An open source software suite for multi-dimensional geoscience data computation and visualization combined with GIS technology

Yaqiang Wang

Chinese Academy of Meteorological Sciences

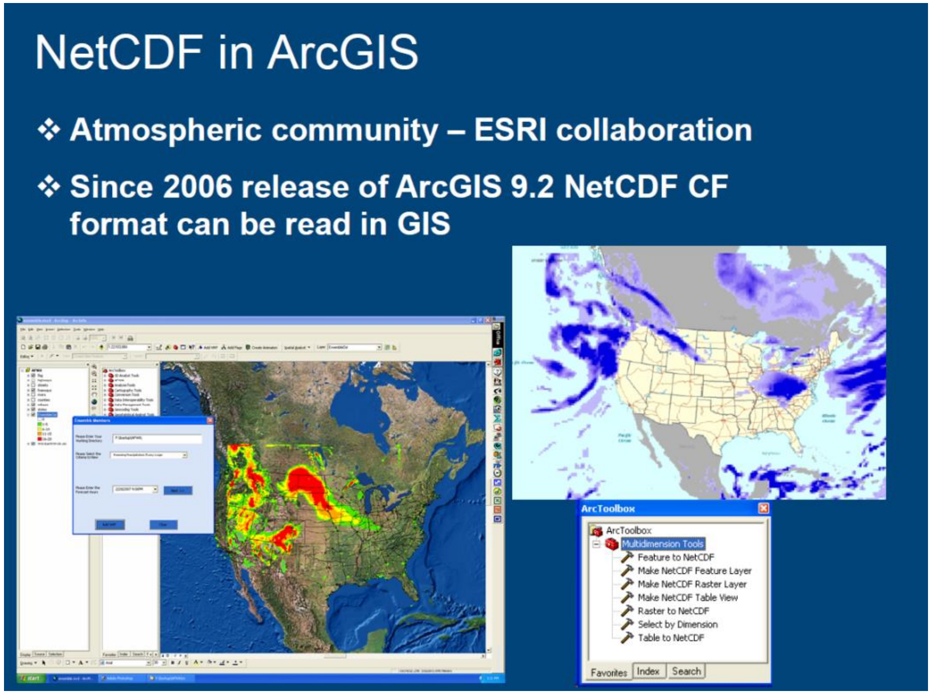
Geoscience data analysis and visualization

- Multi-dimensional array computation
- Geoscience data analysis functions
- Normally used data formats (netCDF, GRIB…)
- Spatial data support, GIS functions
- Data visualization
- Script ability
- Data analysis and visualization library

Commercial software

ArcGIS, MatLab, IDL…

- ➢ Very expensive
- ➢ Not support popular meteorological data formats conveniently

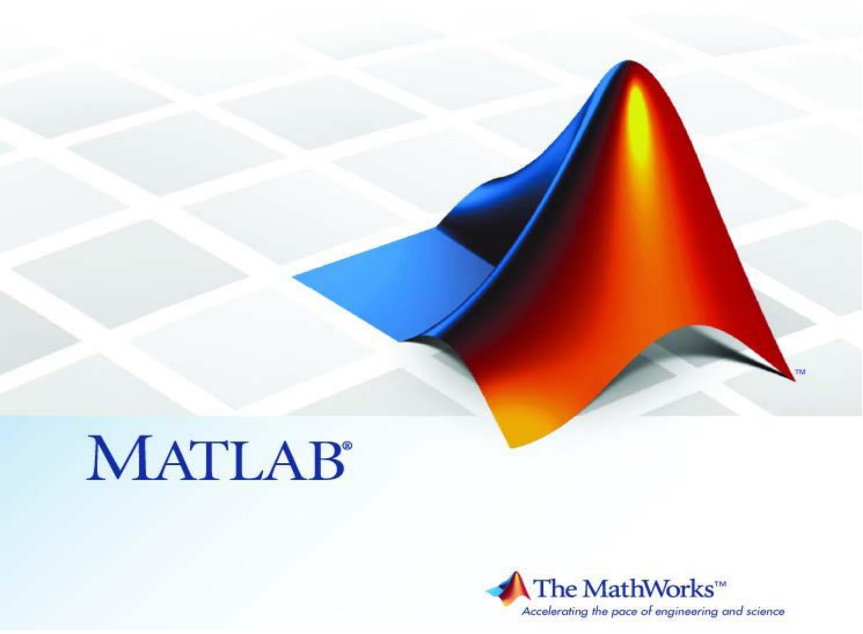


NetCDF in ArcGIS

- ❖ Atmospheric community – ESRI collaboration
- ❖ Since 2006 release of ArcGIS 9.2 NetCDF CF format can be read in GIS

ArcToolbox

- ArcToolbox
  - Multidimension Tools
    - Feature to NetCDF
    - Make NetCDF Feature Layer
    - Make NetCDF Raster Layer
    - Make NetCDF Table View
    - Raster to NetCDF
    - Select by Dimension
    - Table to NetCDF



Freely available software – Atmospheric community

GrADS
NCL
Python: Numpy + Matplotlib + Basemap…
…

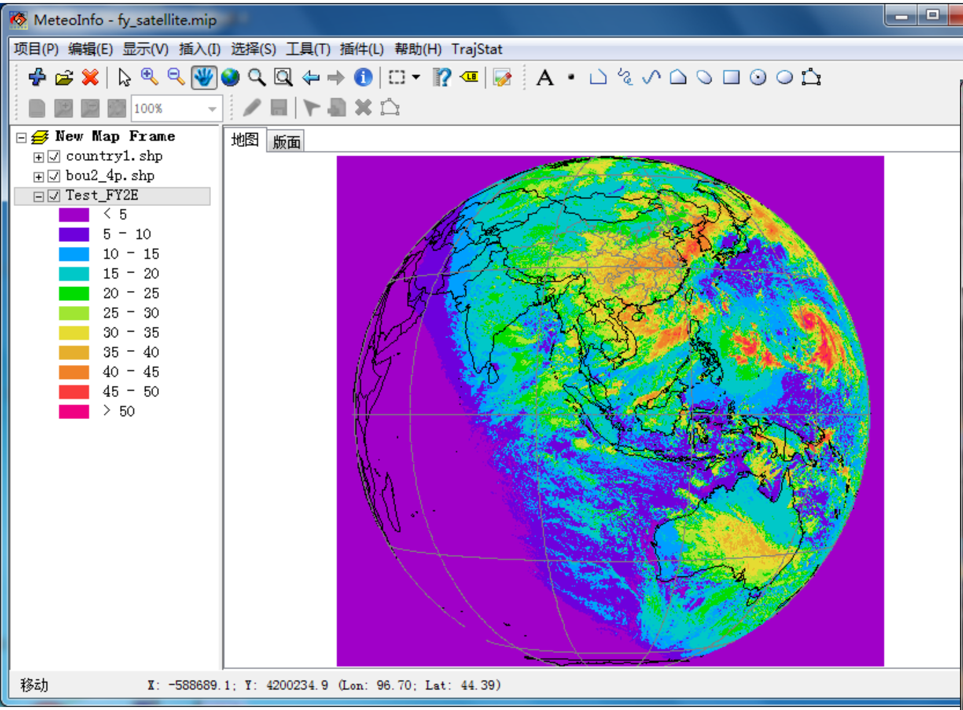
We may need open source “ArcGIS” + “MatLab”

MeteoInfo software package

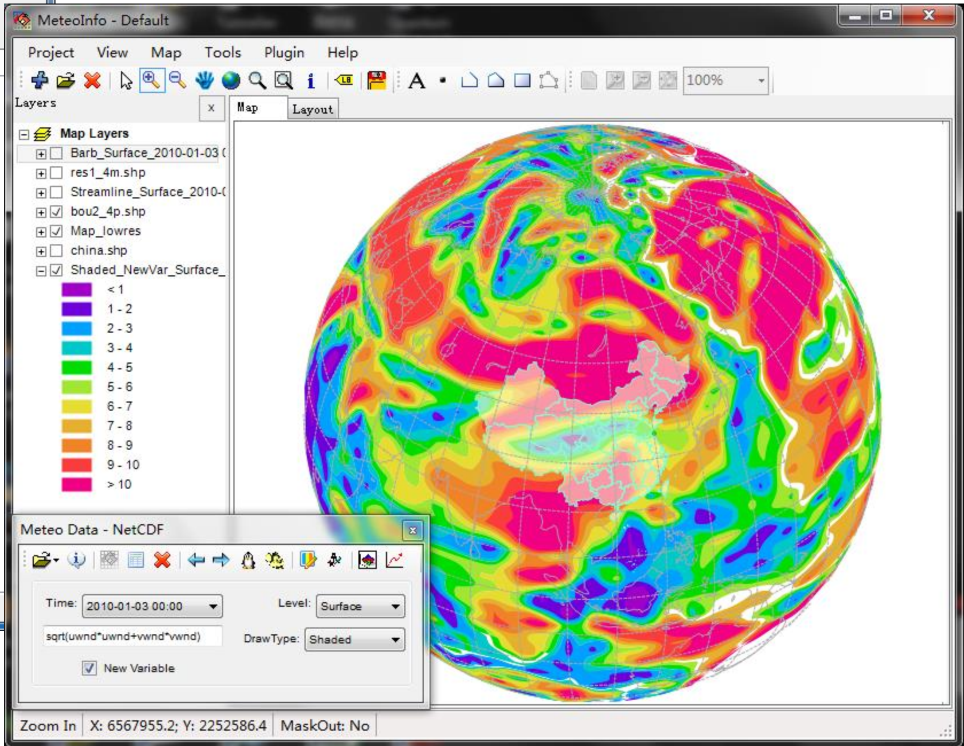
- Motivation: Scientific data analysis and visualization with GIS ability – especially for meteorological community
- MeteoInfo library: Java and C# version
- MeteoInfo GIS desktop: Java and C# version
- MeteoInfoLab: Java and Jython

MeteoInfo： GIS Desktop application

- GIS functions were developed from ground level.
- Two editions: Java and C#.
- Freely and open source from the website http://www.meteothinker.com
- Cross-platform: Windows, Unix, Linux, Mac OS.
- Support netCDF, GRIB 1&2, HDF, GrADS, MICAPS, AWX …
- Script ability.



移动 X: -588689.1; Y: 4200234.9 (Lon: 96.70; Lat: 44.39)



Time: 2010-01-03 00:00 Level: Surface

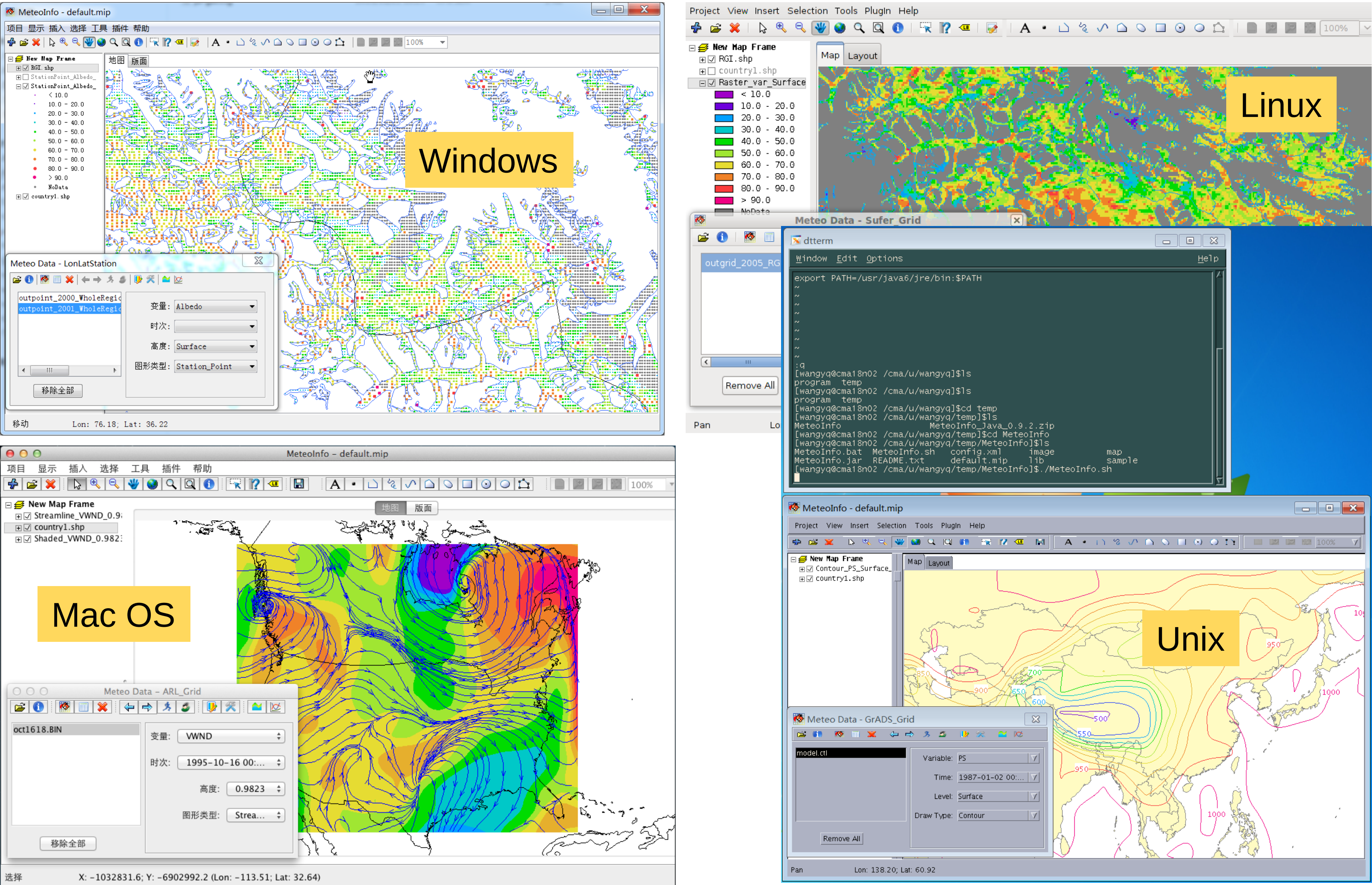
sqrt(uwnd*uwnd+vwnd*vwnd) DrawType: Shaded

☑ New Variable

Zoom In X: 6567955.2; Y: 2252586.4 MaskOut: No

（Wang 2014 Meteorological Applications）

MeteoInfo – Spatial and meteorological data operation in a GIS environment



变量: Albedo
时次:
高度: Surface
图形类型: Station_Point

Lon: 76.18, Lat: 36.22

```
export PATH=/usr/java6/jre/bin:$PATH
~
~
~
~
~
~
~
~
~
:q
[wangyq@cma18n02 /cma/u/wangyq]$ls
program  temp
[wangyq@cma18n02 /cma/u/wangyq]$ls
program  temp
[wangyq@cma18n02 /cma/u/wangyq]$cd temp
[wangyq@cma18n02 /cma/u/wangyq/temp]$ls
MeteoInfo                 MeteoInfo_Java_0.9.2.zip
[wangyq@cma18n02 /cma/u/wangyq/temp]$cd MeteoInfo
[wangyq@cma18n02 /cma/u/wangyq/temp/MeteoInfo]$ls
MeteoInfo.bat  MeteoInfo.sh   config.xml     image           map
MeteoInfo.jar  README.txt     default.mip    lib             sample
[wangyq@cma18n02 /cma/u/wangyq/temp/MeteoInfo]$./MeteoInfo.sh
```

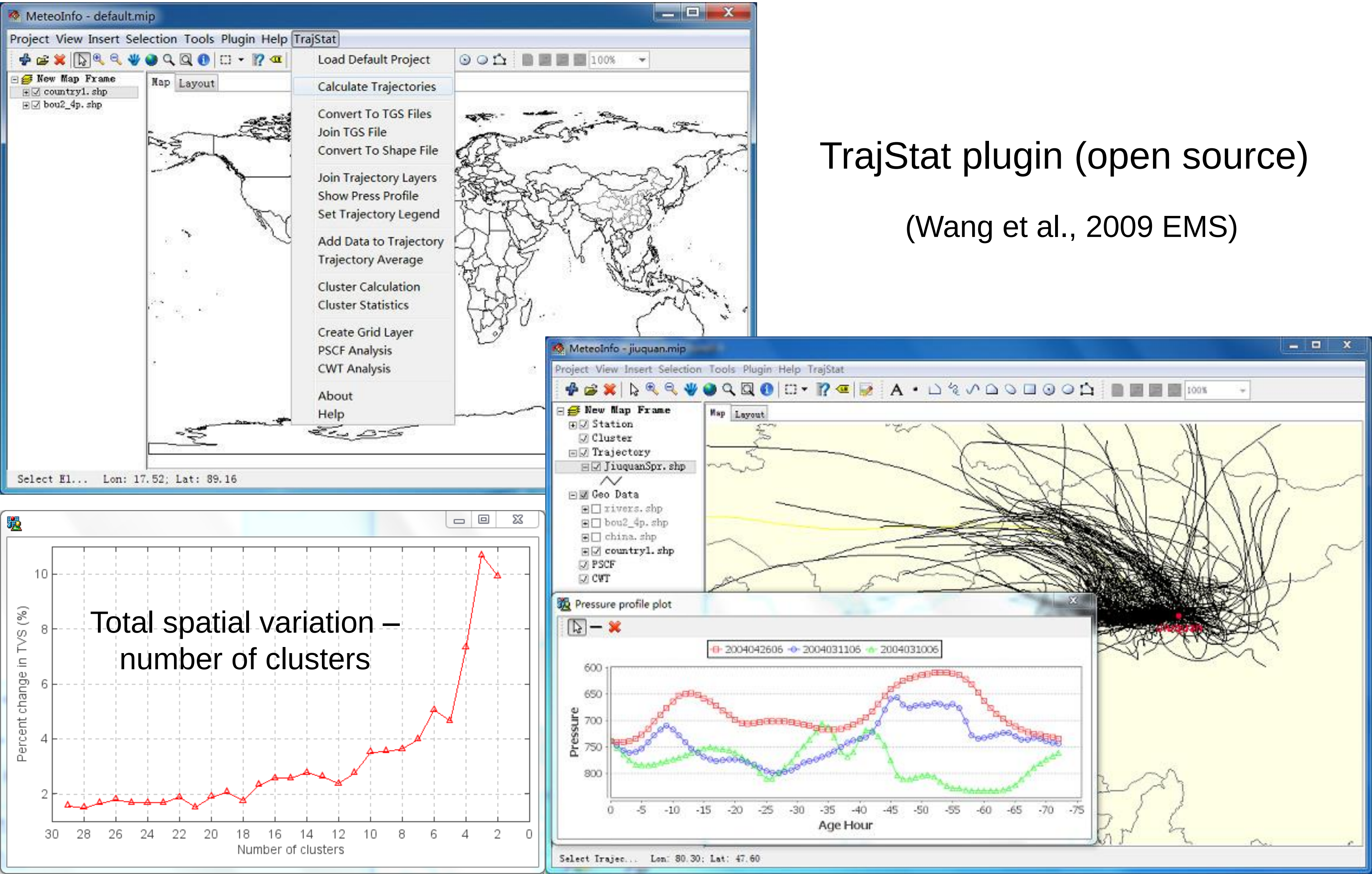
变量: VWND
时次: 1995-10-16 00:...
高度: 0.9823
图形类型: Strea...

X: -1032831.6; Y: -6902992.2 (Lon: -113.51; Lat: 32.64)

Variable: PS
Time: 1987-01-02 00:...
Level: Surface
Draw Type: Contour

Lon: 138.20; Lat: 60.92

MeteoInfo – Write plugins to do specific task

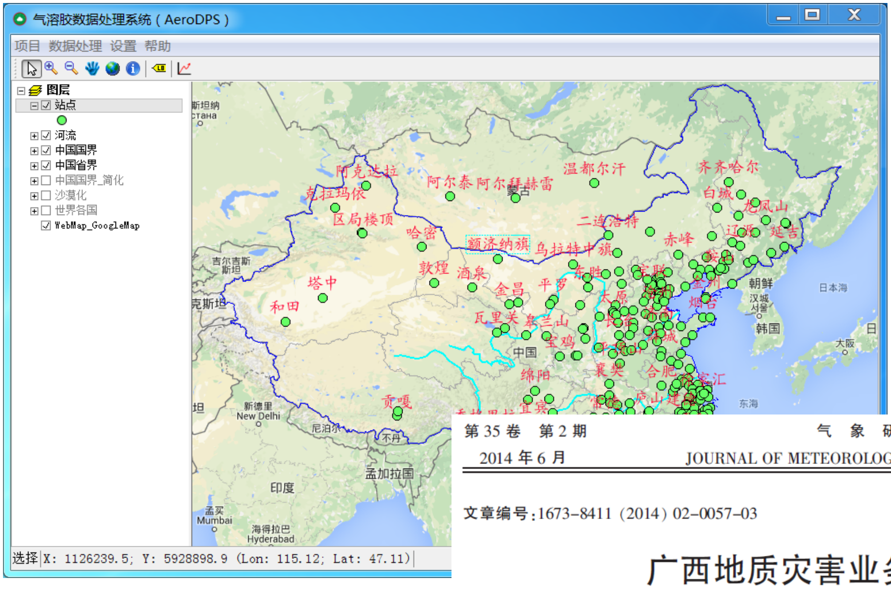


Total spatial variation – number of clusters

TrajStat plugin (open source)

(Wang et al., 2009 EMS)

MeteoInfo – Software development using the library



选择|X: 1126239.5; Y: 5928898.9 (Lon: 115.12; Lat: 47.11)|

第 42 卷第 6 期
2014 年 12 月

气 象 科 技
METEOROLOGICAL SCIENCE AND TECHNOLOGY

Vol 42, No. 6
Dec. 2014

基于 GIS 和 RS 的西藏森林火险等级计算方法

林志强[1] 马艳鲜[1] 德庆[2] 边多[3]

（1 西藏自治区气象服务中心，拉萨 850000；2 西藏自治区气象台，拉萨 850000；
3 西藏自治区高原大气环境科学研究所，拉萨 850000）

摘要 基于遥感技术(RS)和地理信息系统(GIS)，利用由基于 DEM 演算的地面最高温度、最小相对湿度和最大风速等格点化气象要素，FY2 静止气象卫星逐日降水反演产品和 AVHRR 积雪监测产品计算网格森林火险天气等级，结合由植被类型、NDVI、地形要素和公路、人口聚居地等要素评估的森林火险风险等级，综合计算得到网格化的西藏森林火险等级。该项业务程序基于 MeteoInfo 组件建立，能够实现全自动化业务运行。对于森林火灾事件，

的准确性更高，能为西藏林区森林防火工作提供有效参考。

第 35 卷 第 2 期
2014 年 6 月

气 象 研 究 与 应 用
JOURNAL OF METEOROLOGICAL RESEARCH AND APPLICATION

Vol.35 No.2
Jun. 2014

文章编号：1673-8411（2014）02-0057-03

广西地质灾害业务平台的设计与开发

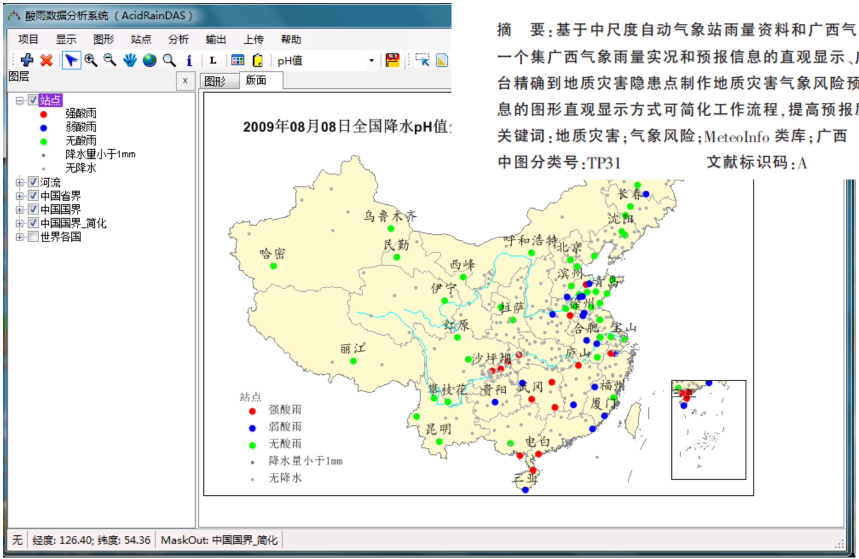
曾 鹏，廖国莲，莫雨淳，李 雄，吴燊先

（广西气象服务中心，广西 南宁 530022）

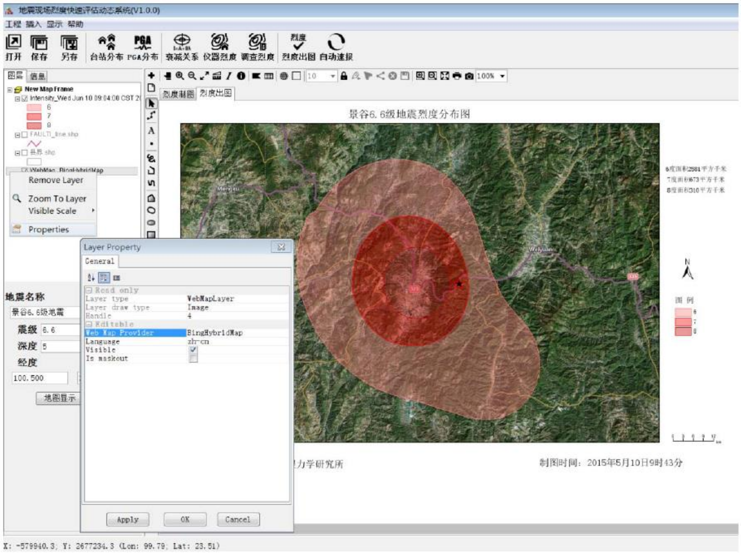
摘 要：基于中尺度自动气象站雨量资料和广西气象台雨量预报等资料，利用 C#、SQL 和 Meteol
一个集广西气象雨量实况和预报信息的直观显示、广西地质灾害预警的制作及会商等功能于一亻
台精确到地质灾害隐患点制作地质灾害气象风险预警，推进了广西地质灾害精细化预报的进程。
息的图形直观显示方式可简化工作流程，提高预报质量和工作效率。

关键词：地质灾害；气象风险；MeteoInfo 类库；广西

中图分类号：TP31 文献标识码：A



无 | 经度: 126.40; 纬度: 54.36 | MaskOut: 中国国界_简化

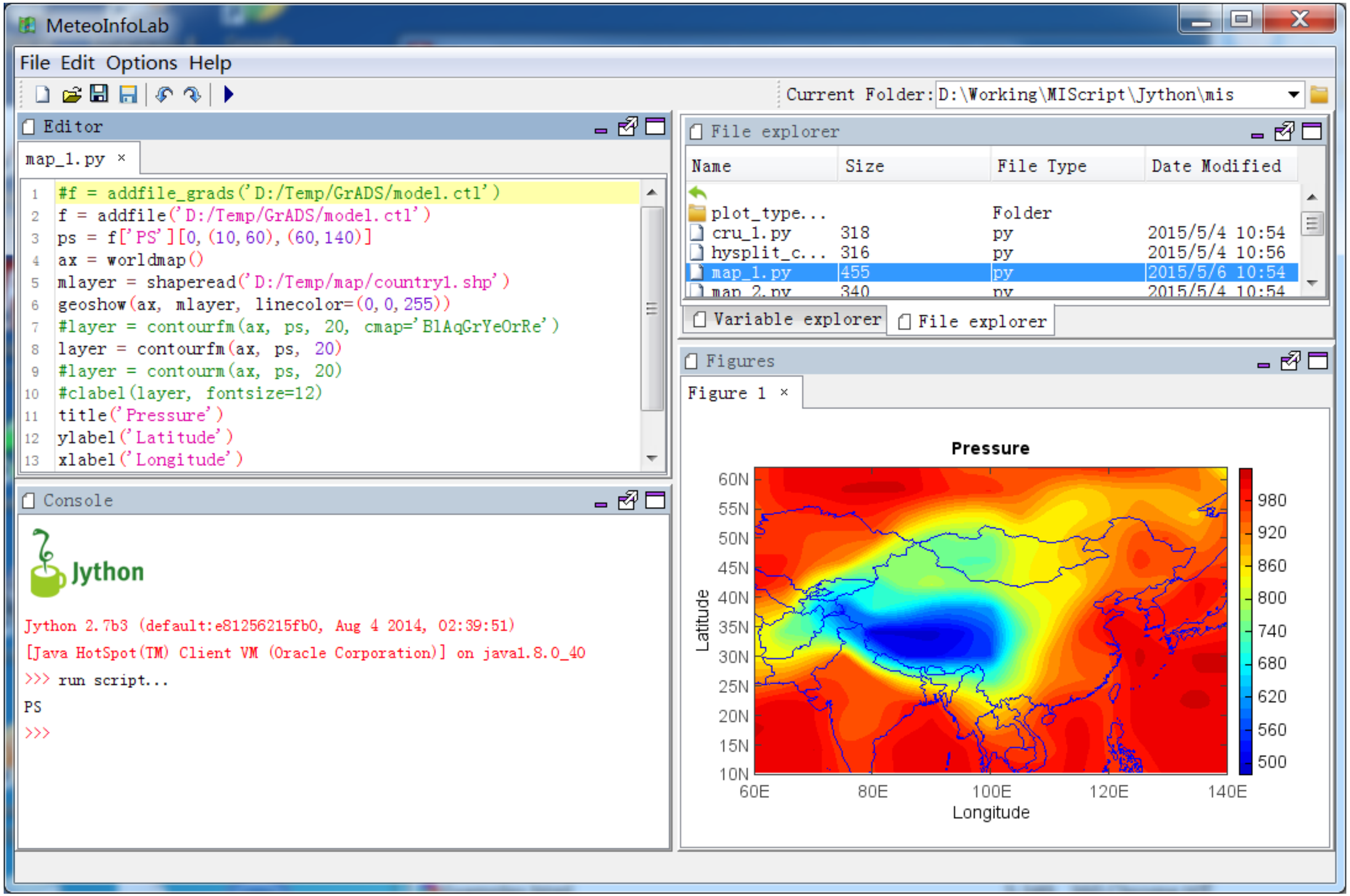


Read only
Layer type: WebMapLayer
Layer draw type: Image
Handle: 4
Editable
Web Map Provider: BingHybridMap
Language: zh-cn
Visible: ☑
Is maskout: □

X: -579940.3; Y: 2677234.3 (Lon: 99.79; Lat: 23.51)

MeteoInfoLab: Scientific data analysis and visualization

MeteoInfoLab is open source software product developed using Java and Jython based on MeteoInfo Java library (Unidata netCDF Java library is used). The purpose is to provide an optional scientific computation and visualization tool similar with MatLab, NCL or Python (Numpy + Matplotlib + Basemap).



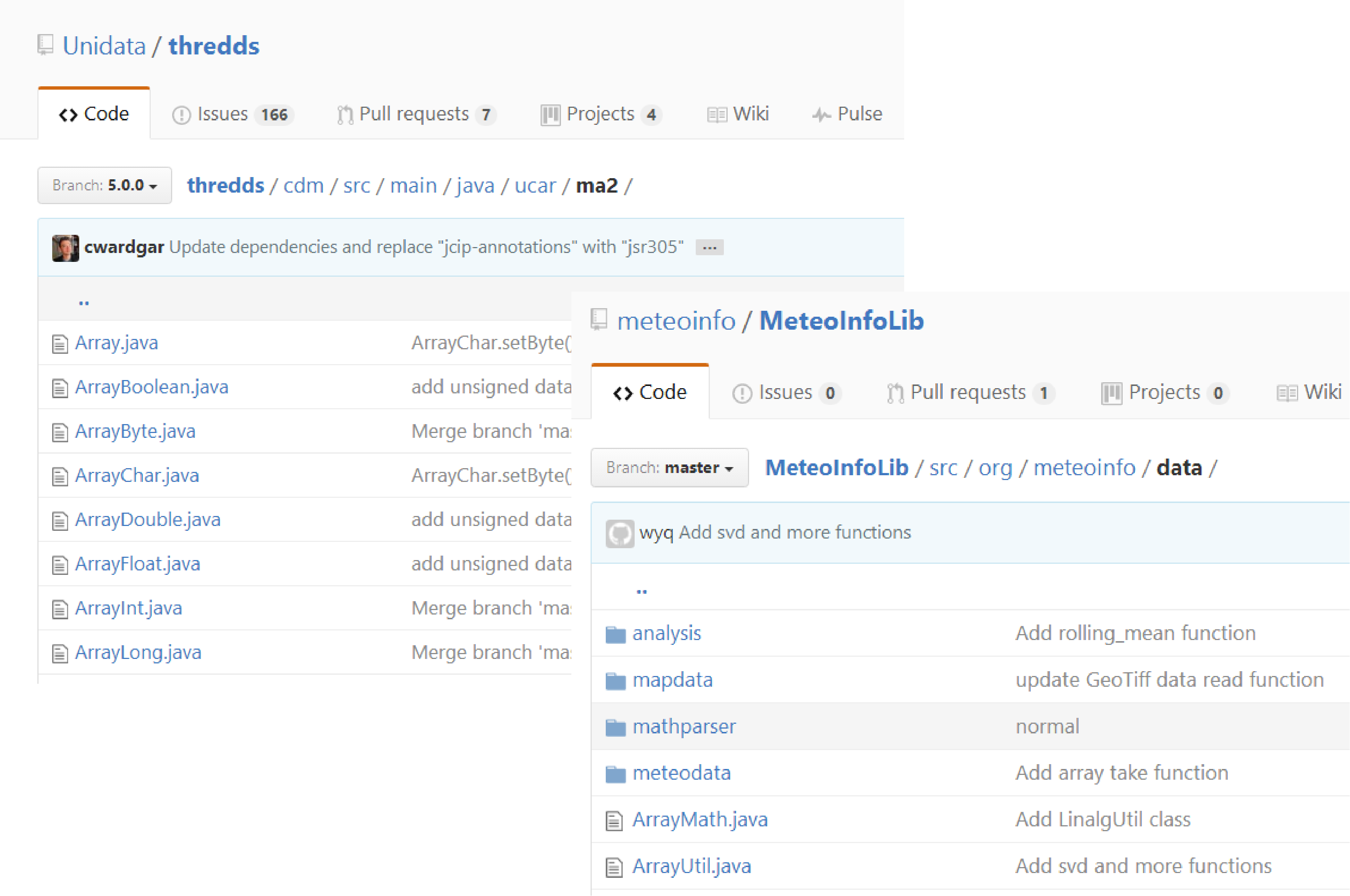
Current Folder: D:\Working\MIScript\Jython\mis

```
#f = addfile_grads('D:/Temp/GrADS/model.ctl')
f = addfile('D:/Temp/GrADS/model.ctl')
ps = f['PS'][0,(10,60),(60,140)]
ax = worldmap()
mlayer = shaperead('D:/Temp/map/country1.shp')
geoshow(ax, mlayer, linecolor=(0,0,255))
#layer = contourfm(ax, ps, 20, cmap='BlAqGrYeOrRe')
layer = contourfm(ax, ps, 20)
#layer = contourm(ax, ps, 20)
#clabel(layer, fontsize=12)
title('Pressure')
ylabel('Latitude')
xlabel('Longitude')
```

Name	Size	File Type	Date Modified
plot_type...		Folder	
cru_1.py	318	py	2015/5/4 10:54
hysplit_c...	316	py	2015/5/4 10:56
map_1.py	455	py	2015/5/6 10:54
map_2.py	340	py	2015/5/4 10:54

```
Jython 2.7b3 (default:e81256215fb0, Aug 4 2014, 02:39:51)
[Java HotSpot(TM) Client VM (Oracle Corporation)] on java1.8.0_40
>>> run script...
PS
>>>
```

Multi-dimensional array support in Java



Unidata / **thredds**

Branch: **5.0.0** ▾ thredds / cdm / src / main / java / ucar / **ma2** /

cwardgar Update dependencies and replace "jcip-annotations" with "jsr305" ...

..	
Array.java	ArrayChar.setByte(
ArrayBoolean.java	add unsigned data
ArrayByte.java	Merge branch 'ma
ArrayChar.java	ArrayChar.setByte(
ArrayDouble.java	add unsigned data
ArrayFloat.java	add unsigned data
ArrayInt.java	Merge branch 'ma
ArrayLong.java	Merge branch 'ma

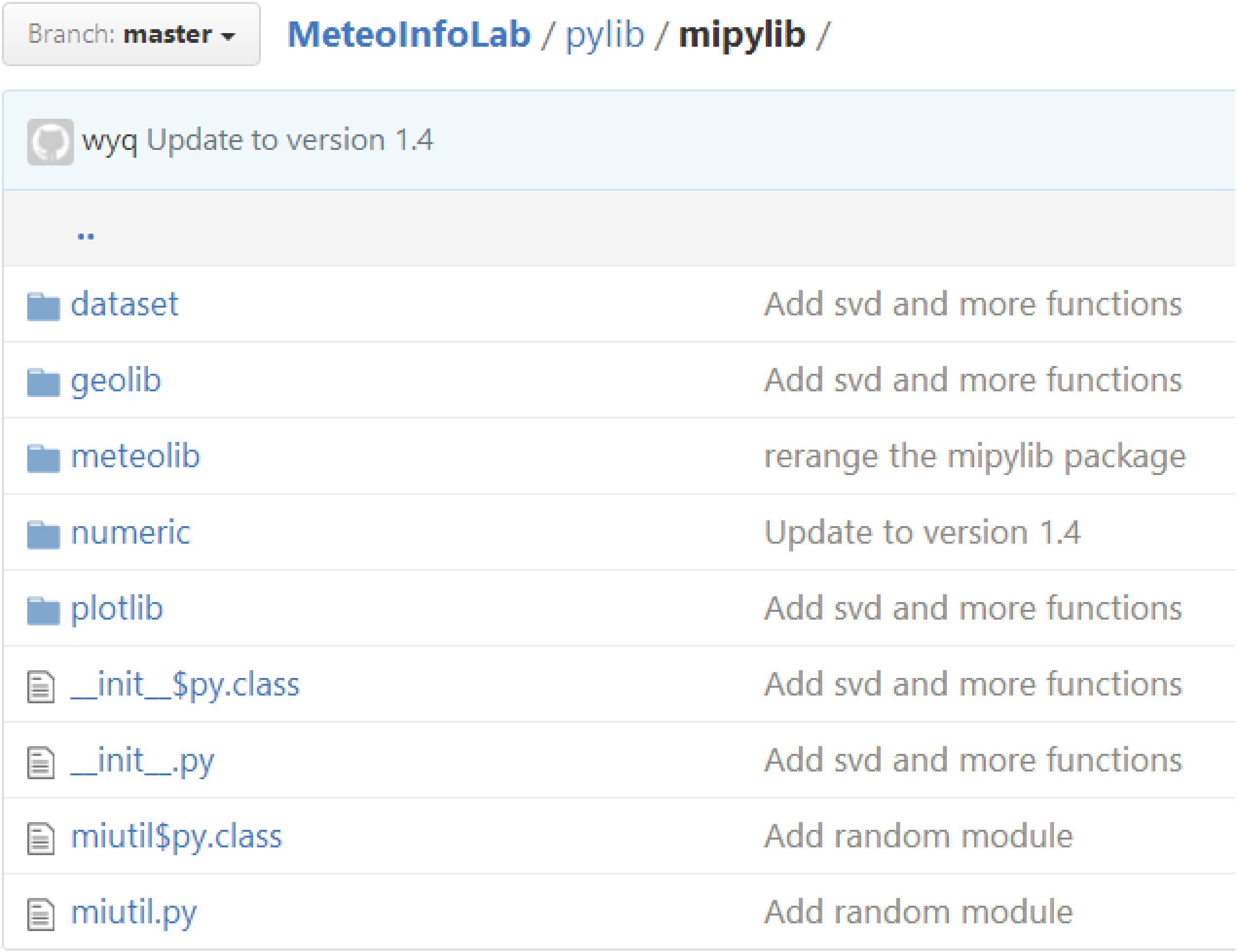
meteoinfo / **MeteoInfoLib**

Branch: **master** ▾ MeteoInfoLib / src / org / meteoinfo / **data** /

wyq Add svd and more functions

..	
analysis	Add rolling_mean function
mapdata	update GeoTiff data read function
mathparser	normal
meteodata	Add array take function
ArrayMath.java	Add LinalgUtil class
ArrayUtil.java	Add svd and more functions

mipylib – Jython library of MeteoInfoLab



Branch: **master** ▾

MeteoInfoLab / pylib / **mipylib** /

wyq Update to version 1.4

..

dataset	Add svd and more functions
geolib	Add svd and more functions
meteolib	rerange the mipylib package
numeric	Update to version 1.4
plotlib	Add svd and more functions
__init__$py.class	Add svd and more functions
__init__.py	Add svd and more functions
miutil$py.class	Add random module
miutil.py	Add random module

minum.py

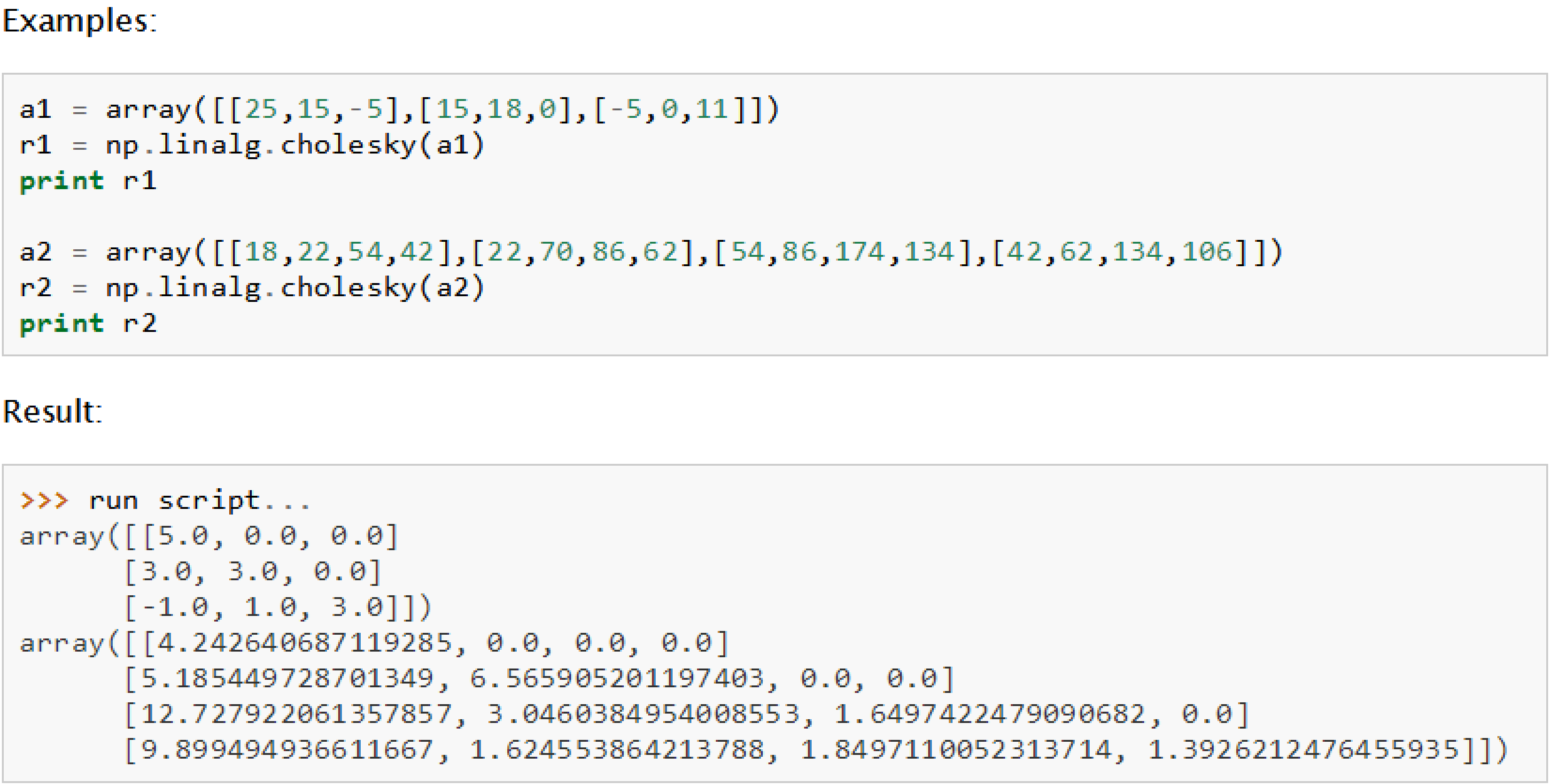
miplot.py

midata.py

Mathematics and statistics components

Apache Commons™
http://commons.apache.org/

commons [Math]™



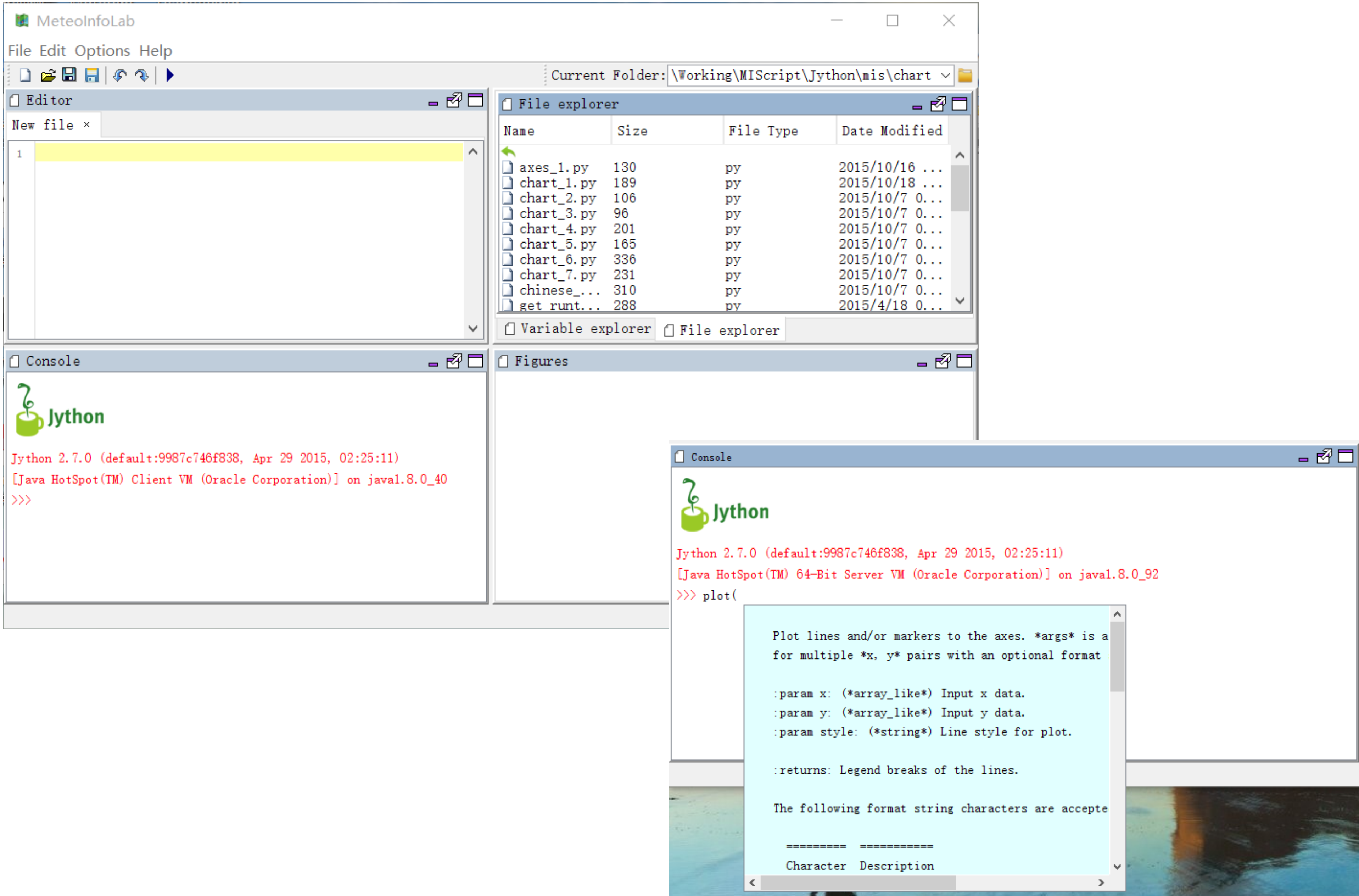
Examples:

```
a1 = array([[25,15,-5],[15,18,0],[-5,0,11]])
r1 = np.linalg.cholesky(a1)
print r1

a2 = array([[18,22,54,42],[22,70,86,62],[54,86,174,134],[42,62,134,106]])
r2 = np.linalg.cholesky(a2)
print r2
```

Result:

```
>>> run script...
array([[5.0, 0.0, 0.0]
      [3.0, 3.0, 0.0]
      [-1.0, 1.0, 3.0]])
array([[4.242640687119285, 0.0, 0.0, 0.0]
      [5.185449728701349, 6.565905201197403, 0.0, 0.0]
      [12.727922061357857, 3.0460384954008553, 1.6497422479090682, 0.0]
      [9.899494936611667, 1.624553864213788, 1.8497110052313714, 1.3926212476455935]])
```

MeteoInfoLab GUI



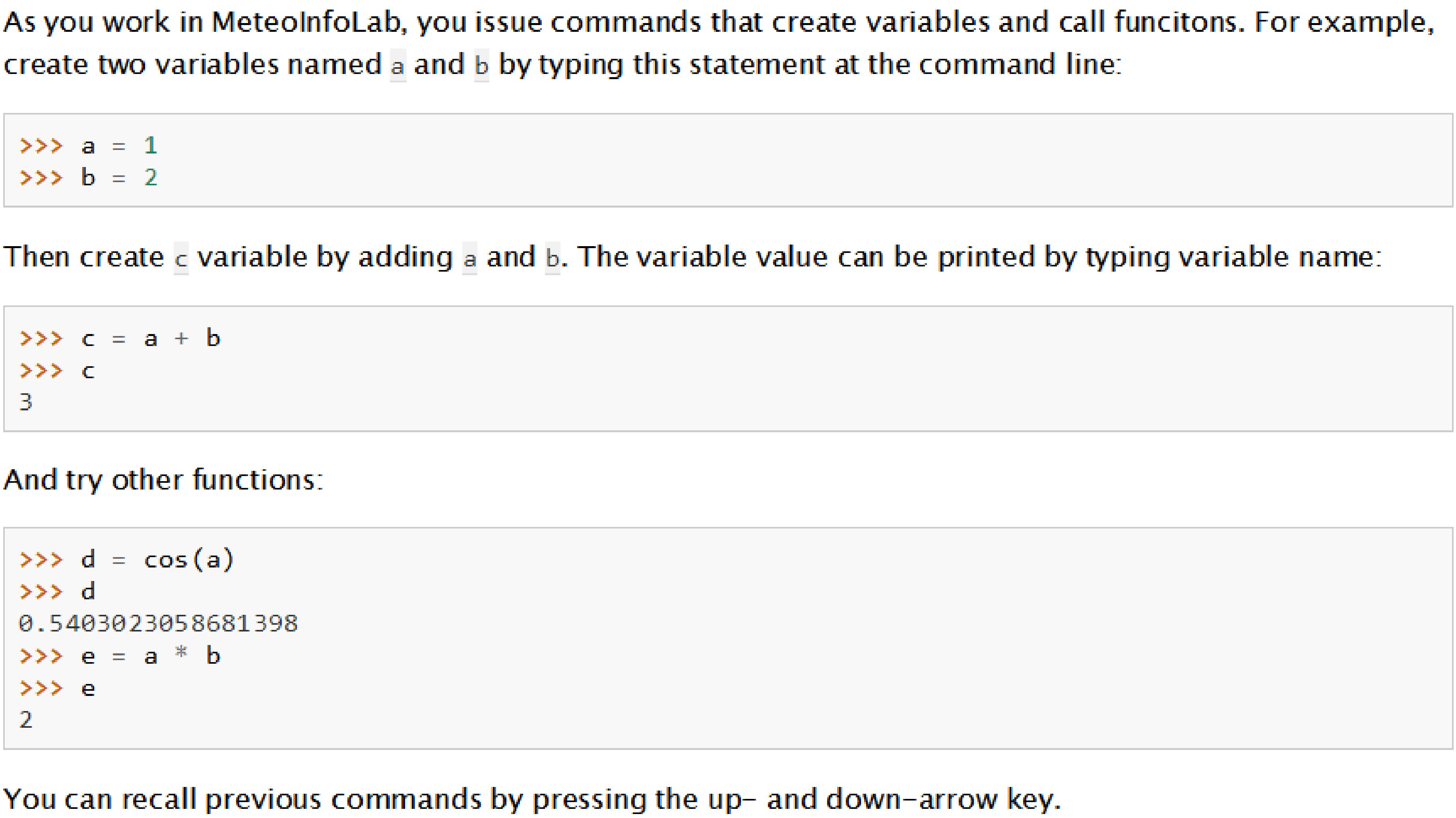
Name	Size	File Type	Date Modified
axes_1.py	130	py	2015/10/16 ...
chart_1.py	189	py	2015/10/18 ...
chart_2.py	106	py	2015/10/7 0...
chart_3.py	96	py	2015/10/7 0...
chart_4.py	201	py	2015/10/7 0...
chart_5.py	165	py	2015/10/7 0...
chart_6.py	336	py	2015/10/7 0...
chart_7.py	231	py	2015/10/7 0...
chinese_...	310	py	2015/10/7 0...
get_runt...	288	py	2015/4/18 0...

```
Jython 2.7.0 (default:9987c746f838, Apr 29 2015, 02:25:11)
[Java HotSpot(TM) Client VM (Oracle Corporation)] on java1.8.0_40
>>>
```

```
Jython 2.7.0 (default:9987c746f838, Apr 29 2015, 02:25:11)
[Java HotSpot(TM) 64-Bit Server VM (Oracle Corporation)] on java1.8.0_92
>>> plot(
```

```
    Plot lines and/or markers to the axes. *args* is a
    for multiple *x, y* pairs with an optional format

    :param x: (*array_like*) Input x data.
    :param y: (*array_like*) Input y data.
    :param style: (*string*) Line style for plot.

    :returns: Legend breaks of the lines.

    The following format string characters are accepte

      =========  ===========
      Character  Description
```

MeteoInfoLab GUI



As you work in MeteoInfoLab, you issue commands that create variables and call funcitons. For example, create two variables named `a` and `b` by typing this statement at the command line:

```
>>> a = 1
>>> b = 2
```

Then create `c` variable by adding `a` and `b`. The variable value can be printed by typing variable name:

```
>>> c = a + b
>>> c
3
```

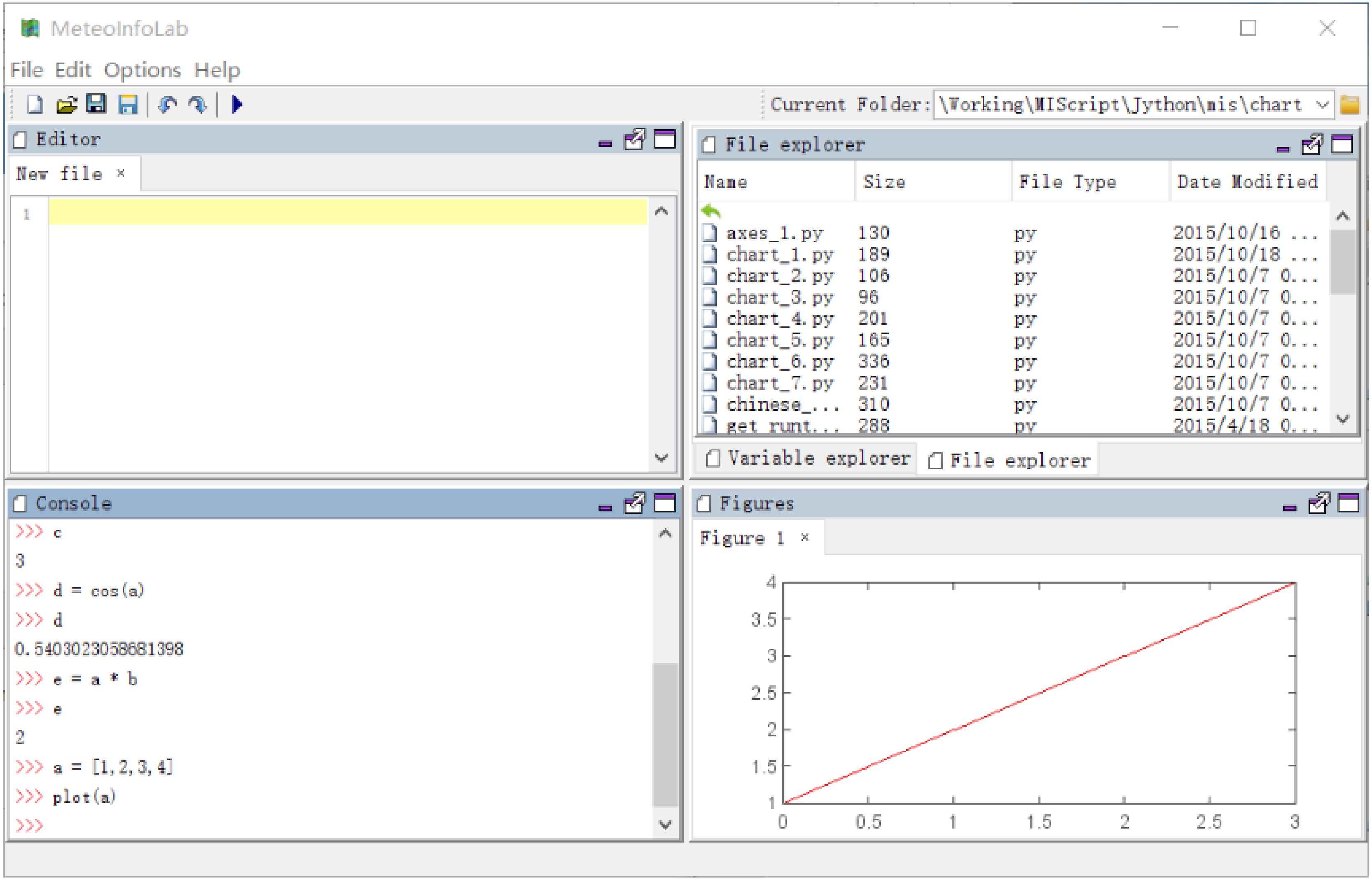
And try other functions:

```
>>> d = cos(a)
>>> d
0.5403023058681398
>>> e = a * b
>>> e
2
```

You can recall previous commands by pressing the up- and down-arrow key.

MeteoInfoLab GUI

```
>>> a = [1,2,3,4]
>>> plot(a)
```

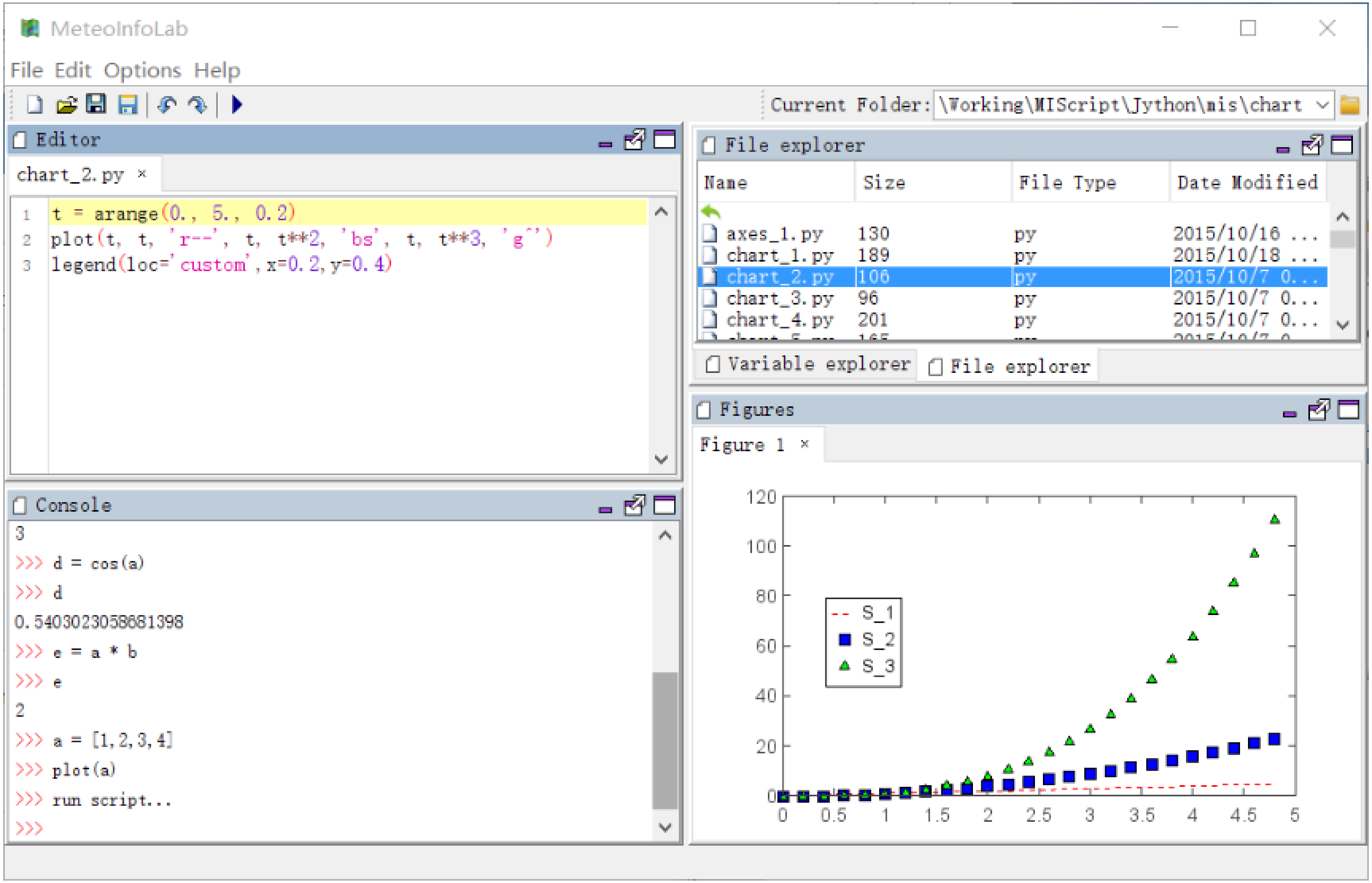


Name	Size	File Type	Date Modified
axes_1.py	130	py	2015/10/16 ...
chart_1.py	189	py	2015/10/18 ...
chart_2.py	106	py	2015/10/7 0...
chart_3.py	96	py	2015/10/7 0...
chart_4.py	201	py	2015/10/7 0...
chart_5.py	165	py	2015/10/7 0...
chart_6.py	336	py	2015/10/7 0...
chart_7.py	231	py	2015/10/7 0...
chinese_...	310	py	2015/10/7 0...
get_runt...	288	py	2015/4/18 0...

```
>>> c
3
>>> d = cos(a)
>>> d
0.5403023058681398
>>> e = a * b
>>> e
2
>>> a = [1,2,3,4]
>>> plot(a)
>>>
```

MeteoInfoLab GUI

Also you can write a script program in editor panel and run it by clicking `Run Script` button in toolbar.

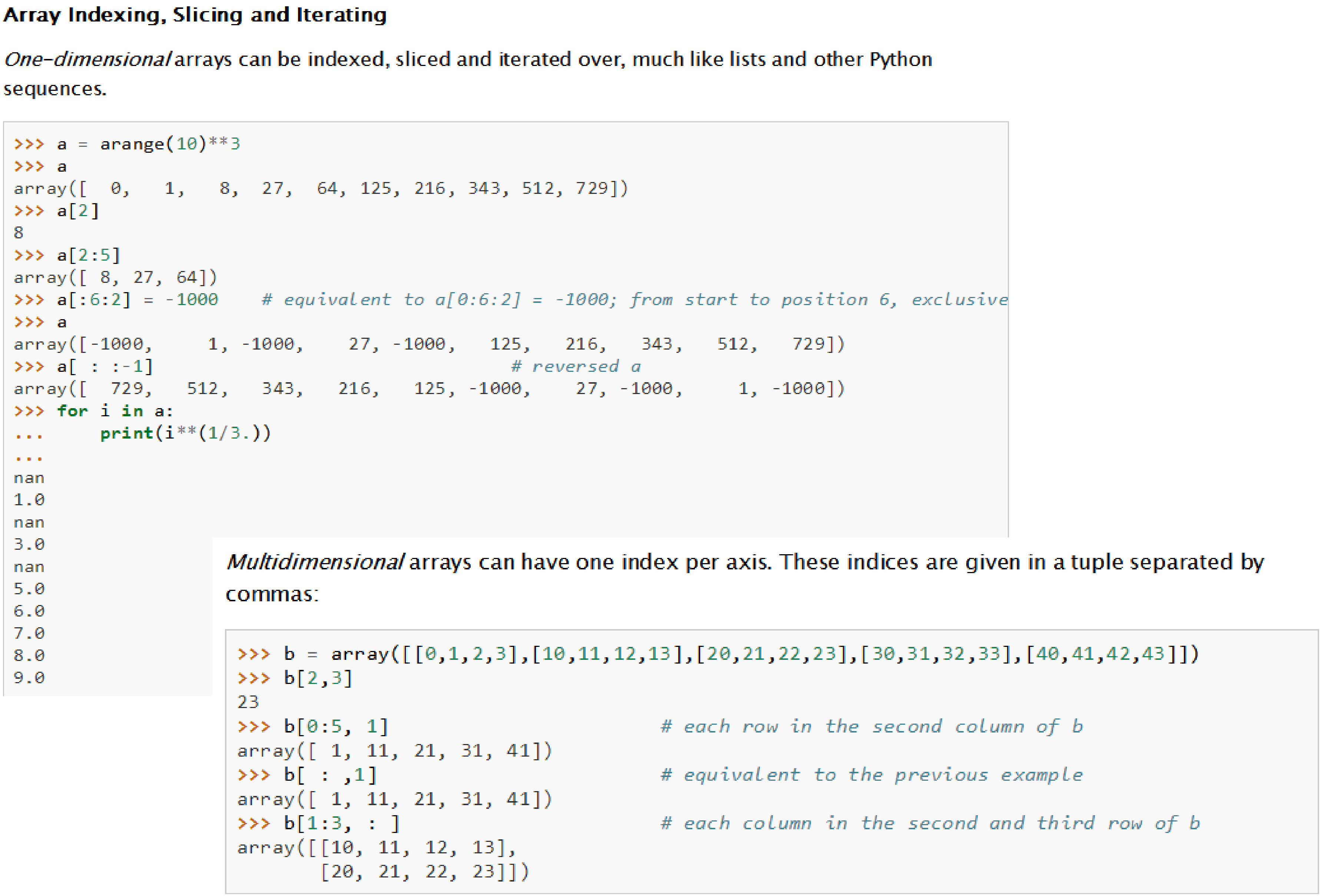


```
t = arange(0., 5., 0.2)
plot(t, t, 'r--', t, t**2, 'bs', t, t**3, 'g^')
legend(loc='custom', x=0.2, y=0.4)
```

Name	Size	File Type	Date Modified
axes_1.py	130	py	2015/10/16 ...
chart_1.py	189	py	2015/10/18 ...
chart_2.py	106	py	2015/10/7 0...
chart_3.py	96	py	2015/10/7 0...
chart_4.py	201	py	2015/10/7 0...

```
3
>>> d = cos(a)
>>> d
0.5403023058681398
>>> e = a * b
>>> e
2
>>> a = [1,2,3,4]
>>> plot(a)
>>> run script...
>>>
```

MeteoInfoLab - Multi-dimensional array



Array Indexing, Slicing and Iterating

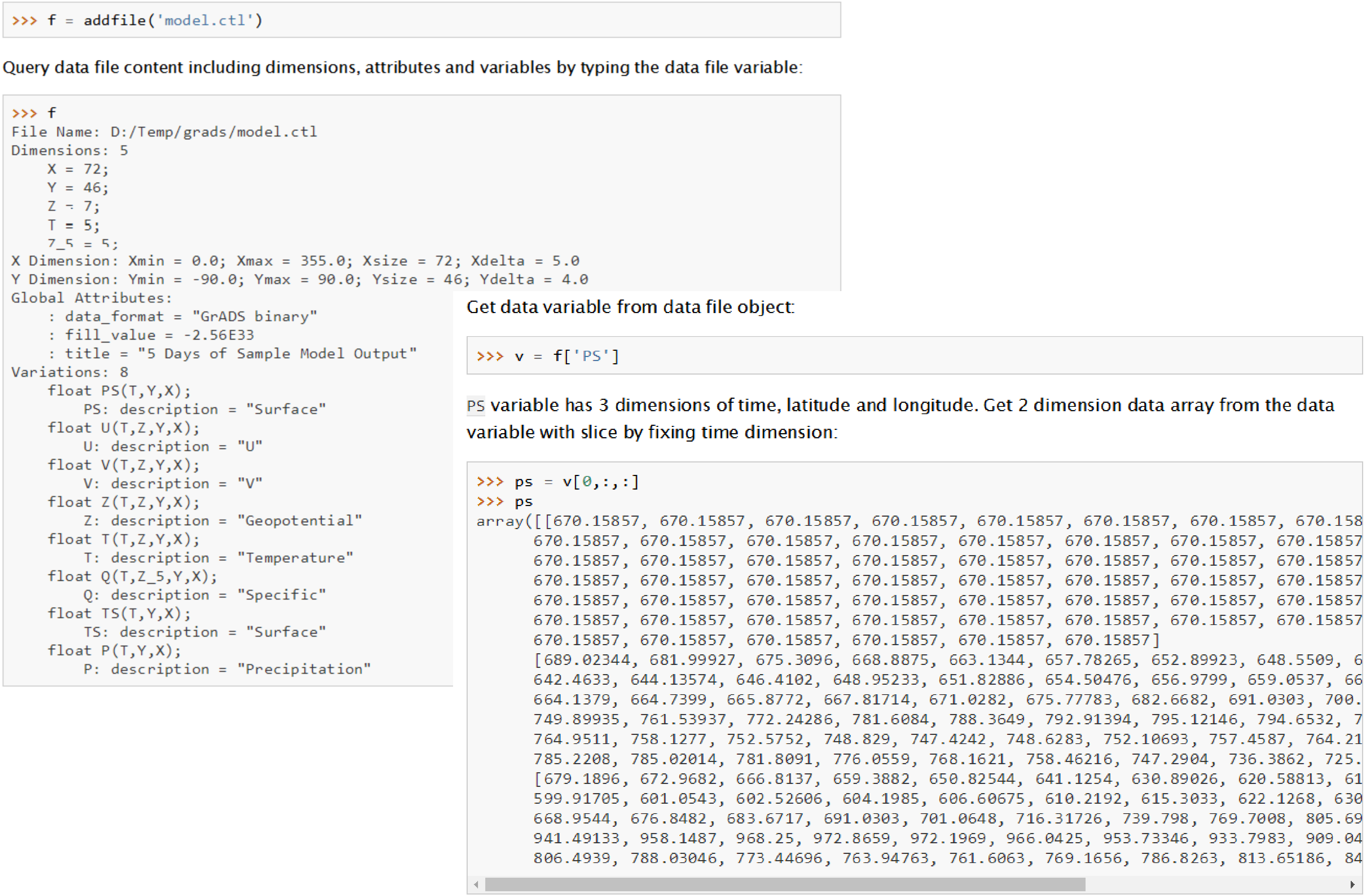
One-dimensional arrays can be indexed, sliced and iterated over, much like lists and other Python sequences.

```
>>> a = arange(10)**3
>>> a
array([  0,   1,   8,  27,  64, 125, 216, 343, 512, 729])
>>> a[2]
8
>>> a[2:5]
array([ 8, 27, 64])
>>> a[:6:2] = -1000    # equivalent to a[0:6:2] = -1000; from start to position 6, exclusive
>>> a
array([-1000,     1, -1000,    27, -1000,   125,   216,   343,   512,   729])
>>> a[ : :-1]                                 # reversed a
array([  729,   512,   343,   216,   125, -1000,    27, -1000,     1, -1000])
>>> for i in a:
...     print(i**(1/3.))
...
nan
1.0
nan
3.0
nan
5.0
6.0
7.0
8.0
9.0
```

Multidimensional arrays can have one index per axis. These indices are given in a tuple separated by commas:

```
>>> b = array([[0,1,2,3],[10,11,12,13],[20,21,22,23],[30,31,32,33],[40,41,42,43]])
>>> b[2,3]
23
>>> b[0:5, 1]                       # each row in the second column of b
array([ 1, 11, 21, 31, 41])
>>> b[ : ,1]                        # equivalent to the previous example
array([ 1, 11, 21, 31, 41])
>>> b[1:3, : ]                      # each column in the second and third row of b
array([[10, 11, 12, 13],
       [20, 21, 22, 23]])
```

MeteoInfoLab – Read meteorological data



```
>>> f = addfile('model.ctl')
```

Query data file content including dimensions, attributes and variables by typing the data file variable:

```
>>> f
File Name: D:/Temp/grads/model.ctl
Dimensions: 5
    X = 72;
    Y = 46;
    Z = 7;
    T = 5;
    Z_5 = 5;
X Dimension: Xmin = 0.0; Xmax = 355.0; Xsize = 72; Xdelta = 5.0
Y Dimension: Ymin = -90.0; Ymax = 90.0; Ysize = 46; Ydelta = 4.0
Global Attributes:
    : data_format = "GrADS binary"
    : fill_value = -2.56E33
    : title = "5 Days of Sample Model Output"
Variations: 8
    float PS(T,Y,X);
        PS: description = "Surface"
    float U(T,Z,Y,X);
        U: description = "U"
    float V(T,Z,Y,X);
        V: description = "V"
    float Z(T,Z,Y,X);
        Z: description = "Geopotential"
    float T(T,Z,Y,X);
        T: description = "Temperature"
    float Q(T,Z_5,Y,X);
        Q: description = "Specific"
    float TS(T,Y,X);
        TS: description = "Surface"
    float P(T,Y,X);
        P: description = "Precipitation"
```

Get data variable from data file object:

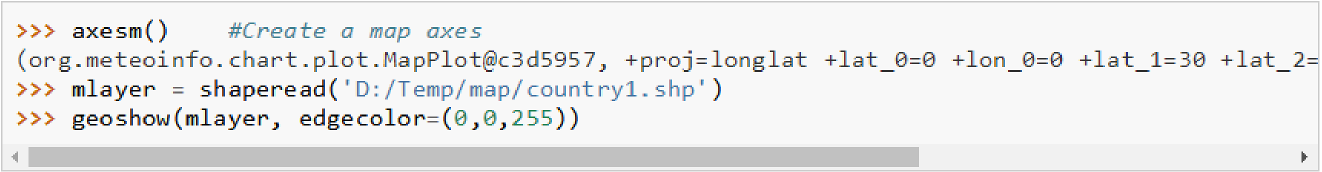
```
>>> v = f['PS']
```

PS variable has 3 dimensions of time, latitude and longitude. Get 2 dimension data array from the data variable with slice by fixing time dimension:

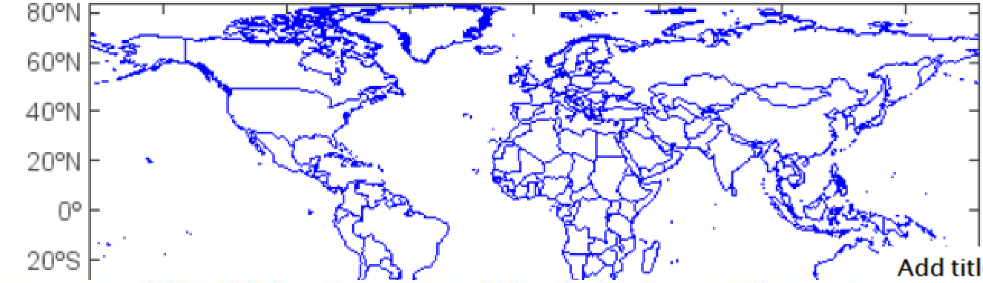
```
>>> ps = v[0,:,:]
>>> ps
array([[670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.158
        670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857
        670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857
        670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857
        670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857
        670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857
        670.15857, 670.15857, 670.15857, 670.15857, 670.15857, 670.15857]
        [689.02344, 681.99927, 675.3096, 668.8875, 663.1344, 657.78265, 652.89923, 648.5509, 6
        642.4633, 644.13574, 646.4102, 648.95233, 651.82886, 654.50476, 656.9799, 659.0537, 66
        664.1379, 664.7399, 665.8772, 667.81714, 671.0282, 675.77783, 682.6682, 691.0303, 700.
        749.89935, 761.53937, 772.24286, 781.6084, 788.3649, 792.91394, 795.12146, 794.6532, 7
        764.9511, 758.1277, 752.5752, 748.829, 747.4242, 748.6283, 752.10693, 757.4587, 764.21
        785.2208, 785.02014, 781.8091, 776.0559, 768.1621, 758.46216, 747.2904, 736.3862, 725.
        [679.1896, 672.9682, 666.8137, 659.3882, 650.82544, 641.1254, 630.89026, 620.58813, 61
        599.91705, 601.0543, 602.52606, 604.1985, 606.60675, 610.2192, 615.3033, 622.1268, 630
        668.9544, 676.8482, 683.6717, 691.0303, 701.0648, 716.31726, 739.798, 769.7008, 805.69
        941.49133, 958.1487, 968.25, 972.8659, 972.1969, 966.0425, 953.73346, 933.7983, 909.04
        806.4939, 788.03046, 773.44696, 763.94763, 761.6063, 769.1656, 786.8263, 813.65186, 84
```

MeteoInfoLab – Plot data

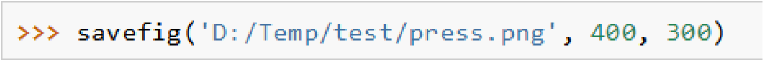
Plot map: create a map axes with axesm function; read shape file, view geodata layer:



```
>>> axesm()    #Create a map axes
(org.meteoinfo.chart.plot.MapPlot@c3d5957, +proj=longlat +lat_0=0 +lon_0=0 +lat_1=30 +lat_2=
>>> mlayer = shaperead('D:/Temp/map/country1.shp')
>>> geoshow(mlayer, edgecolor=(0,0,255))
```

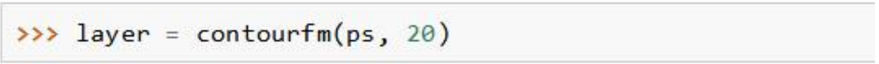


Save figure:

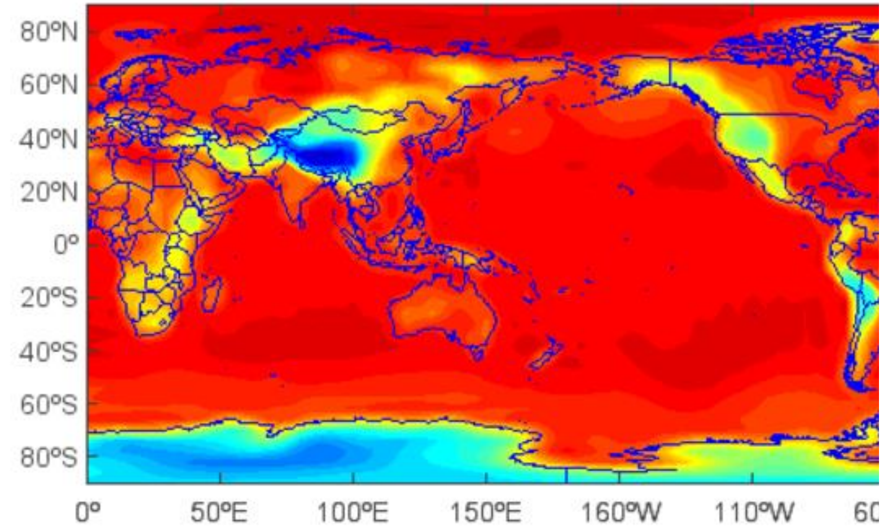


```
>>> savefig('D:/Temp/test/press.png', 400, 300)
```

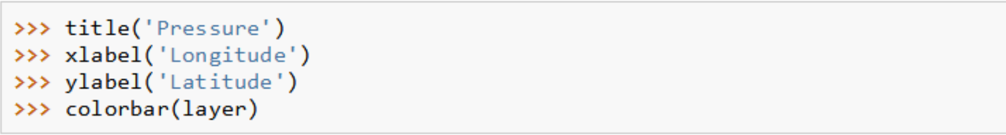
Create and plot filled contour layer from the dimension data array



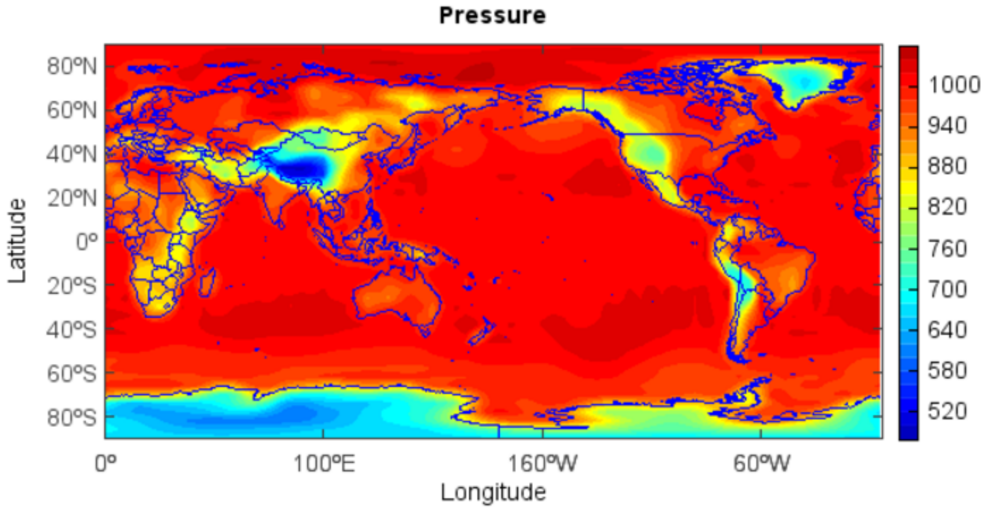
```
>>> layer = contourfm(ps, 20)
```



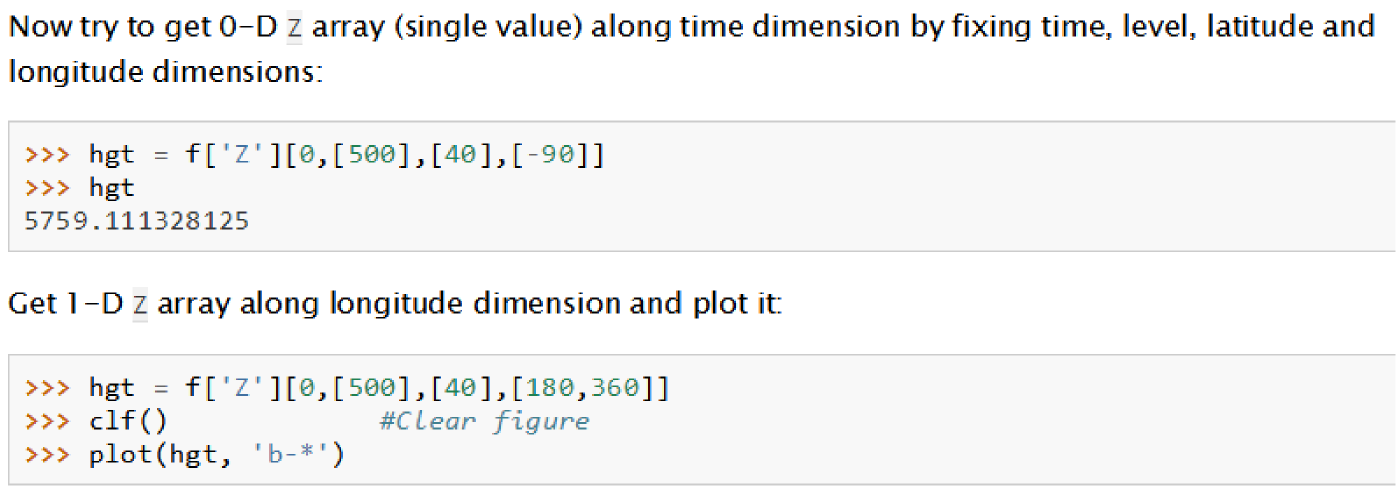
Add title, x and y labels and colorbar:



```
>>> title('Pressure')
>>> xlabel('Longitude')
>>> ylabel('Latitude')
>>> colorbar(layer)
```



MeteoInfoLab – Plot data

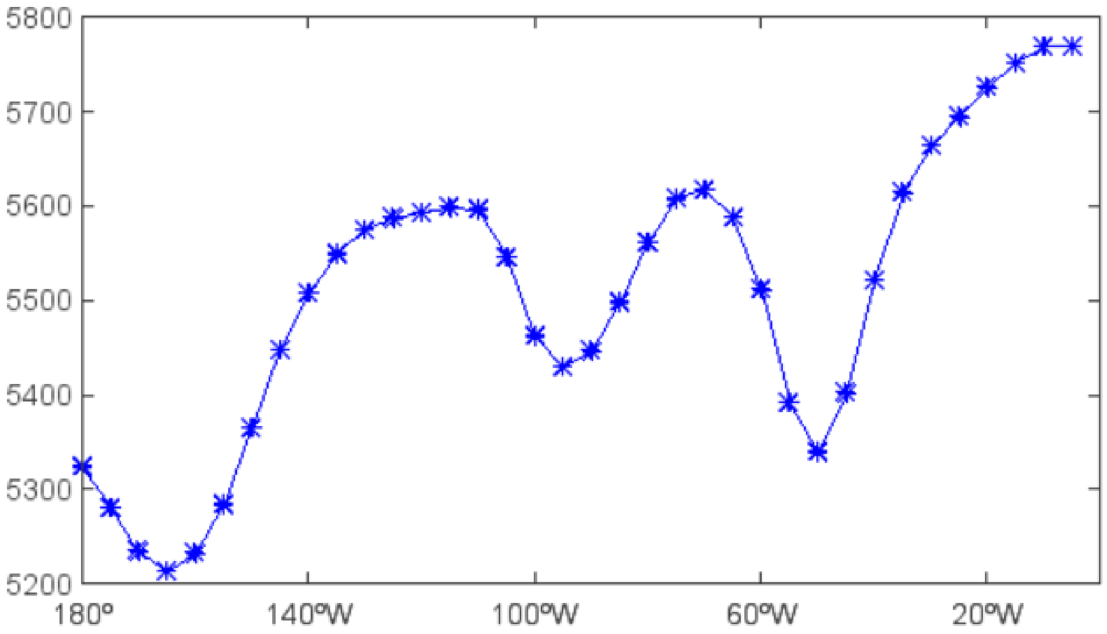


Now try to get 0-D `Z` array (single value) along time dimension by fixing time, level, latitude and longitude dimensions:

```
>>> hgt = f['Z'][0,[500],[40],[-90]]
>>> hgt
5759.111328125
```

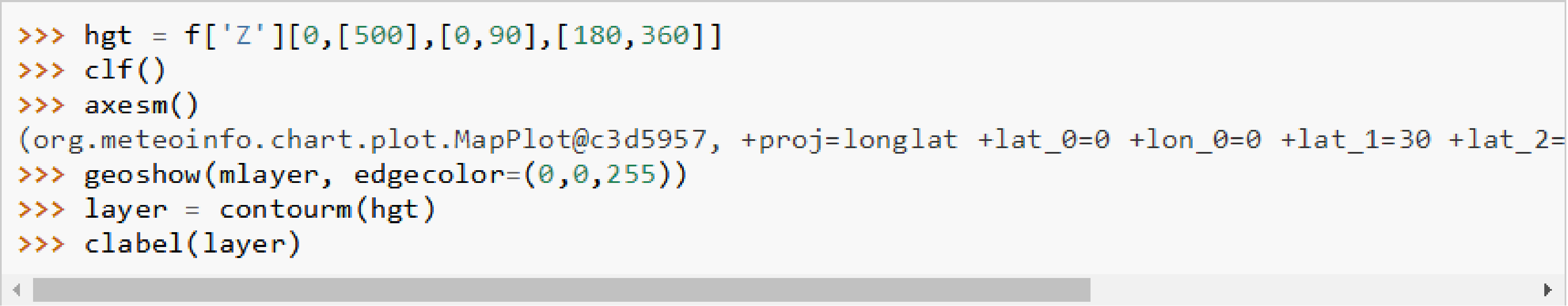
Get 1-D `Z` array along longitude dimension and plot it:

```
>>> hgt = f['Z'][0,[500],[40],[180,360]]
>>> clf()                #Clear figure
>>> plot(hgt, 'b-*')
```

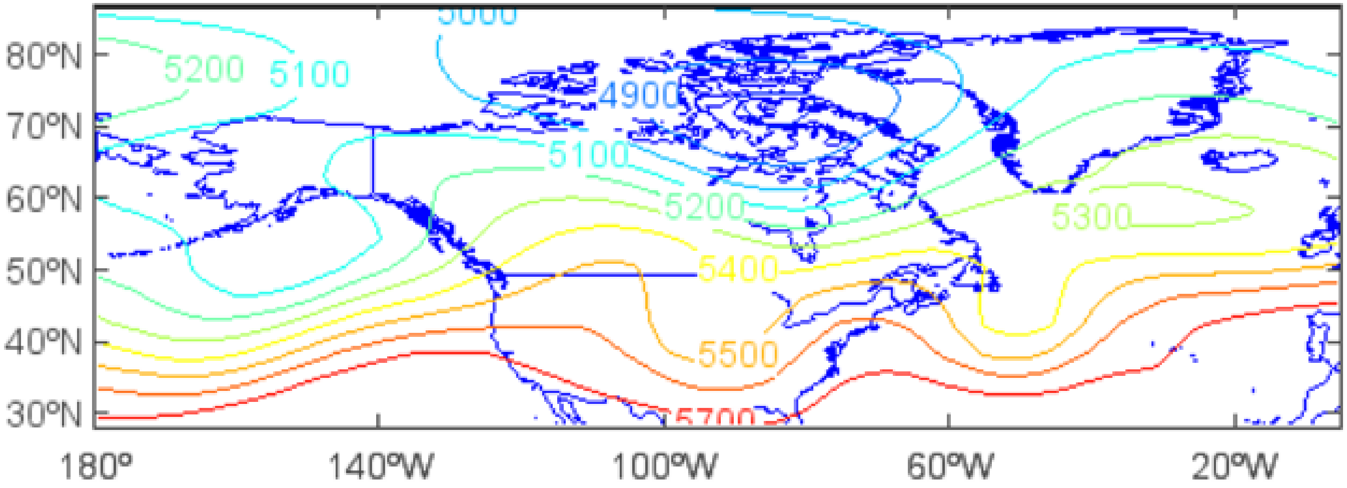


MeteoInfoLab – Plot data

Get and plot 2-D `Z` array with dimensions of latitude and longitude:

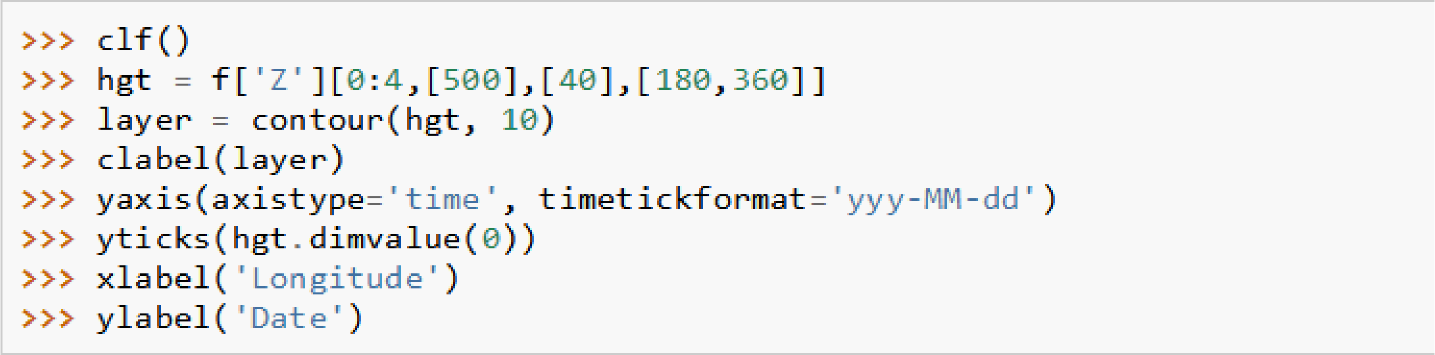


```
>>> hgt = f['Z'][0,[500],[0,90],[180,360]]
>>> clf()
>>> axesm()
(org.meteoinfo.chart.plot.MapPlot@c3d5957, +proj=longlat +lat_0=0 +lon_0=0 +lat_1=30 +lat_2=
>>> geoshow(mlayer, edgecolor=(0,0,255))
>>> layer = contourm(hgt)
>>> clabel(layer)
```

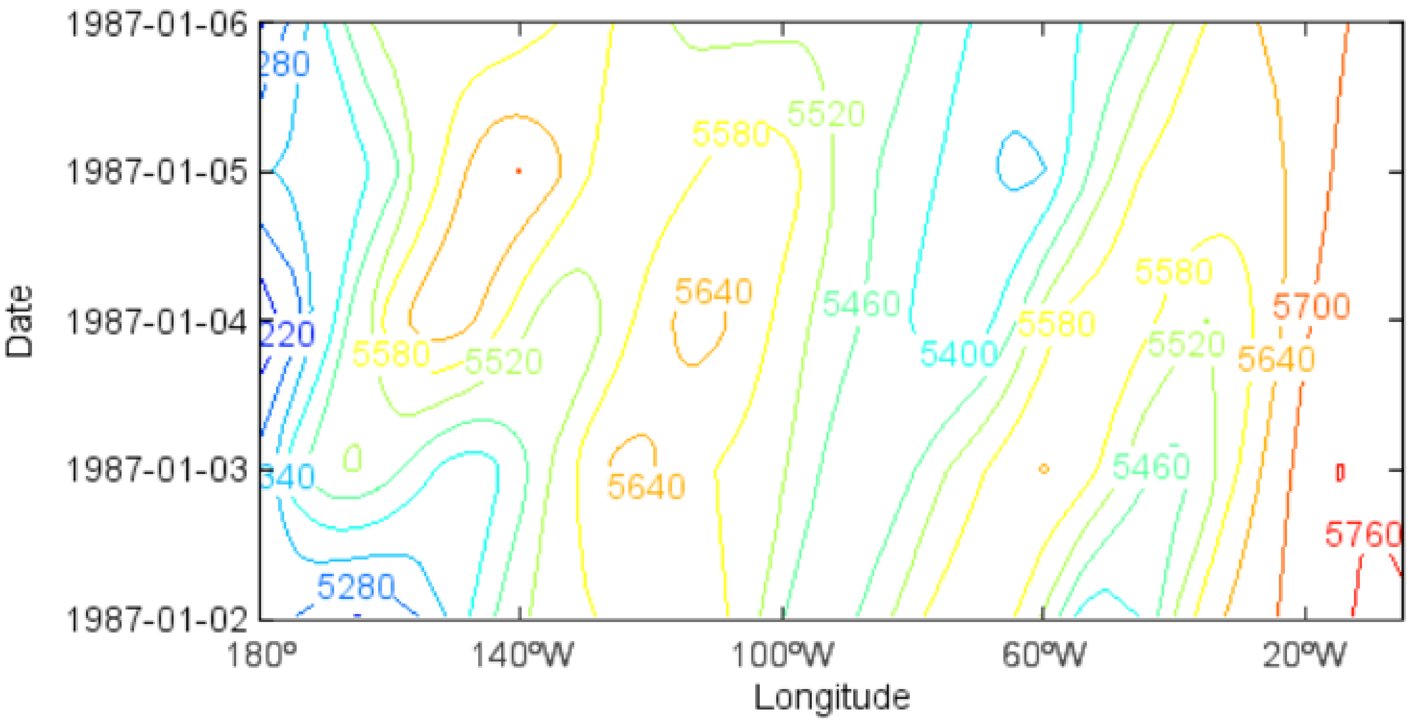


MeteoInfoLab – Plot data

Another example, in this case with X and T varying (Hovmoller plot):

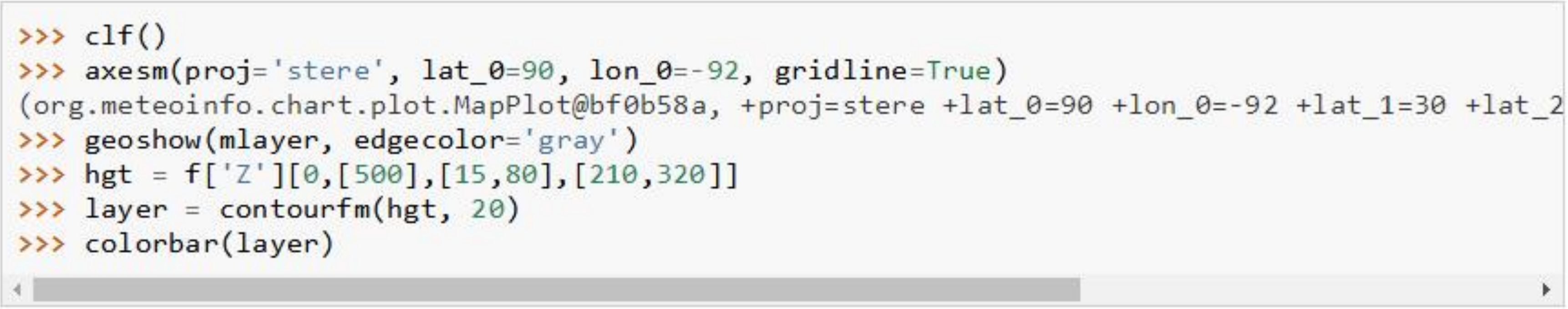


```
>>> clf()
>>> hgt = f['Z'][0:4,[500],[40],[180,360]]
>>> layer = contour(hgt, 10)
>>> clabel(layer)
>>> yaxis(axistype='time', timetickformat='yyy-MM-dd')
>>> yticks(hgt.dimvalue(0))
>>> xlabel('Longitude')
>>> ylabel('Date')
```

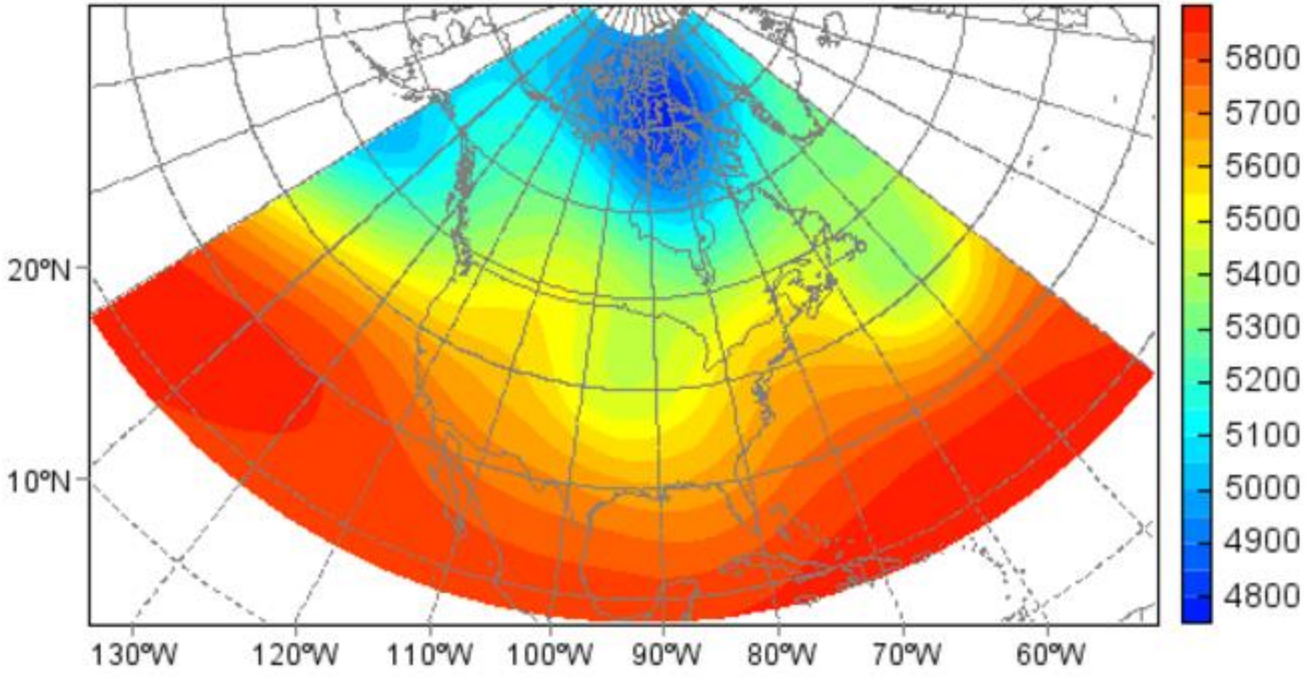


MeteoInfoLab – Plot data

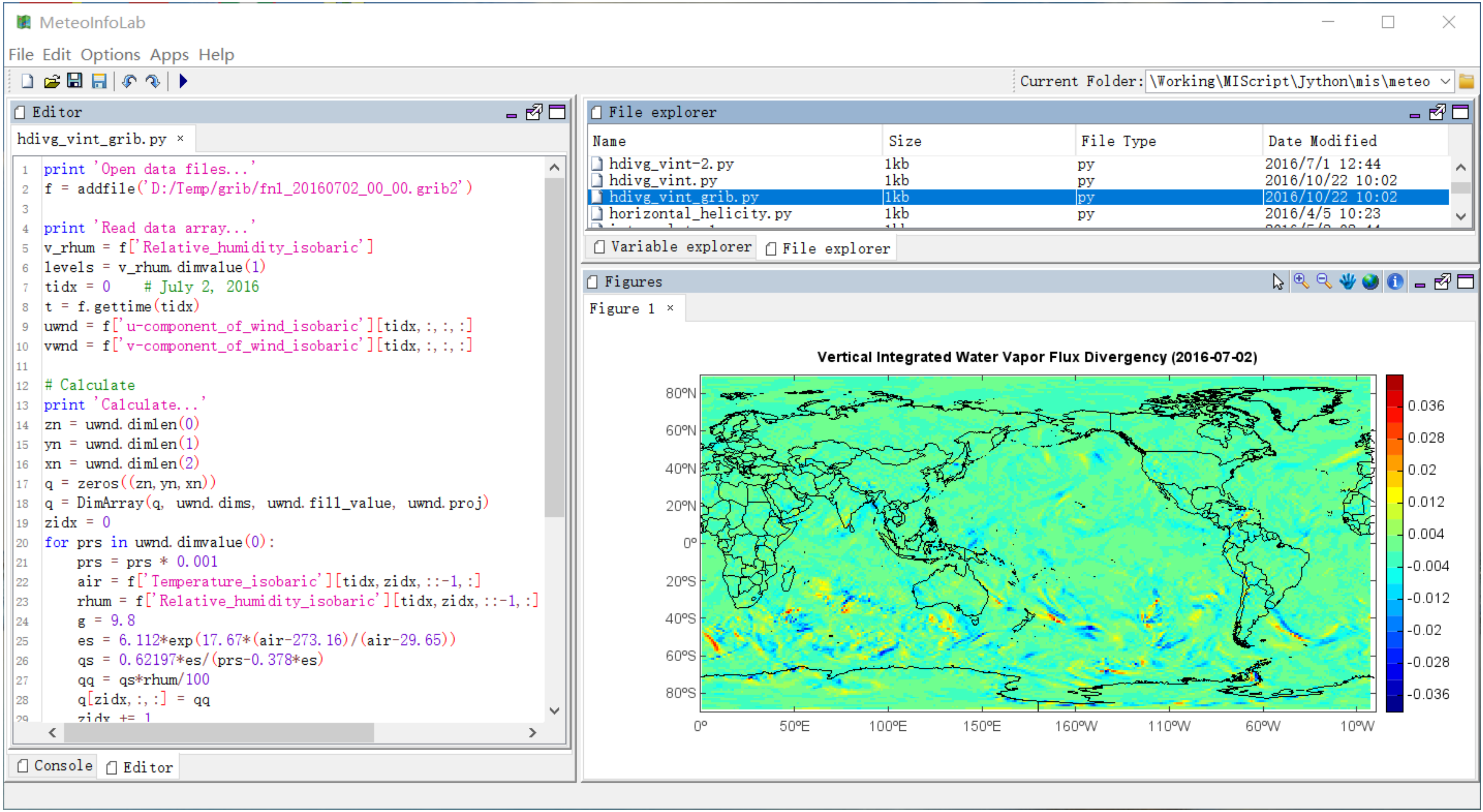
To alter the projection:



```
>>> clf()
>>> axesm(proj='stere', lat_0=90, lon_0=-92, gridline=True)
(org.meteoinfo.chart.plot.MapPlot@bf0b58a, +proj=stere +lat_0=90 +lon_0=-92 +lat_1=30 +lat_2
>>> geoshow(mlayer, edgecolor='gray')
>>> hgt = f['Z'][0,[500],[15,80],[210,320]]
>>> layer = contourfm(hgt, 20)
>>> colorbar(layer)
```



MeteoInfoLab – Meteorological analysis

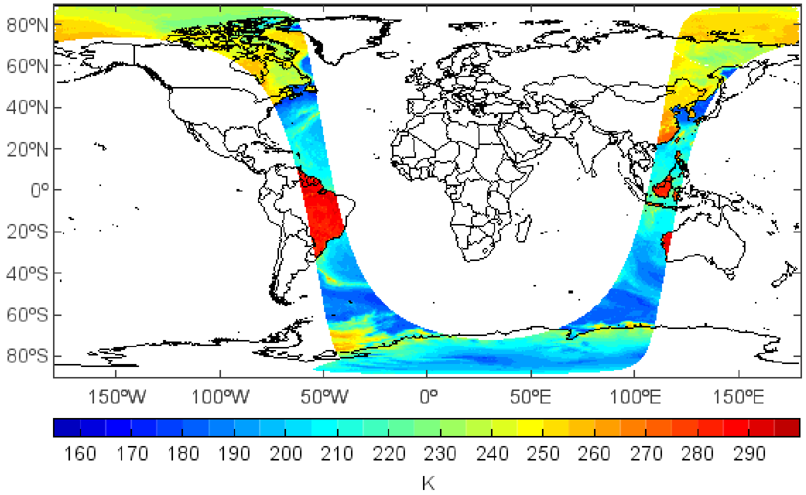


```
print 'Open data files...'
f = addfile('D:/Temp/grib/fnl_20160702_00_00.grib2')

print 'Read data array...'
v_rhum = f['Relative_humidity_isobaric']
levels = v_rhum.dimvalue(1)
tidx = 0    # July 2, 2016
t = f.gettime(tidx)
uwnd = f['u-component_of_wind_isobaric'][tidx,:,:,:]
vwnd = f['v-component_of_wind_isobaric'][tidx,:,:,:]

# Calculate
print 'Calculate...'
zn = uwnd.dimlen(0)
yn = uwnd.dimlen(1)
xn = uwnd.dimlen(2)
q = zeros((zn, yn, xn))
q = DimArray(q, uwnd.dims, uwnd.fill_value, uwnd.proj)
zidx = 0
for prs in uwnd.dimvalue(0):
    prs = prs * 0.001
    air = f['Temperature_isobaric'][tidx,zidx,::-1,:]
    rhum = f['Relative_humidity_isobaric'][tidx,zidx,::-1,:]
    g = 9.8
    es = 6.112*exp(17.67*(air-273.16)/(air-29.65))
    qs = 0.62197*es/(prs-0.378*es)
    qq = qs*rhum/100
    q[zidx,:,:] = qq
    zidx += 1
```

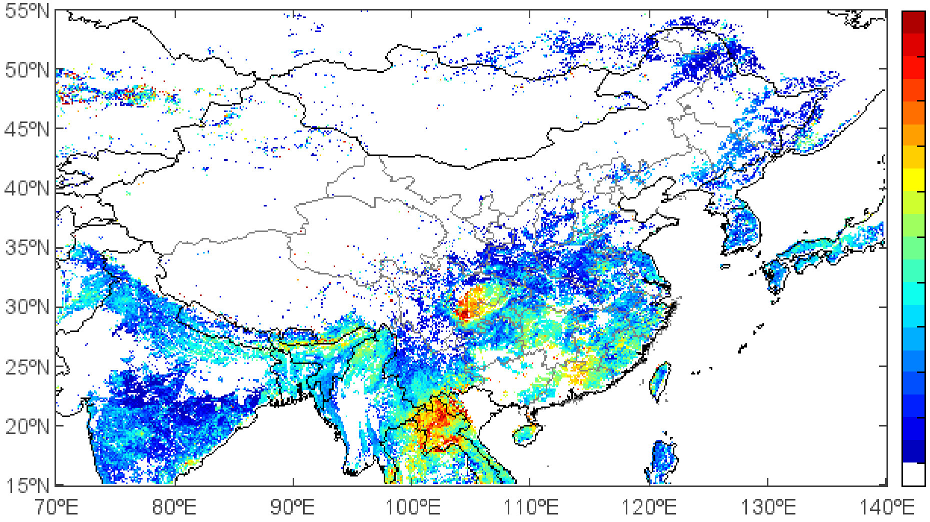
Name	Size	File Type	Date Modified
hdivg_vint-2.py	1kb	py	2016/7/1 12:44
hdivg_vint.py	1kb	py	2016/10/22 10:02
hdivg_vint_grib.py	1kb	py	2016/10/22 10:02
horizontal_helicity.py	1kb	py	2016/4/5 10:23

Vertical Integrated Water Vapor Flux Divergency (2016-07-02)

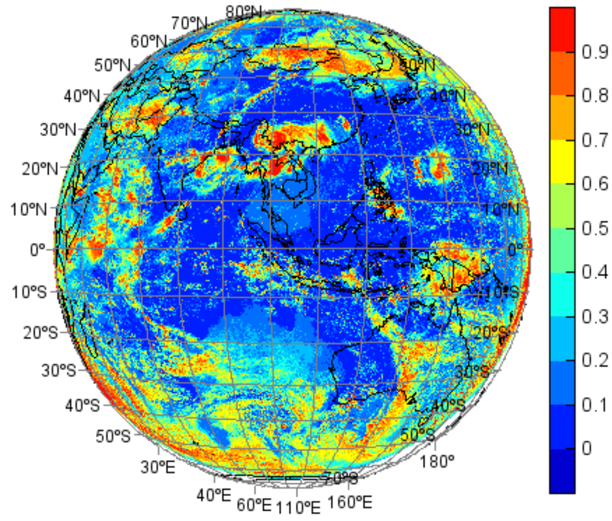
MeteoInfoLab – Satellite data examples



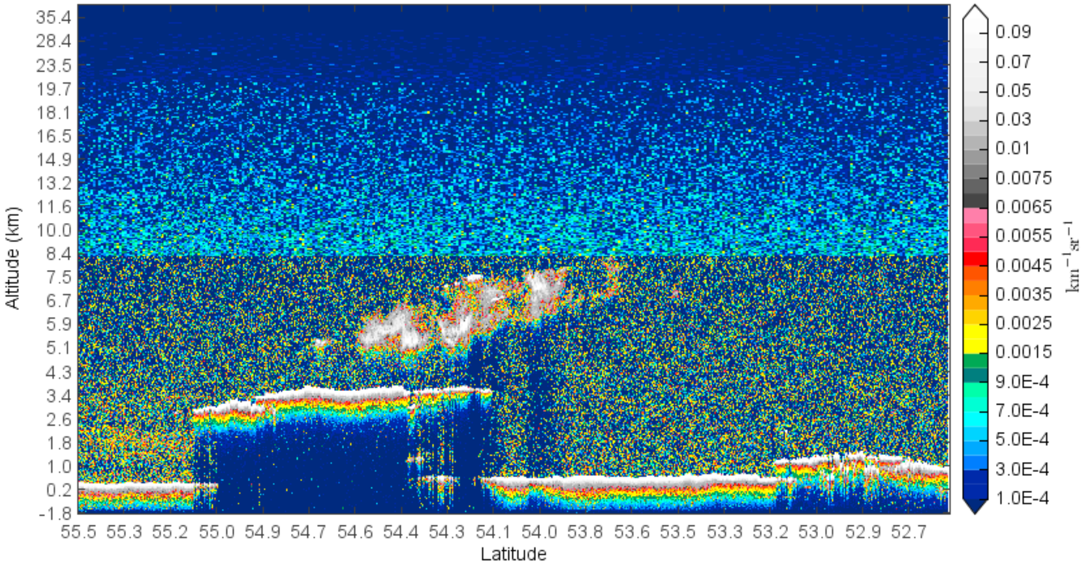
1C.F19.SSMIS.XCAL2015-P.20160105-S214106-E232259.009078.V03A.HDF5
Intercalibrated Tb for channels 1) 19.35 GHz V-Pol 2) 19.35 GHz H-Pol and 3) 22.235 GHz V-Pol
(nchannel1=0)



FY-3A AOD (2014-03-31)

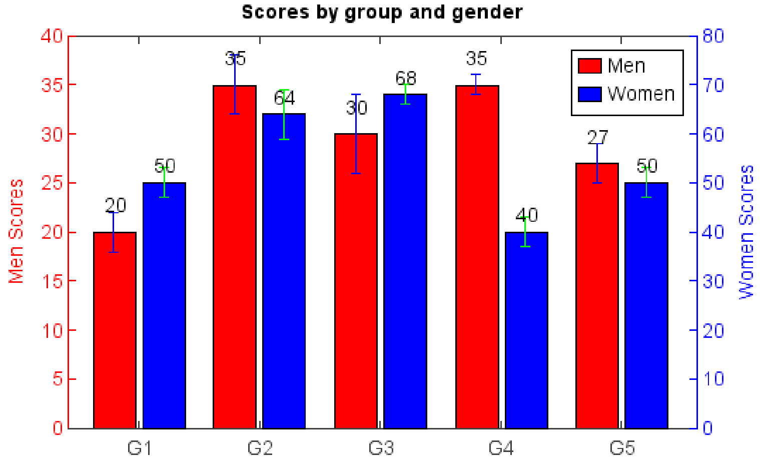


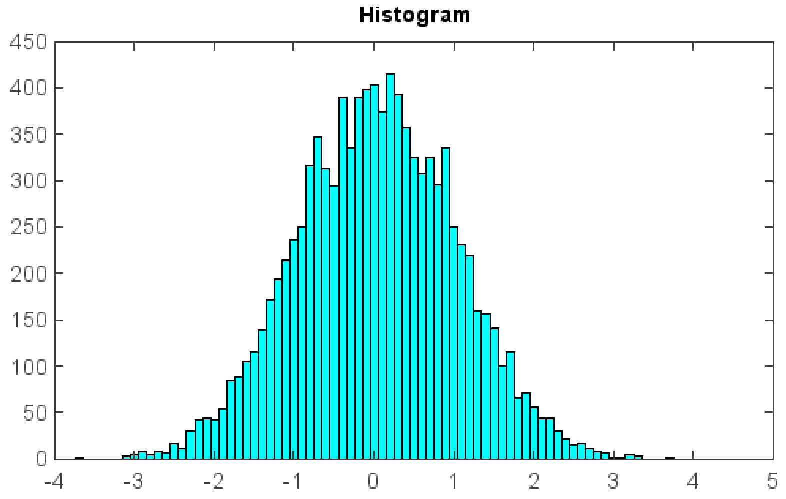
Himawari-8

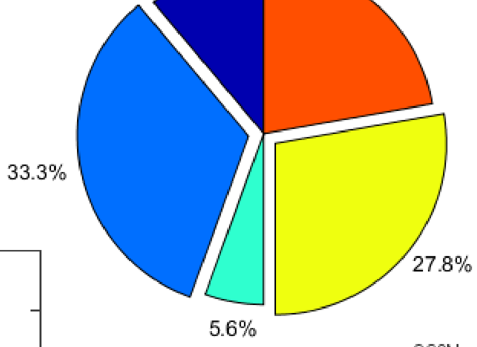


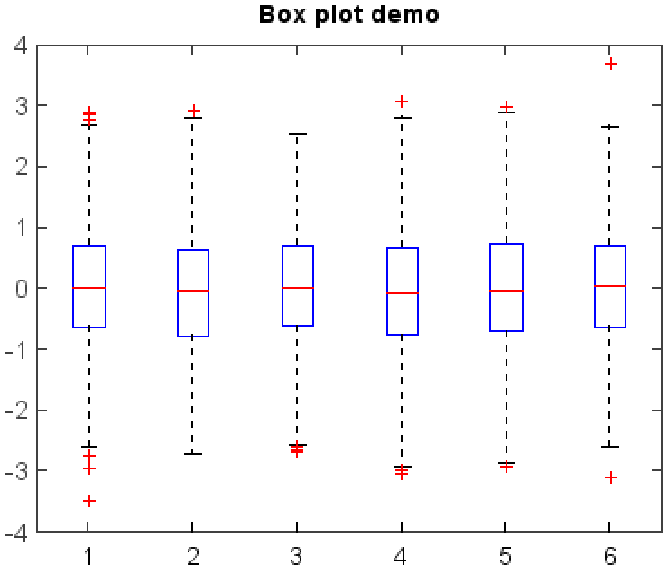
CAL_LID_L1-ValStage1-V3-01.2007-06-12T03-42-18ZN.hdf
Total_Attenuated_Backscatter_532

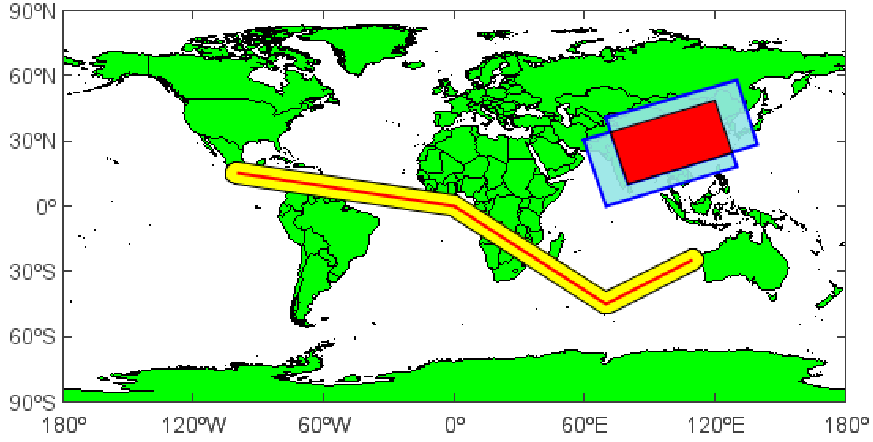
MeteoInfoLab – More examples

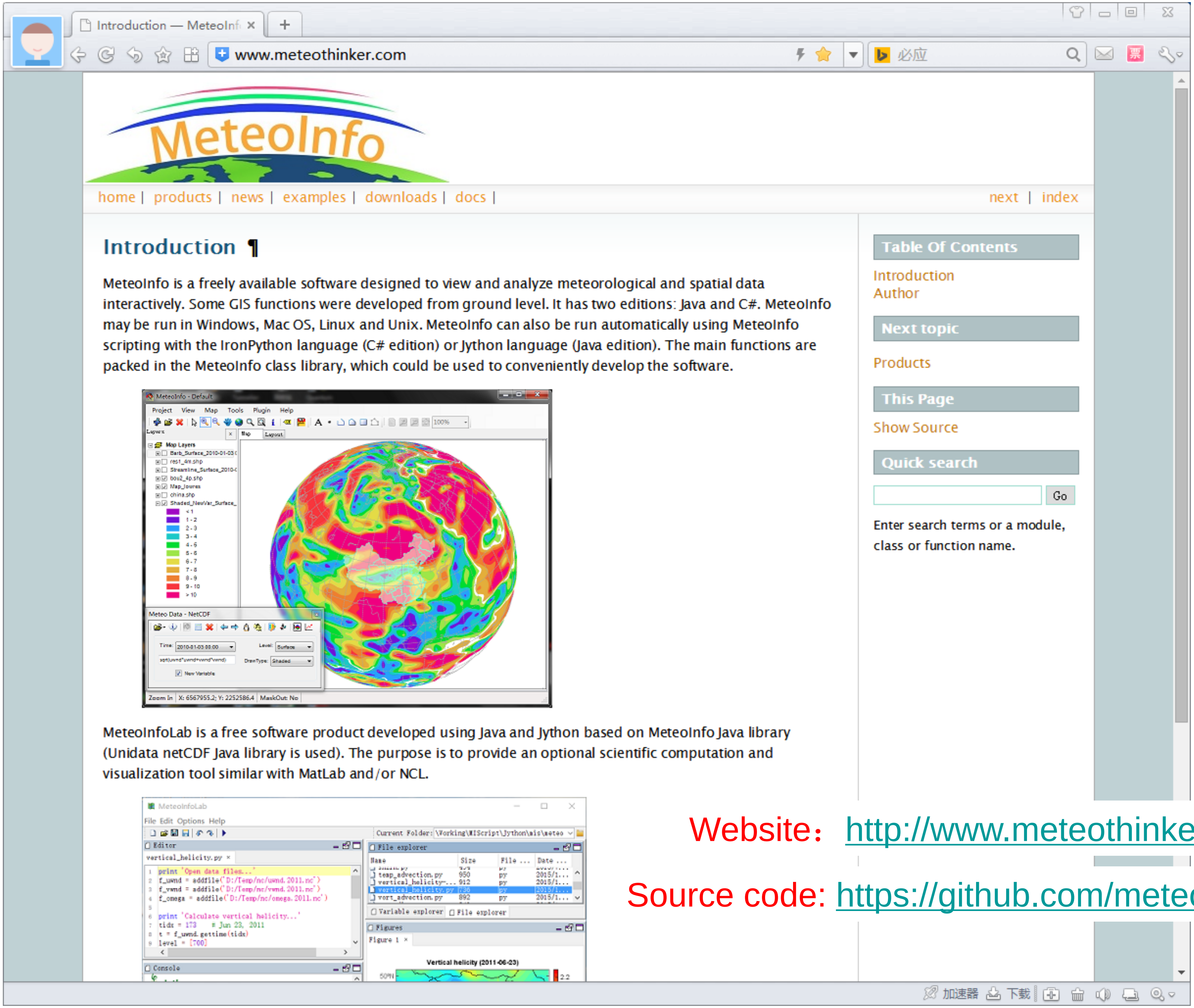












home | products | news | examples | downloads | docs | next | index

Introduction ¶

MeteoInfo is a freely available software designed to view and analyze meteorological and spatial data interactively. Some GIS functions were developed from ground level. It has two editions: Java and C#. MeteoInfo may be run in Windows, Mac OS, Linux and Unix. MeteoInfo can also be run automatically using MeteoInfo scripting with the IronPython language (C# edition) or Jython language (Java edition). The main functions are packed in the MeteoInfo class library, which could be used to conveniently develop the software.

MeteoInfoLab is a free software product developed using Java and Jython based on MeteoInfo Java library (Unidata netCDF Java library is used). The purpose is to provide an optional scientific computation and visualization tool similar with MatLab and/or NCL.

```
print 'Open data files...'
f_uwnd = addfile('D:/Temp/nc/uwnd.2011.nc')
f_vwnd = addfile('D:/Temp/nc/vwnd.2011.nc')
f_omega = addfile('D:/Temp/nc/omega.2011.nc')

print 'Calculate vertical helicity...'
tidx = 173     # Jun 23, 2011
t = f_uwnd.gettime(tidx)
level = [700]
```

Name	Size	File ...	Date ...
temp_advection.py	950	py	2015/1...
vertical_helicity-...	912	py	2015/1...
vertical_helicity.py	736	py	2015/1...
vort_advection.py	892	py	2015/1...

Table Of Contents

Next topic

Products

This Page

Show Source

Quick search

Go

Enter search terms or a module, class or function name.

Website：http://www.meteothinker.com

Source code: https://github.com/meteoinfo

Acknowledgement

- NetCDF Java: Available at www.unidata.ucar.edu/software/netcdf-java
- Jython: Available at http://www.jython.org/
- Common Math: Available at http://commons.apache.org/proper/commons-math/
- Proj4J: Available at http://trac.osgeo.org/proj4j/wiki
- Groovy: Available at http://groovy.codehaus.org/
- wContour: Available at http://www.meteothinker.com/
- L2FProd: Available at http://common.L2FProd.com
- RSyntaxTextArea: Available at http://fifesoft.com/rsyntaxtextarea/
- JLaTeXMath: Available at http://forge.scilab.org/index.php/p/jlatexmath/
- JavaHelp: Available at https://javahelp.java.net/
- BeanShell: Available at http://www.beanshell.org/
- FreeHEP VectorGraphics: Available at http://java.freehep.org/vectorgraphics/
- Docking Frames: Available at http://dock.javaforge.com/
- And more …

Thanks for your attention !

Welcome to use and develop MeteoInfo !