in the DMP run.

Example: To test two 80-cores layouts with 4x5x4 and 2x20x2 partitions, define DLB_NODESI(1)=4, DLB_NODESJ(1)=5, DLB_NODESK(1)=4, and DLB_NODESI(2)=2, DLB_NODESJ(2)=20, DLB_NODESK(2)=2.

DLB_NODESJ(LAYOUT)

Data Type: INTEGER

- $1 \leq Layout \leq 100$

List of grid blocks in y-direction used in Dynamic Load balance (DLB). The DLB will test each partition layout defined by DLB_NODESI, DLB_NODESJ, DLB_NODESK, and choose the one providing the best load balance. For each layout, the product DLB_NODESIxDLB_NODESJxDLB_NODESK must match the number of cores used in the DMP run.

Example: To test two 80-cores layouts with 4x5x4 and 2x20x2 partitions, define DLB_NODESI(1)=4, DLB_NODESJ(1)=5, DLB_NODESK(1)=4, and DLB_NODESI(2)=2, DLB_NODESJ(2)=20, DLB_NODESK(2)=2.

DLB_NODESK(LAYOUT)

Data Type: INTEGER

- $1 \leq Layout \leq 100$

List of grid blocks in z-direction used in Dynamic Load balance (DLB). The DLB will test each partition layout defined by DLB_NODESI, DLB_NODESJ, DLB_NODESK, and choose the one providing the best load balance. For each layout, the product DLB_NODESIxDLB_NODESJxDLB_NODESK must match the number of cores used in the DMP run.

Example: To test two 80-cores layouts with 4x5x4 and 2x20x2 partitions, define DLB_NODESI(1)=4, DLB_NODESJ(1)=5, DLB_NODESK(1)=4, and DLB_NODESI(2)=2, DLB_NODESJ(2)=20, DLB_NODESK(2)=2.

SOLVER_STATISTICS

Data Type: LOGICAL

Print out additional statistics for parallel runs

DEBUG_RESID

Data Type: LOGICAL

Group residuals to reduce global collectives.

ENABLE_DMP_LOG

Data Type: LOGICAL

All ranks write error messages.

DBGPRN_LAYOUT

Data Type: LOGICAL

Print the index layout for debugging.

8.3.18 Batch Queue

MFiX can be used on systems where code execution is controlled through a batch queue submission system instead of interactive or background job type methods as shown in the previous section. Usually, the user specifies the wall clock time duration of the job, and the batch queuing system prioritizes incoming jobs based on their resource allocation requests. In order for MFiX to avoid abrupt and abnormal termination at the end of the batch job session, several keywords need to be entered in mfix.dat. Controlled and clean termination in environments with batch queue is important as the system may terminate the batch job while MFiX is writing out `*.SP` files, which may corrupt the files or cause loss of data.

For this purpose, MFiX checks whether the user-specified termination criteria is reached at the beginning of each time step. However, to avoid performance bottlenecks on small systems where the user is running jobs without a batch queue, this feature is disabled by default. In order to enable this feature the following block of keywords need to be entered into `mfix.dat`.

```
CHK_BATCHQ_END = .TRUE.  ! Enable the controlled termination feature
BATCH_WALLCLOCK = 3600.0 ! Specify the total wall clock duration
                         ! of your job in seconds
TERM_BUFFER = 300.0      ! Specify a buffer time to start
                         ! clean termination of MFiX
```

Setting `CHK_BATCH_END = .TRUE.` in `mfix.dat` will enable the checking of the termination criteria at the beginning of each time step. In the above example, the user has set the total wall clock time for the duration of the batch session to 1 hour (this is specified in seconds in mfix.dat) and a buffer of 300 seconds has been set so that MFiX has sufficient time to terminate cleanly by writing out all `*.SP` and `*.RES` files before the batch session terminates. The duration of the buffer is critical for simulations with large files. MFiX will check if `elapsed time >= (BATCH_WALLCLOCK - TERM_BUFFER)` to start clean termination.

Another way to gracefully terminate MFiX as soon as possible is to create an empty file named MFIX.STOP (filename all uppercase) in the working directory where MFiX runs. At the beginning of each time step if the MFIX.STOP file is

detected, then MFiX will terminate gracefully by saving `*.RES` files. CHK_BATCHQ_END flag must be set to .TRUE. in order to activate this feature.

The following terminal command can be used to gracefully terminate MFiX:

```
> touch MFIX.STOP

Remember to erase the file once MFiX terminates, otherwise the next time MFiX is run
in the same directory it will terminate immediately.
```

```
> rm -f -r ./MFIX.STOP
```

CHK_BATCHQ_END

Data Type: LOGICAL

Enables controlled termination feature when running under batch queue system to force MFiX to cleanly terminate before the end of wall clock allocated in the batch session.

BATCH_WALLCLOCK

Data Type: DOUBLE PRECISION

Total wall-clock duration of the job, in seconds.

TERM_BUFFER

Data Type: DOUBLE PRECISION

Buffer time specified to allow MFiX to write out the files and cleanly terminate before queue wall clock time limit is reached such that (BATCH_WALLCLOCK-TERM_BUFFER) is less than then batch queue wall clock time limit, in seconds.

8.3.19 Cartesian Grid

CARTESIAN_GRID

Data Type: LOGICAL

Activate Cartesian grid cut cell technique.

Table 8.91: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Do not use Cartesian grid cut cell technique.
`.TRUE.`		Use Cartesian grid cut cell technique. one of the following methods must be used to define the geometry:

N_QUADRIC

Data Type: INTEGER

Number of quadric surfaces defining the boundaries (<=100).

USE_STL

Data Type: LOGICAL

Use STL file to describe geometry.

Table 8.92: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Do not use STL file.
`.TRUE.`		Read triangulated geometry (for 3d geometry only) from geometry_####.stl.

USE_MSH

Data Type: LOGICAL

Use .msh file to describe geometry.

Table 8.93: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Do not use .msh file.
`.TRUE.`		Read geometry (for 3d geometry only) from geometry.msh.

USE_POLYGON

Data Type: LOGICAL

Use polygons to describe geometry.

Table 8.94: Valid Values

Name	Default?	Description
`.FALSE.`	◉	Do not use polygons.
`.TRUE.`		Read polygon data (for 2d geometry only) from poly.dat.

N_USR_DEF

Data Type: INTEGER

Number of user-defined functions (currently limited to 0 or 1). If set to 1, the geometry is defined in the user subroutine eval_usr_fct.f.

Table 8.95: Valid Values

Name	Default?	Description
`0`	◉	Do not use user-defined function
`1`		Use one user-defined function

QUADRIC_FORM(QUADRIC ID)

Data Type: CHARACTER

- $1 \leq QuadricID \leq 500$

Form of the quadric surface equation.

Table 8.96: Valid Values

Name	Default?	Description
normal	◉	Use normal form, as defined in equation (1). The LAMDBAs and D must be defined
plane		Plane. Needs to define N_X,N_Y,N_Z (unit normal vector pointing away from fluid cells).
x_cyl_int		Cylinder aligned with x-axis, internal flow. Needs to define RADIUS(QID).
x_cyl_ext		Cylinder aligned with x-axis, external flow. Needs to define RADIUS(QID).
y_cyl_int		Cylinder aligned with y-axis, internal flow. Needs to define RADIUS(QID).
y_cyl_ext		Cylinder aligned with y-axis, external flow. Needs to define RADIUS(QID).
z_cyl_int		Cylinder aligned with z-axis, internal flow. Needs to define RADIUS(QID).
z_cyl_ext		Cylinder aligned with z-axis, external flow. Needs to define RADIUS(QID).
x_cone		Cone aligned with x-axis, internal flow. Needs to define HALF_ANGLE(QID).
y_cone		Cone aligned with y-axis, internal flow. Needs to define HALF_ANGLE(QID).
z_cone		Cone aligned with z-axis, internal flow. Needs to define HALF_ANGLE(QID).
sphere_int		Sphere, internal flow. Needs to define RADIUS(QID).
sphere_ext		Sphere, external flow. Needs to define RADIUS(QID).
C2C		Cylinder-to-cylinder conical junction, internal flow. Needs to be defined between two cylinders.
Torus_int		Torus, internal flow. Needs to define TORUS_R1(QID) and TORUS_R2(QID).A torus is not a quadric surface but is defined as a basic shape.
Torus_ext		Torus, external flow. Needs to define TORUS_R1(QID) and TORUS_R2(QID).
Y_UCOIL_EXT		Pair of parallel cylinders (y-direction), capped at both ends by a cylinder at 90 degree angle to create a U-shaped coil. Needs UCOIL_R1, UCOIL_R2, UCOIL_Y1, UCOIL_Y2.
XY_BEND_INT		Bend between two cylinders in the XY plane, Needs BEND_R1,BEND_R2,BEND_THETA1,BEND_THETA2.
Y_C2C_INT		connects two vertical cylinders by a conical section. Needs C2C_R1,C2C_R2,C2C_Y1,C2C_Y2.
REACTOR1		Reactor, made of two vertical cylinders, connected by a conical section. Each cylinder is rounded and closed by a conical cap. Needs REACTOR1_R1,REACTOR1_R2,REACTOR1_Y1,REACTOR1_Y2, REACTOR1_YR1,REACTOR1_YR2,REACTOR1_RR1,REACTOR1_RR2, REACTOR1_THETA1,REACTOR1_THETA2.

QUADRIC_SCALE

Data Type: DOUBLE PRECISION

Scaling factor, applied to all quadric geometry parameters. Must be a positive number.

LAMBDA_X(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Coefficient LAMBDA_X in equation (1) ('NORMAL' form) or x-component of normal vector defining plane in equation (5) ('DEGENERATE' form).

LAMBDA_Y(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Coefficient LAMBDA_Y in equation (1) ('NORMAL' form) or y-component of normal vector defining plane in equation (5) ('DEGENERATE' form).

LAMBDA_Z(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Coefficient LAMBDA_Z in equation (1) ('NORMAL' form) or z-component of normal vector defining plane in equation (5) ('DEGENERATE' form).

DQUADRIC(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Coefficient D in equation (1).

THETA_X(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Rotation angle with respect to x-axis (degrees).

THETA_Y(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Rotation angle with respect to y-axis (degrees).

THETA_Z(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Rotation angle with respect to z-axis (degrees).

RADIUS(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Cylinder radius (used when QUADRIC_FORM = *_CYL_***)

HALF_ANGLE(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Cone half angle, expressed in degrees (used when QUADRIC_FORM = *_CONE)

TORUS_R1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Torus Radius 1 (used when QUADRIC_FORM = TORUS_*), R1>R2 for a ring.

TORUS_R2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Torus Radius 2 (used when QUADRIC_FORM = TORUS_*), R1>R2 for a ring.

UCOIL_R1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

U-shaped coil Radius 1 (used when QUADRIC_FORM = UCOIL*), UCOIL_R1>UCOIL_R2.

UCOIL_R2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

U-shaped coil Radius 2 (used when QUADRIC_FORM = UCOIL*), UCOIL_R1>UCOIL_R2.

UCOIL_Y1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

U-shaped coil ymax (used when QUADRIC_FORM = UCOIL*), UCOIL_Y2>UCOIL_Y1.

UCOIL_Y2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

U-shaped coil ymin (used when QUADRIC_FORM = UCOIL*), UCOIL_Y2>UCOIL_Y1.

BEND_R1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Bend Radius 1 (used when QUADRIC_FORM = BEND*), BEND_R1>BEND_R2.

BEND_R2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Bend Radius 2 (used when QUADRIC_FORM = BEND*), BEND_R1>BEND_R2.

BEND_THETA1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Bend start angle, in degrees (used when QUADRIC_FORM = BEND*).

BEND_THETA2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Bend end angle, in degrees (used when QUADRIC_FORM = BEND*).

C2C_R1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Cylinder-cone_cylinder Radius 1 (used when QUADRIC_FORM = C2C*).

C2C_R2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Cylinder-cone_cylinder Radius 2 (used when QUADRIC_FORM = C2C*).

C2C_Y1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Cylinder-cone_cylinder Y1 (used when QUADRIC_FORM = C2C*). If Y1=Y2, then R1=R2.

C2C_Y2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Cylinder-cone_cylinder Y2 (used when QUADRIC_FORM = C2C*). If Y1=Y2, then R1=R2.

REACTOR1_R1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, lower cylinder radius.

REACTOR1_R2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, upper cylinder radius.

REACTOR1_Y1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, lower conical transition between cylinders.

REACTOR1_Y2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, upper conical transition between cylinders.

REACTOR1_YR1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, lower rounding below cylinder.

REACTOR1_YR2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, upper rounding above cylinder.

REACTOR1_RR1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, lower rounding radius.

REACTOR1_RR2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, upper rounding radius.

REACTOR1_THETA1(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, lower rounding angle (degrees).

REACTOR1_THETA2(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Reactor 1, upper rounding angle (degrees).

N_X(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

X-component of normal vector defining the plane (used when QUADRIC_FORM = PLANE).

N_Y(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Y-component of normal vector defining the plane (used when QUADRIC_FORM = PLANE).

N_Z(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Z-component of normal vector defining the plane (used when QUADRIC_FORM = PLANE).

T_X(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Translation in x-direction.

T_Y(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Translation in y-direction.

T_Z(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Translation in z-direction.

CLIP_XMIN(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Lower x-limit where the quadric is defined.

CLIP_XMAX(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Upper x-limit where the quadric is defined.

CLIP_YMIN(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Lower y-limit where the quadric is defined.

CLIP_YMAX(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Upper y-limit where the quadric is defined.

CLIP_ZMIN(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Lower z-limit where the quadric is defined.

CLIP_ZMAX(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Upper z-limit where the quadric is defined.

PIECE_XMIN(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Lower x-limit where the quadric is defined in a piecewise group.

PIECE_XMAX(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Upper x-limit where the quadric is defined in a piecewise group.

PIECE_YMIN(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Lower y-limit where the quadric is defined in a piecewise group.

PIECE_YMAX(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Upper y-limit where the quadric is defined in a piecewise group.

PIECE_ZMIN(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Lower z-limit where the quadric is defined in a piecewise group.

PIECE_ZMAX(QUADRIC ID)

Data Type: DOUBLE PRECISION

- $1 \leq QuadricID \leq 500$

Upper z-limit where the quadric is defined in a piecewise group.

FLUID_IN_CLIPPED_REGION(QUADRIC ID)

Data Type: LOGICAL

- $1 \leq QuadricID \leq 500$

Flag defining the type of cells that are outside of the zone defined by [CLIP_XMIN; CLIP_XMAX], [CLIP_YMIN; CLIP_YMAX], [CLIP_ZMIN; CLIP_ZMAX].

Table 8.97: Valid Values

Name	Default?	Description
`.FALSE.`		Remove cells from computational domain.
`.TRUE.`	◉	Treat cells as fluid cells.

BC_ID_Q(QUADRIC ID)

Data Type: INTEGER

- $1 \leq QuadricID \leq 500$

Boundary condition flag.

N_GROUP

Data Type: INTEGER

Number of group(s) of quadrics (<=50).

GROUP_SIZE(GROUP ID)

Data Type: INTEGER

- $1 \leq GroupID \leq DIM_GROUP$

Number of quadrics in the group.

GROUP_Q(GROUP ID, QUADRIC ID)

Data Type: INTEGER

- $1 \leq GroupID \leq DIM_GROUP$
- $1 \leq QuadricID \leq 500$

Quadric ID assigned to a group.

GROUP_RELATION(GROUP ID)

Data Type: CHARACTER

- $1 \leq GroupID \leq DIM_GROUP$

Relation among quadrics of a same group.

Table 8.98: Valid Values

Name	Default?	Description
`OR`	◉	A point belongs to the computational domain if at least one of f(x,y,z) among all quadrics is negative.
`AND`		A point belongs to the computational domain if all of f(x,y,z) among all quadrics are negative.
`PIECEWISE`		When quadrics intersect along planes that are perpendicular to either the x, y, or z-axis, quadrics can be smoothly combined in a piecewise manner.

RELATION_WITH_PREVIOUS(GROUP ID)

Data Type: CHARACTER

- $1 \leq GroupID \leq DIM_GROUP$

Relation between current group and combination of all previous groups.

Table 8.99: Valid Values

Name	Default?	Description
`OR`	◉	A point belongs to the computational domain if f-value for the current group or f-value for the combination of previous groups is negative.
`AND`		A point belongs to the computational domain if f-value for the current group and f-value for the combination of previous groups is negative.

TOL_SNAP(DIRECTION)

Data Type: DOUBLE PRECISION

- $1 \leq Direction \leq 3$

Tolerance used to snap an intersection point onto an existing cell corner (expressed as a fraction of edge length, between 0.0 and 0.5). For stretched grids, three values can be entered in the x, y and z directions.

TOL_DELH

Data Type: DOUBLE PRECISION

Tolerance used to limit acceptable values of normal distance to the wall (expressed as a fraction of cell diagonal, between 0.0 and 1.0).

TOL_SMALL_CELL

Data Type: DOUBLE PRECISION

Tolerance used to detect small cells (expressed as a fraction of cell volume, between 0.0 and 1.0).

TOL_MERGE

Data Type: DOUBLE PRECISION

Tolerance used to remove duplicate nodes (expressed as a fraction of cell diagonal, between 0.0 and 1.0).

TOL_SMALL_AREA

Data Type: DOUBLE PRECISION

Tolerance used to detect small faces (expressed as a fraction of original face area, between 0.0 and 1.0).

ALPHA_MAX

Data Type: DOUBLE PRECISION

Maximum acceptable value of interpolation correction factor.

TOL_F

Data Type: DOUBLE PRECISION

Tolerance used to find intersection of quadric surfaces or user-defined function with background grid.

TOL_POLY

Data Type: DOUBLE PRECISION

Tolerance used to find intersection of polygon with background grid.

ITERMAX_INT

Data Type: INTEGER

Maximum number of iterations used to find intersection points.

TOL_STL

Data Type: DOUBLE PRECISION

Tolerance used to find intersection of STL triangles with background grid.

STL_SMALL_ANGLE

Data Type: DOUBLE PRECISION

Smallest angle accepted for valid STL triangles (in degrees). Triangles having an angle smaller that this value will be ignored.

TOL_STL_DP

Data Type: DOUBLE PRECISION

Dot product tolerance when determining if a point lies in a facet.

DIM_FACETS_PER_CELL

Data Type: INTEGER

Maximum number of STL facets per cell.

OUT_STL_VALUE

Data Type: DOUBLE PRECISION

Defines value of F_STL outside of the STL geometry. A value of 1.0 means the domain outside of the STL geometry is excluded from computation, i.e., an internal flow is computed. Note: This depends on the direction of the facet normal vectors.

Table 8.100: Valid Values

Name	Default?	Description
`-1.0`		model an external flow
`1.0`	◉	model an internal flow

FLIP_STL_NORMALS(BC)

Data Type: LOGICAL

- $1 \leq BC \leq 500$

Option to flip STL facet normals. The index corresponds to the BC ID the STL file is applied to.

Table 8.101: Valid Values

Name	Default?	Description
`.True.`		Flip normals
`.False.`	◉	Do not flip normals

STL_BC_ID

Data Type: INTEGER

Boundary condition flag for the STL geometry

TX_STL

Data Type: DOUBLE PRECISION

Translation in x-direction, applied to the STL geometry.

TY_STL

Data Type: DOUBLE PRECISION

Translation in y-direction, applied to the STL geometry.

TZ_STL

Data Type: DOUBLE PRECISION

Translation in z-direction, applied to the STL geometry.

SCALE_STL

Data Type: DOUBLE PRECISION

Scaling factor, applied to the STL geometry. Note that translation occurs after scaling.

TOL_MSH

Data Type: DOUBLE PRECISION

Tolerance used to find intersection of .msh file with background grid.

OUT_MSH_VALUE

Data Type: DOUBLE PRECISION

Defines value of f outside of the .msh geometry. a value of 1.0 means the domain outside of the .msh geometry is excluded from computation, i.e., an internal flow is computed.

Table 8.102: Valid Values

Name	Default?	Description
`-1.0`		model an external flow
`1.0`	◉	model an internal flow

TX_MSH

Data Type: DOUBLE PRECISION

Translation in x-direction, applied to the .msh geometry.

TY_MSH

Data Type: DOUBLE PRECISION

Translation in y-direction, applied to the .msh geometry.

TZ_MSH

Data Type: DOUBLE PRECISION

Translation in z-direction, applied to the .msh geometry.

SCALE_MSH

Data Type: DOUBLE PRECISION

Scaling factor, applied to the .msh geometry. Note that translation occurs after scaling.

CAD_PROPAGATE_ORDER

Data Type: CHARACTER

Ray propagation order used to determine whether any point is located inside or outside of the STL surface.

Table 8.103: Valid Values

Name	Default?	Description
``	◉	Propagation occurs in unique cell index order, from IJK=IJKSTART3 to IJKEND3, one neighbor at a time (West, East, South, North, Bottom, Top)
`ijk`		Propagation occurs in the I, followed by J, and K directions
`jki`		Propagation occurs in the J, followed by K, and I directions
`kij`		Propagation occurs in the K, followed by I, and J directions

RAY_DIR

Data Type: CHARACTER

Ray direction when propagating CAD value

SET_CORNER_CELLS

Data Type: LOGICAL

Flag to detect and treat corner cells the same way as in the original MFiX version (i.e. without cut cells).

Table 8.104: Valid Values

Name	Default?	Description
`.TRUE.`		Some cut cells may be treated as corner cells.
`.FALSE.`	◉	Do not treat cut cells as corner cells.

FAC_DIM_MAX_CUT_CELL

Data Type: DOUBLE PRECISION

Factor used to allocate cut cell arrays (expressed as a fraction of DIMENSION_3G).

PG_OPTION

Data Type: INTEGER

Option for pressure gradient computation in cut cells.

Table 8.105: Valid Values

Name	Default?	Description
`1`		Use maximum of (east/west), (north/south), and (top/bottom) pairs of velocity cells.
`2`		Use both (east/west), (north/south), and (top/bottom) areas of velocity cells.
`0`	◉	Use east, north and top areas of pressure cell (same as standard cells).

CG_SAFE_MODE

Data Type: INTEGER

Run code in safe mode.

Table 8.106: Valid Values

Name	Default?	Description
1		Performs initial preprocessing but use all original MFiX subroutines during flow solution (using only cell volumes and areas of cut cells).
0	◉	Runs the code with modified subroutines for cut cell treatment.

PRINT_WARNINGS

Data Type: LOGICAL

Prints any warning message encountered during pre-processing on the screen.

CG_UR_FAC

Data Type: DOUBLE PRECISION

Under-relaxation factor used in cut cells (only CG_UR_FAC(2) is used).

PRINT_PROGRESS_BAR

Data Type: LOGICAL

Print a progress bar during each major step of pre-processing stage.

BAR_WIDTH

Data Type: INTEGER

Width of the progress bar (complete status), expressed in number of characters (between 10 and 80).

BAR_CHAR

Data Type: CHARACTER

Character used to create the progress bar.

BAR_RESOLUTION

Data Type: DOUBLE PRECISION

Update frequency of progress bar, expressed in percent of total length (between 1.0 and 100.0).

WRITE_DASHBOARD

Data Type: LOGICAL

Writes the file dashboard.txt at regular intervals. The file shows a summary of the simulation progress.

F_DASHBOARD

Data Type: INTEGER

Frequency, expressed in terms of iterations, at which the dashboard is updated.

RE_INDEXING

Data Type: LOGICAL

Turns on the re-indexing of cells. When true, inactive (dead) cells are removed from computational domain.

ADJUST_PROC_DOMAIN_SIZE

Data Type: LOGICAL

Attempts to adjust grid partition. Each processor will be assigned its own size to minimize load imbalance.

REPORT_BEST_DOMAIN_SIZE

Data Type: LOGICAL

Attempts to adjust grid partition. Each processor will be assigned its own size to minimize load imbalance.

NODESI_REPORT

Data Type: INTEGER

Temporary setting used in serial run to report best domain size for parallel run.

NODESJ_REPORT

Data Type: INTEGER

Temporary setting used in serial run to report best domain size for parallel run.

NODESK_REPORT

Data Type: INTEGER

Temporary setting used in serial run to report best domain size for parallel run.

MINIMIZE_SEND_RECV

Data Type: LOGICAL

Attempts to minimize the size of the send/receive layers.

DWALL_BRUTE_FORCE

Data Type: LOGICAL

Brute force calculation of wall distance.

8.4 Frequently Asked Questions

This document explains how to perform common tasks with MFiX.

8.4.1 How do I ask question, or send feedback?

Please send specific questions and feedback regarding the GUI to the support forum. Please include your version info in all communications (Menu –> About and press the "Copy Version Info" button). Other technical questions regarding MFiX can be sent to the support forum.

8.4.2 How do I change the look & feel of the GUI?

Go to the Settings pane and change the style from the pull down menu.

8.4.3 MFiX (VTK) crashes on RHEL/CentOS 7 with the following error message:

```
ERROR: In ../Rendering/OpenGL2/vtkShaderProgram.cxx, line 431
vtkShaderProgram (0x55555dda47a0):
```

This error message means that the OpenGL driver in RHEL 7.5 does not support the version needed by VTK.

Try using an earlier version of OpenGL:

```
env MESA_GL_VERSION_OVERRIDE=3.2 mfix
```

If this solves the issue for your system, you can add `export MESA_GL_VERSION_OVERRIDE=3.2` to your shell profile.

8.4.4 How do I build the custom solver in parallel?

The custom solver can be built in parallel from the GUI by checking the "Build solver in parallel" check box. This will pass the `-j` option to the `make` command and significantly decrease the time to build the solver.

8.4.5 Play/Pause/Reinit

While the simulation is running, it can be paused by pressing the button. A few settings can be modified, such as the stop time. Settings that cannot be changed (for example, the grid spacing) will be disabled while the simulation is paused. To resume the simulation, save the new settings by pressing the button and press the button.

8.4.6 Restart a simulation from current time

Once a simulation has run to completion, it can be restarted from the current simulation time. Increase the stop time in the "Run" pane, press the button, and press the button. In the "Run solver" dialog box, check the "Restart" check box and select "Resume". Then press the "Run" button.

8.4.7 Restart a simulation from the beginning

Once a simulation has run to completion, or if you pressed the button, the simulation can be restarted from the beginning. This is typically needed if the simulation settings need to be changed and the current results are not needed anymore. First press the to delete the current results. Next change the simulation settings as needed (if any). Press the button, and press the button.

8.4.8 What system of units does MFiX use?

Simulations can be setup using the International System of Units (SI). The centimeter-gram-second system (CGS) is no longer supported.

Property	MFiX SI unit
length, position	meter (m)
mass	kilogram (kg)
time	second (s)
thermal temperature	Kelvin (K)
energy†	Joule (J)
amount of substance‡	kilomole $(kmol)$
force	Newton $(N = kg{\cdot}m{\cdot}s^{-2})$
pressure	Pascal $(Pa = N{\cdot}m^{-2})$
dynamic viscosity	$Pa{\cdot}s$
kinematic viscosity	$m^2{\cdot}s^{-1}$
gas constant	$J{\cdot}K^{-1}{\cdot}kmol^{-1}$
enthalpy	J
specific heat	$J{\cdot}kg^{-1}{\cdot}K^{-1}$
thermal conductivity	$J{\cdot}m^{-1}{\cdot}K^{-1}{\cdot}s$

† The SI unit for energy is the Joule. This is reflected in MFiX through the gas constant. However, entries in the Burcat database are always specified in terms of calories regardless of the simulation units. MFiX converts the entries to joules after reading the database when SI units are used.

‡ The SI unit for the amount of a substance is the mole (mol). MFiX uses the kilomole (kmole). These units are needed when specifying reaction rates:

- amount per time per volume for Eulerian model reactions (kmole.s^{-1}.m^3)
- amount per time for Lagrangian model reactions (kmole.s^{-1})

8.4.9 What do I do if a run does not converge?

Initial non-convergence:

Ensure that the initial conditions are physically realistic. If in the initial time step, the run displays NaN (Not-a-Number) for any residual, reduce the initial time step. If time step reductions do not help, recheck the problem setup.

Holding the time step constant (`DT_FAC=1`) and ignoring the stalling of iterations (`DETECT_STALL=.FALSE.`) may help in overcoming initial nonconvergence. Often a better initial condition will aid convergence. For example, using a hydrostatic rather than a uniform pressure distribution as the initial condition will aid convergence in fluidized bed simulations.

If there are computational regions where the solids tend to compact (i.e., solids volume fraction less than EP_star), convergence may be improved by reducing `UR_FAC(2)` below the default value of 0.5.

Convergence is often difficult with higher order discretization methods. First order upwinding may be used to overcome initial transients and then the higher order method may be turned on. Also, higher-order methods such as van Leer and minmod give faster convergence than methods such as superbee and ULTRA-QUICK.

Non-convergence due to bad cut-cell mesh:

When using cut-cells, non-convergence can arise due to a few bad cut cells, typically small cells (in volume) or bad cut cells coming from poorly resolved geometry in a coarse mesh with geometrical details in the order of one to two cells.

Convergence may be improved by lowering the number of small cut cells. This can be done either by removing small cut cells or by snapping intersection points to cell corners. The settings are available in the Mesh> Mesher pane in the GUI, or through keywords in the .mfx file.

To completely remove a small cut cell, increase the "small cell tolerance" (default value is 0.01), or modify the TOL_SMALL_CELL keyword when editing the .mfx file.

To snap intersection points to a cell corner, increase the "snap tolerance" (default value is 0.00), or modify the TOL_SNAP keyword when editing the .mfx file.

Increasing the "small area tolerance" (TOL_SMALL_AREA keyword) or Normal distance tolerance (TOL_DELH keyword) may also help.

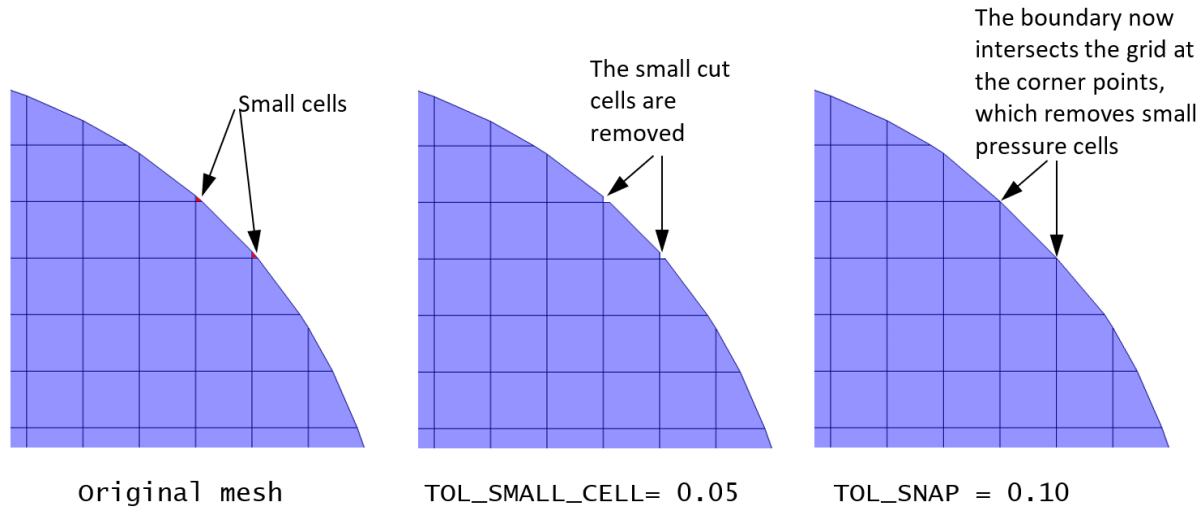


Fig. 8.3: Removing small cut-cells

8.4.10 How to provide initial particle positions in particle_input.dat file?

We can read the initial particle data from a text file, called `particle_input.dat`. The format of this file has changed in the 20.3 release. The old file version can still be used but it is recommended to use the latest format.

To use the file rather than the automatic particle generation (i.e. from Initial Conditions regions), go to the Solids > DEM pane, and leave the "Enable automatic particle generation" checkbox unchecked. Enter a positive number of particles to be read in the "Data file particle count".

Version 2.0 (MFiX 20.3 release)

The new format allows to either read variables or set a constant value for all particles. The file contains a long header where information about the data is specified. This controls how many columns and which variables are read from the file. Next, the data section contains the data for each particle. Each column corresponds to a variable. In 3D, the first 3 columns are always the x,y and z coordinates of the particle's center. In 2D, the first 2 columns are always the x,y coordinates of the particle's center. There are two options for the other variables: either read the data for each particle in the next column, or assign a constant value to all particles.

Below is an example of a `particle_input.dat` file (the line number is shown on the left and it not part of the data itself):

- Line 1 : File version. Do not change this line.
- Lines 2-18 : Instructions to use the file.
- Line 19 : This is a 3D geometry. The first 3 columns in the data section will be x,y,z.
- Line 20 : We will read data for 35361 particles.
- Line 24 : We always read coordinates in the first 2 or 3 columns (do not change)
- Line 25 : We do not read particle's phase ID for each particle. It is assigned a constant value: 1

- Line 26 : We will read the particle's diameter in the next column (column#4).
- Line 27 : We do not read particle's density. It is assigned a constant value: 2500.0 kg/m^3
- Line 25 : We do not read particle's velocity. It is assigned a constant value: (u,v,w) = (0.0,0.0,0.0) m/s
- Lines 29-30: This is a cold, non reacting case, we do not read Temperature or species.
- Line 31 : We do not have user scalars, so we do not read them.
- Lines 32-34: Data header

The data starts on Line 35. Based on the above settings, we need to read, x,y,z, and the diameter of each particles, i.e., 4 columns, for 35361 particles. Only the first few lines of the data is shown.

```
1 Version 2.0
2 ======================================================================
3 Instructions:
4 Dimension: Enter "2" for 2D, "3" for 3D (Integer)
5 Particles: Number of particles to read from this file (Integer)
6 For each variable, enter whether it will be read from the file
7 ("T" to be read (True), "F" to not be read (False)). When not read, enter the
8 default values assign to all particles.
9 Coordinates are always read (X,Y in 2D, X,Y,Z in 3D)
10 Phase_ID, Diameter, Density, Temperature are scalars
11 Velocity requires U,V in 2D, U,V,W in 3D
12 Temperature is only read/set if the energy equation is solved
13 Species are only read/set if the species equations are solved, and requires
14 all species to be set (if phase 1 has 2 species and phase 2 has 4 species,
15 all particles must have 4 entries, even phase 1 particles (pad list with zeros)
16 User scalars need DES_USR_VAR_SIZE values
17 Data starts at line 35. Each column correspond to a variable.
18 ======================================================================
19 Dimension:    3
20 Particles:    35361
21 ======================================================================
22 Variable      Read(T/F)     Default value (when Read=F)
23 ======================================================================
24 Coordinates:     T          Must always be T
25 Phase_ID:        F          1
26 Diameter:        T
27 Density:         F          2500.0
28 Velocity:        F          0.0 0.0 0.0
29 Temperature:     T          (Ignored if energy eq. is not solved)
30 Species:         T          (Ignored if species eq. are not solved)
31 User_Scalar:     T          (Ignored if no user scalars are defined)
32 ======================================================================
33     X (m)            Y (m)            Z (m)        Diameter (m)
34 ======================================================================
35 -5.07749E-02    -7.98147E-01    -2.00612E-01    9.71052E-03
36 -4.86164E-02    -8.16202E-01    -2.25035E-01    8.34185E-03
37 -3.16460E-02    -8.13702E-01    -2.19164E-01    1.09747E-02
38 -3.32833E-02    -8.25178E-01    -2.19393E-01    9.67983E-03
39 -2.48844E-02    -8.30159E-01    -2.16423E-01    9.84182E-03
40 -8.57591E-03    -8.38121E-01    -2.08773E-01    9.92759E-03
41 -2.75312E-03    -8.32665E-01    -2.00770E-01    1.07560E-02
42 1.57276E-02     -8.37839E-01    -2.08196E-01    1.04387E-02
43 1.96605E-02     -8.31447E-01    -2.15105E-01    9.97808E-03
44 2.71083E-02     -8.26450E-01    -2.19492E-01    9.99619E-03
45 3.55440E-02     -8.28885E-01    -2.15687E-01    9.15771E-03
```

(continues on next page)

(continued from previous page)

```
46 4.45244E-02     -8.28239E-01    -2.11771E-01    1.05089E-02
47 4.70488E-02     -8.22092E-01    -2.19129E-01    9.33350E-03
```

The `particle_input.dat` file can be edited in any text editor, including the GUI editor, and be saved in the project directory. Please run the *2D DEM fluid bed with central jet* tutorial from the GUI, or see the *DEM Initial Conditions* last section for an example.

Version 1.0 (Prior to MFiX 20.3 release)

The file stores each particle position, radius, density, and velocity. For 2D geometries, it contains 6 columns: x , y, radius, density, u, v. In 3D, it contains 8 columns: x , y, z, radius, density, u, v, w. The number of lines correspond to the number of particles read, and should match the "Data file particle count" defined in the Solids > DEM tab. The check box "Enable automatic particle generation" must be left unchecked.

Below is an example of a 2D `particle_input.dat` (only 4 particles shown):

```
0.2000D-03  0.2000D-03  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00
0.6000D-03  0.2000D-03  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00
0.1000D-02  0.2000D-03  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00
0.1400D-02  0.2000D-03  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00
```

Below is an example of a 3D `particle_input.dat` (only 4 particles shown):

```
0.2000D-03  0.2000D-03  0.1000D-01  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00  0.
↪0000D+00
0.6000D-03  0.2000D-03  0.1000D-01  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00  0.
↪0000D+00
0.1000D-02  0.2000D-03  0.1000D-01  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00  0.
↪0000D+00
0.1400D-02  0.2000D-03  0.1000D-01  0.5000D-04  0.1800D+04  0.0000D+00  0.0000D+00  0.
↪0000D+00
```

8.4.11 How to verify/flip STL file facet normals?

When using an STL file to define the geometry, a normal vector is associated with each facet (triangle). The direction in which the normal vector points indicates the fluid region. It is important to verify the normal vectors points in the desired direction before running the simulation.

To show the normal vectors in the Model view, expand the geometry view settings on the right and check "Show Normals". It may be necessary to adjust the scale to get a better view. The figure below shows an example of a cylinder with vectors pointing towards the interior region of the cylinder. This will correspond with an internal flow simulation where the fluid region is located inside the cylinder and the outside region is not part of the simulation.

If the normal vectors are not pointing in the correct direction, the normals can be flipped by checking the "Flip normal" box for any selected stl file. To model an external flow using the same geometry as above, flip the normals and the change will be reflected in the Model view (see figure below). Now the normals points towards the outside of the cylinder, and the region inside the cylinder will not be part of the computation.

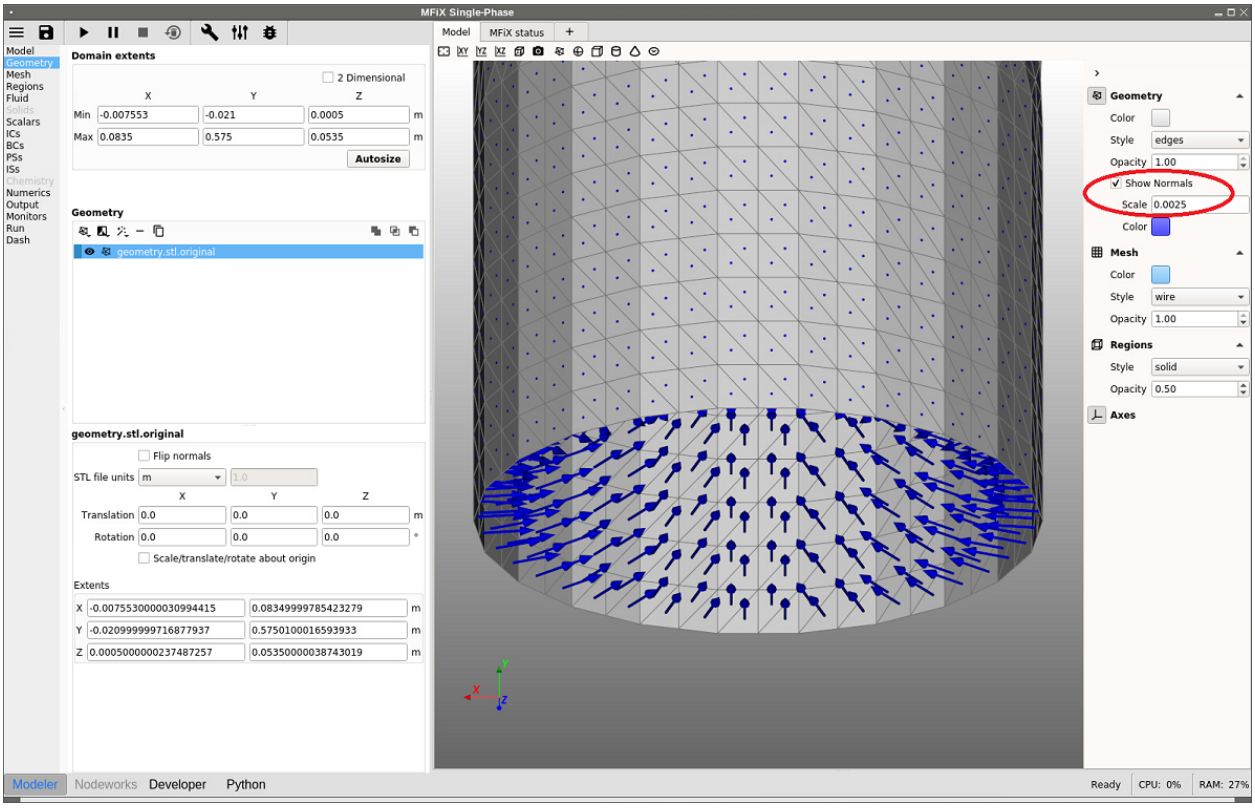


Fig. 8.4: Displaying STL normals, internal flow

8.4.12 How to manually specify a grid partition for DMP runs?

When running a simulation in parallel (DMP) mode, the parallel decomposition is by default uniform in each direction and the number of partitions is set by NODESI, NODESJ, and NODESK in the x,y and z directions, respectively. The default partition may not always be the most efficient. For example, a DEM simulation of a fluidized bed where most particles are at the bottom of the bed may run faster if smaller partitions are set at the bottom, where many particles are located and a large partition is used in the freeboard. This will better distribute particles among processors, and provide a better load balance.

Let's assume the mesh contains 20x100x20 cells (IMAX=20, JMAX=100, KMAX=20) and we use 16 cores with a 2x4x2 partition (NODESI=2, NODESJ=4, NODESK=2). Assume particles are roughly located within the bottom half of the bed throughout the simulation. With the default (uniform) decomposition, each vertical partition will contain JMAX/NODESJ=25 cells (Figure 8.6). This means 8 cores will remain mostly idle with respect to particle tracking and collision.

Instead, we can distribute cells in the y-direction so that the vertical partitions have roughly the same number of particles. This can be achieved with 10, 10, 10, and 70 cells from bottom to top. (Figure 8.7).

Due to the symmetry of the problem in x and z directions, no adjustment is needed in these directions, and a uniform distribution is adequate.

Note that the custom partition will provide better load balance for particles but will make the fluid calculation slower since fluid cells will be unbalanced. However, since a large majority of the time is spent in the DEM loop, it should be beneficial to customize the partition.

To specify this decomposition, the following `gridmap.dat` needs to be saved in the project directory:

The first line specifies the values of NODESI, NODESJ, NODESK.

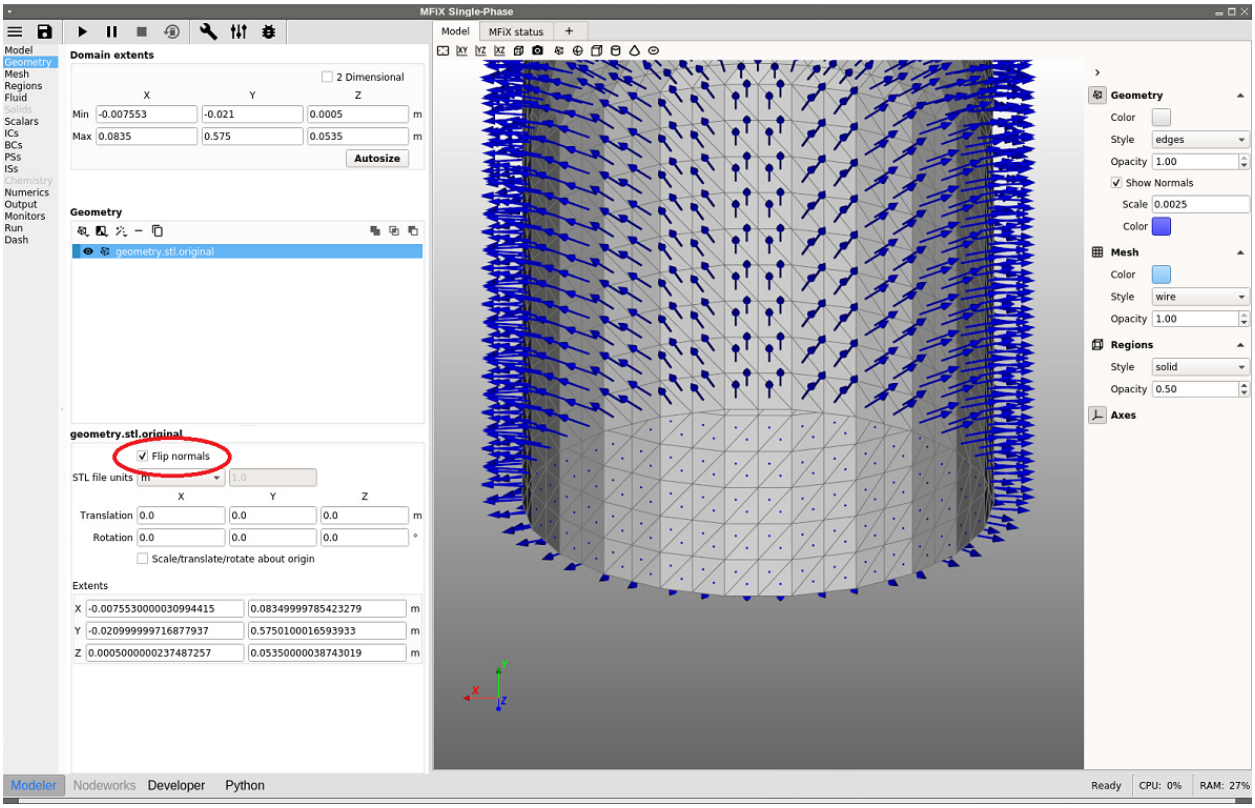


Fig. 8.5: Flipping STL normals to model an external flow

```
2          4          2     ! NODESI, NODESJ, NODESK
```

The next two lines (NODESI=2) specify the partition sizes in the x-direction. Here each partition contains 10 cells:

```
0          10
1          10
```

The next four lines (NODESJ=4) specify the partition sizes in the y-direction. Here three partitions at the bottom contains 10 cells each, and the last partition contains 70 cells:

```
0          10
1          10
2          10
3          70
```

The last two lines (NODESK=2) specify the partition sizes in the z-direction. Here each partition contains 10 cells:

```
0          10
1          10
```

Below is the entire `gridmap.dat` file:

```
2          4          2     ! NODESI, NODESJ, NODESK
0          10
1          10
0          10
```

(continues on next page)

(continued from previous page)

```
1         10
2         10
3         70
0         10
1         10
```

8.5 How to setup PIC simulations

This document explains how to setup PIC simulations (statistical weight, PIC parameters, mesh).

8.5.1 How To Choose a Good Statistical Weight for PIC Runs?

One key to creating efficient and accurate particle in cell (PIC) simulations is choosing an appropriate value for the statistical weight of particles per parcel. The user needs to balance desired solids weight fraction, mesh density and solids mass error with computational speed and physics.

The PIC statistical weight in MFIX (IC_PIC_CONST_STATWT or BC_PIC_CONST_STATWT) represents how many particles are represented by each parcel. The parcels themselves must be of discrete number in a simulation, but because statistical weight is a math construct, partial particles may occur mathematically.

To facilitate statistical weight selection, it is desirable to create a spreadsheet that illustrates how its selection can impact simulation conditions, or the following logic may be prescribed.

1. Collect the following information:
   a. Diameter of an equivalent single spherical particle (m)
   b. Mass density of particles (kg/m ^{3})
   c. Max-pack solids fraction of particles for simulation
2. Choose a mesh for the simulation. Separately, calculate an average cell volume (m ^{3}) and a minimum cell volume (m ^{3}).
3. Using max-pack solids fraction (which must be defined for PIC, 1-EP_STAR), calculate an average cellular volume available for solid (m ^{3}) and a minimum cellular volume available for solid (m ^{3}).
4. Using particle diameter, calculate the number of spherical particles these average cellular volumes available for solid represent. Assuming a uniform mesh, this calculated number, as a rounded-down integer, would represent the maximum statistical weight one can choose in a PIC simulation, although it is highly unlikely one would choose this value. In a variable mesh, the value associated with the minimum cell volume is the maximum statistical weight. (Note: using an average cellular volume for this calculation allows a balance of parcel load in both large and small cells.) Note that a user does not need to know the solids fraction distribution that they expect in a simulation, only the max-pack solids fraction, which is a constant.
5. Now the balancing act between parcel count, actual solids fraction and mass error begins. First, choose a statistical weight value. An easy way to make a first approximation is to use the maximum statistical weight from (4) and divide by an integer which in your mind represents the number of parcels you want per cell. Statistical weights derived like this will naturally give a solids fraction very close to whatever the original desired value from (1) was, and minimize mass error in a simulation. The statistical weight does not need to be an integer, because partial particles make sense in PIC. While the parcel count in an individual cell will initially begin as an integer, there is no limitation that says an integer number of particles must make up a parcel. Choosing an integer value for statistical weight does make it simple to imagine the parcels as grouped entities of particles, but it may interfere with expectations for representative solids volume fraction per cell. Note, however, that regardless of what the user

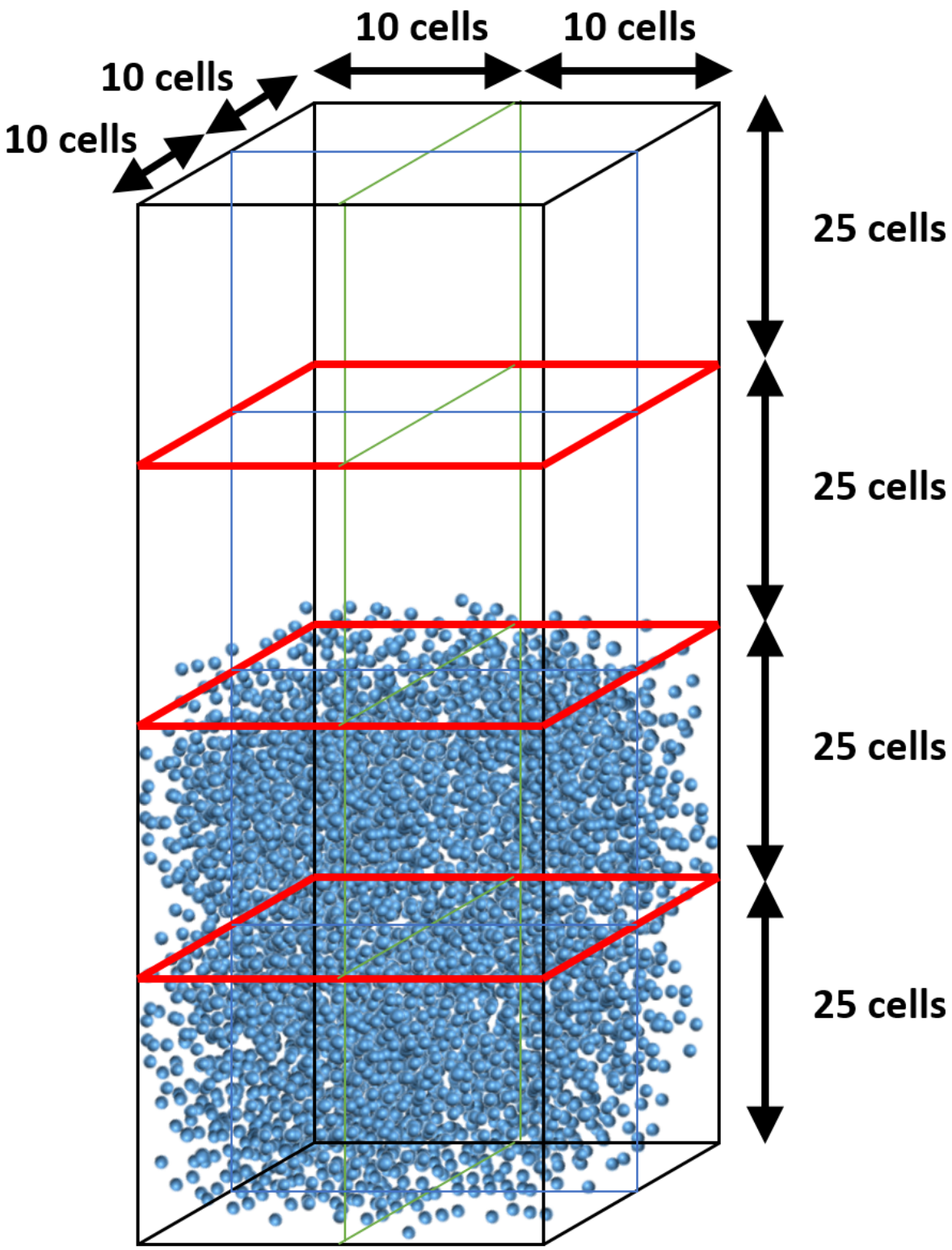


Fig. 8.6: Default partition (uniform distribution)

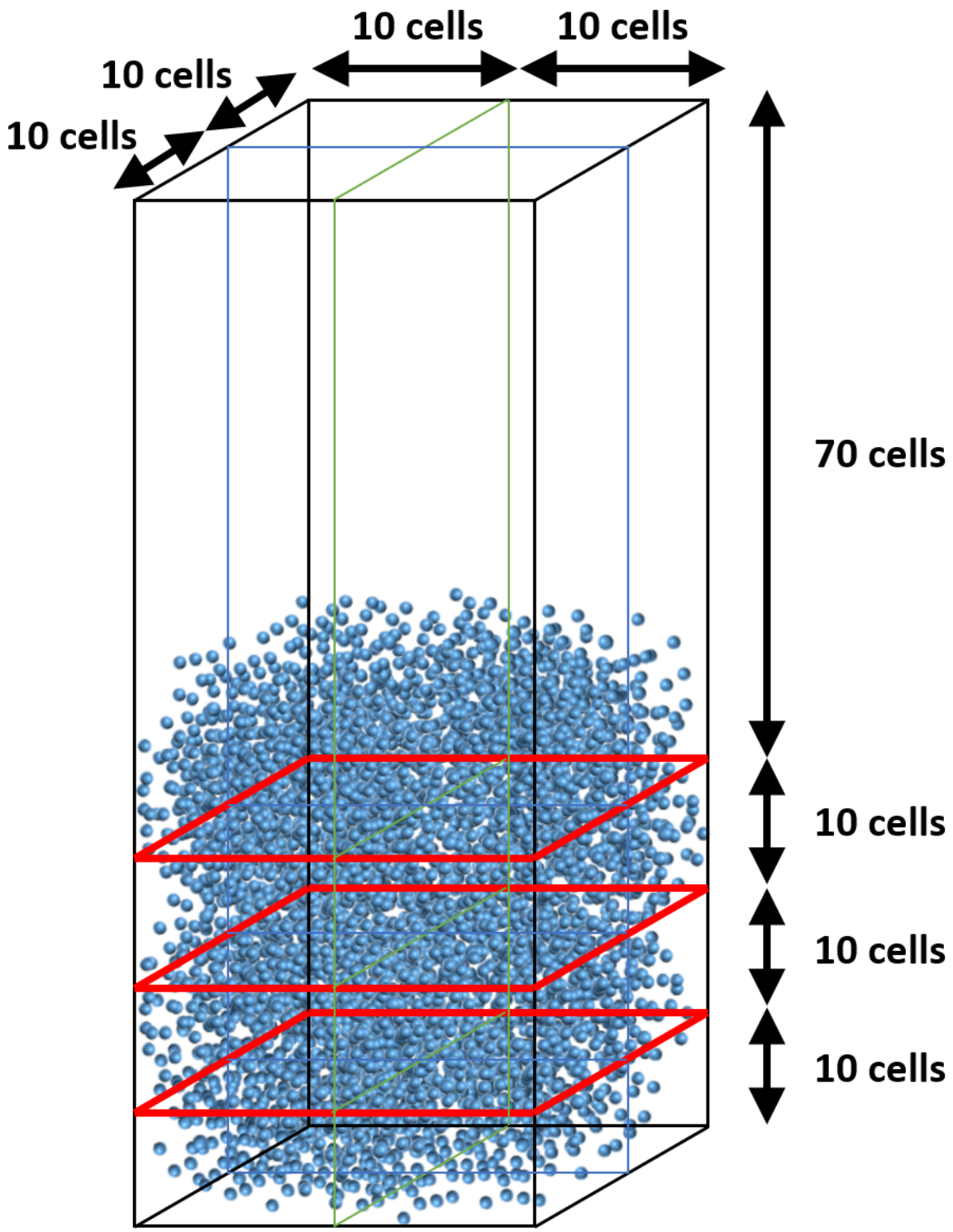


Fig. 8.7: Adjusted partition in vertical direction (specified with gridmap.dat)

specifies as a statistical weight, the MFIX program will collect truncated mass values in each cell until enough mass to represent a full parcel accumulates, whereupon a single extra parcel will be placed in the domain.

6. It is advisable to inspect your fully populated PIC parcel domain to make sure you have not applied a value that does not make visual or physical sense. This is particularly true for very dilute flow and highly packed flow. For dilute flows, choosing a statistical weight that is too large will result in a simulation that cannot possibly capture expected flow dynamics. In a highly packed flow, choosing a statistical weight that is too low may result in non-physical packing, and an overly large local solids fraction. As such, be certain to note the values reported in the MFIX output for desired solids fraction and calculated solids fraction, then assure yourself that the numbers are within some reasonable error band.

8.5.2 Extra info and Examples:

During initialization, the user can choose any region to fill with parcels. If a distinct mass of particles is needed in a region at start-up, the user must pre-conceive an appropriate volume of cells and associated solids fraction to achieve that mass.

For example, if a user wants :

- 0.1 kg of 0.001 m particles with density 1000 kg/m ^{3}
- in a volume that is 0.1 x 0.1 x 0.1 m ^{3} defined by a 10 x 10 x 10 mesh
- with max-pack of 0.4 (theoretical limit of random packing of spheres is 0.36)

First, calculate the volume of 1 particle: V_particle= π/6 $(0.001)^3$ = 5.236E-10 m ^{3}

Calculate the mass of 1 particle: m_particle=(1000 kg/m ^{3})(5.236E-10 m ^{3}) = 5.236E-7 kg

Calculate single cellular volume: V_cell=(0.1m) ^{3}/((10×10×10))=1.0E-6 m ^{3}

Calculate available volume for parcels: (1-0.4)(1.0E-6 m ^{3}) = 6.0E-7 m ^{3}

For simplicity, we assume that the particles are evenly distributed among the 1000 cells. That means, each cell needs 0.100 kg/1000cells = 0.0001 kg of solid. In turn, that means each cell needs 0.0001 kg / 5.236E-7 kg = 190.985 particles. From this particle loading, you can choose a statistical weight based on the number of particles. For example, you could choose 190.985 as the statistical weight and get 1 big parcel (unwise), or you could choose, say, a statistical weight of 10 and get 19 parcels with a small local mass error. The small errors are accumulated, and when large enough, an extra parcel will be placed locally in the mesh to account for the local loss. Note that if the user does not want to fill an initial area at maximum packing, simply choose a different value for solids fraction and recalculate. Having a preconception of how "loose" a particle bed should be is very helpful for reducing the transient in a simulation start-up.

Likewise, if a mass inflow boundary is prescribed in the PIC model, a similar logic must be followed. The key is to make sure that the statistical weight used for the boundary flow condition is reasonable (as in, the number of particles per parcel will fit into the cell(s) that abut the boundary condition specification). MFIX will control the mass itself by accounting individual particle mass and randomly assigning parcels through the BC window. Note that smaller statistical weights produce less local mass error, but the user must choose their own balance between statistical weight and computational speed.

8.5.3 Meshing Basics for PIC Runs

There are three common mistakes a user may make when setting up a PIC run. The first represents an entire misunderstanding of MFIX-PIC which is necessarily a 3-D model. That means, you cannot specify a 2-D simulation with MFIX-PIC. Second, the interpolation algorithm which undergirds the MFIX-PIC model requires a minimum of 3-cells in each direction through a geometry to give a robust solution. It is important to check small geometric regions of the mesh that might pass through an orifice or channel to assure that there are at least 3 cells available for calculations to work in a hydrodynamically stable way. Smaller cell distributions will not crash the model, but you may get unexpected flow behavior. Finally, choosing statistical weights that in comparison to mesh give less than one parcel per mesh cell will give errant solutions. (See section on choosing good statistical weights.) Note that one of the benefits of PIC is that it allows for a somewhat coarse mesh with a large particle count, but a user needs to balance mesh and parcel count in a reasonable way.

8.5.4 Choosing Primary PIC Parameters

PIC parcels do not behave like DEM particles. There are no Newtonian physical laws that govern collisions. Instead, PIC utilizes a solids stress (τ_p) field that influences solids motion through a coupling with the momentum equation. Specifically, the solids stress is calculated as:

$\tau_p = \frac{P_p \epsilon_p^{\gamma}}{\max[\epsilon_{cp} - \epsilon_p, \delta(1-\epsilon_p)]}$

In MFIX, the parameters that the user can control to define this solids stress are an empirical pressure constant (PSFAC_FRIC_PIC (P_p), a volume fraction exponential scale factor (FRIC_EXP_PIC (γ), void fraction at maximal close packing(EP_STAR (ϵ_{cp}) and a non-singularity constant (FRIC_NON_SING_FAC (δ). The only other variable at play within the solids stress model is solids void fraction (ϵ_p) which is calculated dynamically as the simulation progresses.

The parameters used in PIC are very robust. The default values of P_p =100, γ =3, δ =1.0E-7 should work well for most systems. There should be no reason to change δ. The value for ϵ_{cp} will vary depending on your application, although it is worth noting that there is a theoretical limit of ~0.36 for the random packing of equal spheres. It is important to note that ϵ_{cp} represents the void or gas fraction per cell, and not the solids fraction. The effects of changing the values of P_p and γ are very problem specific. For example, sedimentation cases (on which the solids model was originally built) will model best with P_p =1, γ =2 so that gravity has the most effect on the system through the momentum equation. But, very dynamic cases (U_mf>10) where we expect the solids to more greatly affect the hydrodynamics of the system may respond better with selections like P_p =1000, γ =3. Essentially, like all CFD models, it may be necessary to tune the PIC model to your specific case. As a rule of thumb, larger P_p at constant γ will result in particles that move more dynamically (as the selection will linearly increase the solids stress). But, larger γ at constant P_p will reduce the solids stress exponentially and dampen overall solids motion. The largest values of solid stress will occur when P_p is large and γ is small.

8.5.5 Choosing Secondary PIC parameters

Like DEM, the user can control restitution factors that affect parcel motion. MPPIC_COEFF_EN_WALL applies a scale factor to the normal component of velocity after a parcel interacts with a wall. Likewise, MPPIC_COEFF_ET_WALL applies a scale factor to the tangential components of velocity after a parcel interacts with a wall. There are two other empirical factors available in MFIX-PIC. The first is MPPIC_COEFF_EN1 which is an overall scale factor that can dampen the effect of the entire solids stress model, and MPPIC_VELFAC_COEFF which is a scale factor that effects the value of bulk solids velocity through an evaluation of gas-solids slip velocity. These last two factors should remain at default values unless you are working as an advanced user and understand the open source PIC code implicitly.

8.5.6 Collision Damping

An empirical collision damping feature has been added to the MFIX-PIC model. The feature includes a calculation for collision frequency based on statistical likelihood. From this frequency, a small change in velocity is calculated and applied to parcels locally. The effect is an overall damping of solids velocity, particularly in areas where dense particle flow is apparent. To invoke the feature, the user must select "Collision Damping" in the GUI (Solids>PIC pane) or set the keyword is PIC_COLLISION_DAMPING=.TRUE. in the .mfx file.

8.5.7 Frequently Asked Questions

FAQ: Why can't I define parcels per cell instead of particles per parcel?

When creating the MFIX-PIC model, programmers chose to define particles per parcel for two reasons.

1. The user has full control over the statistical weights of the PIC parcels and is not reliant on the computer code to calculate those values without knowledge of the simulation being run.
2. In most CFD simulations, mesh can be highly variable. When a user defines parcels/cell, the smallest cell will limit the number of parcels that can be placed in any cell at initialization. By defining particles/parcel, the check we suggest for examining the smallest cell only limits the overall size of the parcel, and a more even solids void fraction can be achieved at start-up. An argument can be made that multiple initialization regions can be created to smooth solids void fraction using the parcel/cell method, but this only complicates set-up.

FAQ: What is a good number (or range) for the number of parcels per cells

It depends on the kind of simulation you are trying to run, how many processors you have available to you, and what kind of fidelity you expect. Avoid going below 3 parcels/cell in a serial run where there is very little dynamic (like settling).

Typically, around 10 parcels/cell works smoothly. If the simulation looks chunky, reduce the number of particles/parcel, thus upping the parcels/cell count.

FAQ: What solids model parameter values work best for the PIC model when running 2 densities that are roughly 10x different in magnitude?

A density difference like this (say 300 kg/m ^{3} and 3,000 kg/m ^{3}) will cause natural physical separation in most systems. To capture the best physics in a PIC model where density drives strong gravitational effects, choose P_p =1 and γ =5 to begin. You may still need to adjust these parameters but these choices will give you a good place to start.

CHAPTER

NINE

RUNNING MFIX ON JOULE

This section is for MFiX users on NETL's HPC system Joule. MFiX is installed as an environment module, so installing it yourself is not necessary. The following sections document which compiler modules to use on Joule to run MFiX and build the solver.

To build the solver from the GUI with a particular compiler on Joule, you need the environment module for that compiler loaded when before starting MFiX. If that compiler is not loaded, exit MFiX to go back to your shell, load the appropriate module, and start MFiX again.

To avoid confusion, use `module list` and `module unload` to check that you only have one version (e.g. 8.2, 6.5, 6.4) of one particular compiler (e.g. `gnu`, `intel`) loaded at any given time!

9.1 Running the GUI

1. Load the MFiX 20.4 module:

```
> module load mfix/20.4
```

2) Run the GUI:

```
> vglrun mfix
```

3) To build from the GUI, see: *Building Custom Interactive Solver*.

4) To build from the command line, run:

```
> build_mfixsolver
```

With just `mfix/20.4` environment module loaded, you can build the MFiX solver in serial with GCC 4.8 (the CentOS system compiler). For building the solver with other configurations, see below.

9.2 Building Solver with GCC

At the time of this writing, Joule has modules for GCC 6.4, 6.5, 8.2, 8.4, and 9.3. GCC 8.2 is recommended.

```
> module load gnu/8.2.0                                          # if only␣
↪using serial
> module load gnu/8.2.0 openmpi/4.0.1_gnu8.2                     # if␣
↪using DMP
```

(continues on next page)

(continued from previous page)

```
> module load gnu/6.5.0                                          # if only␣
↪using serial
> module load gnu/6.5.0 openmpi/3.1.3_gnu6.5                     # if using␣
↪DMP
```

When building from the command line:

```
> build_mfixsolver -s none           # for serial, no interactive support
> build_mfixsolver -s none --dmp     # for DMP, no interactive support
```

When building from the GUI:

- For DMP support, check the DMP checkbox (on the Build Dialog)
- Default interactive support is python (recommended)

9.3 Building Solver with Intel Fortran Compiler

Load module:

```
> module load intel/2019.2.053                      # if only using serial
> module load intel/2019.2.053 intelmpi/2019.2.144  # if using DMP
```

When building from the command line:

```
> build_mfixsolver -s none        # for serial
> build_mfixsolver -s none --dmp  # for DMP
```

Building an Interactive solver with Intel compiler is not supported.

When building from the GUI:

- Check "Enable developer mode" under Settings, then select Interactive support: "None" (on the Build Dialog)
- For DMP support, check the DMP checkbox (on the Build Dialog)

9.4 Building Solver from Source Tarball

To build the solver from the MFiX source tarball or from a Git repo, you do not need to load the `mfix/20.4` module. Instead, you only need to load the CMake module:

```
> module load cmake/3.14.1
```

Also load the GCC or Intel module for the compiler you want to use. For complete instructions on building the solver from source, see *Build from Source (for Developers)*.

BIBLIOGRAPHY

[SB1988] Syamlal, M, and O'Brien, T.J. (1988). Simulation of granular layer inversion in liquid fluidized beds, *International Journal of Multiphase Flow*, Volume 14, Issue 4, Pages 473-481, https://doi.org/10.1016/0301-9322(88)90023-7.

[HKL2001] Hill, R., Koch, D., and Ladd, A. (2001). Moderate-Reynolds-number flows in ordered and random arrays of spheres. *Journal of Fluid Mechanics*, Volume 448, Pages 243-278. https://doi.org/10.1017/S0022112001005936

[DG1990] Ding, J. and Gidaspow, D. (1990). A bubbling fluidization model using kinetic theory of granular flow, *AIChE Journal*, Volume 36, Issue 4, Pages 523-538, https://doi.org/10.1002/aic.690360404

[LB2000] Lathouwers, D. and Bellan J. (2000). Modeling of dense gas-solid reactive mixtures applied to biomass pyrolysis in a fluidized bed, Proceedings of the 2000 U.S. DOE Hydrogen Program Review, https://www.nrel.gov/docs/fy01osti/28890.pdf

[WY1966] Wen C.Y., and Yu Y.H. (1966). Mechanics of fluidization, *The Chemical Engineering Progress Symposium Series*, Volume 62, Pages 100-111.

[BVK2007] Beetstra, R., van der Hoef, M.A., and Kuipers, J.A.M. (2007). Numerical study of segregation using a new drag force correlation for polydisperse systems derived from lattice-Boltzmann simulations, *Chemical Engineering Science*, Volume 62, Issues 1–2, Pages 246-255. https://doi.org/10.1016/j.ces.2006.08.054.

[HYS2010] Holloway, W., Yin, X., and Sundaresan, S. (2010). Fluid-particle drag in inertial polydisperse gas–solid suspensions, *AIChE Journal*, Volume 56, Issue 8, Pages 1995-2004. https://doi.org/10.1002/aic.12127

[HBK2005] Hoef, M., Beetstra, R., and Kuipers, J. (2005). Lattice-Boltzmann simulations of low-Reynolds-number flow past mono- and bidisperse arrays of spheres: Results for the permeability and drag force. *Journal of Fluid Mechanics*, Volume 528, Pages 233-254. https://doi.org/10.1017/S0022112004003295

[BVK2007a] Beetstra, R. , van der Hoef, M. A. and Kuipers, J. A. (2007), Drag force of intermediate Reynolds number flow past mono- and bidisperse arrays of spheres. AIChE J., 53: 489-501. https://doi.org/10.1002/aic.11065

[BVK2007b] (2007), Erratum. AIChE J., 53: 3020-3020. https://doi.org/10.1002/aic.11330

[IPBS2012] Igci, Y., Pannala, S., Benyahia, S., and Sundaresan, S. (2012). Validation studies on filtered model equations for gas-particle flows in risers, *Industrial & Engineering Chemistry Research*, Volume 54, Issue 4, Pages 2094-2103. https://doi.org/10.1021/ie2007278

[MMHAS2013] Milioli, C.C., Milioli, F.E., Holloway, W., Agrawal, K. and Sundaresan, S. (2013), Filtered two-fluid models of fluidized gas-particle flows: New constitutive relations. *AIChE J.*, Volume 59, Issue 9, Pages 3265-3275. https://doi.org/10.1002/aic.14130