PROGRAM POLE (version 8d → POLE8d)

Version 20/MAY/2014
****

Developed by: Carlos Tomé
MST-8 MS G755
Los Alamos National Laboratory
Los Alamos NM 87545 USA
e-mail: tome@lanl.gov

GNUPLOT script by: Arul Kumar Mariyappan (May 30, 2014)
****

1- GENERAL DESCRIPTION

POLE8 is a double-precision Fortran code for calculating pole figures and inverse pole figures. User has to provide a Texture File, consisting of a discrete set of crystal orientations, defined by their associated Euler angles and volume fractions. The poles to be plotted are read from the input file POLE8.IN either in three-index notation (for cubic, tetragonal, orthogonal, monoclinic & triclinic crystals) or four-index notation (for hexagonal & trigonal crystals).

POLE8d also has the capability of plotting pole figures from an Orientation Distribution Function *.SOD file generated by popLA (or having the same format as popLA files). It is also possible to plot experimental pole figures with popLA format: *.WPF, *.EPF, *.GPF (pole figures given by intensities in a polar grid).

POLE8 creates a set of output ASCII files with (X,Y) coordinates for importing into and plotting with any graphic software. The strategy is to use POLE8 as a purely numerical code which generates (X,Y) data files for plotting with any graphic software available to the user (such as ORIGIN or GNUPLOT or EXCEL). In this way, the code becomes independent of the graphic hardware/software, except for the requirement of having a Fortran compiler (I use GFORTRAN) and graphic software (I use GNUPLOT and ORIGIN).

The executable version of POLE8 is not included in the distribution because of filter issues and because it is dependent on the operating system and hardware. I run Windows 7 in a 64 Byte PC. I compile and run POLE8d from the Command Window. When you compile the source code you may need to run it from Windows depending on the compiler used.

2 – COMPILING/RUNNING POLE8 and USING GNUPLOT FOR PLOTTING OUTPUT

Step#1
For compiling pole8 with GFORTRAN open the Command Window in the directory where POLE8.FOR resides and type

gfortran pole8.for –o pole8

(Works for both Windows and Mac OS)

The files provided should compile without error or warnings using the above, and should create an executable named

pole8.exe *(for Windows OS)*
pole8 *(for Mac OS; Unix executable file)*

It should not be necessary, but you can add compilation options such as

-fno-align-commons, -fdefault-real-8, -fdefault-double-8,

(Works for both Windows and Mac OS)

Step#2
Before running the code make sure that you have entered the correct information in the input file **POLE8.IN**, with the right path and name for the texture files to plot.
To run the pole8 code, type:

pole8.exe *(for Windows OS)*

./pole8 *(for Mac OS)*

in the same command terminal. Then it will ask to enter a five character label (e.g., DUMMY) and some plot parameters.
At the end of the run you will see the output files DUMMY***.*** along with the GNUPLOT script file (DUMMY.plt)

Step#3
To plot the pole figure(s) or Inverse Pole Figure(s) you need to open GNUPLOT. To do so type
start gnuplot *(for Windows OS)*
gnuplot *(for Mac OS)*
It will lead to the gnuplot terminal (window). Once in the gnuplot terminal type
load 'DUMMY.plt' *(for Windows OS)*
load "DUMMY.plt" *(for Mac OS)*

It will display the PF or IPF in the monitor and also save the figure in the same directory as an Encapsulated Post Script (EPS) file **DUMMY.eps**

3- CODE CAPABILITIES

- Reads texture files in Bunge, Kocks or Roe conventions of Euler angles.
- Plots dots or intensity lines. When plotting intensity lines divides the Pole Figure or Inverse Pole Figure in cells using a grid, assigns weights to the cells of the grid, calculates intensities associated with the center of each cell, and interpolates intensities to derive isointensity lines.
- Plots equal area or stereographic projections.
- When plotting lines uses a (phi,theta) or a (phi,cos(theta)) grid.
- Optional symmetrization of the original poles: centro-symmetry (inversion), X mirror plane, Y mirror plane, X & Y mirror planes (orthotropic), axisymmetry (cylindrical).
- 'Gaussian' spreading of the poles for smoother representation.
- Plots intensity lines at equal increments, at geometric increments, or for arbitrary intensity values.
- Plots pole figures for all crystal symmetries (cubic, hexag, trigo, tetra, ortho, monoc, tricl).
- Plots inverse pole figures for arbitrary crystal symmetry.
- Can read several files containing several textures each, and can plot several poles for each texture.
- Converts Orientation Distribution Function (popLA format *.SOD) into discrete texture file and plots poles.
- Reads and plots experimental pole figures (popLA format *.EPF, *.WPF, *.GPF)

4 - USER DEFINED PARAMETERS

User has to provide plotting parameters using one of the following options:

- Interactively through the screen using mostly default values (**default**=0).
- Interactively through the screen choosing every parameter (**default**=1).
- Reading the parameters from a batch file (POLE8.PRO) which is created in each run, contains the parameters of the run, and can be edited by the user if desired (**default**=-1).

The following are the options for representing Pole Figures and Inverse Pole Figures. Most parameters control plotting of intensity lines and do not apply when plotting dots. Default is always the zero value of the parameter when appropriate.

- **mgrid, ngrid**: partition of [0< phi <2π] and [0< theta <π/2] intervals to be used in the PF or IPF. PF default is: mgrid=36, ngrid= 9, corresponding to 10 deg cells. IPF default is: mgrid=72, ngrid= 18, corresponding to 5 deg cells.

- **spread**: introduces a Gaussian spread around each pole which 'smoothes' the PF. **spread** represents the solid angle (in degrees) swept by the Gaussian. Default is spread=π /mgrid.

- **icros**: plot crosses in PF cells where intensity is below the minimum (icros= 0). If (icros=1) it does not plot crosses. Default is icross=0.

- **crysym**: this parameter is read from single crystal file and gives information about the crystal symmetry: **crysym**=CUBIC, HEXAG, TRIGO, TETRA, ORTHO, MONOC, TRICL identifies cubic, hexagonal, trigonal, tetragonal, orthogonal, monoclinic, triclinic, respectively.

- **ifull**: plot pole figure using a full circle (ifull=0) or plot only a reduced circular section depending on the PF symmetrization defined by **isym** (ifull=1). Default is ifull=0.

- **igrid**: use equal partition of (phi,theta) (igrid=0) or (phi, cos(theta)) (igrid=1) in polar grid. Default is igrid=0.

- **iper1, iper2, iper3**: the user is prompted to enter the position of the sample axes in the PF. Specifically: which axis goes at the right (iper1), which at the top (iper2) and which at the center of the PF (iper3). For example, it is usual to plot rolling textures with the TD at the right, the RD at the top, and the ND at the center of the pole figure. If the texture is referred to a set of axes where RD=1, TD=2, and ND=3 then one would enter iper1=2, iper2=1, iper3=3. It is allowed to enter negative axes, such as iper1=-2 or iper3=-3, for example.

- **ipfig**: controls whether plotting a Direct PF or an Inverse PF from a discrete texture file, or a *.SOD, or a *.EPF file.

  - if ipfig=0 plots a direct PF using the poles read from POLE8.IN and the orientations read from one or more discrete texture files.

  - if ipfig=1 the vectors (100) or (010) or (001) read from POLE8.IN will be plot in an Inverse PF. Reads orientations from discrete texture file(s)

  - if ipfig=-1 reads an Orientation Distribution Function (popLA format *.SOD), creates three discrete texture files based on a large regular Euler grid (*.FUL), a small regular Euler grid (*.RED), and an intermediate random Euler grid (*.RAN). Afterwards, it plots the poles passed by the user through POLE8.IN.
    The default (iformat=0) is for ODF files with popLA format and is hardwired inside SUBROUTINE READ_POPLA_SOD. Internally, there is an option for ODF files with Beartex format (iformat=1) or with OIM format (iformat=2). Neither option is thoroughly tested and user needs to activate them inside the subroutine.

  - if ipfig=-2 reads intensities of experimental poles associated with a regular polar grid (popLA format *.EPF) and plots them. Assumes intensity given every 5° in the intervals 0°≤φ≤355° and 0°≤ ɵ ≤90° and sets mgrid=72 , ngrid=18.

- **iproj**: use equal area projection (iproj=0) or stereographic projection (iproj=1). Default is iproj=0.

- **irepr**: represent PF or IPF using intensity lines (irepr=0) or points (irepr=1). Default is irepr=0.

- **isepa**: plot intensity levels using equal increments on a linear scale (isepa=1) or equal increments on a logarithmic scale (isepa=0). Default is isepa=0.

- **step**: increment used to define the intensity line values. Default is step=0.5.
  isepa=0 → geometric separation: level(l)=2^(step*(l-2))
  isepa=1 → arithmetic separation: level(l)=step*l

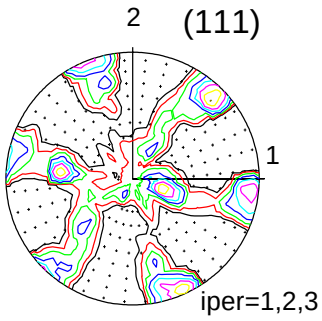
- **ishft**: introduce a half-cell rotation in phi in order to make possible for maxima to coincide with the main axes (ishft=1) or do not rotate (ishft=0). Default is ishft=0.
- **ismth**: a Gaussian filter smooths the central part of the PF. Used mostly to avoid the rather artificial 'spiky' appearance at the center. Default is ismth=0.
  ismth=0 → does not do any smoothing (default)
  ismth=1,2,3…10 → represent increasing levels of smoothing

- **isym**: enforces a specific symmetrization in the PF (both, points or intensity line representation). Default is isym=0.
  isym=0 : do not symmetrize the PF
  isym=1 : enforce a centro-symmetric pole figure
  isym=2 : enforce a mirror plane perpendicular to the horizontal axis of the PF
  isym=3 : enforce a mirror plane perpendicular to the vertical axis of the PF
  isym=4 : enforce two mirror planes (orthotropic)
  isym=-1 : enforce axial symmetry of the PF (usually used for deriving volume fractions)

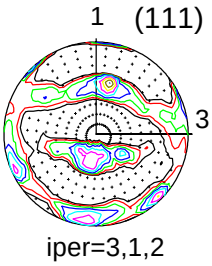
- **pfrot**: a rotation angle in degrees. Allows user to rotate the poles around the center of the PF. Used to 're-position' experimental pole figures when the sample has not been properly positioned in the goniometer. Default is pfrot=0˚.

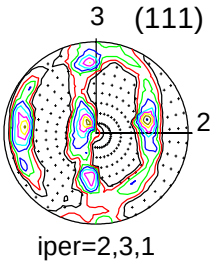
EXAMPLES OF HOW CHANGING PLOT OPTIONS AFFECTS POLE FIGURES

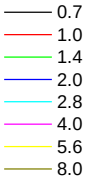
Copper (4030 orientations) - ECAE - POLE8 parametric options



PERMUTATION OPTIONS



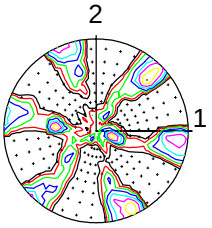


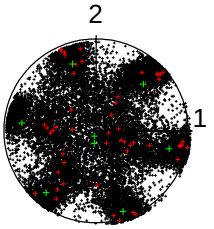


Default Parameters

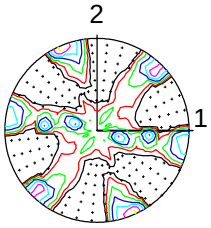
ifull=0	ishft=0
igrid=0	ismth=0
iproj=0	isym=0
irepr=0	pfrot=0
isepa=0	step=0.5

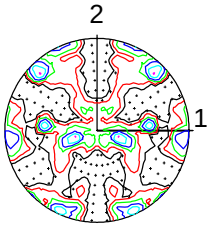
PROJECTION OPTIONS

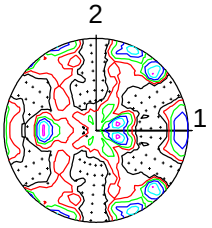


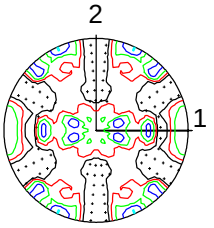


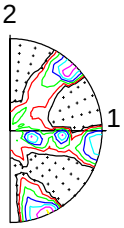
SYMMETRIZATION OPTIONS

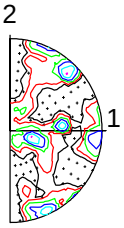


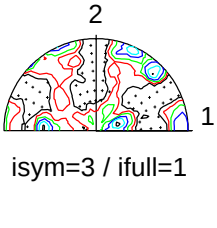












Number of columns and rows to plot
If NTEXTURES and NPOLES are the number of textures read and the number of poles to plot for each texture, then POLE8 arranges the pole figures in a pattern of NROWS (one for each texture) and NCOLS (one for each pole) containing the poles (or viceversa, whichever is larger).

Setting the maximum number of poles to plot
The maximum number of different textures is MAXT, the maximum number of different poles is MAXP, the maximum number of orientations in each texture file is NGMAX. They are defined as 36, 36 and 70000, respectively, in POLE8.DIM (shown below). These parameters can be changed if necessary and the code needs to be recompiled.

```
c ************************************************************************
c     POLE8.DIM      --->      VERSION OF 04/MAR/2009
c     COMMON STATEMENTS AND PARAMETERS TO BE INCLUDED DURING COMPILATION
c ************************************************************************

      parameter (ngmax= 70000)      ! maximum # of grains in texture file
      parameter (nlmax=    15)      ! maximum # of level lines
      parameter (npmax=400000)      ! maximum # of poles in pole figure
      parameter (nsmax= 50000)      ! maximum # of segments forming intensity lines
      parameter (maxt=36)       ! maximum number of different textures to plot
      parameter (maxp=36)       ! maximum number of different poles to plot
```

5- INPUT FILES

POLE8 requires three input files: POLE8.IN, a texture file and a crystal file. Optionally, POLE8.PRO may be used as the parameter input file (see description in OUTPUT files)

****

POLE8.IN

(Notice that this file contains information about several crystal symmetries. Only one symmetry is used at a time. The corresponding lines have to be moved 'up-front'. See examples)

****

```
* number of texture files 'ntextfiles' to read below
 3
* path & name of texture files (one per line)
c:\textures\mg\extr8378.tex
c:\textures\mg\1616.tex
c:\textures\mg\2679.tex
* index 'ipfig': 0 for pole fig plot, 1 for inverse pole fig plot
  0
* path & name of file with symmetry & cell parameters of single crystal
datasx\mg.sx
* # of crystallographic poles (PF) or sample axes (IPF) to read below
  3
* Miller indices of poles
   0   0   0   1      !  basal pole
   1   0  -1   0      ! prism 1st order poles
   1   0  -1   1      ! pyramidal 1st order poles


******** cut & paste above for different symmetries and different poles **************

* Miller indices of poles for cubics
   1   1   0      !  6   (110) poles
   1   0   0      !  3   (100) poles
   2   1   1      ! 12   (112) poles

* indices of sample axis for (IPF)
   1   0   0      !  x1 axis for IPF
   0   1   0      !  x2 axis for IPF
   0   0   1      !  x3 axis for IPF

* Miller indices of poles for monoclinic
   1   1   1      ! 2   (111) poles
   1   3   1      ! 2   (131) poles
  -1   3   1      ! 2   (-131) poles
   1   0   0      ! 1   (100) pole
   0   1   0      ! 1   (010) pole
   0   0   1      ! 1   (001) pole

* Miller indices of poles for hexag and trigonal
   0   0   0   1   ! 1 basal pole
   1   0  -1   0   ! 3 prism 1st order poles
   2  -1  -1   0   ! 3 prism 2nd order poles
   1   0  -1   1   ! 6 pyramidal 1st order poles
   1   0  -1   2   ! 6 pyramidal 2nd order poles

* Miller indices of poles for orthorhombic
```

```
  1   1   1      ! 4   (111) poles
  1   0   0      ! 1   (100) pole
  0   1   0      ! 1   (010) pole
  0   0   1      ! 1   (001) pole
*******************************************************************************
```

Line 1: reminder
Line 2: number of texture files to read below whose PFs or IPFs are going to be plot. POLE8 allows to plot several poles (up to MAXP) for several textures (up to MAXT). Each texture file may contain one or several textures, separated by appropriate heading.
Line 3: reminder.
Line 4: path and name of texture files. If more than one, use one line for each file listed. When ipfig=-1 this is the path and name of the *.SOD file (only one can be processed in one run). When ipfig=-2 this is the path and name of the *.EPF file (more than one can be processed in one run)
Line 5: reminder
Line 6: parameter 'ipfig' indicating whether Direct or Inverse Pole Figures are plot. And whether they correspond to discrete texture files, *.SOD files, or *.EPF files.
Line 7: reminder
Line 8: path and name of file providing the symmetry and lattice parameters of the single crystal.
Line 9: reminder
Line 10: number of different poles (NPOLES) to be read below and to be plot. When plotting IPF's this number indicates the number of IPF's to be plot.
Line 11: reminder
Line 12+: Miller indices of poles to be plotted (3-index notation for cubic, ortho, tetra, monoc, tricl; 4-index notation for hexagonal and trigonal). These lines contain the coordinates (100) and/or (010) and/or (001) of the sample axes to be plotted when plotting IPF's.

```
*******************************************************************************
```

TEXTURE FILE(s) (path and name declared in POLE8.IN)

```
*******************************************************************************
TEXTURE AT STRAIN =    0.5000
 1.555     1.000     0.630  <-- axes length of macro ellipsoid
90.00     90.00      0.00  <-- Euler angles of macro ellipsoid (deg)
B       500
   -107.05    105.24    -41.51     0.0020000
   -145.57     41.09     64.31     0.0020000
     62.21    127.40   -116.27     0.0020000
    -58.49    123.53    -52.69     0.0020000
    -58.36     73.80    109.54     0.0020000
           ---         ---         ---         ---
*******************************************************************************
```

Line 1: reminder
Line 2: reminder
Line 3: reminder
Line 4: convention used to represent grain orientation (B=Bunge, R=Roe, K=Kocks) and lines that have to be read until end of texture.
Lines 5+: angles phi1, PHI, phi2 (Bunge) and volume fraction (weight) of each orientation

A similar block with the same structure may follow in the same file. POLE8 will read and store all textures until reaching the end of the file. It will then open and read from the next texture file in the list, if appropriate.
WARNING: make sure that the last line in the texture file is not blank! Else, POLE8 will interpret that another texture is to be read and an error will occur.

****

ORIENTATION DISTRIBUTION FILE (ODF) (alternative to texture file when IPFIG=-1)

****

```
demo     Cu rol.90%,pt.reX 13 Bwimv iter: 6.4%Fon=  0 31-MAR-** strength= 2.99
 SODB  5.0 90.0  5.0 90.0 1 1 2 1 3   42     phi1=  0.0
 11941172  15   4   4   0   0   0   0   0   1   0   0   0   1   1 10515781194
   44  10  10   1   4   0   9   9   1   0   1   0   1   3   2   3  17  20  44
   12  23 120  29  13  24   9  13   4  10  14  10  10  74  32  31 304   9  12
   14   9  16   8 107  19  10  26  16   6   8  30   4   0  19   3  14   0  14
    ---     ---         ---         ---
```

****

Line 1: reminder

Line 2: the 4th character provides the convention used for the Euler angles (B=Bunge) or (K=Kocks)

Line 3+: intensities associated with a regular grid in Euler space (phi1, PHI, phi2).

Several blocks follow, one for each parametric value of phi1. See EXAMPLE 5 for full file. When the *.SOD was calculated assuming orthotropic texture, the file contains 4 times less blocks. POLE8 can deal with it if DEFAULT=1 is used.

****

EXPERIMENTAL POLE FILE (alternative to texture file when IPFIG=-2)

****

```
demo     Cu rol.90%,pt.reX 13 BWIMV iter: 6.4%FON=  0 31-MAR-** strength= 2.99
(111)  5.0 90.0  5.0360.0 1 1 2 1 3  100   17     PROJ
    2   1   7   0   2   9   1   3   2   5   6   7   4   3   4   5   3   5
    8   6   8   6   8   5   9   5   5   4   4   5   4   8   4   4   4   5
    3   3   2   2   3   4   7   4   2   2  12   3   2   1   2   3   4   5
   12   5   3   7   1   3   3   8   6   2   0   1   2   0   1   3   5   2
    5  10  22   2  23   5  25  13  30  30  40  57  48 104  94  56 108  31
   30  36  46  62  43  63  91 140  13  69  19  10  12   5  14  15   4  12
    ---     ---         ---         ---
```

****

Line 1: reminder

Line 2: read with FORMAT(1x,3i1,1x,2f5.1) → gives the Miller indices of the poles, the separation in angles of the grid points, and the maximum goniometer angle where intensities were measured (for incomplete pole figures). These are all read and used by POLE8.

Line 3+: intensities associated with the regular polar grid (phi, theta).

Several blocks follow, one for each pole measured for this texture (see EXAMPLE 4 for full file). POLE8 assumes intensity given every 5° in the intervals $0° \le \varphi \le 355°$ and $0° \le \theta \le 90°$.

****

SINGLE CRYSTAL FILE (path and name declared in POLE8.IN)

****

POLE8 reads the symmetry of the single crystal (CRYSYM=cubic, hexag, orhto, tricl, etc) and the parameters of the unit cell, namely, the length of the three vectors (a,b,c) and the angles that they form (α,β,γ). This data is used for calculating the symmetry operations and the normal vectors associated with the different poles.

The single crystal file has the same format used by other codes (VPSC & EPSC). A few examples are shown below. The first line is a reminder. The second lines gives CRYSYM. The third line gives the lattice parameters.

Mg - AZ31B
HEXAGONAL crysym
1. 1. 1.61 90. 90. 120. cdim(i),cang(i)

---

*Material: fcc AUSTENITIC STEEL
cubic crysym
1.0 1.0 1.0 90. 90. 90. unit cell axes and angles

---

* orthrhombic Uranium
orthorhombic
1.0 2.056 1.736 90. 90. 90.

---

* monoclinic uranium
monoclinic
2.906 5.737 4.971 90. 90. 92.89

---

* olivine
trigonal
6.01 4.78 10.3 90. 90. 120.

6- OUTPUT FILES

When launching POLE8 the user is prompted to enter a 5 character string (e.g., DUMMY) for labeling the output files. All output files start with these 5 characters.
All the output files used for plotting consist of two columns of (x,y) data. Different data groups (either another intensity line, or another pole figure, or another texture) are separated by a blank line to indicate a break in the point-joining operation. Examples of plotting files are:

DUMMY_CC.DAT: Contains coordinates for drawing the circle, half-circle, or quarter circle that encloses the PF or IPF. Plot by joining the points consecutively.

```
 0.0000   0.9389
 0.0000   0.0000
 0.9389   0.0000
 0.0000   0.0000
 0.9028   0.0000
 0.9026   0.0158
    .        .
    .        .
```

For line plots:
* **DUMMY_XX.DAT**: Coordinates of points where the intensity in the grid cell is lower than the minimum line intensity to be drawn. Permits to plot dots or crosses in the areas of low intensity, and helps to read the PF.

```
-0.1197  -0.0105
-0.1160  -0.0311
-0.1089  -0.0508
-0.0984  -0.0689
-0.0849  -0.0849
-0.0689  -0.0984
    .        .
    .        .
    .        .
```

* **DUMMYnnn.DAT**: Coordinates of points defining the intensity line of value 'nnn'. Join points sequentially. A blank line in the file separates different intensity lines having the same level.

```
-0.64674   0.00000
-0.64611  -0.01265
-0.64697  -0.05660
-0.62689  -0.08075
-0.61130  -0.10779
-0.59544  -0.13198
    .        .
    .        .
    .        .
```

* **DUMMY_MX.DAT**: a file giving the x,y coordinates of the grid point with maximum intensity, and the intensity value. One line per pole and per texture.

```
0.019 -0.218  9.220
```

```
 2.114 -0.608  8.721
 3.760 -0.401  5.421
```

---

* **DUMMY.PLT:** a file giving the script to plot PF or IPF using GNUPLOT.

```
# GNUPLOT program to plot the output of POLE8.FOR
# POLE8d developed by C. N. Tome
set noborder
set size ratio -1
set key outside
set noxtics
set noytics
set label "( 1 1 1 )" at -0.30,  1.10
set label "( 1 1 0 )" at  1.87,  1.10
set label "( 1 0 0 )" at  4.03,  1.10
set label " 2" at  5.24, -0.00
set label " 1" at  4.27,  1.00
plot "lines_CC.DAT" using 1:2 notitle  with lines lt 1 lc -1, \
"lines007.DAT" using 1:2 title "   0.7" with lines lt 1 lc  1,\
"lines010.DAT" using 1:2 title "   1.0" with lines lt 1 lc  2,\
"lines014.DAT" using 1:2 title "   1.4" with lines lt 1 lc  3,\
"lines020.DAT" using 1:2 title "   2.0" with lines lt 1 lc  4,\
"lines028.DAT" using 1:2 title "   2.8" with lines lt 1 lc  5,\
"lines040.DAT" using 1:2 title "   4.0" with lines lt 1 lc 18,\
"lines056.DAT" using 1:2 title "   5.7" with lines lt 1 lc  7,\
"lines080.DAT" using 1:2 title "   8.0" with lines lt 1 lc  8,\
"lines_XX.DAT" using 1:2 notitle with dots lt 0
set term postscript eps color blacktext
set output "lines.eps"
replot;
```

For dot plots:

* **DUMMY_Xn** (with n=0,9): Coordinates of poles for plotting them as points or crosses (as opposed to intensity lines). The poles are distributed in ten bins, depending on the relative weight of the grain from which they originate. n=0 represents the minimum and n=9 the maximum. The user may want to plot them using symbols proportional to the value 'n'.

```
   0.3456    0.3981
  -0.3776    0.7955
  -0.0199   -0.2831
  -0.7563   -0.0779
   0.1090    0.2791
   0.8145    0.0094
      .         .
      .         .
      .         .
```

---

* **DUMMY.PLT:** a small file with the script to plot POLE or INVERSE POLE FIGURES using GNUPLOT.

```
# GNUPLOT program to plot the output of POLE8.FOR
# POLE8d developed by C. N. Tome
```

```
set noborder
set size ratio -1
set key outside
set noxtics
set noytics
set label "( 1 1 1 )" at -0.30,  1.10
set label "( 1 1 0 )" at  1.87,  1.10
set label "( 1 0 0 )" at  4.03,  1.10
set label " 2" at  5.24, -0.00
set label " 1" at  4.27,  1.00
plot "dotss_CC.DAT" using 1:2 notitle with lines lt 1 lc -1 ,\
“dotss_X9.DAT" using 1:2 notitle with points   lt 1 lc -1
set term postscript eps color blacktext
set output "dotss.eps"
replot;
```

5-1 OUTPUT FILES WHICH MAY NOT BE NECESSARY

POLE8.OUT: it is meant to be read by another program (PLOTPF8 which links with the graphic library PLOT88. Unless you use PLOT88, neither POLE8.OUT nor PLOTPF8.FOR are going to be of much help, except for giving the user an idea of a possible plotting strategy.

POLE8.PRO: a file listing the plot parameters used in the last run. When running the code, the user is prompted to choose parameters (DEFAULT=1), run default parameters (DEFAULT=0), or read them from the batch file POLE8.PRO (DEFAULT=-1). POLE8.PRO is overwritten after each run. It comes handy when making several plots using the same parameters, or to save plotting profiles to be used for different cases, or to read arbitrary values for the intensity lines (as opposed to equally spaced). See description of parameters in Section 3. This file looks like:

```
ifull iproj irepr  isym ncmax
    0     0     0     0     6
 iper1 iper2 iper3
    2     1     3
 igrid mgrid ngrid
    0    36     9
 ishft ismth  pfrot
    0     0    0.00
 icros isepa  spread
    0     0    5.00
 nlevels
   15
 rlevel(l)
    0.71     1.00     1.41     2.00     2.83     4.00     5.66     8.00
   11.31    16.00    22.63    32.00    45.25    64.00    90.51
```

PROFILE.OUT: Intensity profiles as a function of angle θ (declination from pole to equator). Calculated only when axi-symmetrizing the PF (ISYM=-1). Used, for estimating volume fraction of orientations within a range of the angle θ.

Column 1: angle θ

Column 2: intensity $I(\theta)$

Column 3: $I(\theta)*\sin(\theta)$

Column 4: $\int I(\theta)*\sin(\theta)\, d\theta$ volume fraction within a ring around the center of the PF.

Column 5: $\int I(\theta)*\sin(\theta)\, d\theta$ accumulated volume fraction from the center of the PF up to a ring.

text_1.dat
poles 0 0 0 1

ang	int	vdens	vfrac	vfacc
0.81	0.13421	0.00190	0.00051	0.00051
10.80	0.13417	0.02514	0.00373	0.00424
19.72	0.04564	0.01540	0.00247	0.00671
28.97	0.05568	0.02696	0.00439	0.01110
38.30	0.12960	0.08033	0.01315	0.02425
47.68	0.06495	0.04803	0.00788	0.03214
57.07	0.11367	0.09541	0.01569	0.04782
66.47	0.79140	0.72561	0.11940	0.16722
75.88	2.05859	1.99641	0.32864	0.49586
85.29	3.07233	3.06197	0.50414	1.00000

GRIDINT.OUT: Intensity at each phi (longitude), theta (latitude) position in the PF. May be used if you want to numerically process the PF, or plot intensity lines with a graphic software that can read this format. The reason why -185<phi<185 and 0<theta<95 is to allow the user to interpolate values within the correct interval -180<phi<180 and 0<theta<90

text_1.dat
poles 1 1 1

phi	theta	int
-185.000	0.810	0.000
-175.000	0.810	0.000
-165.000	0.810	0.000
-155.000	0.810	0.000
……..		
135.000	94.706	0.569
145.000	94.706	0.401
155.000	94.706	0.366
165.000	94.706	1.029
175.000	94.706	2.900
185.000	94.706	2.170

PF_DIFF.OUT: When plotting PF's or IPF's for two texture files, POLE8 calculates a parameter that measures the difference between the poles of each texture. Given the volume fraction associated with each grid element (fvfr(i,j,1) and fvfr(I,j,2)), the Pole Figure Difference Parameter is the sum of absolute values of the differences, normalized by the sum of the volume fractions in each pole figure.

$$PFD = \frac{\sum_{i,j} \left| fvfr(i,j,1) - fvfr(i,j,2 \right|}{\sum_{i,j} \left| fvfr(i,j,1) \right| + \sum_{i,j} \left| fvfr(i,j,2 \right|} \qquad and \qquad 0 \le P \le 1$$

An example of output for example 3 is shown below:

poles 0 0 0 1
sum1 6.28319 sum2 6.28319
PF difference parameter 0.60697

poles 1 0 -1 1
sum1 6.28319 sum2 6.28319
PF difference parameter 0.10143

poles 1 0 -1 0
sum1 6.28319 sum2 6.28319
PF difference parameter 0.34133

7- EXAMPLES

Five examples follow which include the corresponding plots. The input texture, the input file POLE8.IN, and the output file POLE8.PRO are included in the corresponding folder. The ASCII output files used for making the plots are also included.

7-1 Example 1:

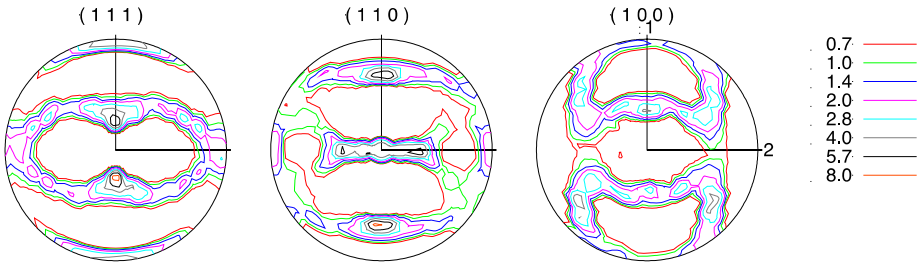
A simulated texture of rolled Cu (fcc). (111), (110), (100) poles plotted as points and intensity lines in the pole figure. In this example several poles are plot for a single texture.

POLE8.IN → same is used for intensity lines and dots

```
* # of texture files 'ntextfiles' to read below
  1
* path & name of texture files (one per line)
example1/fcc_roll.tex
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
  0
* single crystal unit cell (N/A if ipfig=-2)
datasx/fcc.sx
* number of X-tall poles (PF) or sample axes (IPF) to read (N/A if ipfig=-2)
  3
*Miller indices of poles (N/A if ipfig=-2)
   1   1   1     ! 3   (100) poles
   1   1   0     ! 6   (110) poles
   1   0   0     ! 4   (111) poles
```

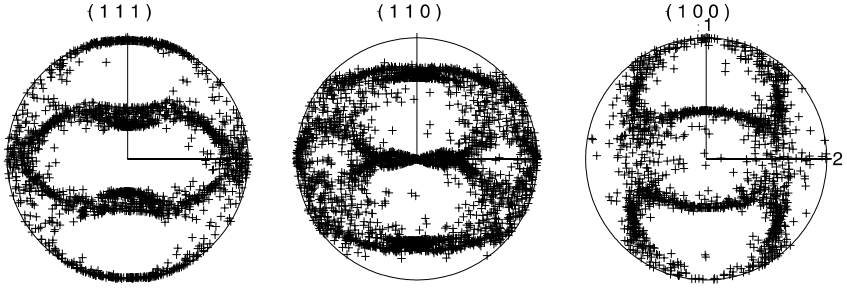
POLE8.PRO → for lines

```
ifull iproj irepr  isym ncmax
    0     0     0     0     6
 iper1 iper2 iper3
    2     1     3
 igrid mgrid ngrid
    0    36     9
 ishft ismth  pfrot
    0     0    0.00
 icros isepa  spread
    0     0    5.00
 nlevels
   15
 rlevel(l)
    0.71     1.00     1.41     2.00     2.83     4.00     5.66     8.00
   11.31    16.00    22.63    32.00    45.25    64.00    90.51
```



POLE8.PRO → for dots

```
ifull iproj irepr  isym ncmax
    0     0     1     0     6
 iper1 iper2 iper3
    2     1     3
 igrid mgrid ngrid
    0    36     9
 ishft ismth  pfrot
    0     0    0.00
 icros isepa  spread
    0     0    5.00
 nlevels
    0
 rlevel(l)
```



7-2 Example 2:

Simulated textures of Cu (fcc) deformed by axial compression and tension.
Axis (001) is plot in Inverse Pole Figures, using lines and points.
In this example two textures and one pole are plot. In the case of intensity lines, the option default=-1 was used in order to plot less lines and make the IPF look less cluttered. In the case when crosses were plot, default=1 had to be used in order to choose 'dots' over 'intensity lines'.

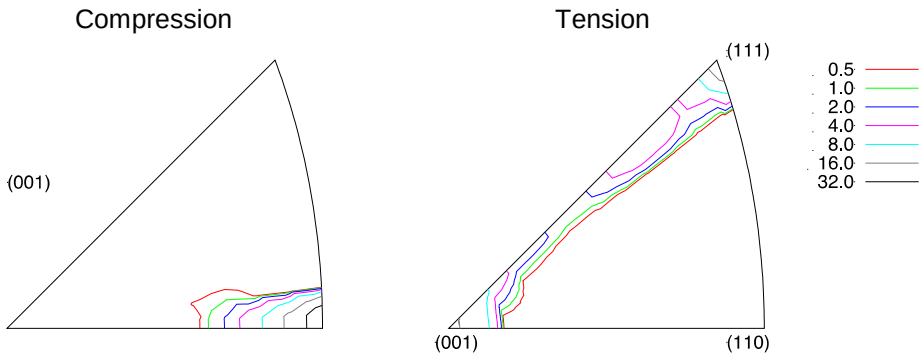
POLE8.IN → same for lines and dots

```
* # of texture files 'ntextfiles' to read below
  2
* path & name of texture files (one per line)
example2/fcc_comp.tex
example2/fcc_tens.tex
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
  1
* single crystal unit cell (N/A if ipfig=-2)
datasx/fcc.sx
* number of X-tall poles (PF) or sample axes (IPF) to read (N/A if ipfig=-2)
  1
*Miller indices of poles (N/A if ipfig=-2)
  0   0   1      !  x3 axis for IPF
```

POLE8.PRO → for lines

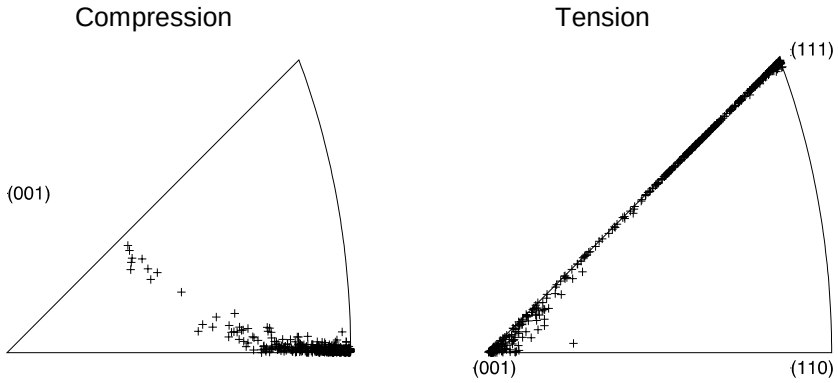
```
ifull iproj irepr  isym ncmax
    0     0     0     0     6
 iper1 iper2 iper3
```

```
   1     2     3
igrid mgrid ngrid
   0    72    18
ishft ismth  pfrot
   0     0    0.00
icros isepa  spread
   0     0    2.50
nlevels
  15
rlevel(l)
   0.50     1.00     2.00     4.00     8.00    16.00    32.00    64.00
 128.00   256.00   512.00  1024.00  2048.00  4096.00  8192.00
```



POLE8.PRO → for dots

```
ifull iproj irepr  isym ncmax
   0     0     1     0     6
iper1 iper2 iper3
   1     2     3
igrid mgrid ngrid
   0    72    18
ishft ismth  pfrot
   0     0    0.00
icros isepa  spread
   0     0    2.50
nlevels
   0
rlevel(l)
```



7-3 Example 3:

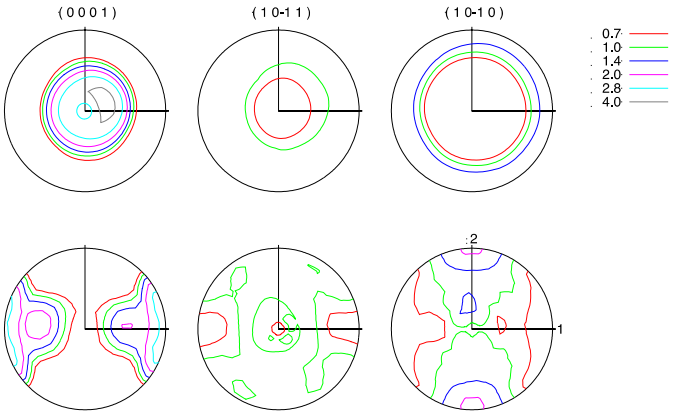
Experimental textures for clock-rolled and annealed Zr (hcp) and the same Zr after 27% In-Plane Compression. Basal (0002), pyramidal (10-11) and prism (10-10) poles are plot for each texture in one single run of POLE8.

POLE8.IN

```
* # of texture files 'ntextfiles' to read below
  2
* path & name of texture files (one per line)
example3/5678.tex
example3/7185.tex
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
  0
* single crystal unit cell (N/A if ipfig=-2)
datasx/zr.sx
* number of X-tall poles (PF) or sample axes (IPF) to read (N/A if ipfig=-2)
  3
*Miller indices of poles (N/A if ipfig=-2)
  0   0   0   1   ! 1 basal pole
  1   0  -1   1   ! 6 pyramidal 1st order poles
  1   0  -1   0   ! 3 prism 1st order poles
```

POLE8.PRO

```
 ifull iproj irepr  isym ncmax
    0     0     0     0     6
 iper1 iper2 iper3
    1     2     3
 igrid mgrid ngrid
    0    36     9
 ishft ismth  pfrot
    0     0    0.00
 icros isepa  spread
    0     0    5.00
 nlevels
   15
 rlevel(l)
    0.71     1.00     1.41     2.00     2.83     4.00     5.66     8.00
   11.31    16.00    22.63    32.00    45.25    64.00    90.51
```

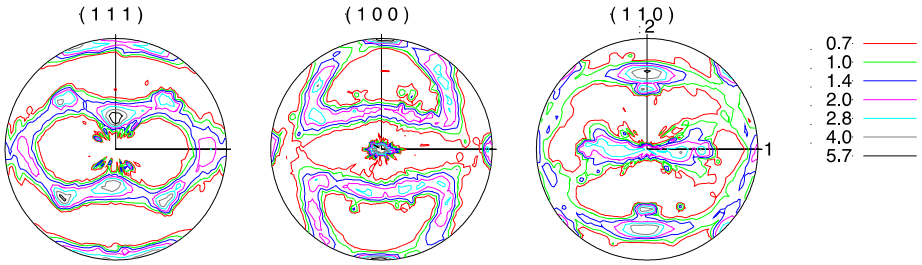


First row: initial annealed Zr

Second row: Zr after 27% in-plane-compression

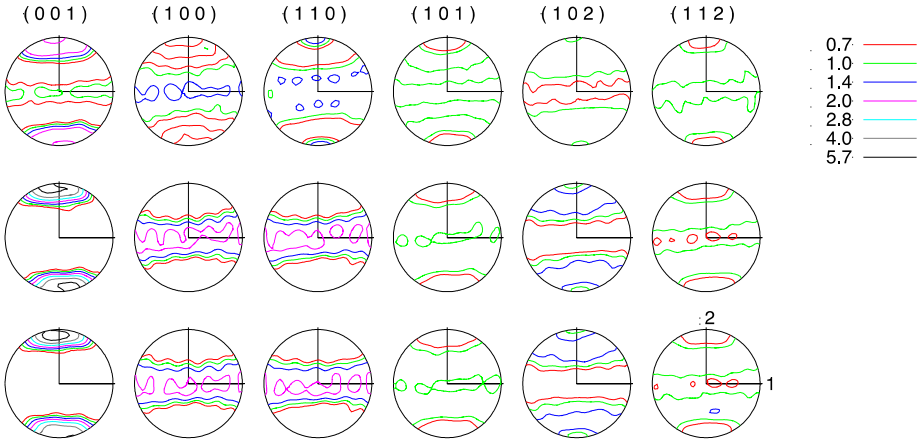
7-4a Example 4(a):
Experimental pole figures of rolled Cu (fcc).

POLE8.IN
* # of texture files 'ntextfiles' to read below
1
* path & name of texture files (one per line)
example4a/Cu_roll.wpf
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
-2



7-4b Example 4(b):
Experimental pole figures of clock-rolled Zr (hcp) deformed 27% by In-Plane Compression at 76K, 300K and 450K. This case illustrates plotting multiple poles for multiple experimental pole figures.

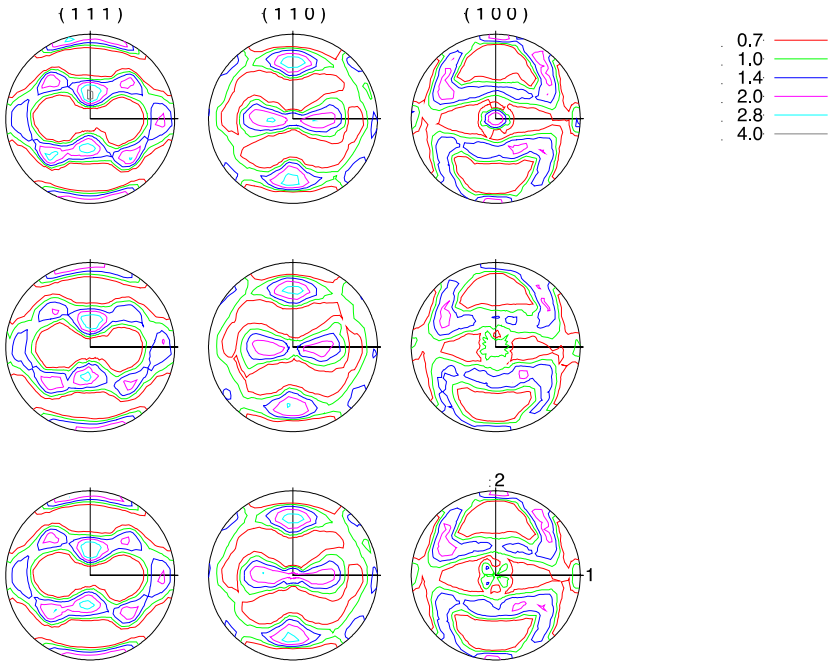
POLE8.IN
* # of texture files 'ntextfiles' to read below
3
* path & name of texture files (one per line)
example4b/ZrTTC76K.gpf
example4b/ZrTTC300.gpf
example4b/ZrTTC450.gpf
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
-2



7-5a Example 5(a):
Plots of the same poles shown in Example 4, but now we do so from first reading the calculated Distribution Orientation Function (*.SOD), producing three discrete texture files with different number of grains, and plotting the poles from such files.
Case of rolled Cu (fcc). Notice that the correct position of sample axes in the PF is (1 2 3) in order to obtain the same 'look' for the pole figures.

POLE8.IN

```
* # of texture files 'ntextfiles' to read below
  1
* path & name of texture files (one per line)
example5a/Cu_roll.sod
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
  -1
* single crystal unit cell (N/A if ipfig=-2)
datasx/fcc.sx
* number of X-tall poles (PF) or sample axes (IPF) to read (N/A if ipfig=-2)
  3
*Miller indices of poles (N/A if ipfig=-2)
   1   1   1     ! 3   (100) poles
   1   1   0     ! 6   (110) poles
   1   0   0     ! 4   (111) poles
```

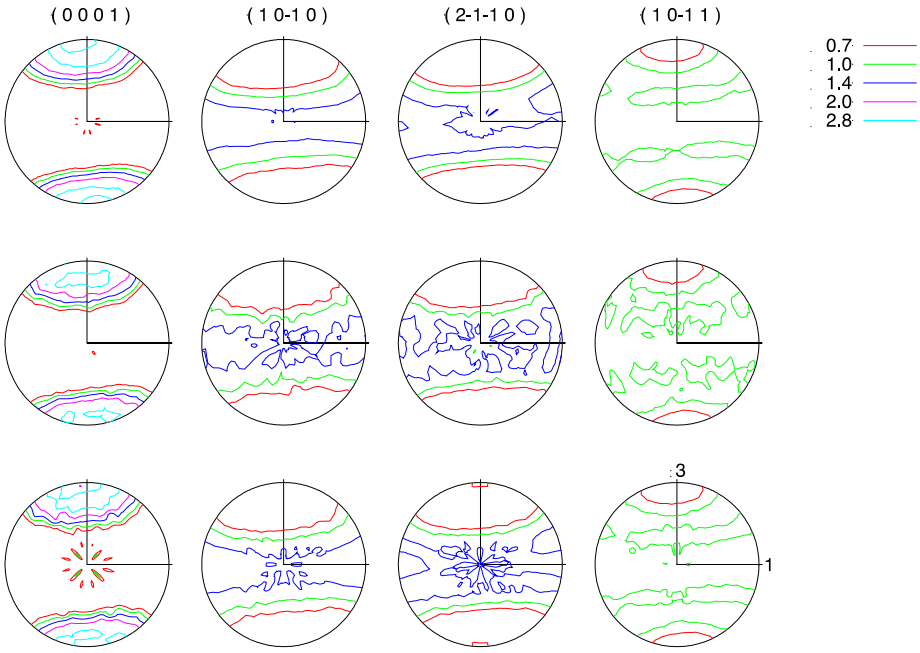


First row: *.FUL (large size regular orientations texture)
Second row: *.RAN (medium size random orientations texture)
Third row: *.RED (small size regular orientations texture)

7-5b Example 5(b):
Same as Example 5(a), except this one is for clock-rolled Zr (hcp) deformed 27% by In-Plane Compression. And the position of sample axes in the PF is (1 3 2) in order to obtain the same 'look' for the pole figures.

POLE8.IN

```
* # of texture files 'ntextfiles' to read below
  1
* path & name of texture files (one per line)
example5b/ZrTTC300.sod
* 'ipfig': 0 for PF plot, 1 for IPF plot, -1 for *.SOD file, -2 for *.EPF file
  -1
* single crystal unit cell (N/A if ipfig=-2)
datasx/Zr.sx
* number of X-tall poles (PF) or sample axes (IPF) to read (N/A if ipfig=-2)
  4
*Miller indices of poles (N/A if ipfig=-2)
   0   0   0   1   ! 1 basal pole
   1   0  -1   0   ! 3 prism 1st order poles
   2  -1  -1   0   ! 3 prism 2nd order poles
   1   0  -1   1   ! 6 pyramidal 1st order poles
```



First row: *.FUL (large size regular orientations texture)
Second row: *.RAN (medium size random orientations texture)
Third row: *.RED (small size regular orientations texture)