# Table of Contents

```
% This code is copyright Alex Foden and Ben Britton 09/04/2019
% Do not distribute.
%
% This RTI-EBSD (refined template indexing) code and its associated scripts
% may only be shared with express and direct permission of
% Alex Foden & Ben Britton
%
% If you would like a copy or access to the repository
% Please contact b.britton@imperial.ac.uk or a.foden16@imperial.ac.uk
%
% Ben Britton has a list of authorised users of this code
%
% Do not fork
%
% THE SOFTWARE IS PROVIDED "AS IS", WITHOUT WARRANTY OF ANY KIND,
% EXPRESS OR IMPLIED, INCLUDING BUT NOT LIMITED TO THE WARRANTIES
% OF MERCHANTABILITY, FITNESS FOR A PARTICULAR PURPOSE AND
% NONINFRINGEMENT. IN NO EVENT SHALL THE AUTHORS OR COPYRIGHT HOLDERS
% BE LIABLE FOR ANY CLAIM, DAMAGES OR OTHER LIABILITY, WHETHER IN AN
% ACTION OF CONTRACT, TORT OR OTHERWISE, ARISING FROM, OUT OF OR IN
% CONNECTION WITH THE SOFTWARE OR THE USE OR OTHER DEALINGS IN THE SOFTWARE.

% Requirements:
% MATLAB R2018a or above
% MTEX version 5.2.beta2 or above
% Created by Alex Foden and Ben Britton 28/03/2019
% If you are using a CIF file not in the MTEX toolbox, you will need to add
% the full file path to the cif file to the phase file you are using

clear % clear any previous data
home % clear the command window
close all %Close all open figures
time1=clock; %create a clock for pTime to work
```

# setup

```
RTI_loc='C:\Users\af2416\Documents\GitHub\RTI_to_share\'; % Change to
 RTI directory
Astro_loc='C:\Users\af2416\Documents\GitHub\AstroEBSD\'; % Change to
 AstroEBSD location
MTEX_loc='C:\Users\af2416\Desktop\mtex-5.2.beta2\'; %change the MTEX
 location
```

# Start the three sub programs

```
run(fullfile(RTI_loc,'start_RTI.m')); %start up RTI
run(fullfile(MTEX_loc,'startup_mtex.m')); %start MTEX
run(fullfile(Astro_loc,'start_AstroEBSD.m')); %start astro
```

```
Loading Refined Template MatchingrTemplateMatch file paths loaded
initialize MTEX 5.2.beta2  .... done!

<strong>MTEX 5.2.beta2</strong> (<a href="matlab:MTEXdoc('mtex')">show
 documentation</a>)
  <a href="matlab:import_wizard('PoleFigure')">Import pole figure
 data</a>
  <a href="matlab:import_wizard('EBSD')">Import EBSD data</a>
  <a href="matlab:import_wizard('ODF')">Import ODF data</a>

  <a href="matlab:install_mtex">Install MTEX for future sessions</a>

Loading AstroEBSD 1.0  AstroEBSD file paths loaded
```

# Inputs

```
InputUser.HDF5_folder='V:\Example_Data'; %folder location for the BCF/
HDF5 file
InputUser.EBSD_File = 'Demo_Ben'; %name of the BCF/HDF5 file
InputUser.Phase_Input  = {'Ferrite'}; %name of the phase you are
 indexing

Bin_loc = fullfile(RTI_loc,'masterpatterns'); %location of the binary
 files used for
Phase_Folder = fullfile(Astro_loc,'phases'); %location of the
 AstroEBSD phases

%Set the screen size
screensize = 128; %size of the library patterns and the resize of the
 raw EBSPs

%Set the SO(3) sampling freq
Sampling_Freq=7; %in degrees

%Select the type of correlation you want to use. 1 = NDP, 2 = FFT
XCF_type = 2; %Need to remove this variable as we now only use the FFT
```

```
%Set the number of iterations to do in the refinement step
iterations = 4;

%LPT size used
LPTsize = 500; %in pixels
```

# Set up backgroud corrections for reference image

```
%From AstroEBSD

%background correction
Settings_CorX.gfilt=1; %use a high pass filter (do you mean high
 pass?)
Settings_CorX.gfilt_s=5; %low pass filter sigma

%radius mask
Settings_CorX.radius=0; %use a radius mask
Settings_CorX.radius_frac=0.85; %fraction of the pattern width to use
 as the mask

%hold pixel
Settings_CorX.hotpixel=1; %hot pixel correction
Settings_CorX.hot_thresh=1000; %hot pixel threshold

%resize
Settings_CorX.resize=1; %resize correction
Settings_CorX.size=screensize; %image height

Settings_CorX.RealBG=0; %use a real BG
Settings_CorX.EBSP_bgnum=30; %number of real pattern to use for BG

Settings_CorX.SquareCrop = 1; %make square the EBSP

Settings_CorX.SplitBG=1; %deal with a split screen
```

# now run the code

```
pTime('Starting RTI matching',time1);
RTI_run;
```

```
Time since start =  [  0 d  0 h  0 m  7s] - Starting RTI matching
Time since start =  [  0 d  0 h  0 m 10s] - Reading HDF5 / BCF data
Time since start =  [  0 d  0 h  0 m 11s] - Updating the PC map to
 adjust for square data
Time since start =  [  0 d  0 h  0 m 11s] - Template library
 generation
Starting parallel pool (parpool) using the 'local' profile ...
connected to 20 workers.
Time since start =  [  0 d  0 h  2 m  6s] - Matching patterns
```

# Save the data

```
pTime('Saving RTI data',time1);
RTI_save;
```

```
Time since start =  [  0 d  3 h 19 m 14s] - Saving RTI data
Overwriting /Demo_Ben/EBSD/Data/DD
Overwriting /Demo_Ben/EBSD/Data/PCX
Overwriting /Demo_Ben/EBSD/Data/PCY

Undefined function or variable 'RTI_peakheight'.

Error in RTI_save (line 62)
RTI_peakheight=RTI_peakheight(:); %form as a column

Error in deck_iron (line 106)
RTI_save;
```

# Now read the H5 File

```
%lots of warnings will appear - these can be ignored!
[ RTI_MapData,RTI_MicroscopeData,RTI_PhaseData,RTI_EBSPData ] =
 bReadHDF5( OutputUser );

%form into a 2D map
% [RTI_Data] = EBSD_Map(RTI_MapData,RTI_MicroscopeData);

%rebuild the phase data
[~,~,~,~,~,RTI_newinfo ] =
 Phase_Builder_RTI( OutputUser.Phase_Input,Phase_Folder, Bin_loc );

%load the crystal structure from the CIF file - this will be used for
 the symmetry elements
cs=loadCIF(RTI_newinfo.cif_file); %CIF file for the cyrstal you aare
 indexing. This is for symmetry in generation of the library

%set MTEX properties
propsT.x=RTI_MapData.XSample;
propsT.y=RTI_MapData.YSample;
propsT.quality=RTI_MapData.PeakHeight;
phasesT=propsT.quality*0;

rot_Template=rotation('Euler',RTI_MapData.phi1*degree,RTI_MapData.PHI*degree,RTI_M

EBSD_template=EBSD(rot_Template,phasesT,cs,'options',propsT);
```

# Now plot the MTEX data

```
setMTEXpref('xAxisDirection','west');
setMTEXpref('zAxisDirection','outOfPlane');

%plot the peak height map
```

```
figure;
plot(EBSD_template,EBSD_template.prop.quality)
colorbar;

oM=ipfTSLKey(EBSD_template(cs.mineral));

%plot the IPF maps
figure;
oM.inversePoleFigureDirection = xvector;
plot(EBSD_template(cs.mineral),
 oM.orientation2color(EBSD_template(cs.mineral).orientations))
title('IPF - X')

nextAxis;
oM.inversePoleFigureDirection = yvector;
plot(EBSD_template(cs.mineral),
 oM.orientation2color(EBSD_template(cs.mineral).orientations))
title('IPF - Y')

nextAxis;
oM.inversePoleFigureDirection = zvector;
plot(EBSD_template(cs.mineral),
 oM.orientation2color(EBSD_template(cs.mineral).orientations))
title('IPF - Z')

pTime('Code complete',time1);
```

*Published with MATLAB® R2018b*