## **General Download Instructions**

You need to download following software to start the preprocessing steps– (i) **mtex-5.1.1.zip** (ii) **DREAM3D-6.4.197-Win64.zip** and (iii) **ParaView-5.5.0-RC4-Qt5-Windows-64bit.zip.**

# **DREAM3D**

1. Browse to the folder **DREAM3D-6.4.197-Win64** and double-click on **DREAM3D** executable. In the toolbar, click on **File -> Open** and browse to the directory **Training\_Materials->Pre-Processing->HCP**, and load the pipeline **PRISMS\_pipeline\_HCP.json**. Two pop-up windows will appear one after the other. Ignore them by clicking **Cancel** for both. **(Caution!!! : Please don’t try to save the pipeline by pressing Ctrl + S because that will attempt to save the huge EBSD data set which will take up time and the application will freeze.)**
2. Click on the **StatsGenerator** filter in the pipeline window. Click on the **ODF** button and click on the radiobox under **ODF parameters** displaying **Bulk Load From File**. Click on the **Select** button and choose the file containing the EBSD data with path **Training\_Materials/Pre-Processing/HCP/mgdata.txt**. Choose the angle representation as **Euler** and the delimiter as **Space**, as in Fig. 1. Then click on the button displaying **Load Data** to generate the pole figures.

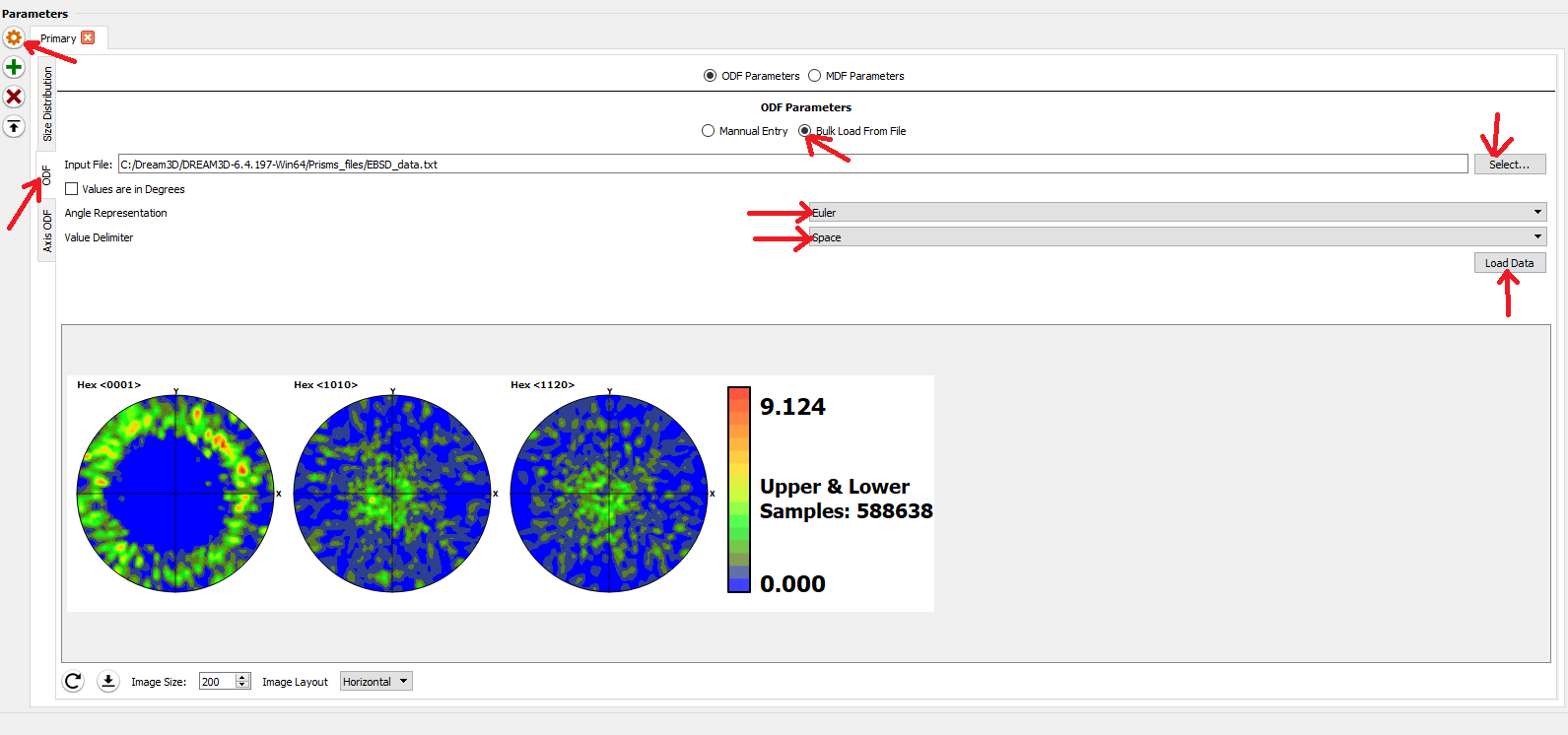


Fig. 1

1. Click on the **Size Distribution** tab and then on **Create Data** as in Fig. 2. This completes the process of generating the size, shape and orientation distributions.

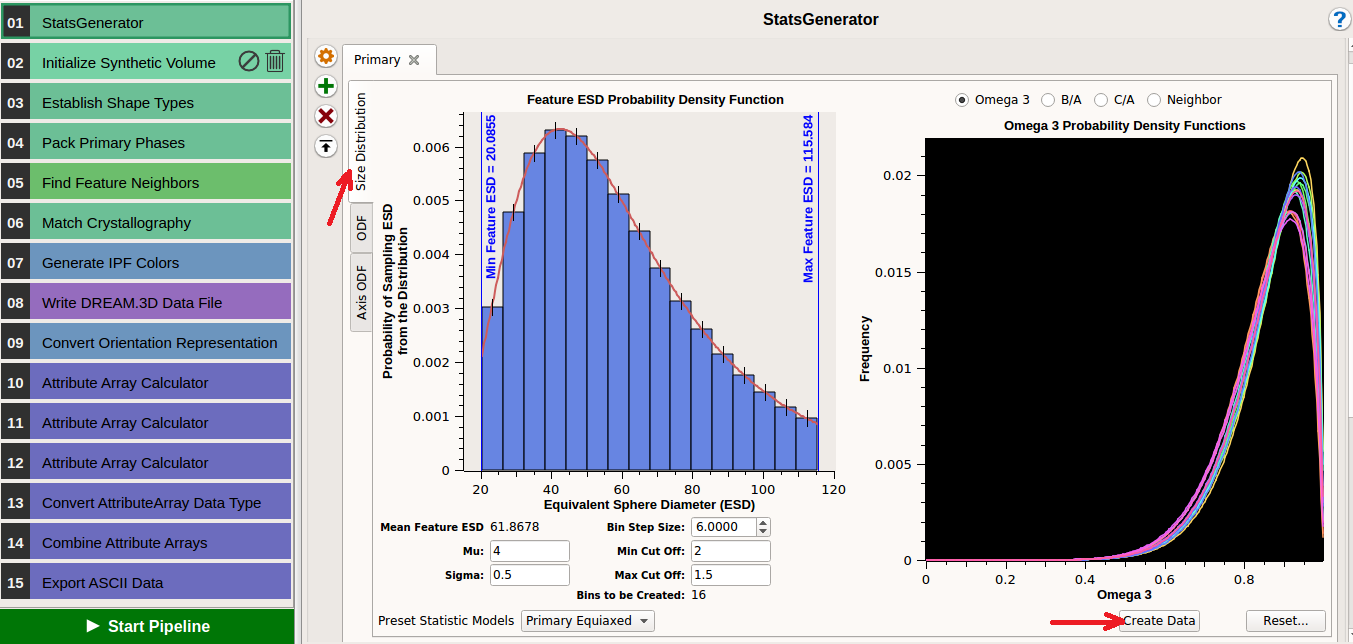


Fig. 2

1. Click on the **Initialize Synthetic Volume** filter in the pipeline window. Here the box dimensions can be specified. Choose dimensions of 32 voxels along x, y and z respectively(Fig. 3). Choose the resolution as 11 along each direction. Check the box next to **Estimate Number of Features** to get the approximate number of grains.

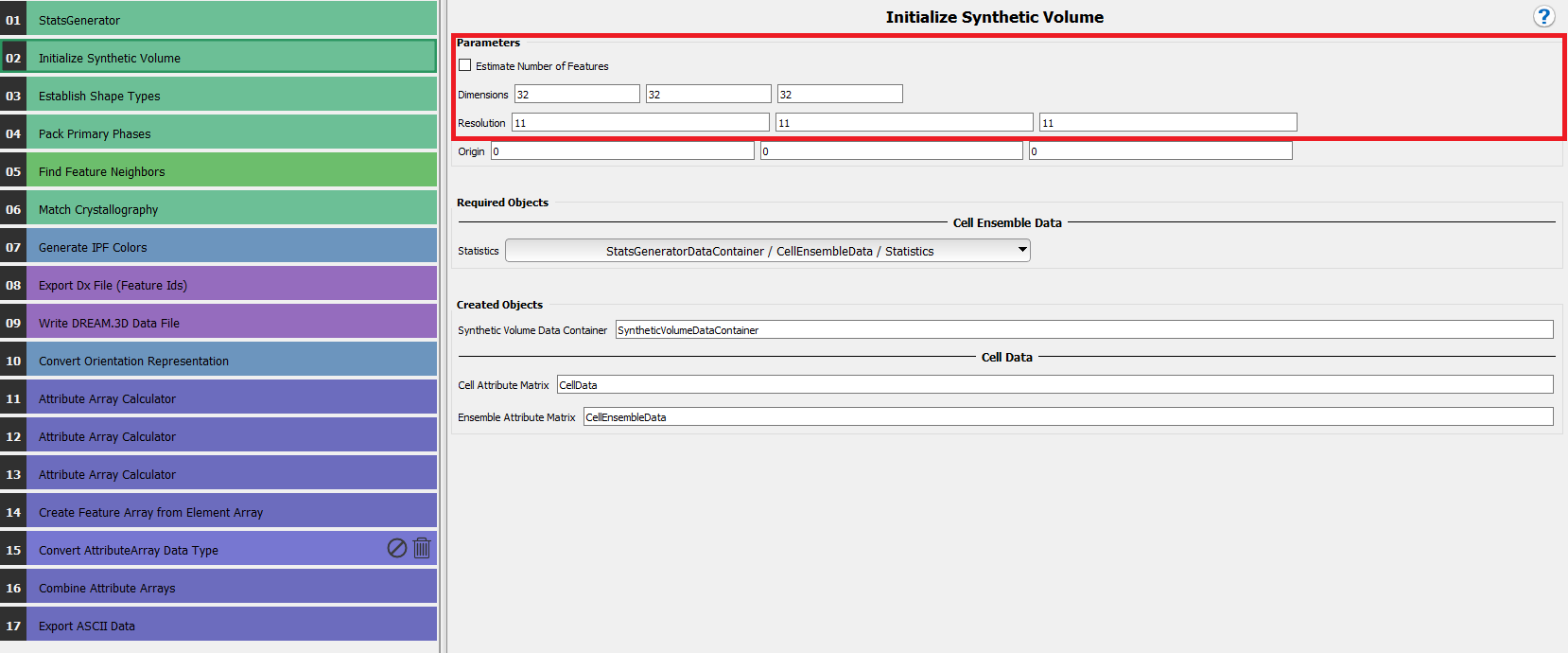


Fig. 3

1. Click on the **Export Dx File (Feature Ids)** filter in the pipeline window. This filter will generate an output file containing the grain-ids for all the voxels in the format required as input to PRISMS-CPFE. In the present case, specify the output file address (you have to specify a path) and file name as **grainID.txt**(Fig. 4).

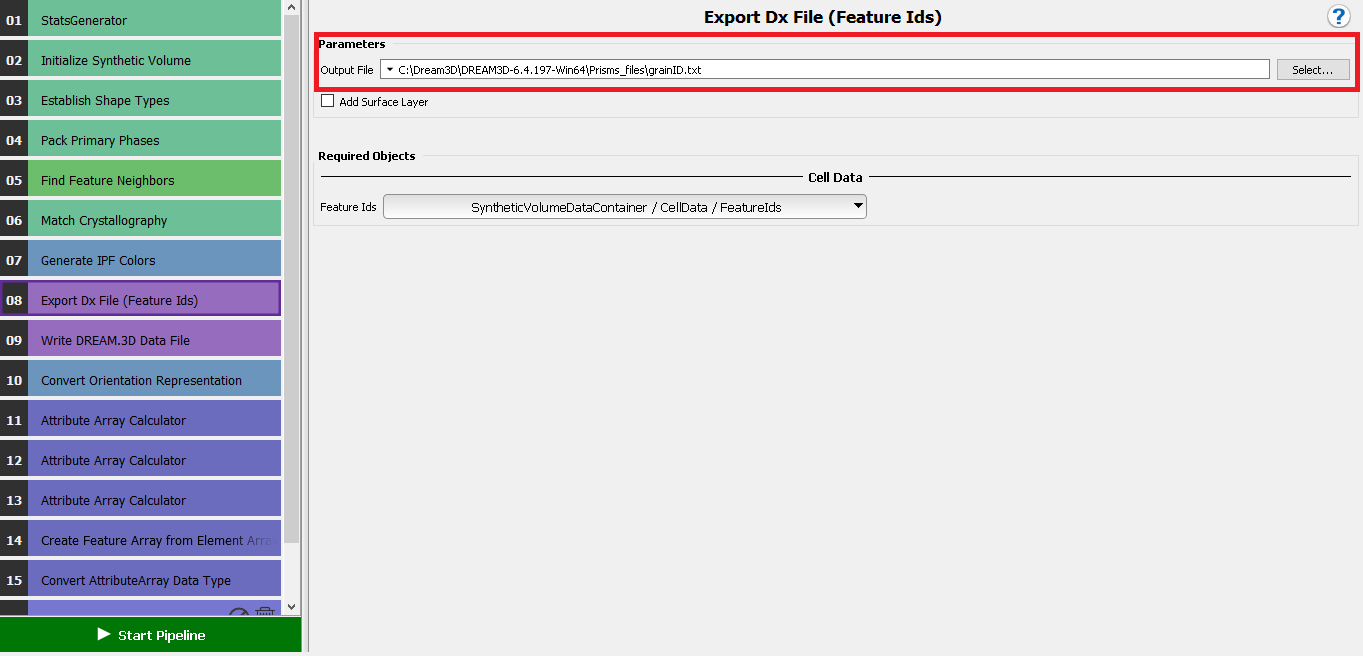


Fig. 4

1. Click on the **Write DREAM3D Data File** filter in the pipeline window(Fig. 5). This is performed so that the microstructure can be visualized in a visualization software like Paraview. Name the file **Magnesium.dream3d** and check the box next to **Write Xdmf File**.

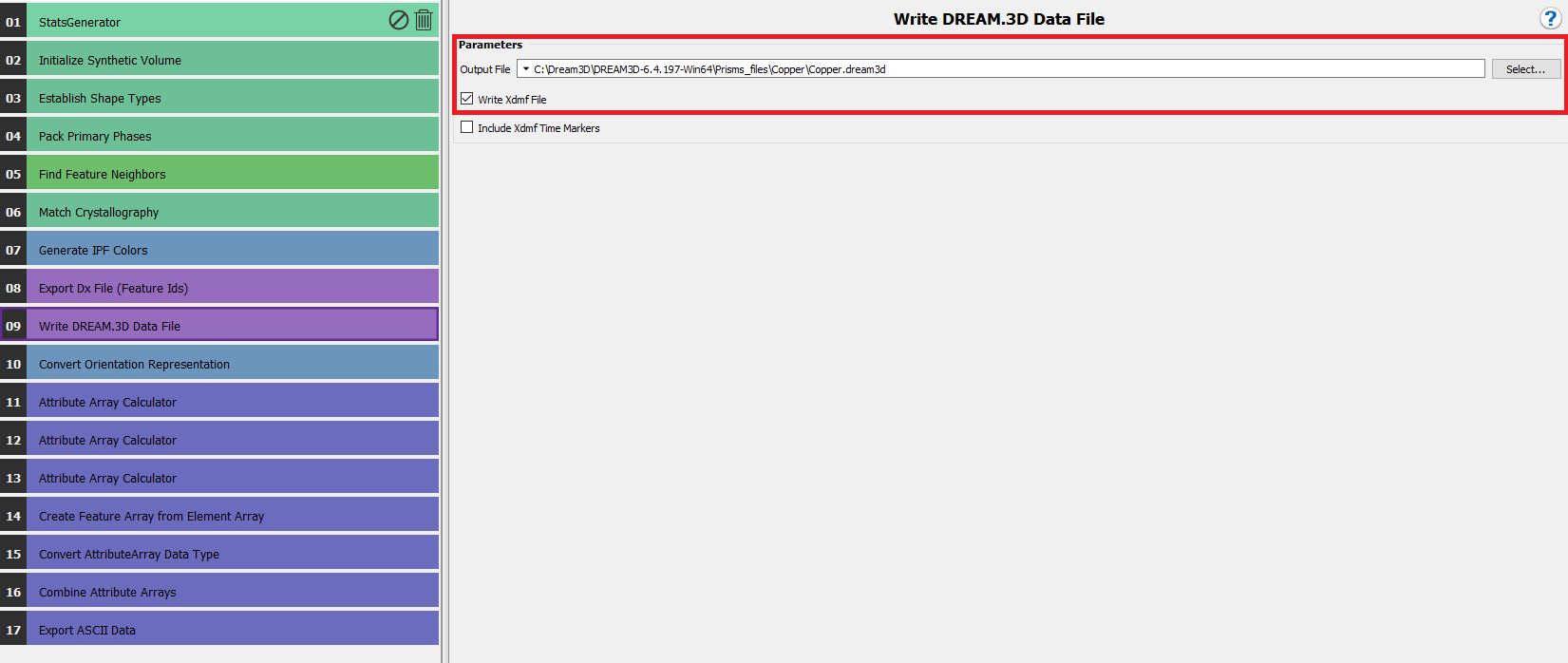


Fig. 5

1. Click on the **Export ASCII Data** filter in the pipeline window. Specify the output path as required, file extension as **.txt**, **maximum tuples per line** as **1** and **(space)** as the delimiter. For **Attribute Arrays to Export** choose **SyntheticVolumeDataContainer / CellFeatureData**, and then double-click on the data container **orientations**( Fig. 6). This will eventually generate the file **orientations.txt**.

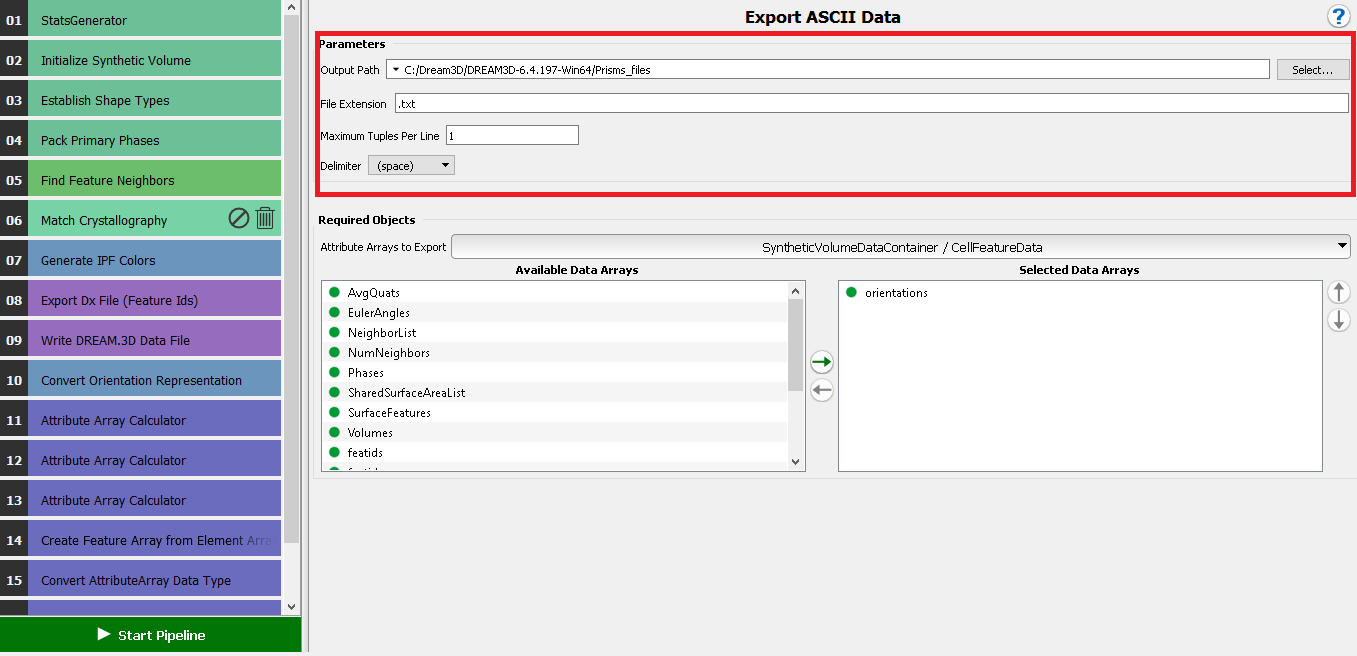


Fig. 6

1. Click on **Start pipeline** (Fig. 6).

## **Viusalization of microstructure in Paraview**

To visualize the outputs from CPFE we can go through the following steps.

1. Browse to **ParaView-5.5.0-RC4-Qt5-Windows-64bit** -> **ParaView-5.5.0-RC4-Qt5-Windows-64bit** -> **bin** in the main directorycontaining the paraview installation, and double-click on the **Paraview icon** .
2. Then click on **File-> Open,** and choose the file with **.xdmf** extension that you wish to visualize, in this case, **Magnesium.xdmf**. A window will pop up asking for a reader to open it with. Choose **Xdmf Reader** and the file will get loaded in Paraview.
3. The filename is visible in a small section to the left hand side of the window. Next, click on **Apply** under the **Properties** tab. Additionally, it has next to it the symbol of a closed eye, which means that it is visible or not. If it is not, click on the closed eye icon to open it. Fig. 7 depicts the window before and after opening the file.

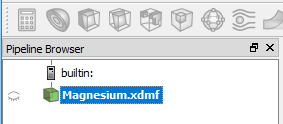
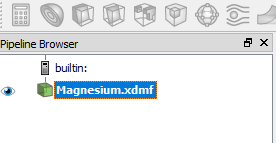
 

Fig. 7

1. In the toolbar, there is a drop down menu that reads **Solid Color**. Click on that menu and choose **Feature Ids**. Adjacent to it is another drop down men that reads **Outline**. Click on it and choose **Surface**. The microstructure is now visible with a colorbar for reference, which denotes the grain IDs (Fig. 8).

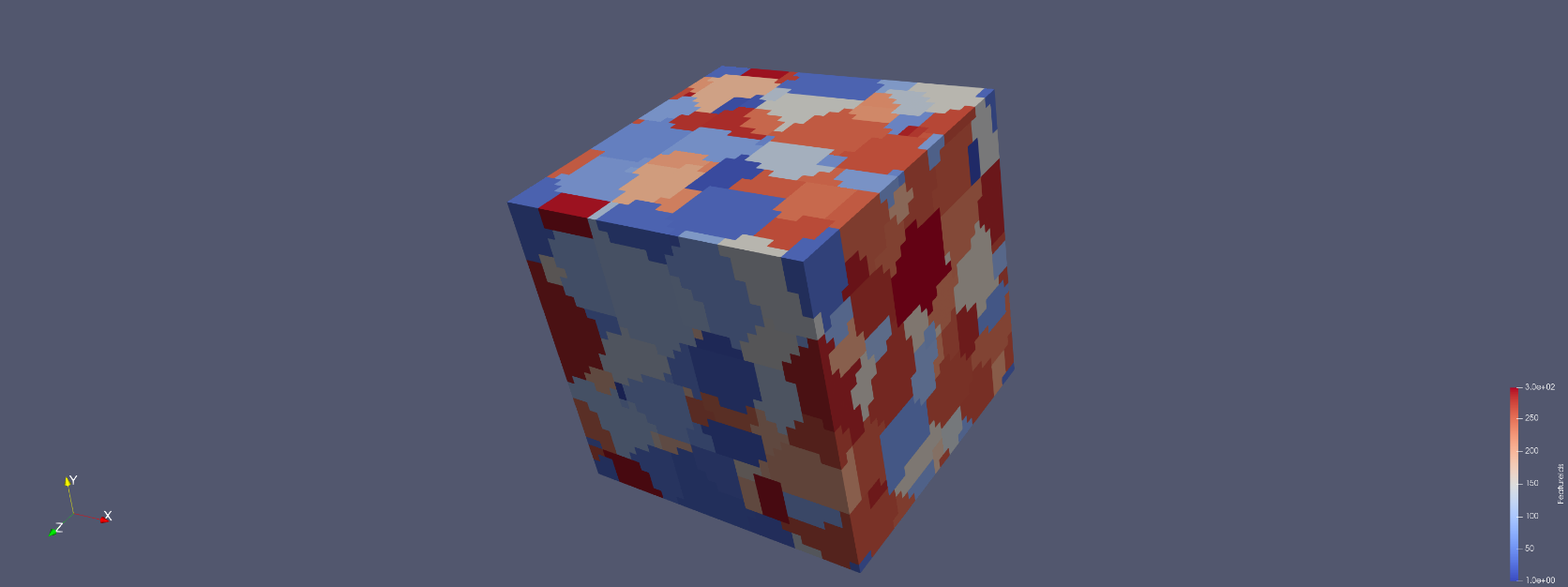


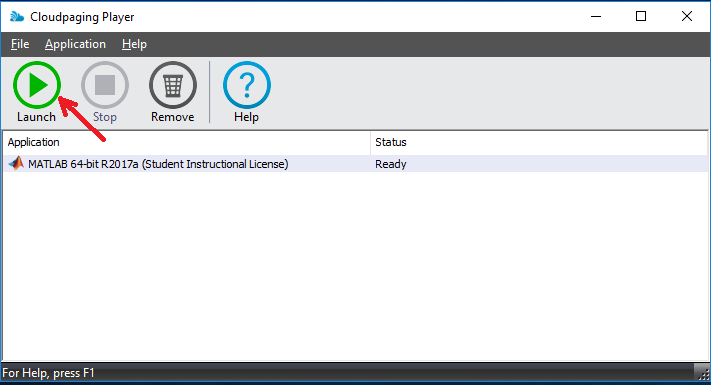
Fig. 8

# **MATLAB and Set Path**

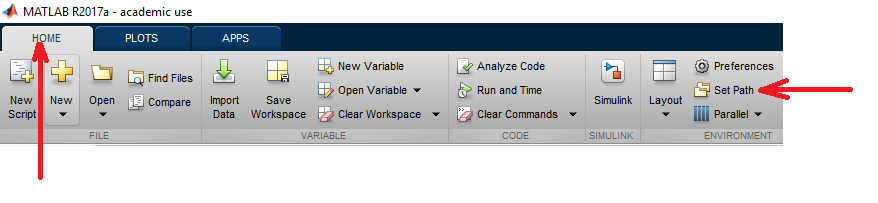
1. In Google Chrome you should already have a tab that reads **CAEN Software**. Else open a new tab and in the address bar type [**https://appsanywhere.engin.umich.edu/**](https://appsanywhere.engin.umich.edu/)**.** This should load the webpage.
2. In the top right corner, a search bar is present that by default reads **Search Apps**. Click there and type MATLAB. A list of software packages will appear in the space below. Look for the application **MATLAB 64-bit R2017a** and click on the button **Launch.**
3. The taskbar at the bottom of the screen appears, as in the following image. Click on the blue icon denoted by the arrow.



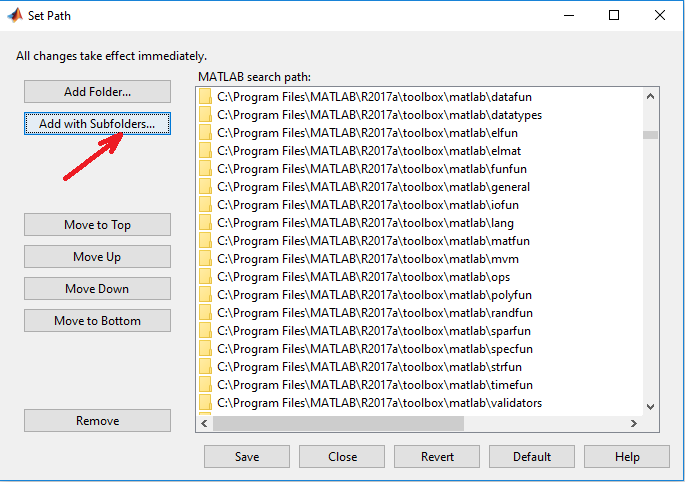
1. A prompt should appear as in the following image. Click on the green icon that shows **Launch.** This should start MATLAB on your PC, which would take a couple of moments to initialize.

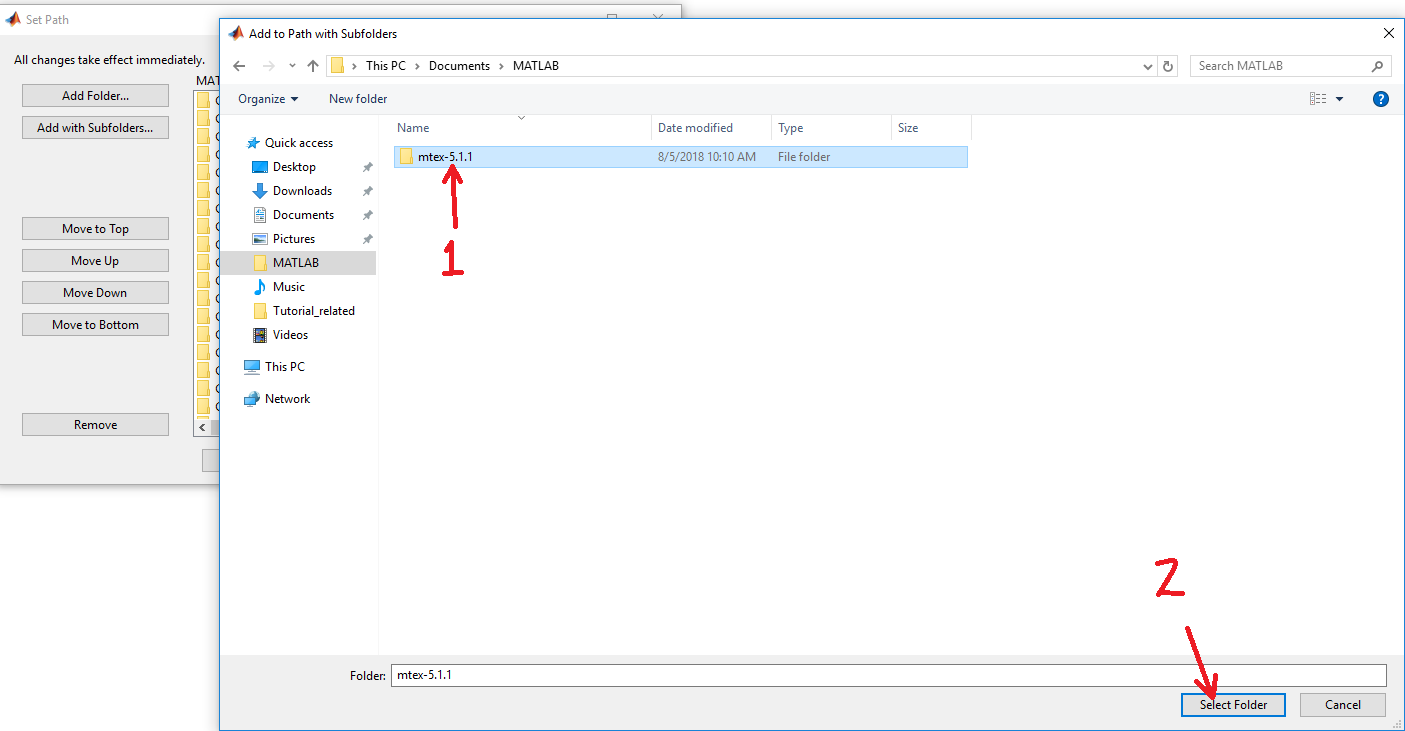


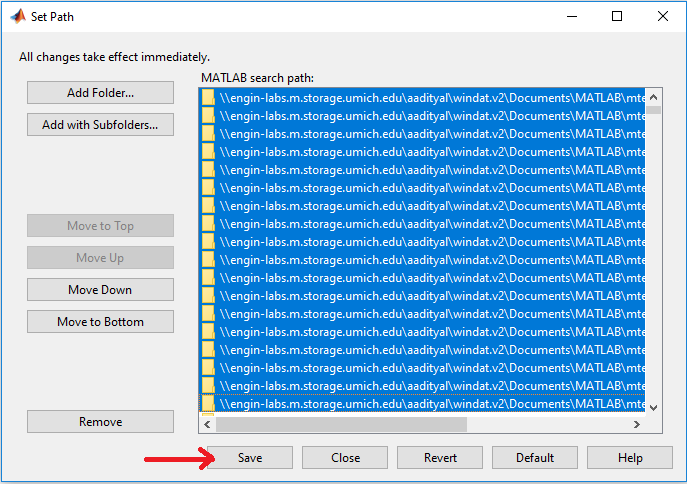
1. Once initialized we need to add the MTEX directory to the MATLAB path. Click on the **HOME** tab on the top left corner of the MATLAB window, and on the icon **Set Path** as depicted in the figure below.



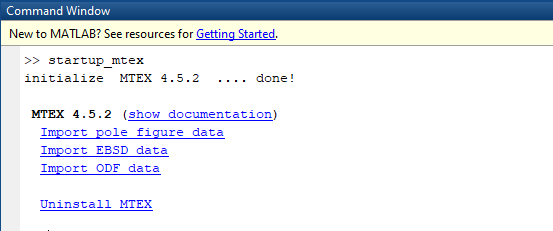
1. A window will then appear to modify MATLAB’s search path. Click on the button **Add with Subfolders…** and then choose the folder to be added. In this case, the folder to be added is **path-to-mtex/mtex-5.1.1.** Choosing that folder will add it and all its subfolders to the MATLAB search path. Then click on **Save** to retain those settings permanently. This is depicted in the figure below.







1. Then in the MATLAB command window type startup\_mtex, to initialize MTEX. The command prompt will look as follows.

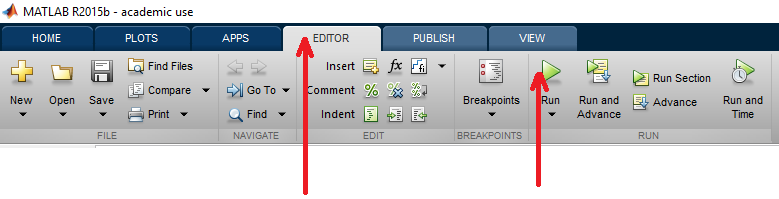


This completes the process of opening MATLAB and adding the MTEX directory so that we can readily access it’s functions. Now change the MATLAB working directory to **Training\_Materials->Pre-Processing->HCP.**

# **Pole Figures with MTEX – Experimental data for Magnesium**

Here we generate the pole figures for the microstructure generated by DREAM3D.

1. Copy the **plot\_polefromori\_Mg.m** script from **PRISMS\_workshop\_final\PRISMS\_workshop\_final\Input\HCP**. Ensure that the files **mgdata.txt** and **plot\_polefromori\_Mg.m** are in the same folder.
2. Click on the **EDITOR** tab and run the script by clicking on the green button denoting **Run** on the toolstrip**,** as shown in the following image.



That should generate the pole figures from the available orientation data from experiments.

# **Pole Figures with MTEX – DREAM3D RVE for Magnesium**

Here we generate the pole figures for the microstructure generated by DREAM3D.

1. Copy the **polefigure\_initial\_Mg.m** script from **PRISMS\_workshop\_final\PRISMS\_workshop\_final\Input\HCP**. Ensure that the files **grainID.txt**, **orientations.txt** and **polefigure\_initial\_Mg.m** are in the same folder.
2. Open the script **polefigure\_initial\_Mg.m** in MATLAB. In lines 5, 6 and 7 ensure that the number of voxels in the x, y and z directions matches with those specified in the line 1 of **grainID.txt**.

% Number of voxels in x,y and z directions

xnum = 32 ;

ynum = 32 ;

znum = 32 ;

1. Ensure the crystal symmetry and sample symmetry appear as follows

%% Specify crystal symmetry. In this case it is Magnesium

cs = crystalSymmetry('6/mmm', [3.21 3.21 5.213], 'X||a\*', 'Y||b', 'Z||c\*');

%% Specify sample symmetry

ss = specimenSymmetry('triclinic');

1. The <0 0 0 1>, <-1 0 1 0> and <-2 1 1 0> are computed here and appear as the following lines in the script

%% Calculate pole figure

pf1 = calcPoleFigure(odf,Miller({0,0,0,1},ori.CS),'resolution',2\*degree,'complete') ;

pf2 = calcPoleFigure(odf,Miller({-1,0,1,0},ori.CS),'resolution',2\*degree,'complete') ;

pf3 = calcPoleFigure(odf,Miller({-2,1,1,0},ori.CS),'resolution',2\*degree,'complete') ;

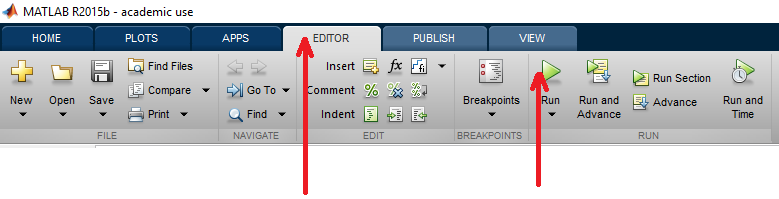
The result of the run is the <0 0 0 1>, <-1 0 1 0> and <-2 1 1 0> pole figures for the initial microstructure. The miller indices can be changed to anything else by just changing one line of code. For example, if we need the <1 -3 2 1> pole figure, the command for pole figure calculation would look as follows

pf = calcPoleFigure(odf,Miller({1,-3,2,1},ori.CS),'resolution',2\*degree,'complete') ;

# **Pole Figures with MTEX – Experimental data for Copper**

Here we generate the pole figures for the microstructure generated by DREAM3D.

1. Copy the **plot\_polefromori\_Cu.m** script from **PRISMS\_workshop\_final\PRISMS\_workshop\_final\Input\FCC**. Ensure that the files **copperdata.txt** and **plot\_polefromori\_Cu.m** are in the same folder.
2. Click on the **EDITOR** tab and run the script by clicking on the green button denoting **Run** on the toolstrip**,** as shown in the following image.



That should generate the pole figures from the available orientation data from experiments.

# **Pole Figures with MTEX – DREAM3D RVE for Copper**

Here we generate the pole figures for the microstructure generated by DREAM3D.

1. Copy the **polefigure\_initial\_Cu.m** script from **PRISMS\_workshop\_final\PRISMS\_workshop\_final\Input\FCC**. Ensure that the files **grainID.txt**, **orientations.txt** and **polefigure\_initial\_Cu.m** are in the same folder.
2. Open the script **polefigure\_initial\_Cu.m** in MATLAB. In lines 5, 6 and 7 ensure that the number of voxels in the x, y and z directions matches with those specified in the line 1 of **grainID.txt**.

% Number of voxels in x,y and z directions

xnum = 32 ;

ynum = 32 ;

znum = 32 ;

1. Ensure the crystal symmetry and sample symmetry appear as follows

%% Specify crystal symmetry. In this case it is Magnesium

cs = crystalSymmetry('cubic');

%% Specify sample symmetry

ss = specimenSymmetry('triclinic');

1. The <0 0 1>, <0 1 1> and <1 1 1> are computed here and appear as the following lines in the script

%% Calculate pole figure

pf1 = calcPoleFigure(odf,Miller({0,0,1},ori.CS),'resolution',2\*degree,'complete') ;

pf2 = calcPoleFigure(odf,Miller({0,1,1},ori.CS),'resolution',2\*degree,'complete') ;

pf3 = calcPoleFigure(odf,Miller({1,1,1},ori.CS),'resolution',2\*degree,'complete') ;

The result of the run is the <0 0 1>, <0 1 1> and <1 1 1> pole figures for the initial microstructure. The miller indices can be changed to anything else by just changing one line of code. For example, if we need the <1 2 1> pole figure, the command for pole figure calculation would look as follows

pf = calcPoleFigure(odf,Miller({1,2,1},ori.CS),'resolution',2\*degree,'complete') ;