# MagIC_workshop_example

May 13, 2014

## 1 Example notebook

This simple example notebook accompanies a short talk given at the 2014 MagIC science and database workshop. The PDF of the slides is availible here: https://github.com/Swanson-Hysell/PmagPy_IPython/raw/master/2014_MagIC_workshop/Swanson-Hysell_MagIC2014.pdf

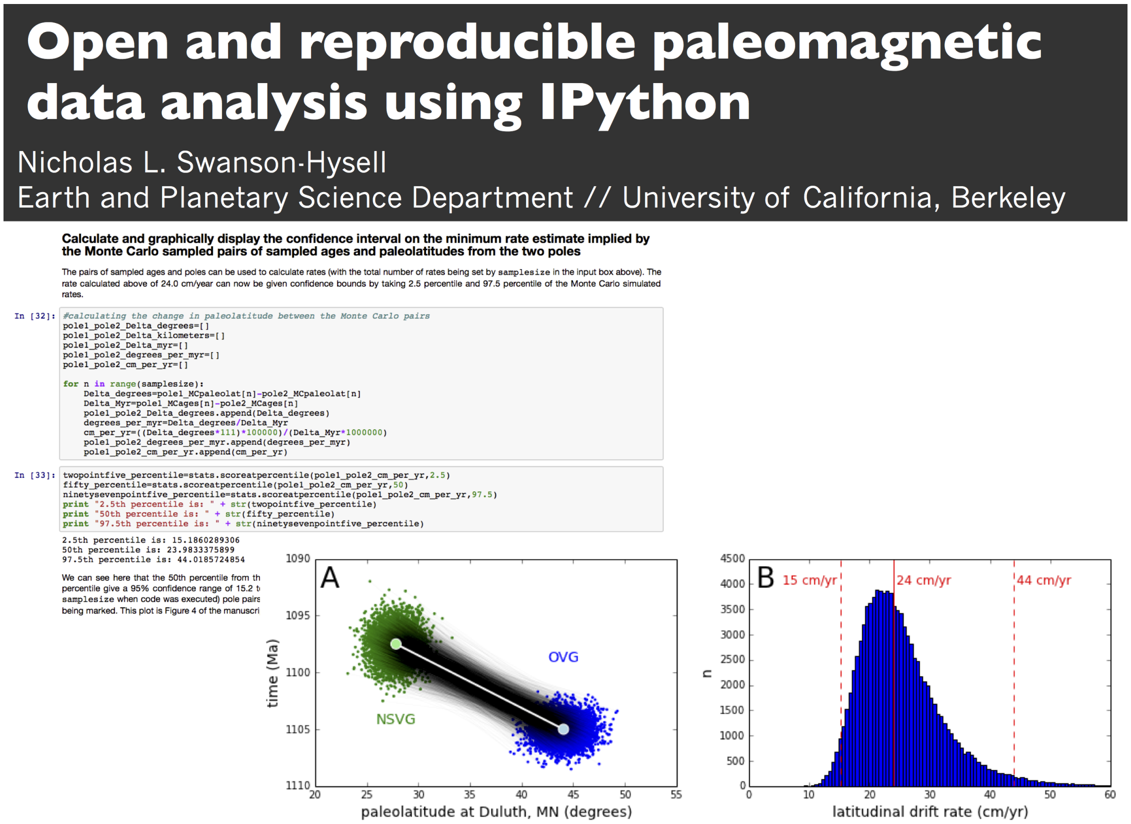
### 1.1 Import function libraries

IPmag.py is a function library for paleomagnetic data analysis and plotting developed for interactive use within IPython notebooks. The IPmag.py library relies on the pmag.py and pmagplotlib.py libraries of the PmagPy project (https://github.com/ltauxe/PmagPy) and adds new functions and modifies stand alone PmagPy programs for use in the interactive IPython environment. The pmag.py library is included in the IPmag.py repository as a very slightly modified version from that availible in the main PmagPy repository. The version of pmag.py being used here from pmagpy-2.206.

```
In [1]: import pmag, pmagplotlib, IPmag
        import numpy as np
        import pandas as pd
        import matplotlib.pyplot as plt
        from IPython.core.display import Image
        %matplotlib inline
```

```
In [2]: Image(filename='2014_MagIC_workshop/MagIC2014.png')
```

Out[2]:




## 1.2 Example developing Fisher distributed data, plotting the data and comparing two data sets using a test for a common mean.

Let's start by generating a set of directions from a Fisher distribution using the `IPmag.ifishrot` function.

```
In [3]: data1 = IPmag.ifishrot(k=42,n=40,Dec=204,Inc=-42)

In [4]: data1

Out[4]: [[218.11569556120406, -50.274554383096415, 1.0],
         [208.42469985977803, -44.983564159695767, 1.0],
         [230.88947292192859, -46.624745623253688, 1.0],
         [202.18890087664008, -22.717319275764698, 1.0],
         [216.82955908650555, -28.907758493266957, 1.0],
         [196.85975068187085, -46.435678919773139, 1.0],
         [213.48898715459782, -43.464854834824145, 1.0],
         [220.88017633760342, -48.523977578214293, 1.0],
         [204.66975050330208, -44.219106889261496, 1.0],
         [212.07833966805737, -45.226257597348543, 1.0],
         [218.52170309556112, -45.655245287636959, 1.0],
         [171.10001720046739, -32.625265305237932, 1.0],
         [209.62896427763317, -27.200382379310234, 1.0],
         [202.02376663609701, -31.373611015918822, 1.0],
         [219.858098969628, -26.181647562457623, 1.0],
```

```
 [201.51084707027249, -46.029599137054497, 1.0],
 [196.93792773272156, -32.375759325022891, 1.0],
 [219.15260167019571, -46.979181620208472, 1.0],
 [199.03833390387695, -36.189397551310819, 1.0],
 [211.22798485822273, -33.793088122986404, 1.0],
 [213.30212668033019, -29.480685522187063, 1.0],
 [193.4351277787006, -37.781229447382039, 1.0],
 [205.47753870083065, -53.551303539771652, 1.0],
 [189.82561803175119, -45.872681086591527, 1.0],
 [197.9108349577906, -24.471405779328361, 1.0],
 [214.68041398680762, -37.105229923997953, 1.0],
 [208.66798870032272, -22.849704879992224, 1.0],
 [209.74777009800195, -39.414303337458669, 1.0],
 [188.02618595628297, -43.09564330913679, 1.0],
 [203.12121565256541, -29.344790964074413, 1.0],
 [213.58458948006586, -45.960383273200165, 1.0],
 [211.60375134783246, -51.953865045454421, 1.0],
 [200.38259909486771, -32.842769713928099, 1.0],
 [199.74223825029202, -32.53772194490282, 1.0],
 [187.56945652997442, -45.430366188485181, 1.0],
 [202.25418233704036, -49.545747549334799, 1.0],
 [198.70069062957035, -37.795954796139377, 1.0],
 [208.96878897950532, -26.498413076478503, 1.0],
 [197.6424438647874, -45.158617064842765, 1.0],
 [185.35850695695143, -47.496901204778986, 1.0]]
```

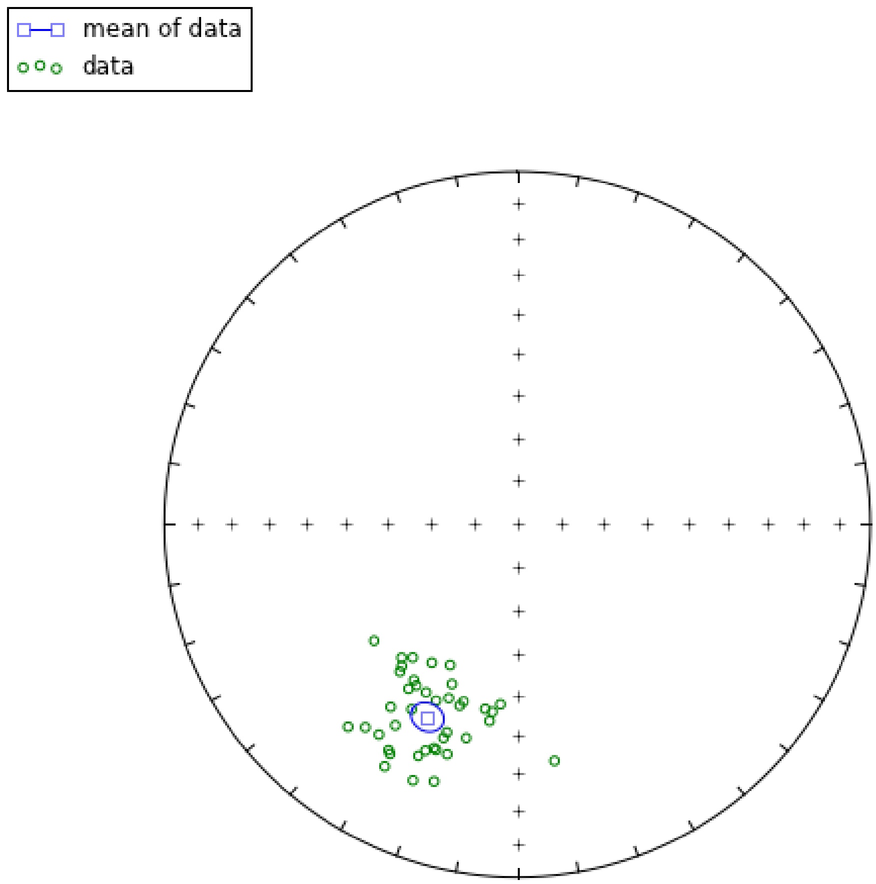
Let's use the `pmag.fisher_mean` function to calculated the Fisher statistical parameters for the data.

In [5]:
```
data1_mean = pmag.fisher_mean(data1)
data1_mean
```

Out[5]:
```
{'alpha95': 3.5406501384077265,
 'csd': 12.534764892280513,
 'dec': 204.99558465507852,
 'inc': -39.509178691305621,
 'k': 41.757804112425852,
 'n': 40,
 'r': 39.066042843273102}
```

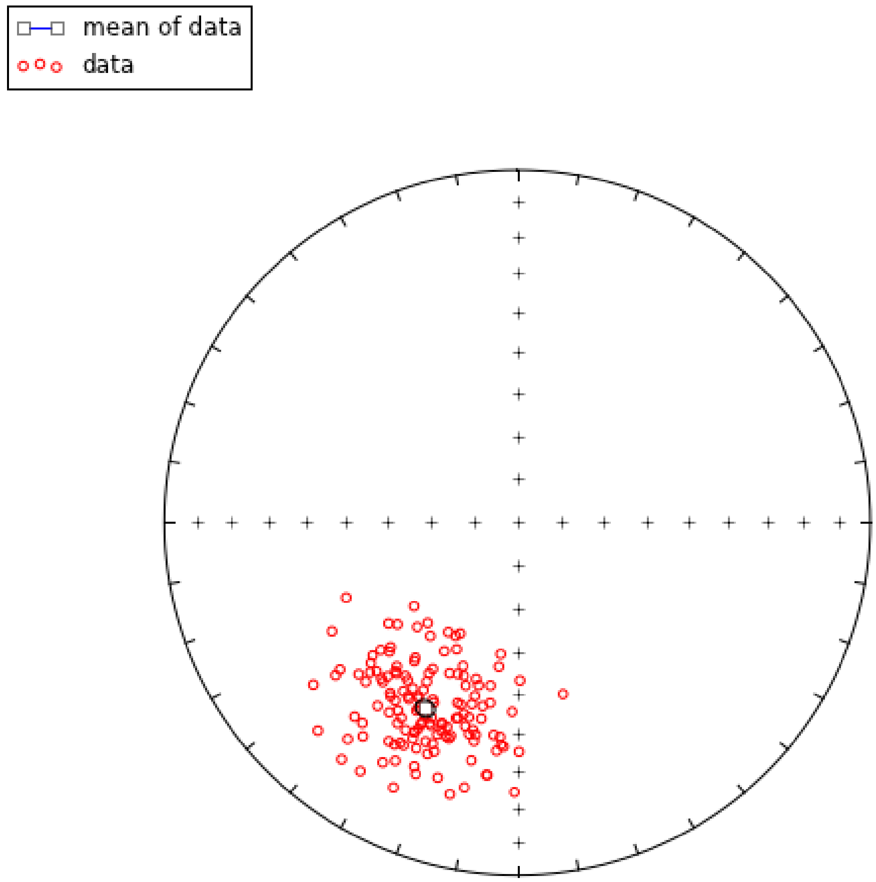
In [6]:
```
pmag.fisher_mean??
```

In [7]:
```
fignum = 1
plt.figure(num=fignum,figsize=(8,8),dpi=160)
IPmag.iplotNET(fignum)
IPmag.iplotDI(data1,color='g',label='data',legend='yes')
IPmag.iplotDImean(data1_mean['dec'],data1_mean['inc'],data1_mean['alpha95'],
                  color='b',marker='s',label='mean of data',legend='yes')
```

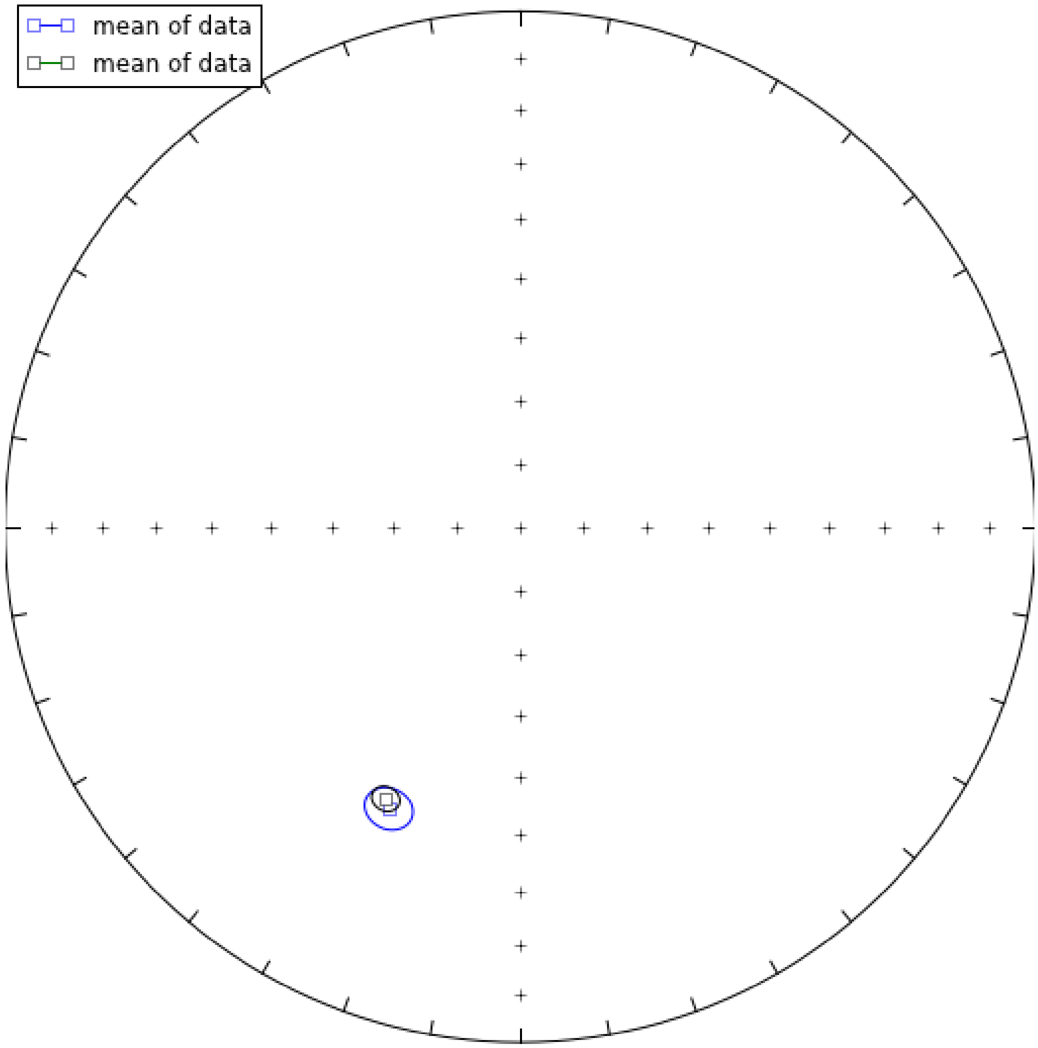



```
In [8]: data2 = IPmag.ifishrot(k=36,n=140,Dec=206,Inc=-41)

In [9]: data2_mean = pmag.fisher_mean(data2)
        data2_mean

Out[9]: {'alpha95': 2.1017754545938985,
         'csd': 14.069339483629621,
         'dec': 206.29443911646595,
         'inc': -40.835567769695842,
         'k': 33.145350796410476,
         'n': 140,
         'r': 135.80635001108351}

In [10]: fignum = 1
         plt.figure(num=fignum,figsize=(8,8),dpi=160)
         IPmag.iplotNET(fignum)
```

```
IPmag.iplotDI(data2,color='r',label='data',legend='yes')
IPmag.iplotDImean(data2_mean['dec'],data2_mean['inc'],data2_mean['alpha95'],
                  color='k',marker='s',label='mean of data',legend='yes')
```

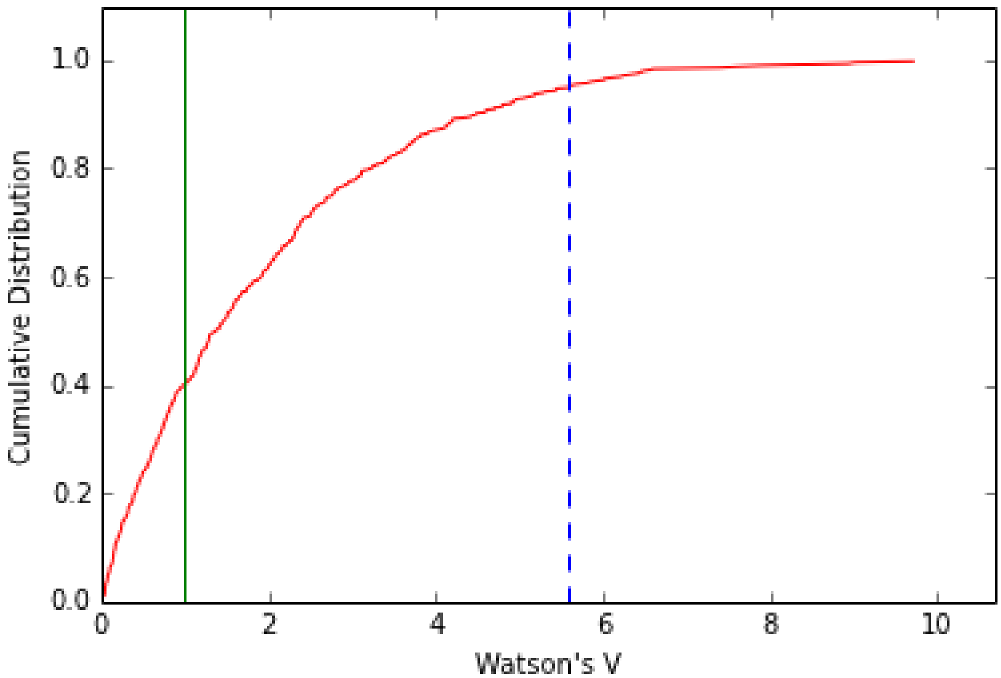



```
In [11]: fignum = 1
         plt.figure(num=fignum,figsize=(8,8),dpi=160)
         IPmag.iplotNET(fignum)
         IPmag.iplotDImean(data1_mean['dec'],data1_mean['inc'],data1_mean['alpha95'],
                           color='b',marker='s',label='mean of data',legend='yes')
         IPmag.iplotDImean(data2_mean['dec'],data2_mean['inc'],data2_mean['alpha95'],
                           color='k',marker='s',label='mean of data',legend='yes')
```




```
In [12]: IPmag.iWatsonV(data1,data2,NumSims=500,plot='yes')
```

```
Results of Watson V test:

Watson's V:           1.0
Critical value of V:  5.6
"Pass": Since V is less than Vcrit, the null hypothesis
that the two populations are drawn from distributions
that share a common mean direction can not be rejected.

M&M1990 classification:

Angle between data set means: 1.7
Critical angle for M&M1990:   3.9
The McFadden and McElhinny (1990) classification for
this test is: 'A'
```




```
In [13]: IPmag.iWatsonV?
```

## 1.3 Reading data from MagIC format results files (pmag.magic_read)

PmagPy has built in functions to deal with data in MagIC database format files. One such function is the `pmag.magic_read` function. In the code block below, this function is used to read in data from a `pmag_results.txt` file. The `pmag.magic_read` function creates two outputs:

1. the data from the Magic template file in a list of dictionaries
2. the type of data in the file (e.g. 'pmag_results')

Let's look at data from:

Tauxe, L., Luskin, C., Selkin, P., Gans, P. and Calvert, A. (2004). Paleomagnetic results from the Snake River Plain: Contribution to the time[U+2010]averaged field global database. Geochemistry Geophysics Geosystems 5: doi: 10.1029/2003GC000661. http://onlinelibrary.wiley.com/doi/10.1029/2003GC000661/

Downloaded from the MagIC database:

http://earthref.org/MAGIC/9207/

```
In [14]: data,file_type=pmag.magic_read('Example_Data/pmag_results.txt')
         results = pd.DataFrame(data)
         results.average_dec = results.average_dec.astype(float)
         results.average_inc = results.average_inc.astype(float)
         results.average_alpha95 = results.average_alpha95.astype(float)
         results
```

```
Out[14]:   antipodal average_age average_age_sigma average_age_unit  average_alpha95  \
         0                   3.4              0.03               Ma              2.3
         1                     1                 1               Ma              3.0
         2                     2              0.05               Ma              4.3
```

| | | | | | |
|---|---|---|---|---|---|
| 3 | | 0.095 | 0.05 | Ma | 3.0 |
| 4 | | 0.395 | 0.01 | Ma | 3.0 |
| 5 | | 0.373 | 0.06 | Ma | 5.1 |
| 6 | | 0.404 | 0.07 | Ma | 4.0 |
| 7 | | 1.76 | 0.016 | Ma | 2.6 |
| 8 | | 2.26 | 0.05 | Ma | 3.3 |
| 9 | | 1.69 | 0.02 | Ma | 2.5 |
| 10 | | 0.591 | 0.015 | Ma | 4.2 |
| 11 | | 0.291 | 0.016 | Ma | 2.6 |
| 12 | | 1.73 | 0.01 | Ma | 4.9 |
| 13 | | 2.94 | 0.025 | Ma | 3.2 |
| 14 | | 1 | 1 | Ma | 1.6 |
| 15 | | 5.75 | 0.125 | Ma | 2.7 |
| 16 | | 0.184 | 0.014 | Ma | 1.3 |
| 17 | | 0.191 | 0.008 | Ma | 2.4 |
| 18 | | 0.124 | 0.039 | Ma | 5.7 |
| 19 | | 1 | 1 | Ma | 5.2 |
| 20 | | 1 | 1 | Ma | 4.2 |
| 21 | | 1.5055 | 1.6253 | Ma | 7.7 |
| 22 | | 1.0757 | 0.91686 | Ma | 8.7 |
| 23 | 177.2 | 1.3418 | 1.3874 | Ma | 5.4 |

| | average_dec | average_inc | average_int | average_int_n | average_int_sigma | \ |
|---|---|---|---|---|---|---|
| 0 | 330.1 | 64.9 | 3.61e-05 | 5 | 1.32e-05 | |
| 1 | 151.8 | -57.5 | | | | |
| 2 | 16.5 | 54.6 | | | | |
| 3 | 14.6 | 77.9 | | | | |
| 4 | 346.3 | 73.8 | | | | |
| 5 | 15.3 | 44.8 | | | | |
| 6 | 347.4 | 65.6 | | | | |
| 7 | 172.5 | -66.9 | | | | |
| 8 | 6.3 | 29.7 | | | | |
| 9 | 14.5 | 45.5 | | | | |
| 10 | 163.3 | -62.2 | | | | |
| 11 | 2.9 | 61.9 | | | | |
| 12 | 358.6 | 62.5 | | | | |
| 13 | 197.4 | -51.2 | | | | |
| 14 | 9.5 | 62.8 | | | | |
| 15 | 7.5 | 66.7 | 4.88e-05 | 2 | 1.7e-06 | |
| 16 | 23.2 | 54.0 | | | | |
| 17 | 205.8 | -49.4 | 7.85e-05 | 3 | 4.7e-06 | |
| 18 | 197.6 | -65.5 | | | | |
| 19 | 188.1 | -47.2 | 5.02e-05 | 2 | 2.7e-06 | |
| 20 | 192.9 | -60.7 | | | | |
| 21 | 6.1 | 59.6 | | | | |
| 22 | 185.1 | -58.8 | | | | |
| 23 | 5.7 | 59.3 | | | | |

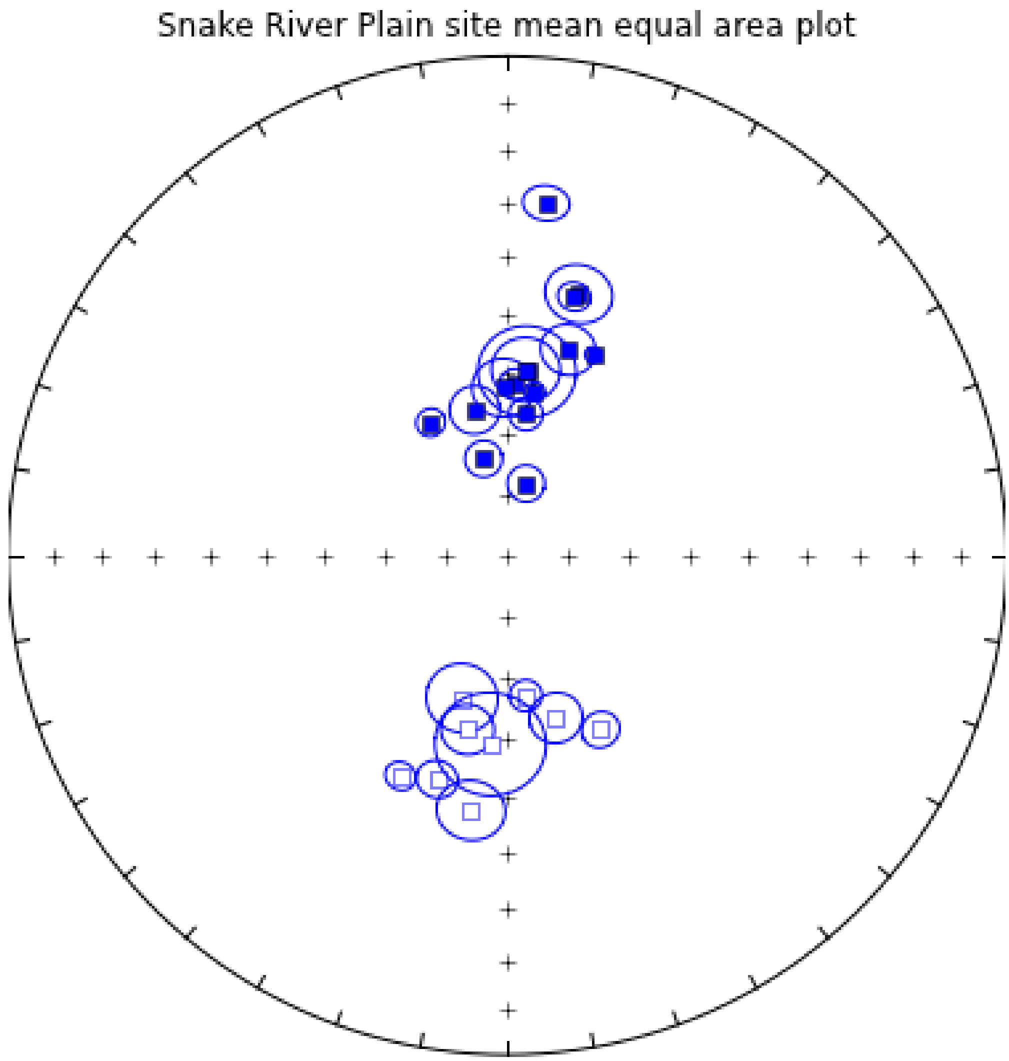
| | average_k | average_lat | average_lat_sigma | average_lon | average_lon_sigma | \ |
|---|---|---|---|---|---|---|
| 0 | 572.7 | 42.6026 | | 245.602 | | |
| 1 | 408.8 | 42.6035 | | 245.599 | | |
| 2 | 313.1 | 42.601 | | 245.596 | | |
| 3 | 407.3 | 42.7366 | | 245.165 | | |
| 4 | 428.4 | 42.8418 | | 245.112 | | |

```
5      227.8     42.8657                        245.127
6      228.8     42.9203                        244.923
7      892.1     42.4996                        245.851
8      333.8        42.5                         245.85
9      970.1     42.5287                        245.979
10     325.7     42.4556                        245.636
11     872.4     42.4892                          245.6
12     153.9     42.4619                        245.615
13     296.1     42.6529                        246.988
14    1253.3      43.305                        247.732
15     488.7     43.3682                        247.345
16    1820.7     43.4213                        247.256
17     642.1     43.8859                        247.199
18     138.9     43.8427                        247.376
19     168.2     43.5749                        248.266
20       335     44.1566                        248.574
21      30.2     42.8154             0.346    245.9077             0.9269
22      41.9      43.225            0.7166      246.95             1.1686
23      35.4     42.9714            0.5414    246.3048             1.1236

   average_n average_nn average_r conglomerate_test contact_test
0          8          8    7.9878                ND           ND ...
1          7          7    6.9853                ND           ND ...
2          5          5    4.9872                ND           ND ...
3          7          7    6.9865                ND           ND ...
4          7          7    6.9883                ND           ND ...
5          5          5    4.9824                ND           ND ...
6          7          7    6.9738                ND           ND ...
7          5          5    4.9955                ND           ND ...
8          7          7    6.9835                ND           ND ...
9          5          5    4.9964                ND           ND ...
10         5          5    4.9877                ND           ND ...
11         5          5    4.9954                ND           ND ...
12         7          7    6.9643                ND           ND ...
13         8          8    7.9764                ND           ND ...
14         8          8    7.9944                ND           ND ...
15         7          7    6.9877                ND           ND ...
16         8          8    7.9962                ND           ND ...
17         7          7    6.9914                ND           ND ...
18         6          6     5.964                ND           ND ...
19         6          6    5.9703                ND           ND ...
20         5          5    4.9881                ND           ND ...
21        13               12.602                ND           ND ...
22         8                7.833                ND           ND ...
23        21              20.4344                ND           ND ...

[24 rows x 42 columns]
```

Now that we have imported the data, we can do other things with it such as plot it using `IPmag.iplotDImean`.

In [15]:
```
fignum = 1
plt.figure(num=fignum,figsize=(6,6),dpi=160)
IPmag.iplotNET(fignum)
plt.title('Snake River Plain site mean equal area plot')
```

```
for n in range(0,len(results)):
    IPmag.iplotDImean(results['average_dec'][n],results['average_inc'][n],results['average_alpl
                  color='b',marker='s',label='mean of data',legend='no')
```




This notebook is availible in this Github repository: https://github.com/Swanson-Hysell/PmagPy_IPython

The notebook can be viewed as raw html here: http://nbviewer.ipython.org/github/Swanson-Hysell/PmagPy_IPython/blob/master/IPmag_examples.ipynb

This command can be used to to convert this notebook into a PDF document:

`nbconvert --to latex MagIC_workshop_example.ipynb --post PDF`

In []: