# Description of MTEX twin analysis code

## Introduction

MTEX has become a powerful, widely used texture microstructure characterization tool with the latest release having over 4000 downloads from github. In addition to its extensive functionality and documentation, MTEX has an advantage over many other codes for microstructure characterization in that its code is ready to use in Matlab without special environment setup or compiling, and it is easily understood and debugged in the Matlab environment. Many students are already familiar with Matlab due to extensive use in Academic and research institutions. In parallel to the rise of MTEX there have been significant advances in automated twin characterization presented in recent papers such as (Pradalier, Juan et al. 2018) and (Marshall, Proust et al. 2010). Incorporating the functionality described in these papers into the MTEX framework could be major boon to researchers who have already invested in learning MTEX and would provide a platform to quickly prototype advances in twin analysis. The final motivation for developing a twin analysis code in MTEX is that the code will benefit from community support – i.e. in finding bugs, community explanation on using the code, and extending functionality. The community framework will likely result largest impact on the microstructure characterization field.

## Twin definition

The twin definitions are handled by two sets of inputs. The first is the crystal symmetry which is defined for pure titanium as



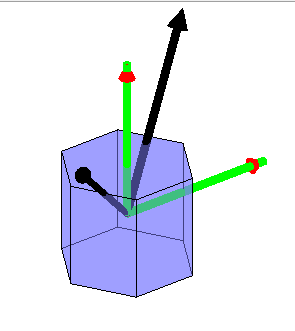
where the ‘notIndexed’ phase in CS{1} is used for EBSD with unindexed data. The twinning frame is defined with respect to the a-axis and c-axis of the unit cell. These values are used in computing the misorientation, variants, etc… for each twin mode.

The second input are the twin specific parameters. The parameter actType defines if it is a type 1 or type 2 twin, k1 is the twin plane, eta1 is the shear direction in k1, name is used during plotting, and the tnum is the index of the twin.



Care should be taken when selecting the plane and direction used to define k1 and eta1. Specifically, if the k1 direction is pointing in the opposite direction of the c-axis, assumptions about the sign of the Schmid factor for the parent and twin will be inverted. The definition can be easily checked using the following code wherein the Rtw twin rotation matrix which transform the black k1 and eta1 vectors defining the twin frame into the crystal frame.





The transformation between the twin and crystal frame along with the misorientation, twin slip system (corresponds to the variants), axis, angle, and variants are defined in a call to



in which the following definitions are used



The tType specifies whether a rotation around k1 (tType{1}) or eta1 (tType{2}) will be used.

Reference the work of Li et. al 2015 for the twin definitions along with the work of the twin table guys for list of common twins in various systems. Adding that the benefit of this method for defining twins are that the axis and angle are taken care of by an appropriate cell definition.

## Applied stress

The applied stress used during Schmid factor calculations is defined as the second order tensor sigma.



## Building the graph data structure and stored parameters

### Description of edge and nodes

### Description of accessing variables

## Initial clustering of grains

### Options: mean orientation, seeded mean orientation, boundary

## Group fragments into families

## Weight family Votes

### Describe each weighted value

## Family tree cleanup

Inputs: defined for each cluster of grains formed by merging boundaries

nFamily: the Family Id linking fragments of similar orientation

nId: the fragment id consistent all the way back to the original graph

eType: each edge at this point is a twin, eType the twin mode

eVote: the edge vote is based on the voting scheme weighting Schmid, boundary ratio, etc..

ePairs: A list of all pairs nx2 where entries are the nId

eGlobalId: A global edge idea goes back to the original graph

Move this next section to a function

The first step is to compute the family relationship list. This list is a max(nFamily) x max(eType) cell array each cell containing a logical array of size(ePairs) that is only true if the Family id and edge type are both true.

The next step is to sum the votes for a given family and edge type using the family relationship list to index eVote. The resulting array eVotesSummed is a double array of size max(nFamily) x max(eType) containing a vote for each edge. If multiple edges share the familyId and eType they receive the same vote. The Parent array is a logical array of size(ePairs) defining which node of an edge is the parent node and which is the twin node.

Move this next section to a function

### Description of dealing with circular relationships and incorrectly added grains

Add some special twin filtering here

## Twin Filtering

## Manual Corrections

### Changing Parent

### Deleting edge

### Adding edge (twin)

## Build Family Tree

Algorithm used

## Twin Geometry Reconstruction

Algorithms used

## Twin Statistics

Various metrics that are computed

## References

Marshall, P. E., G. Proust, J. T. Rogers and R. J. McCabe (2010). "Automatic twin statistics from electron backscattered diffraction data." J Microsc **238**(3): 218-229.

Pradalier, C., P.-A. Juan, R. J. McCabe and L. Capolungo (2018). A Graph Theory-Based Automated Twin Recognition Technique for Electron Backscatter Diffraction Analysis.