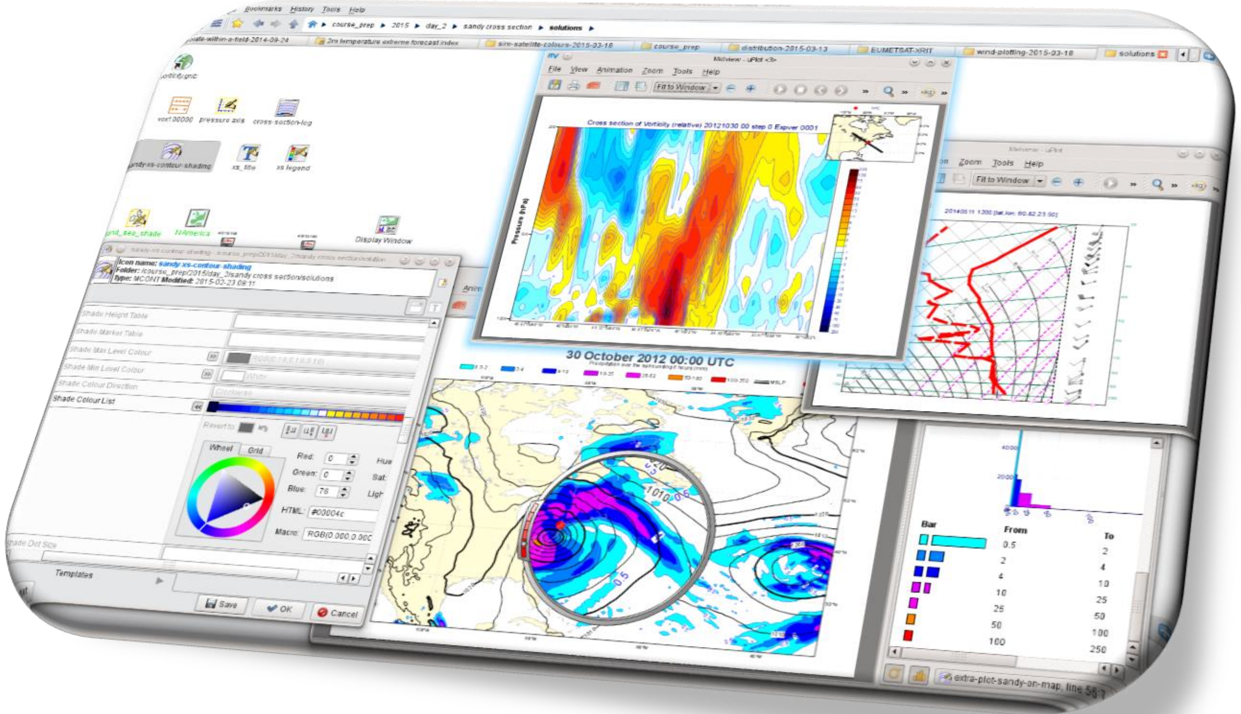
Metview’s new Python interface – first results and roadmap for further developments

python™

EGOWS 2018, ECMWF

Iain Russell

Development Section, ECMWF



Thanks to

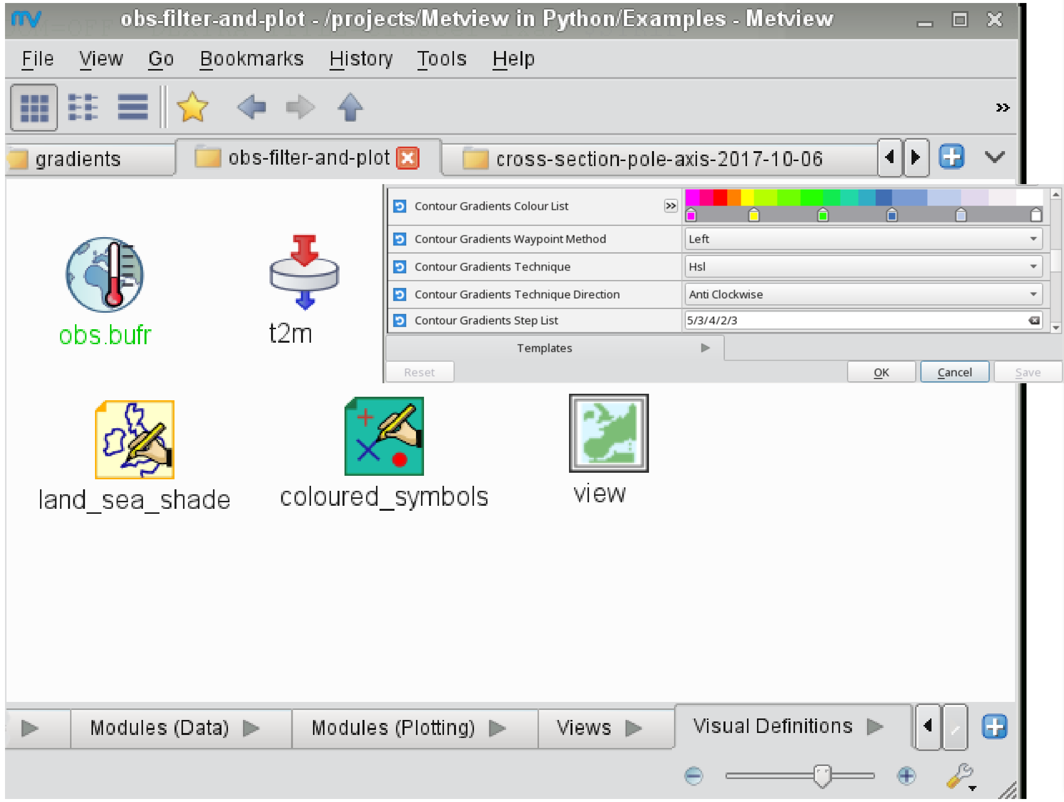
Sándor Kertész

Fernando Ii

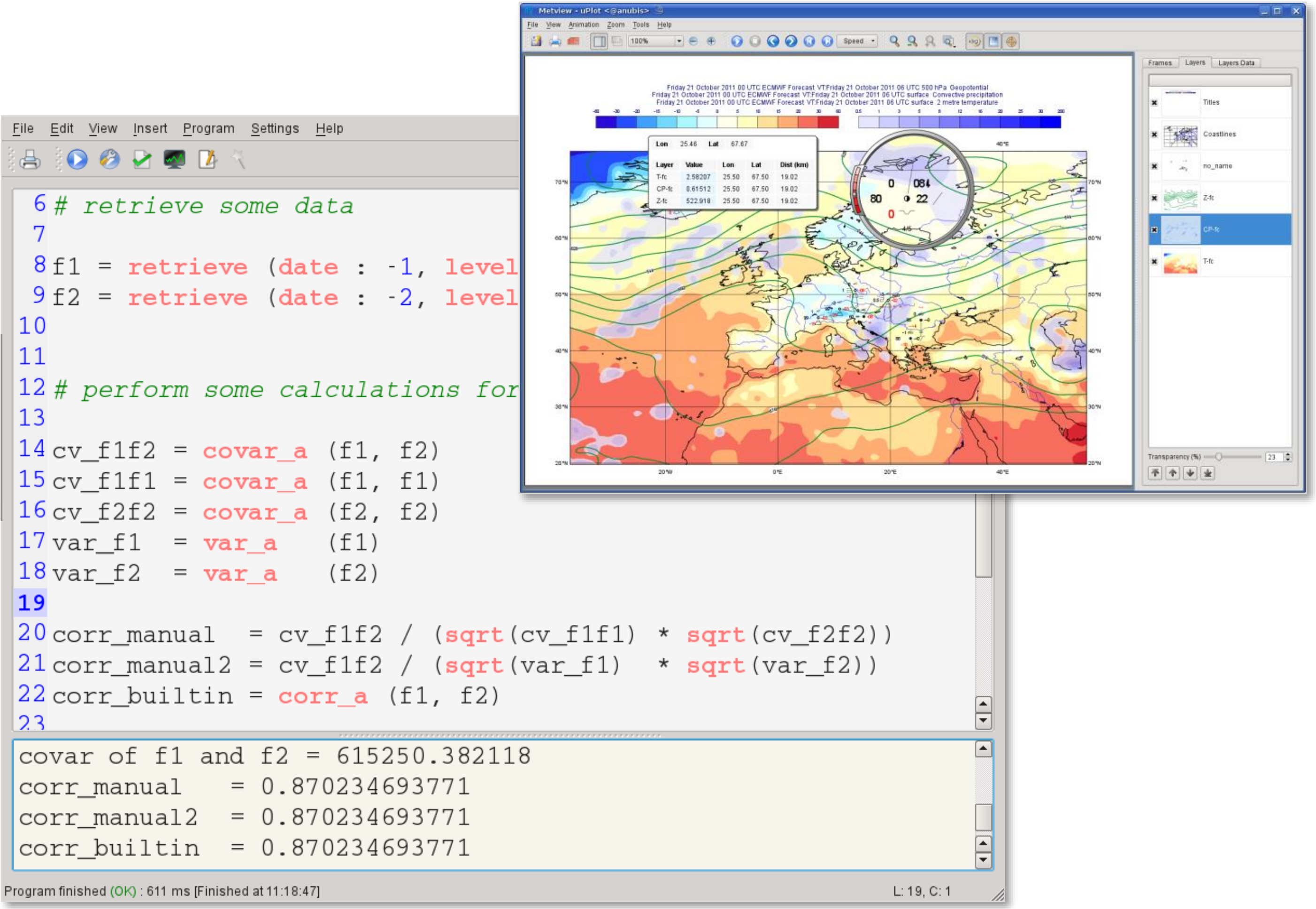
Stephan Siemen

ECMWF

What is Metview?



Contour Gradients Colour List
Contour Gradients Waypoint Method: Left
Contour Gradients Technique: Hsl
Contour Gradients Technique Direction: Anti Clockwise
Contour Gradients Step List: 5/3/4/2/3


File Edit View Insert Program Settings Help
Metview - uPlot <@anubis>
File View Animation Zoom Tools Help
100%
Speed
Frames Layers Layers Data
Titles
Coastlines
no_name
Z-fc
CP-fc
T-fc
Transparency (%)
Friday 21 October 2011 00 UTC ECMWF Forecast VT:Friday 21 October 2011 06 UTC 500 hPa Geopotential
Friday 21 October 2011 00 UTC ECMWF Forecast VT:Friday 21 October 2011 06 UTC surface Convective precipitation
Friday 21 October 2011 00 UTC ECMWF Forecast VT:Friday 21 October 2011 06 UTC surface 2 metre temperature
Lon 25.46 Lat 67.67
70°N
60°N
50°N
40°N
30°N
20°N
20°W
0°E
20°E
40°E

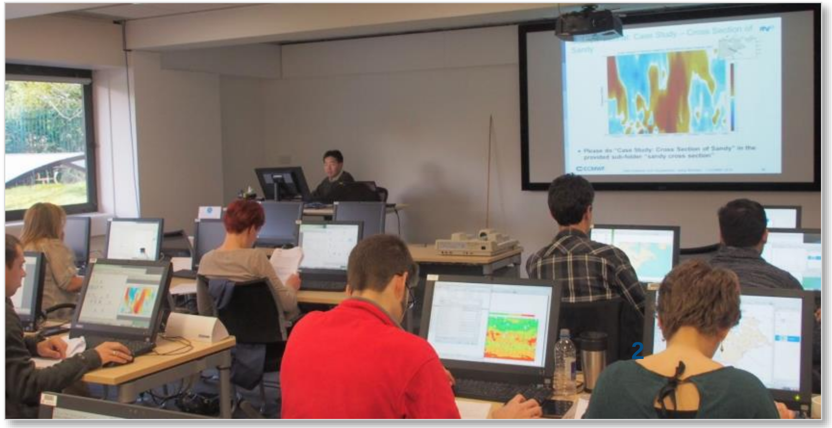
Layer	Value	Lon	Lat	Dist (km)
T-fc	2.58207	25.50	67.50	19.02
CP-fc	0.61512	25.50	67.50	19.02
Z-fc	522.918	25.50	67.50	19.02

```
# retrieve some data

f1 = retrieve (date : -1, level
f2 = retrieve (date : -2, level


# perform some calculations for

cv_f1f2 = covar_a (f1, f2)
cv_f1f1 = covar_a (f1, f1)
cv_f2f2 = covar_a (f2, f2)
var_f1  = var_a   (f1)
var_f2  = var_a   (f2)

corr_manual  = cv_f1f2 / (sqrt(cv_f1f1) * sqrt(cv_f2f2))
corr_manual2 = cv_f1f2 / (sqrt(var_f1)  * sqrt(var_f2))
corr_builtin = corr_a (f1, f2)
```

```
covar of f1 and f2 = 615250.382118
corr_manual   = 0.870234693771
corr_manual2  = 0.870234693771
corr_builtin  = 0.870234693771
```

Program finished (OK) : 611 ms [Finished at 11:18:47]   L: 19, C: 1

- UNIX, Open Source under Apache Licence 2.0
- Metview is a co-operation project with INPE (Brazil)

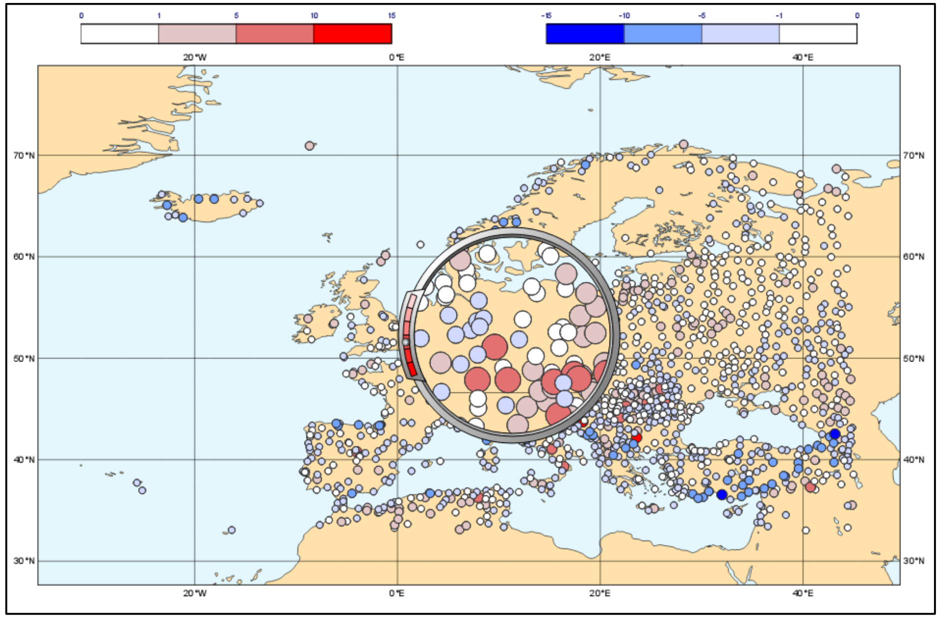


High-level data processing with Metview Macro

- We already have the Macro language…

Forecast – observation difference:

```
forecast = retrieve (…) # GRIB from MARS
obs      = retrieve (…) # BUFR from MARS
t2m_obs  = obsfilter(data:      obs,
                     parameter: 012004,
                     output:    'geopoints’)
plot(forecast – t2m_obs)
```



Why create a Python interface to Metview?

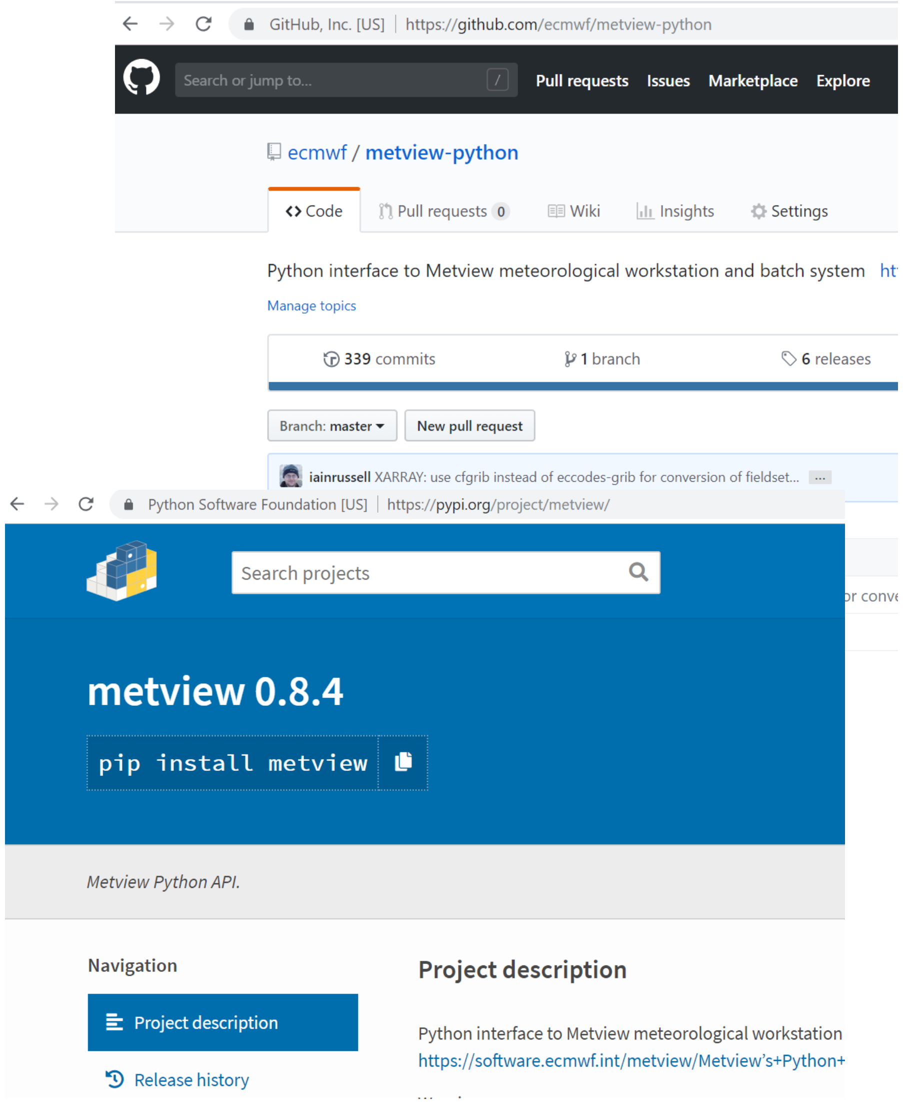
- Metview’s Macro language is great; Python has even more language features
- Not everyone knows Metview Macro!
  - Less learning curve for people who already know Python
  - Better for community-building
  - Learning Python is a good investment in general!
- Enable Metview to work seamlessly within the Python eco-system
  - Bring Metview’s data processing and interactive data inspection tools into Python sessions ; interact with Python data structures
  - Use existing solutions where possible (e.g. for multi-dimensional data arrays, data models)
- Enable Metview to be a component of the Copernicus Climate Data Store Toolbox

Current Status

- Alpha release (developed with B-Open)
  - Beta release just around the corner

- Available on github and PyPi
  - https://github.com/ecmwf/metview-python
  - `pip install metview`

- Python layer only – this still requires the Metview binaries to be installed too



GitHub, Inc. [US] | https://github.com/ecmwf/metview-python

Python interface to Metview meteorological workstation and batch system ht

iainrussell XARRAY: use cfgrib instead of eccodes-grib for conversion of fieldset...

Python Software Foundation [US] | https://pypi.org/project/metview/

metview 0.8.4

```
pip install metview
```

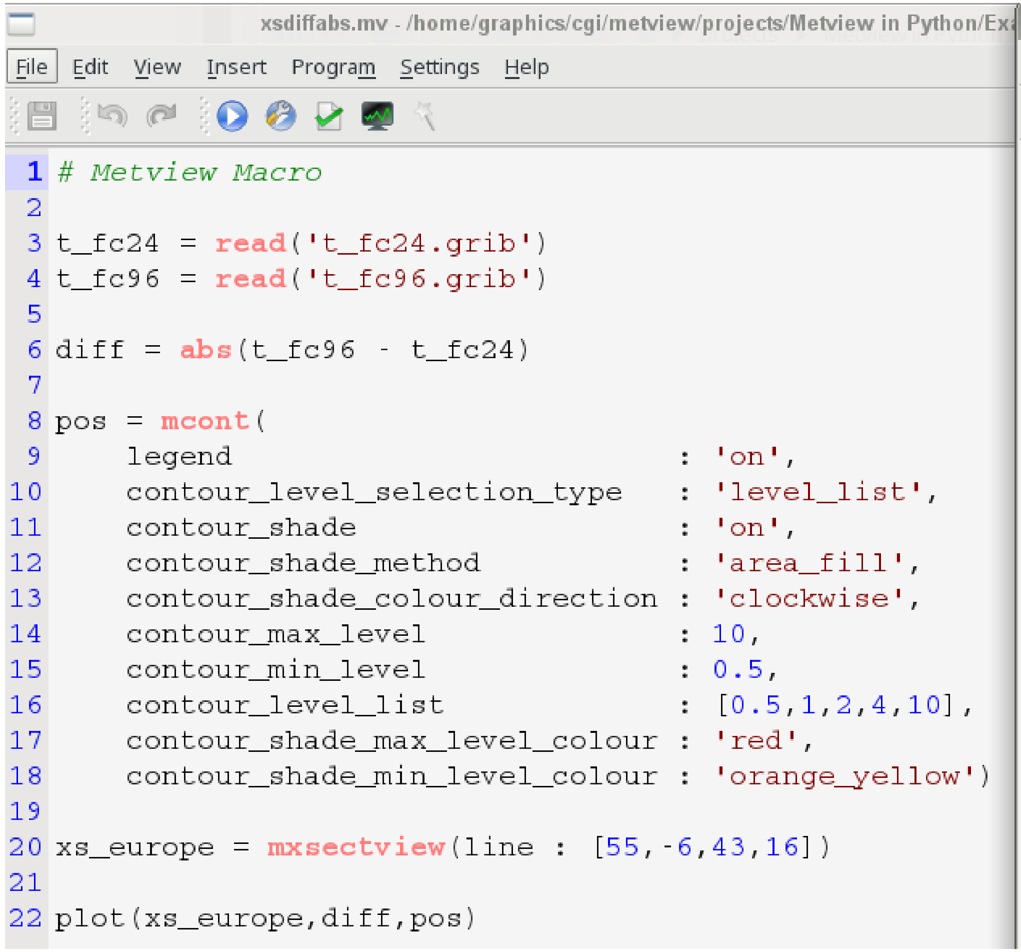
Metview Python API.

Project description

Python interface to Metview meteorological workstation
https://software.ecmwf.int/metview/Metview's+Python+

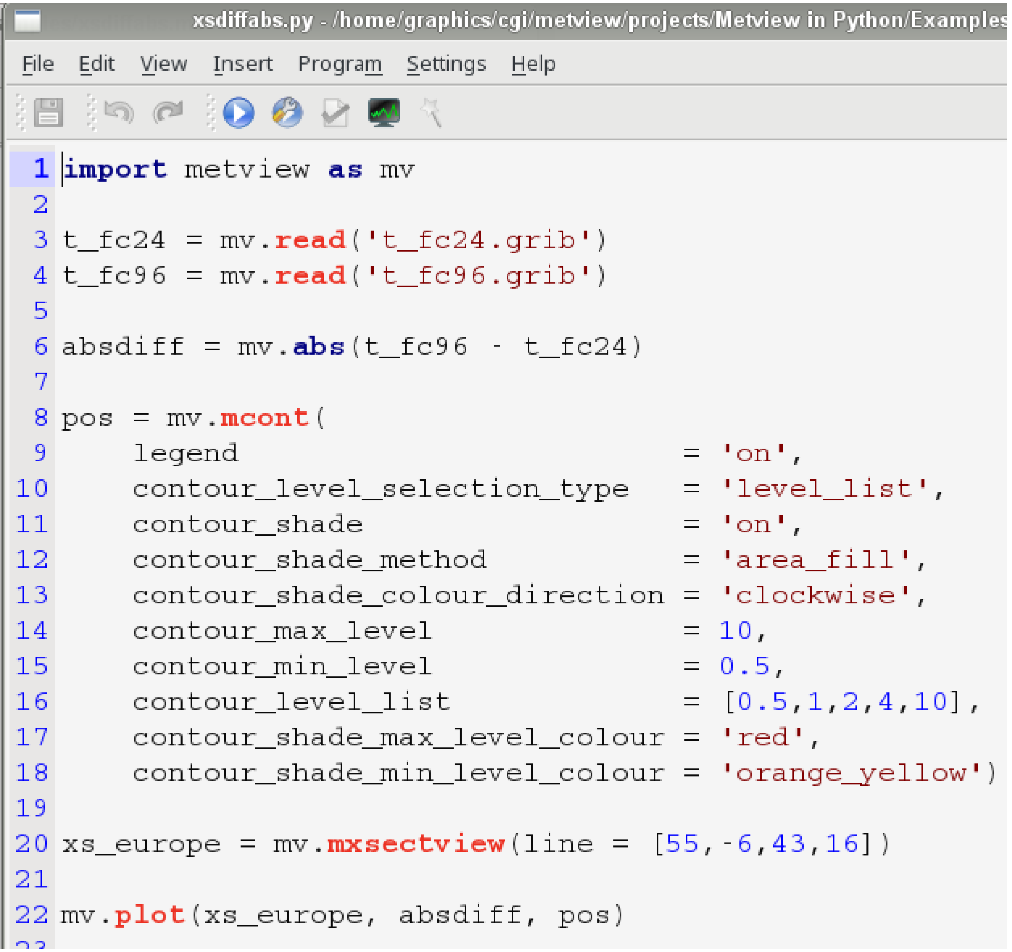
Macro / Python comparison

Macro


xsdiffabs.mv - /home/graphics/cgi/metview/projects/Metview in Python/Ex
File
Edit
View
Insert
Program
Settings
Help

```
# Metview Macro

t_fc24 = read('t_fc24.grib')
t_fc96 = read('t_fc96.grib')

diff = abs(t_fc96 - t_fc24)

pos = mcont(
    legend                         : 'on',
    contour_level_selection_type   : 'level_list',
    contour_shade                  : 'on',
    contour_shade_method           : 'area_fill',
    contour_shade_colour_direction : 'clockwise',
    contour_max_level              : 10,
    contour_min_level              : 0.5,
    contour_level_list             : [0.5,1,2,4,10],
    contour_shade_max_level_colour : 'red',
    contour_shade_min_level_colour : 'orange_yellow')

xs_europe = mxsectview(line : [55,-6,43,16])

plot(xs_europe,diff,pos)
```

Python

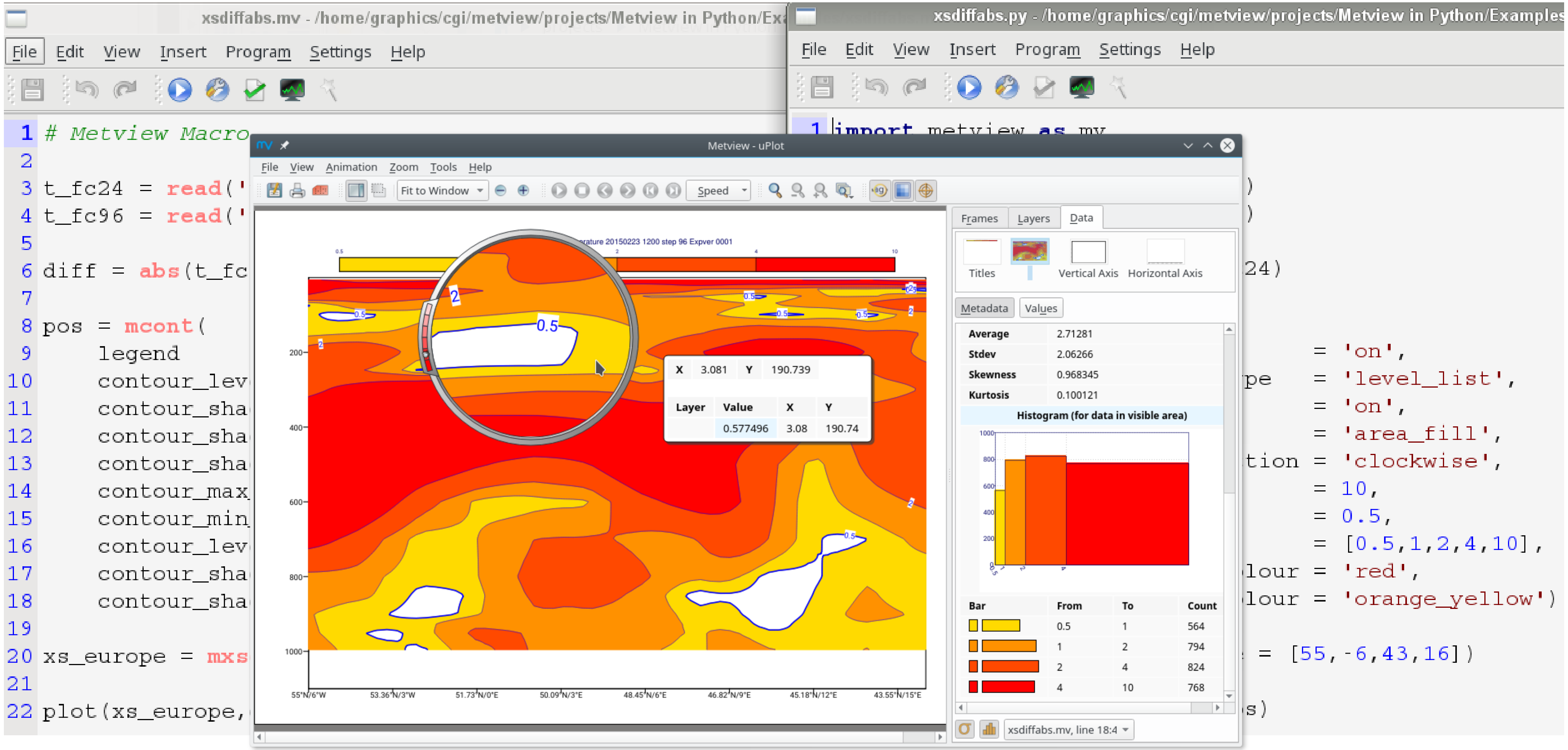

xsdiffabs.py - /home/graphics/cgi/metview/projects/Metview in Python/Examples
File
Edit
View
Insert
Program
Settings
Help

```
import metview as mv

t_fc24 = mv.read('t_fc24.grib')
t_fc96 = mv.read('t_fc96.grib')

absdiff = mv.abs(t_fc96 - t_fc24)

pos = mv.mcont(
    legend                         = 'on',
    contour_level_selection_type   = 'level_list',
    contour_shade                  = 'on',
    contour_shade_method           = 'area_fill',
    contour_shade_colour_direction = 'clockwise',
    contour_max_level              = 10,
    contour_min_level              = 0.5,
    contour_level_list             = [0.5,1,2,4,10],
    contour_shade_max_level_colour = 'red',
    contour_shade_min_level_colour = 'orange_yellow')

xs_europe = mv.mxsectview(line = [55,-6,43,16])

mv.plot(xs_europe, absdiff, pos)
```

Macro / Python comparison

Macro

Python



xsdiffabs.mv - /home/graphics/cgi/metview/projects/Metview in Python/Ex

```
# Metview Macro

t_fc24 = read('
t_fc96 = read('

diff = abs(t_fc

pos = mcont(
      legend
      contour_lev
      contour_sha
      contour_sha
      contour_sha
      contour_max
      contour_min
      contour_lev
      contour_sha
      contour_sha

xs_europe = mxs

plot(xs_europe,
```

xsdiffabs.py - /home/graphics/cgi/metview/projects/Metview in Python/Examples

```
import metview as mv
)
)

24)


             = 'on',
pe           = 'level_list',
             = 'on',
             = 'area_fill',
tion         = 'clockwise',
             = 10,
             = 0.5,
             = [0.5,1,2,4,10],
lour         = 'red',
lour         = 'orange_yellow')

 = [55,-6,43,16])

s)
```

Metview - uPlot

X	3.081	Y	190.739

Layer	Value	X	Y
	0.577496	3.08	190.74

Average	2.71281
Stdev	2.06266
Skewness	0.968345
Kurtosis	0.100121

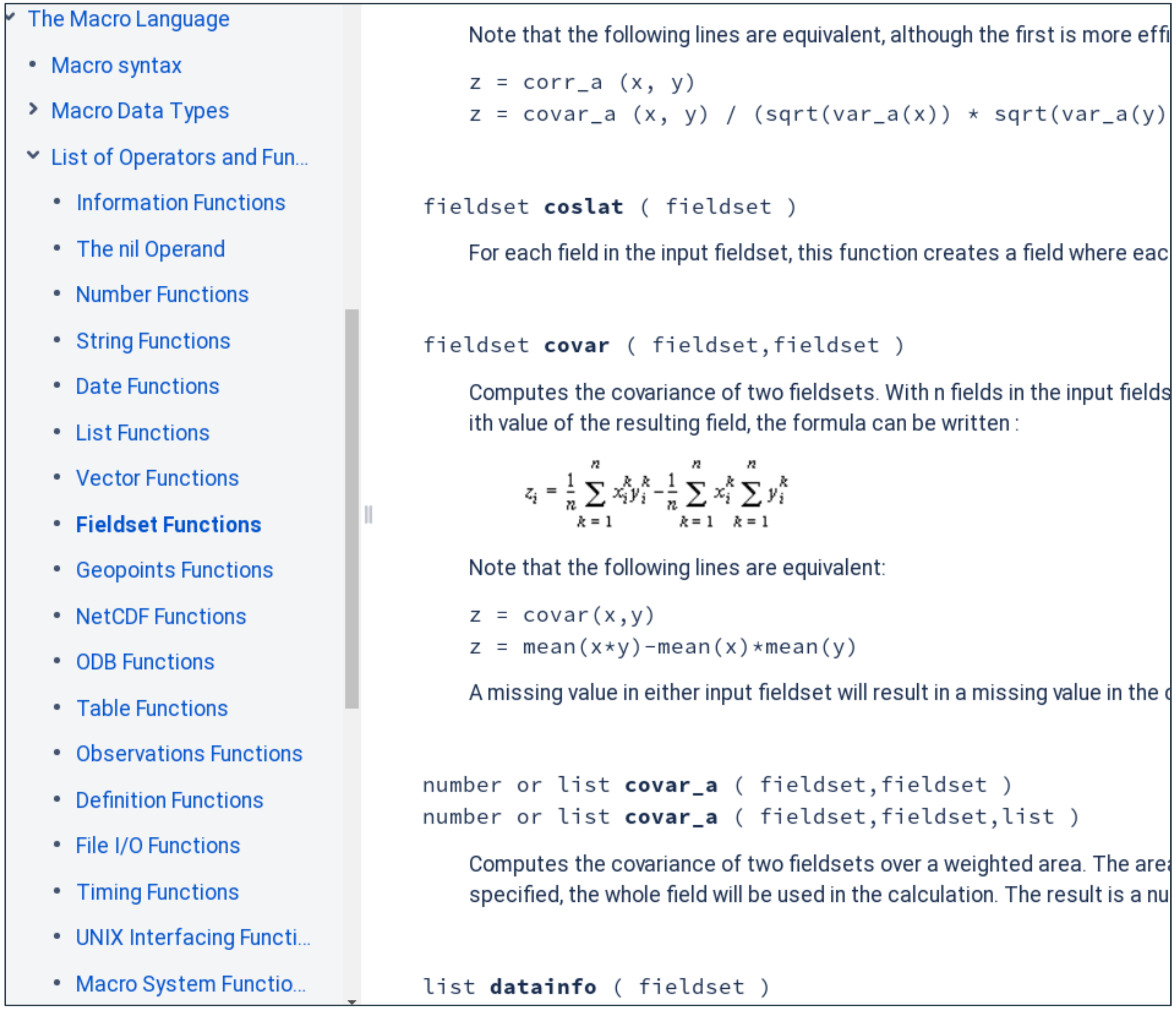
Histogram (for data in visible area)

Bar	From	To	Count
🟨	0.5	1	564
🟧	1	2	794
🟧	2	4	824
🟥	4	10	768

Data types (1)

- All Metview Macro functions can be called from Python, e.g. mv.covar(f1, f2)
- Data types returned are either standard Python types (numbers, lists, strings, datetimes), numpy arrays
- Or… thin class wrappers around more complex objects such as fieldsets, geopoints or ODB
  - Allows **`a+b`** on Fieldsets, etc
  - Allows thin object-oriented layer, e.g.

  -
    ```
    data = mv.read('file.grib')
    av = mv.integrate(data)# equivalent
    av = data.integrate()  # equivalent
    ```



Note that the following lines are equivalent, although the first is more effi

```
z = corr_a (x, y)
z = covar_a (x, y) / (sqrt(var_a(x)) * sqrt(var_a(y)
```

`fieldset **coslat** ( fieldset )`

For each field in the input fieldset, this function creates a field where eac

`fieldset **covar** ( fieldset,fieldset )`

Computes the covariance of two fieldsets. With n fields in the input fields ith value of the resulting field, the formula can be written :

$$z_i = \frac{1}{n}\sum_{k=1}^{n} x_i^k y_i^k - \frac{1}{n}\sum_{k=1}^{n} x_i^k \sum_{k=1}^{n} y_i^k$$

Note that the following lines are equivalent:

```
z = covar(x,y)
z = mean(x*y)-mean(x)*mean(y)
```

A missing value in either input fieldset will result in a missing value in the c

`number or list **covar_a** ( fieldset,fieldset )`
`number or list **covar_a** ( fieldset,fieldset,list )`

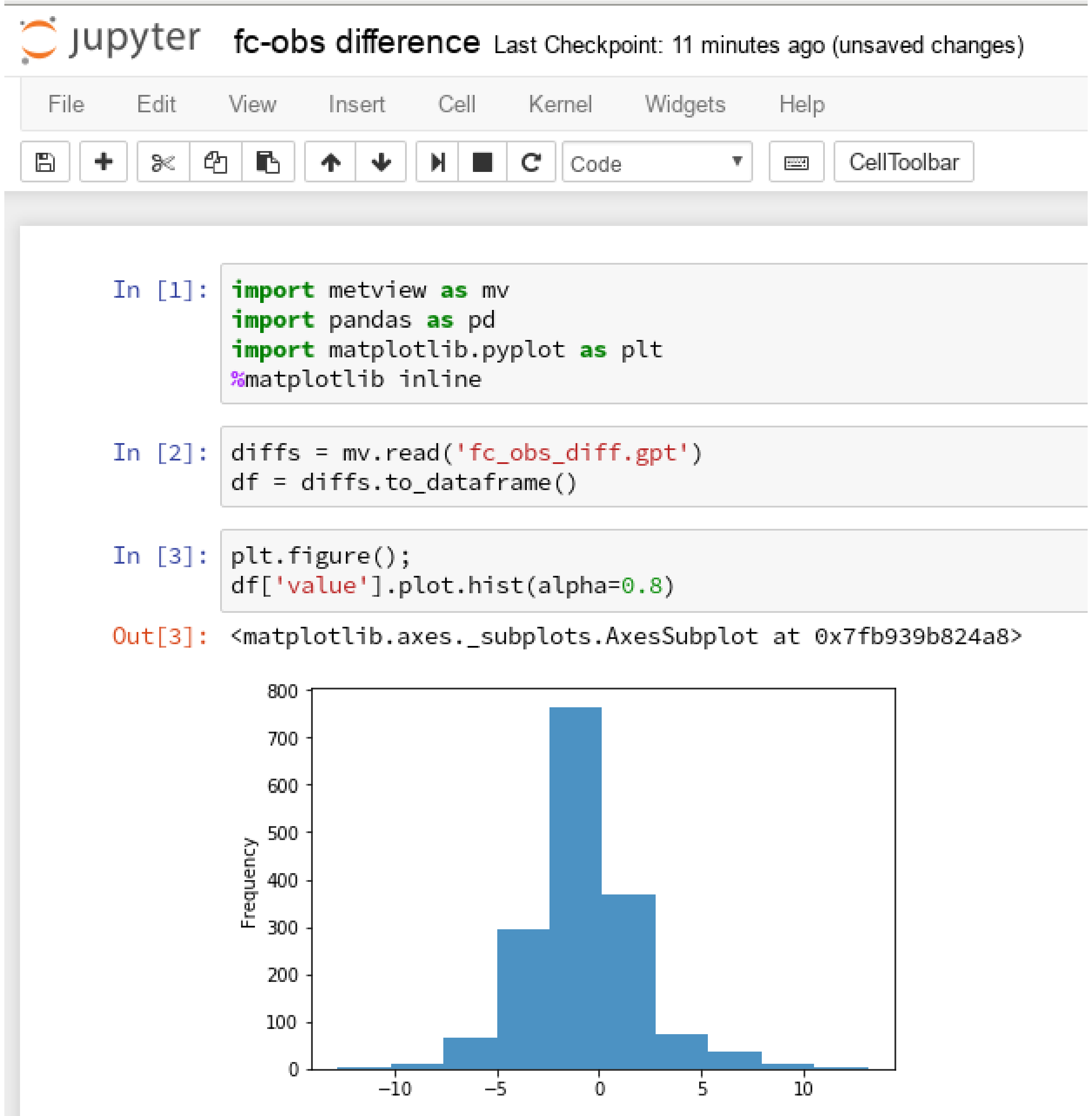
Computes the covariance of two fieldsets over a weighted area. The area specified, the whole field will be used in the calculation. The result is a nu

`list **datainfo** ( fieldset )`

Data types (2)

- Can also export Metview Geopoints (and BUFR via the filter), ODB and Table data types to **pandas** Dataframes (table-like format, common in scientific data processing)

- Example: load a geopoints file with forecast-obs differences (generated earlier), convert to Pandas Dataframe and plot a histogram of the values



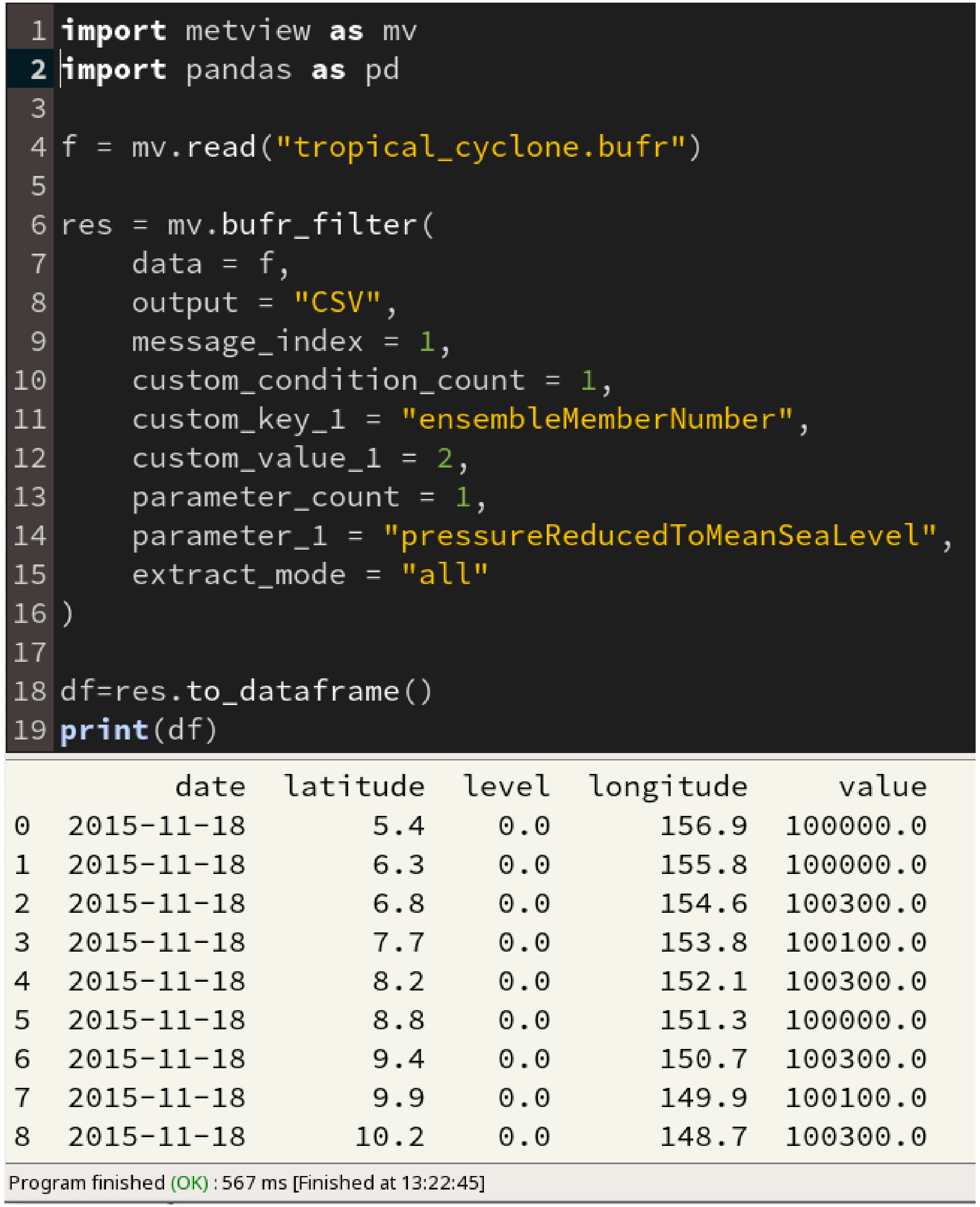
```
In [1]: import metview as mv
        import pandas as pd
        import matplotlib.pyplot as plt
        %matplotlib inline
```

```
In [2]: diffs = mv.read('fc_obs_diff.gpt')
        df = diffs.to_dataframe()
```

```
In [3]: plt.figure();
        df['value'].plot.hist(alpha=0.8)

Out[3]: <matplotlib.axes._subplots.AxesSubplot at 0x7fb939b824a8>
```

Data types (3)

- Example: tropical cyclone track encoded in BUFR
  - Run a filter to produce a Table (CSV)
  - Convert to pandas dataframe



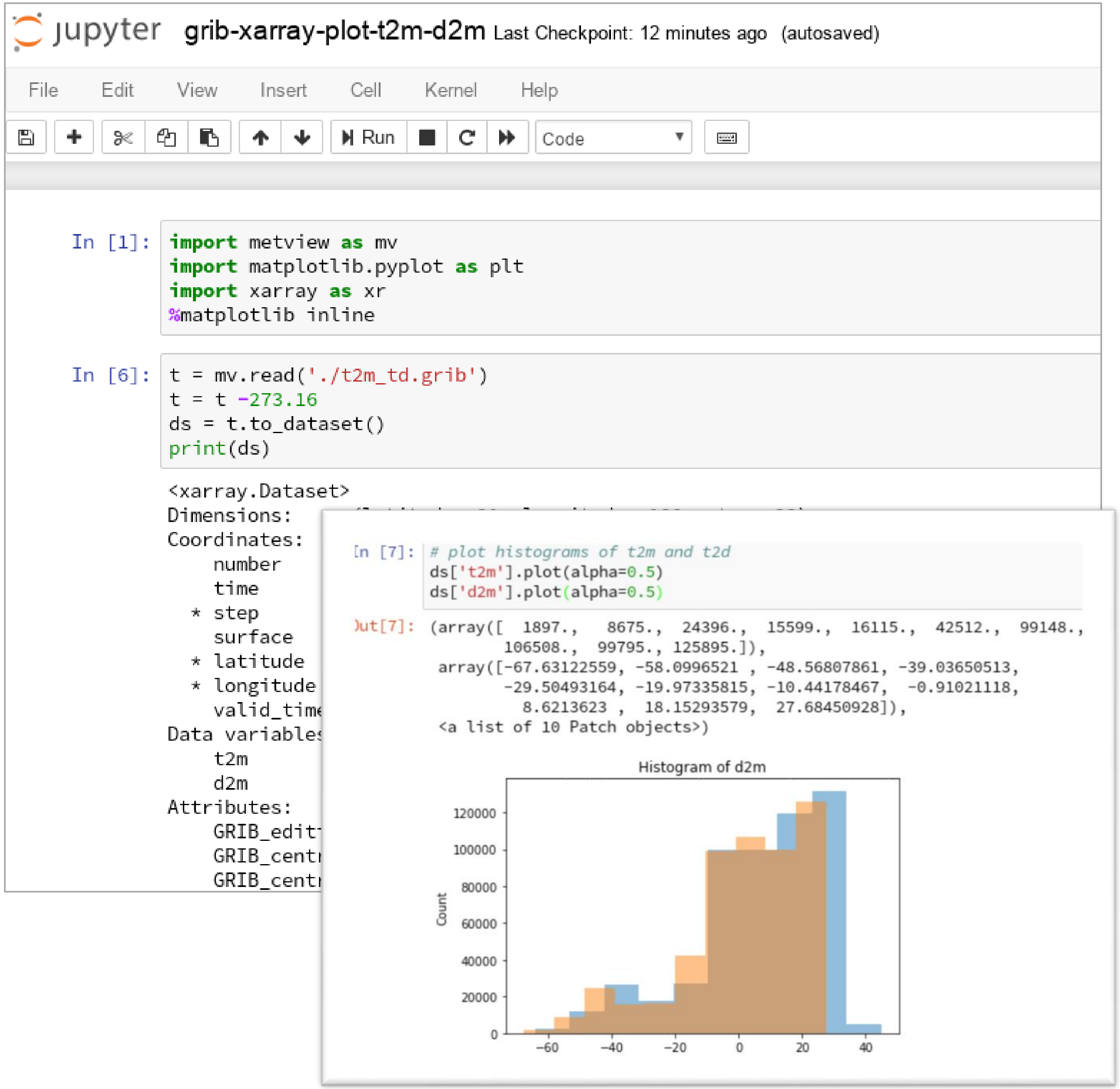
```python
import metview as mv
import pandas as pd

f = mv.read("tropical_cyclone.bufr")

res = mv.bufr_filter(
    data = f,
    output = "CSV",
    message_index = 1,
    custom_condition_count = 1,
    custom_key_1 = "ensembleMemberNumber",
    custom_value_1 = 2,
    parameter_count = 1,
    parameter_1 = "pressureReducedToMeanSeaLevel",
    extract_mode = "all"
)

df=res.to_dataframe()
print(df)
```

```
         date  latitude  level  longitude     value
0  2015-11-18       5.4    0.0      156.9  100000.0
1  2015-11-18       6.3    0.0      155.8  100000.0
2  2015-11-18       6.8    0.0      154.6  100300.0
3  2015-11-18       7.7    0.0      153.8  100100.0
4  2015-11-18       8.2    0.0      152.1  100300.0
5  2015-11-18       8.8    0.0      151.3  100000.0
6  2015-11-18       9.4    0.0      150.7  100300.0
7  2015-11-18       9.9    0.0      149.9  100100.0
8  2015-11-18      10.2    0.0      148.7  100300.0
```

Program finished (OK) : 567 ms [Finished at 13:22:45]

Data types (4)

- Can also export Metview Fieldsets to **xarray** Datasets (N-dimensional array (data cube) based on Pandas)
  - Provides data in the Common Data Model used by netCDF and the CDS
- Uses the **cfgrib** package developed by B-Open, available on github and PyPi:
- *“Python interface to map GRIB files to the NetCDF Common Data Model following the CF Conventions.”*
- This has an xarray GRIB driver, which is in the process of being integrated into the main xarray package



```
In [1]: import metview as mv
        import matplotlib.pyplot as plt
        import xarray as xr
        %matplotlib inline
```

```
In [6]: t = mv.read('./t2m_td.grib')
        t = t -273.16
        ds = t.to_dataset()
        print(ds)
```

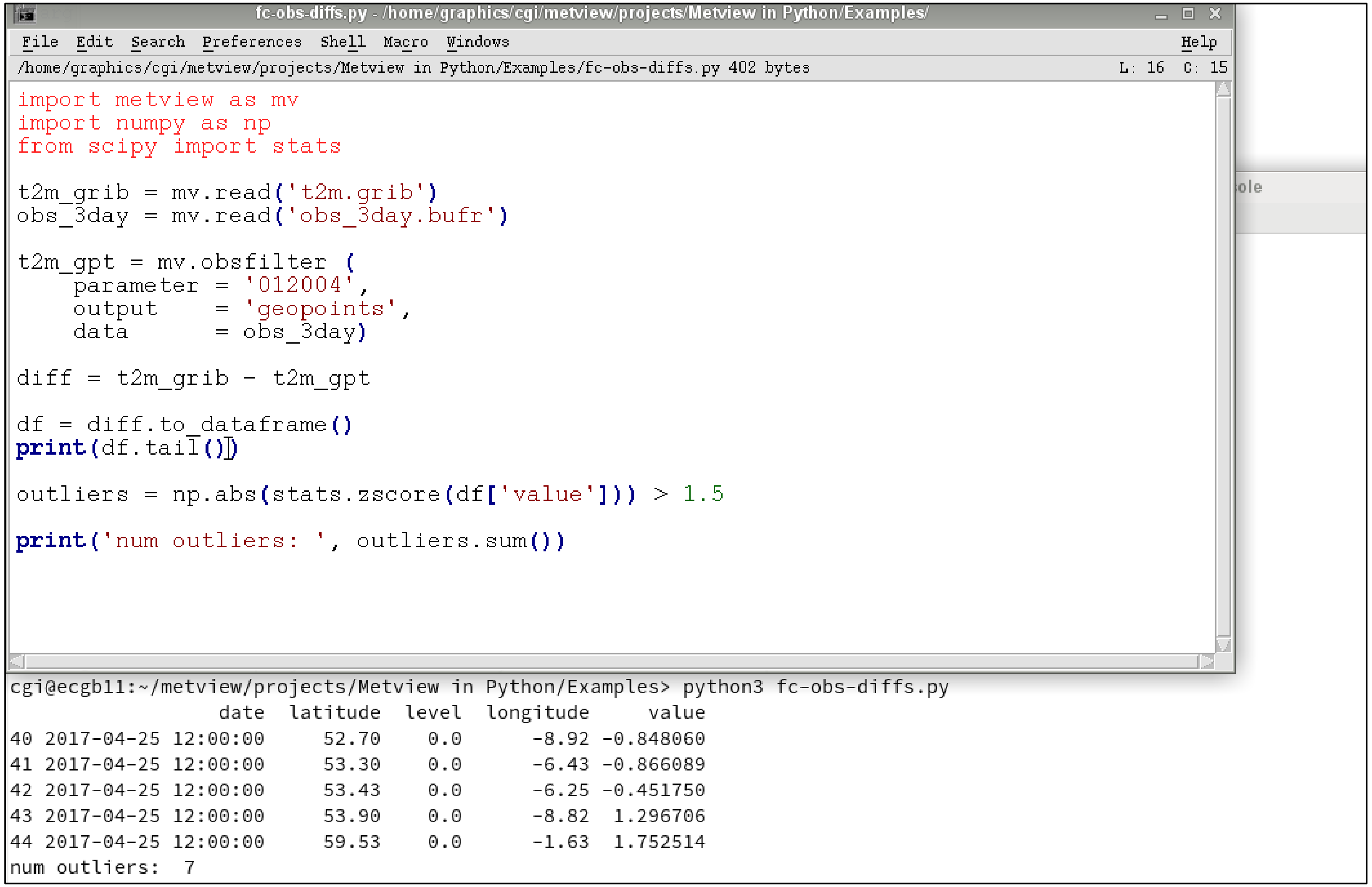
```
<xarray.Dataset>
Dimensions:
Coordinates:
    number
    time
  * step
    surface
  * latitude
  * longitude
    valid_time
Data variables
    t2m
    d2m
Attributes:
    GRIB_edit
    GRIB_centr
    GRIB_centr
```

```
In [7]: # plot histograms of t2m and t2d
        ds['t2m'].plot(alpha=0.5)
        ds['d2m'].plot(alpha=0.5)
```

```
Out[7]: (array([  1897.,   8675.,  24396.,  15599.,  16115.,  42512.,  99148.,
         106508.,  99795., 125895.]),
         array([-67.63122559, -58.0996521 , -48.56807861, -39.03650513,
                -29.50493164, -19.97335815, -10.44178467,  -0.91021118,
                  8.6213623 ,  18.15293579,  27.68450928]),
         <a list of 10 Patch objects>)
```

Ways to run a Metview Python script (1)

- This is a standard Python package, so it can be run the same way as any other package, for example…
- Create a script in a text editor and run from the command line



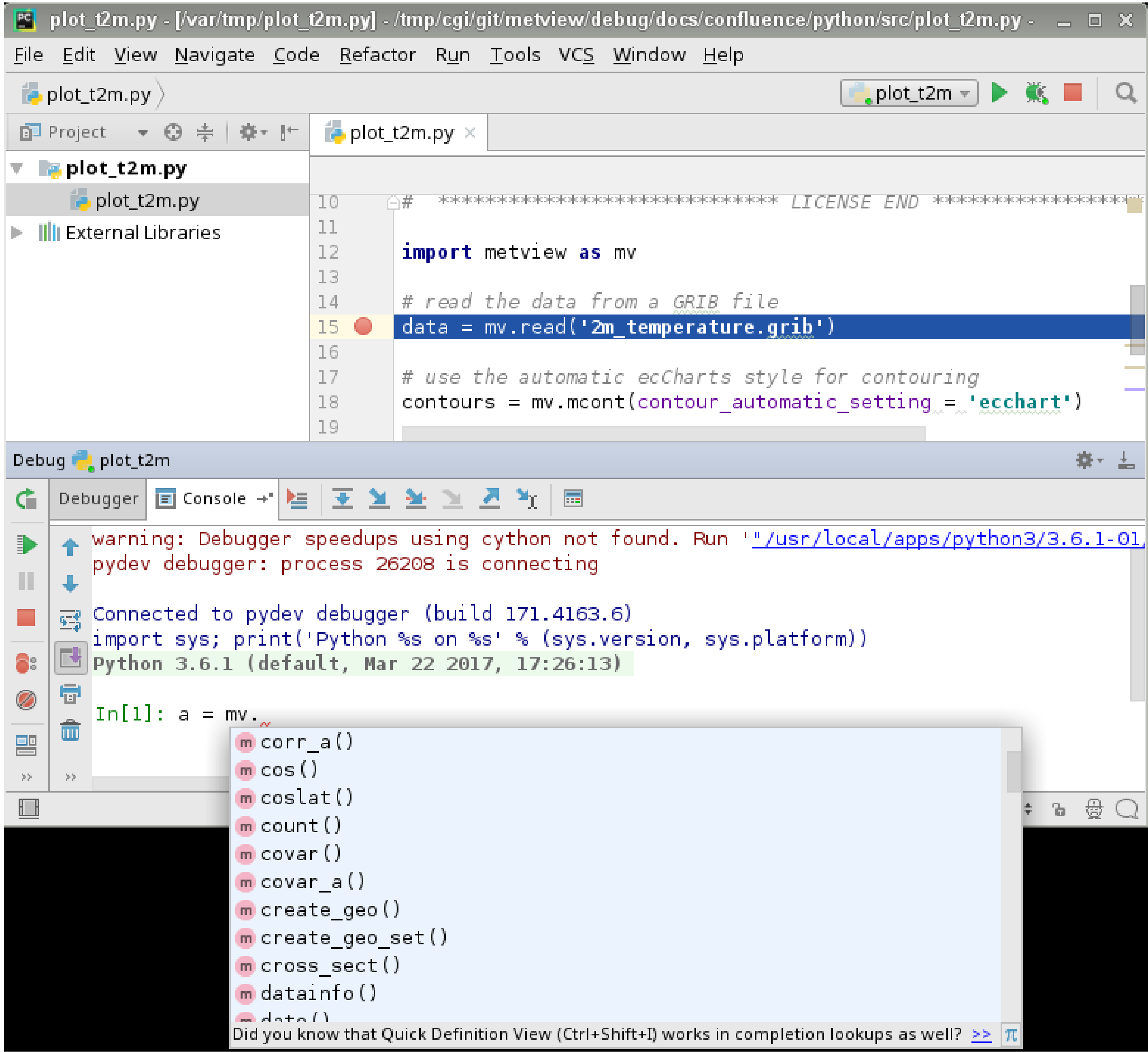
fc-obs-diffs.py - /home/graphics/cgi/metview/projects/Metview in Python/Examples/
File
Edit
Search
Preferences
Shell
Macro
Windows
Help
/home/graphics/cgi/metview/projects/Metview in Python/Examples/fc-obs-diffs.py 402 bytes
L: 16
C: 15

```
import metview as mv
import numpy as np
from scipy import stats

t2m_grib = mv.read('t2m.grib')
obs_3day = mv.read('obs_3day.bufr')

t2m_gpt = mv.obsfilter (
    parameter = '012004',
    output    = 'geopoints',
    data      = obs_3day)

diff = t2m_grib - t2m_gpt

df = diff.to_dataframe()
print(df.tail())

outliers = np.abs(stats.zscore(df['value'])) > 1.5

print('num outliers: ', outliers.sum())
```

```
cgi@ecgb11:~/metview/projects/Metview in Python/Examples> python3 fc-obs-diffs.py
                  date  latitude  level  longitude     value
40 2017-04-25 12:00:00     52.70    0.0      -8.92 -0.848060
41 2017-04-25 12:00:00     53.30    0.0      -6.43 -0.866089
42 2017-04-25 12:00:00     53.43    0.0      -6.25 -0.451750
43 2017-04-25 12:00:00     53.90    0.0      -8.82  1.296706
44 2017-04-25 12:00:00     59.53    0.0      -1.63  1.752514
num outliers:  7
```

Ways to run a Metview Python script (2)

- From a Python IDE, e.g. PyCharm – can provide code completion and debugging facilities



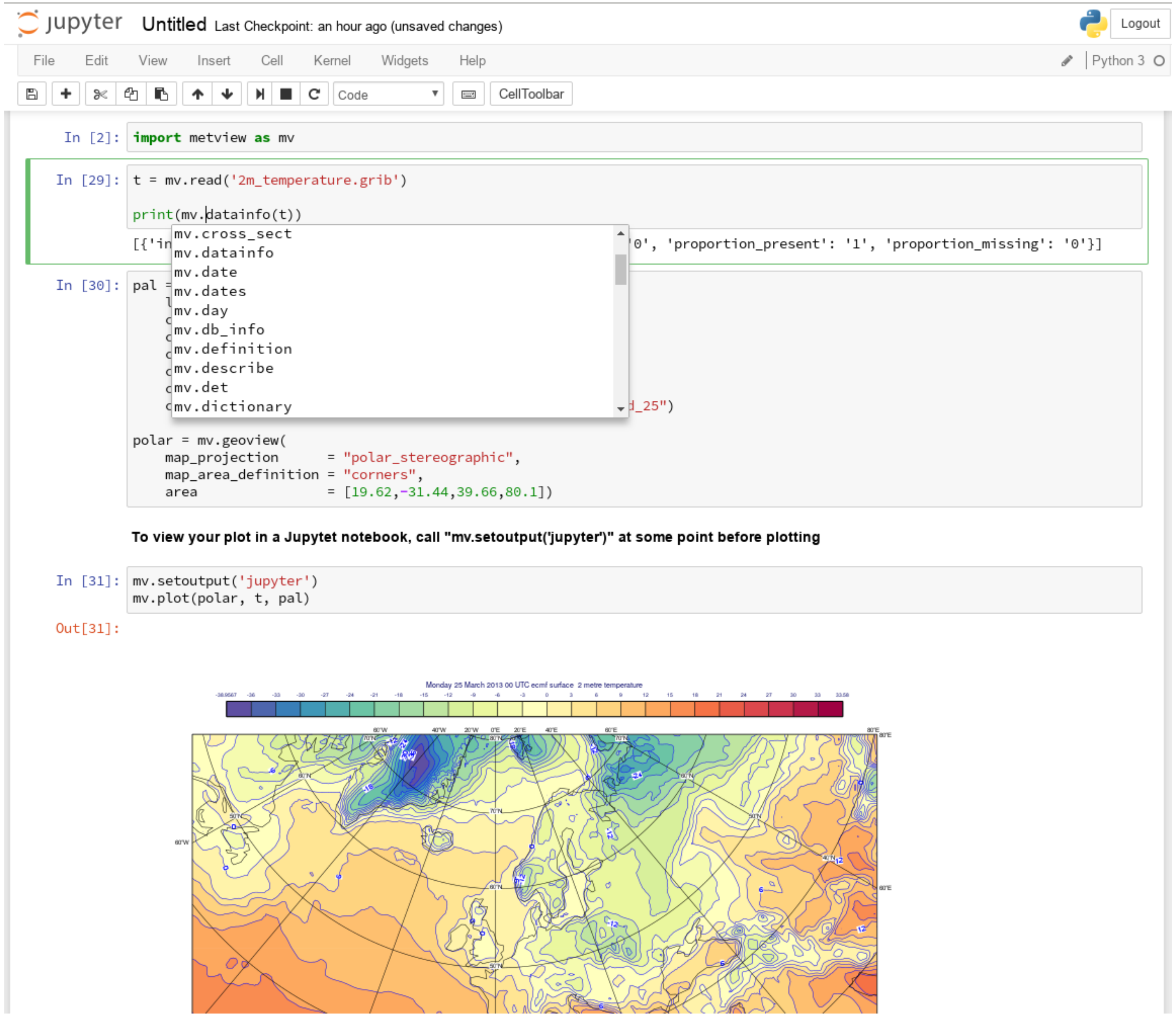
```
# ****************************** LICENSE END ******************

import metview as mv

# read the data from a GRIB file
data = mv.read('2m_temperature.grib')

# use the automatic ecCharts style for contouring
contours = mv.mcont(contour_automatic_setting = 'ecchart')
```

```
warning: Debugger speedups using cython not found. Run '"/usr/local/apps/python3/3.6.1-01,
pydev debugger: process 26208 is connecting

Connected to pydev debugger (build 171.4163.6)
import sys; print('Python %s on %s' % (sys.version, sys.platform))
Python 3.6.1 (default, Mar 22 2017, 17:26:13)

In[1]: a = mv.
```

Did you know that Quick Definition View (Ctrl+Shift+I) works in completion lookups as well? >>

Ways to run a Metview Python script (3)

- Jupyter notebook
- Runs in a web browser
- Provides interactive environment combining code, documentation and plots
- A way of sharing and reproducing work
- The concept can be expanded using JupyterLab and JupyterHub



```
In [2]:  import metview as mv
```

```
In [29]: t = mv.read('2m_temperature.grib')

         print(mv.datainfo(t))

         [{'in                                                    '0', 'proportion_present': '1', 'proportion_missing': '0'}]
```

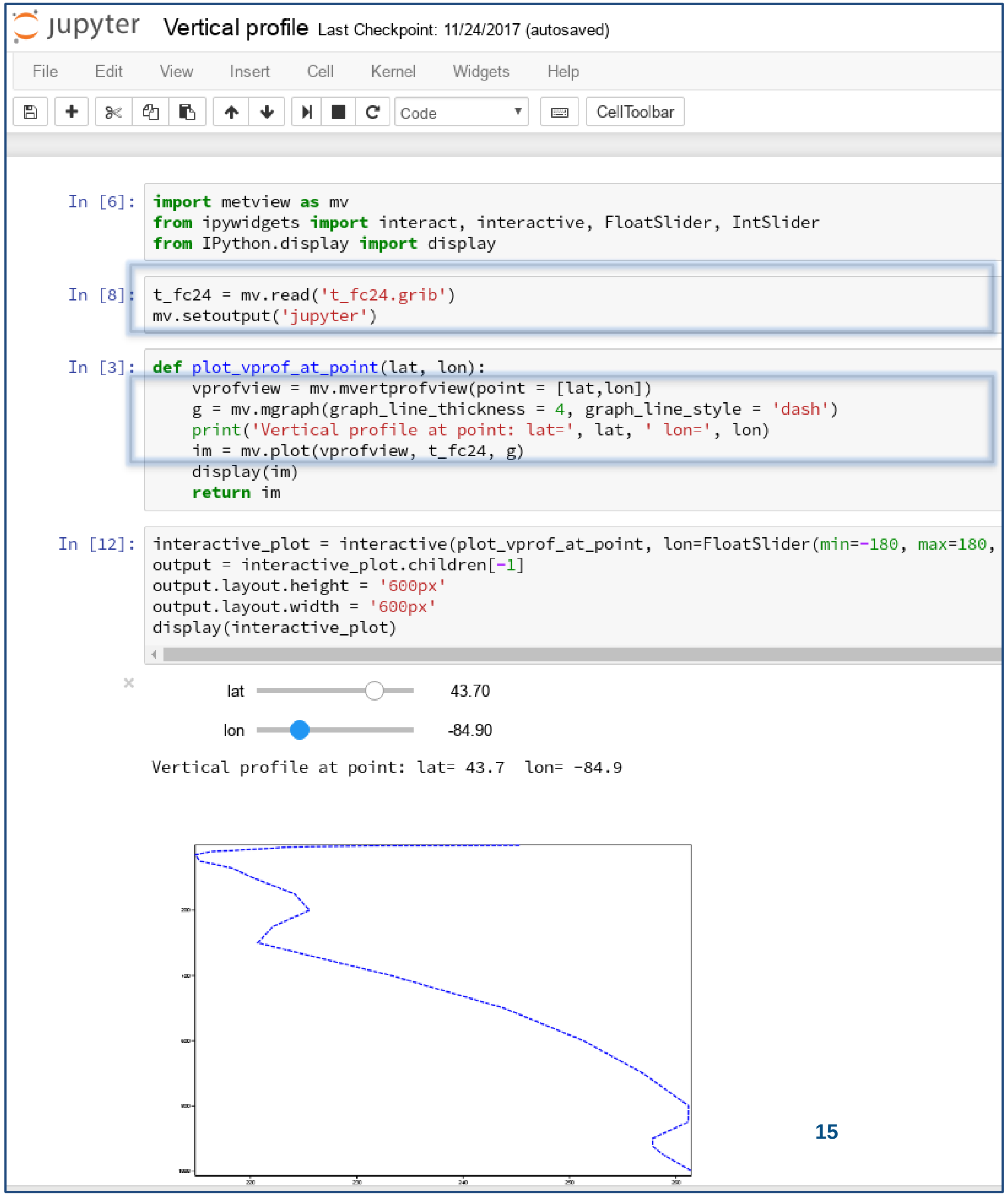
```
In [30]: pal =
                                                                 d_25")

         polar = mv.geoview(
             map_projection      = "polar_stereographic",
             map_area_definition = "corners",
             area                = [19.62,-31.44,39.66,80.1])
```

To view your plot in a Jupytet notebook, call "mv.setoutput('jupyter')" at some point before plotting

```
In [31]: mv.setoutput('jupyter')
         mv.plot(polar, t, pal)

Out[31]:
```

Ways to run a Metview Python script (4)

- Jupyter notebooks also offer some interactive tools that can be used in conjunction with Metview
- The highlighted code is Metview, the rest is boilerplate code for interactive plots in notebooks



```
In [6]: import metview as mv
        from ipywidgets import interact, interactive, FloatSlider, IntSlider
        from IPython.display import display
```

```
In [8]: t_fc24 = mv.read('t_fc24.grib')
        mv.setoutput('jupyter')
```

```
In [3]: def plot_vprof_at_point(lat, lon):
            vprofview = mv.mvertprofview(point = [lat,lon])
            g = mv.mgraph(graph_line_thickness = 4, graph_line_style = 'dash')
            print('Vertical profile at point: lat=', lat, ' lon=', lon)
            im = mv.plot(vprofview, t_fc24, g)
            display(im)
            return im
```

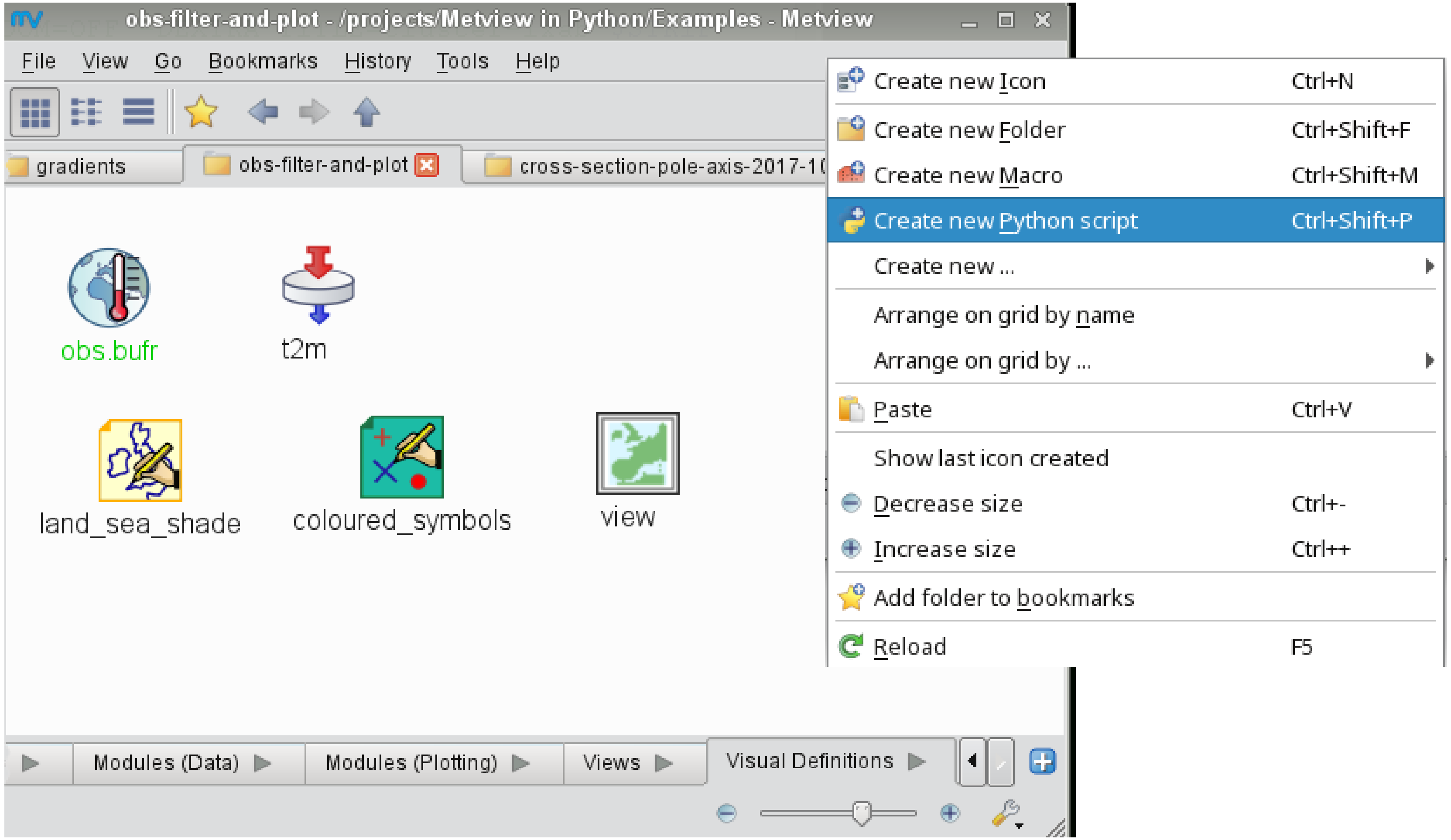
```
In [12]: interactive_plot = interactive(plot_vprof_at_point, lon=FloatSlider(min=-180, max=180,
         output = interactive_plot.children[-1]
         output.layout.height = '600px'
         output.layout.width = '600px'
         display(interactive_plot)
```

lat 43.70

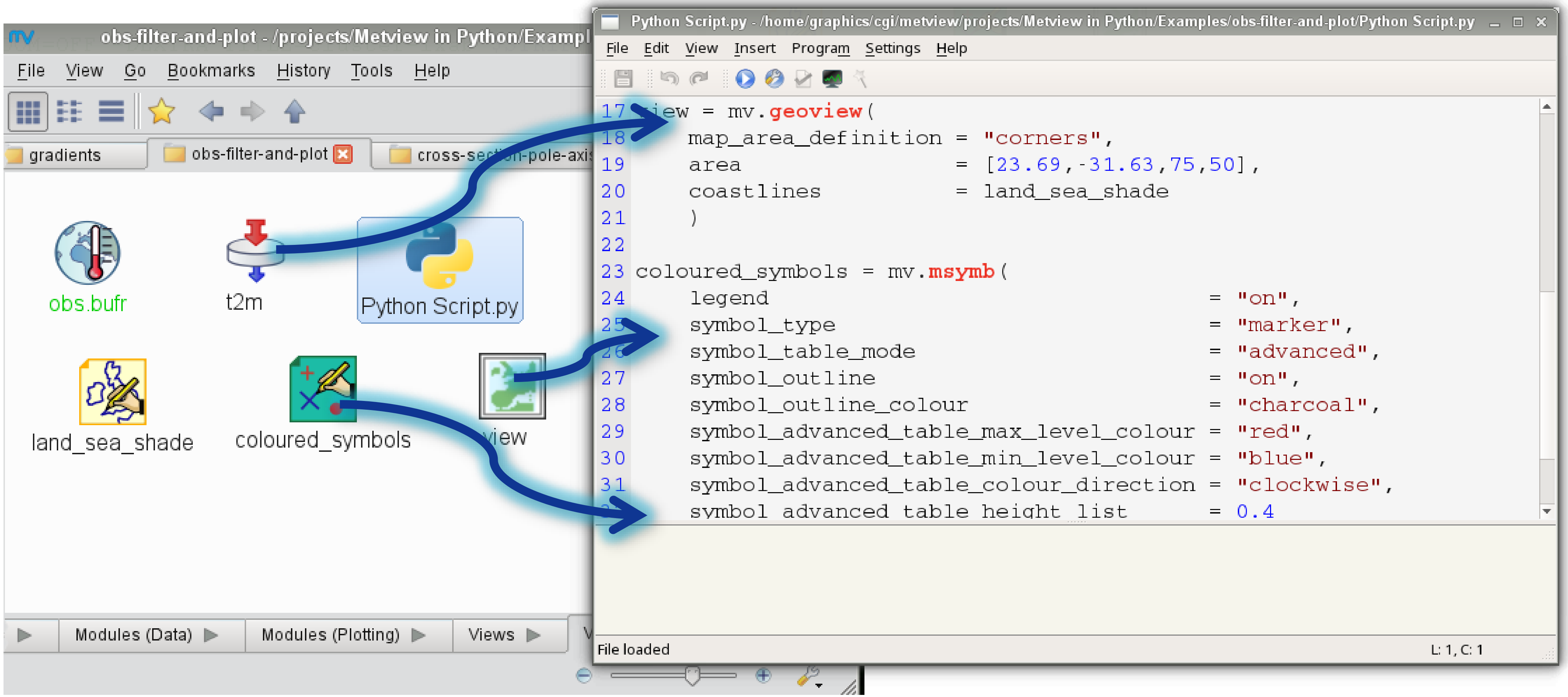
lon -84.90

```
Vertical profile at point: lat= 43.7  lon= -84.9
```

Ways to run a Metview Python script (5)

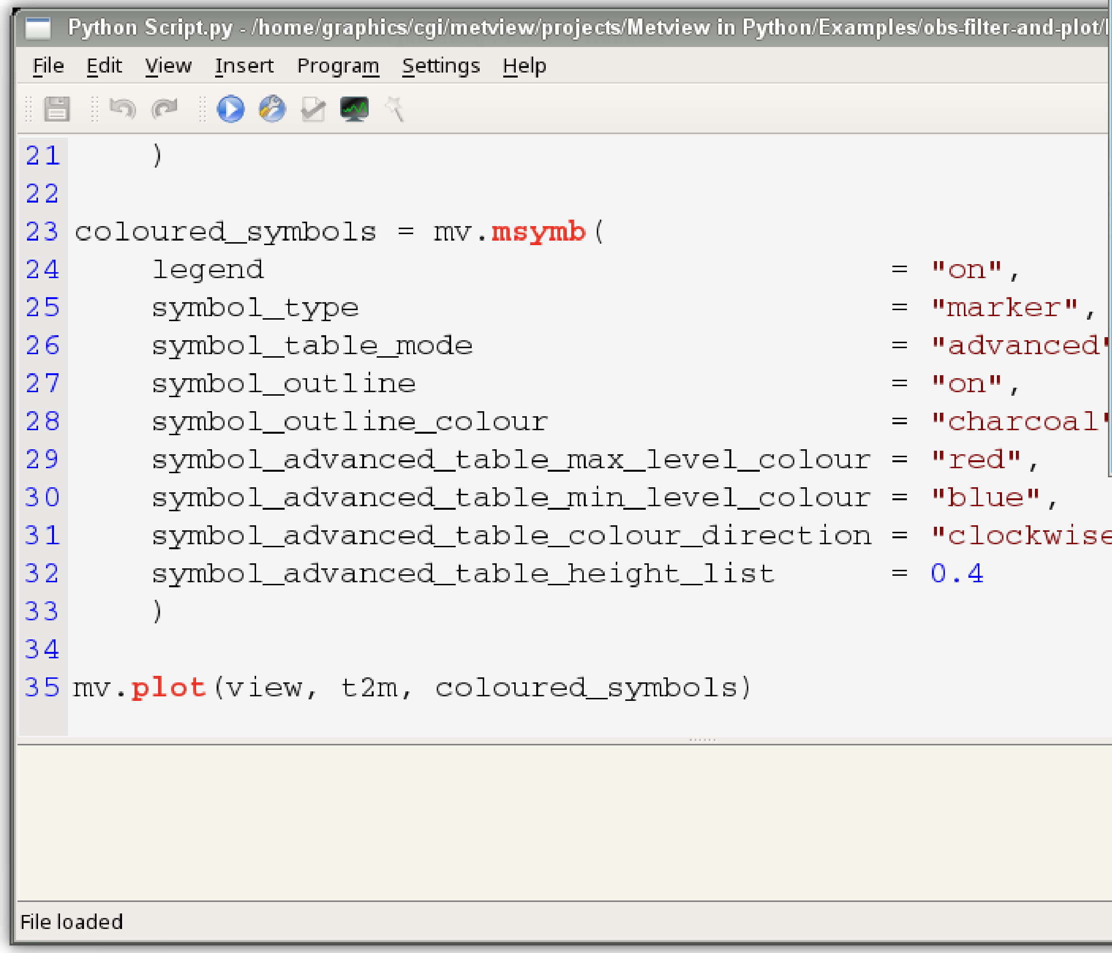


Ways to run a Metview Python script (5)


obs-filter-and-plot - /projects/Metview in Python/Exampl
File View Go Bookmarks History Tools Help
gradients
obs-filter-and-plot
cross-section-pole-axis
obs.bufr
t2m
Python Script.py
land_sea_shade
coloured_symbols
view
Modules (Data)
Modules (Plotting)
Views
Python Script.py - /home/graphics/cgi/metview/projects/Metview in Python/Examples/obs-filter-and-plot/Python Script.py
File Edit View Insert Program Settings Help
File loaded
L: 1, C: 1

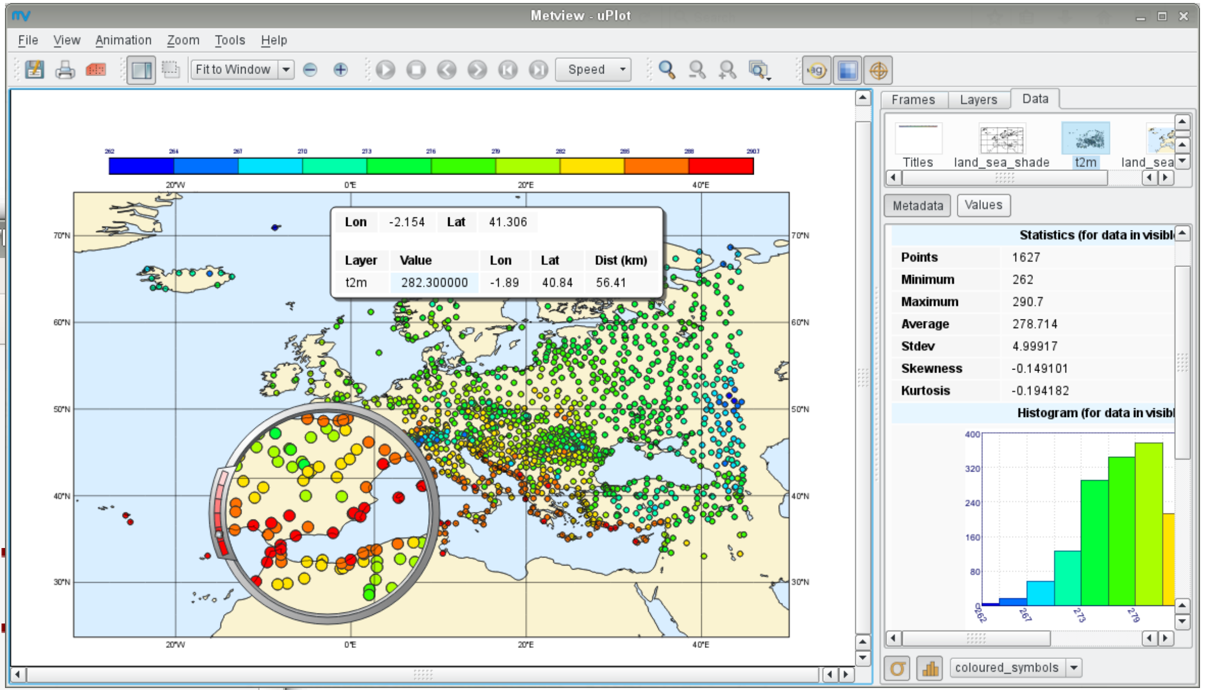
```
view = mv.geoview(
    map_area_definition = "corners",
    area                = [23.69,-31.63,75,50],
    coastlines          = land_sea_shade
    )

coloured_symbols = mv.msymb(
    legend                                 = "on",
    symbol_type                            = "marker",
    symbol_table_mode                      = "advanced",
    symbol_outline                         = "on",
    symbol_outline_colour                  = "charcoal",
    symbol_advanced_table_max_level_colour = "red",
    symbol_advanced_table_min_level_colour = "blue",
    symbol_advanced_table_colour_direction = "clockwise",
    symbol_advanced_table_height_list      = 0.4
```

Ways to run a Metview Python script (5)


Python Script.py - /home/graphics/cgi/metview/projects/Metview in Python/Examples/obs-filter-and-plot/
File Edit View Insert Program Settings Help

```
    )

coloured_symbols = mv.msymb(
    legend                                = "on",
    symbol_type                           = "marker",
    symbol_table_mode                     = "advanced'
    symbol_outline                        = "on",
    symbol_outline_colour                 = "charcoal'
    symbol_advanced_table_max_level_colour = "red",
    symbol_advanced_table_min_level_colour = "blue",
    symbol_advanced_table_colour_direction = "clockwise",
    symbol_advanced_table_height_list     = 0.4
    )

mv.plot(view, t2m, coloured_symbols)
```

File loaded L: 1, C: 1

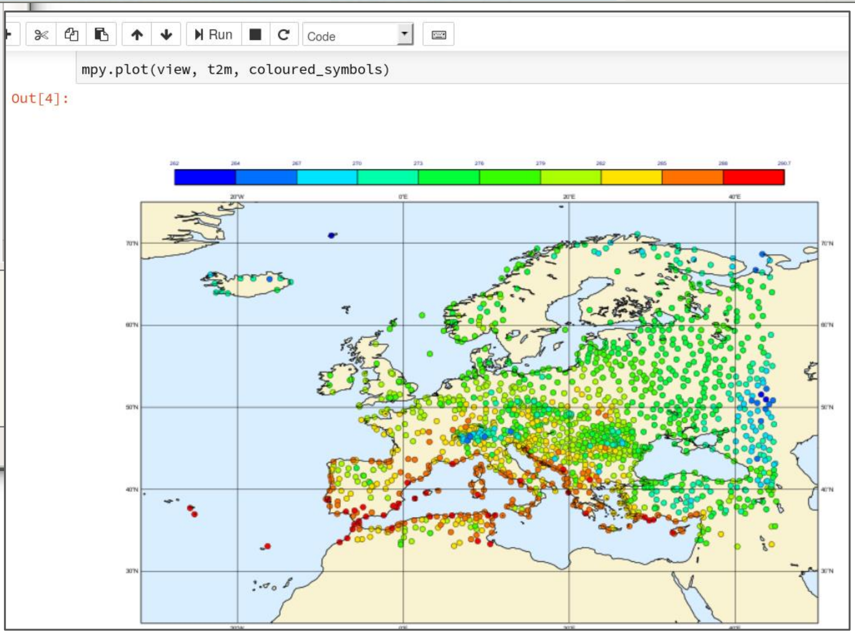


Lon	-2.154	Lat	41.306	
Layer	Value	Lon	Lat	Dist (km)
t2m	282.300000	-1.89	40.84	56.41

Statistics (for data in visibl

Points	1627
Minimum	262
Maximum	290.7
Average	278.714
Stdev	4.99917
Skewness	-0.149101
Kurtosis	-0.194182

Histogram (for data in visibl



```
mpy.plot(view, t2m, coloured_symbols)
```

Out[4]:

How to set up

- Requirements:
  - An installation of Metview 5
  - Python 3
  - Metview’s python package installed

e.g. (Fedora 28)

```
dnf config-manager --add-repo
    https://download.opensuse.org/repositories/home:SStepke/Fedora_28/home:SStepke.repo
dnf install Metview # install binaries
pip install metview # install Python layer
python3 -m metview selfcheck # available in next release, 0.8.5
Hello world - printed from Metview!
Trying to connect to a Metview installation...
Metview version 5.3.0 found
Your system is ready.
```

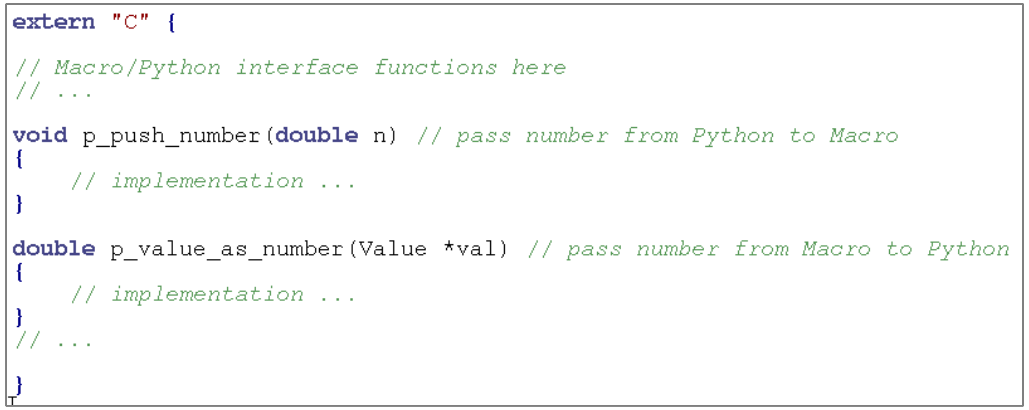
  - Ubuntu has Metview in its core repositories, but version 5 is not properly there

To run on internal ECMWF machines or ecgate:

```
module load python3/new
module swap metview/new
```

Implementation details (1)

- We use the **cffi** package to bridge C++/Python
  - We needed to extract the Macro functions into a shared library for cffi to open
- One single source file in Metview – python.cc – with all the interface functions (extern C)
  - For passing various data types between Python and Macro
  - For pushing function operands onto the stack
  - For calling Macro functions (by name)
  - The Python module has the corresponding header file
- Macro has a function that returns a list of its functions
  - Create a Python function for each
  - Put into the module namespace
  - So we can add new functions to Macro, and the Python module will automatically see them (no need to update the Python layer)



```
extern "C" {

// Macro/Python interface functions here
// ...

void p_push_number(double n) // pass number from Python to Macro
{
    // implementation ...
}

double p_value_as_number(Value *val) // pass number from Macro to Python
{
    // implementation ...
}
// ...

}
```

Implementation details (2)

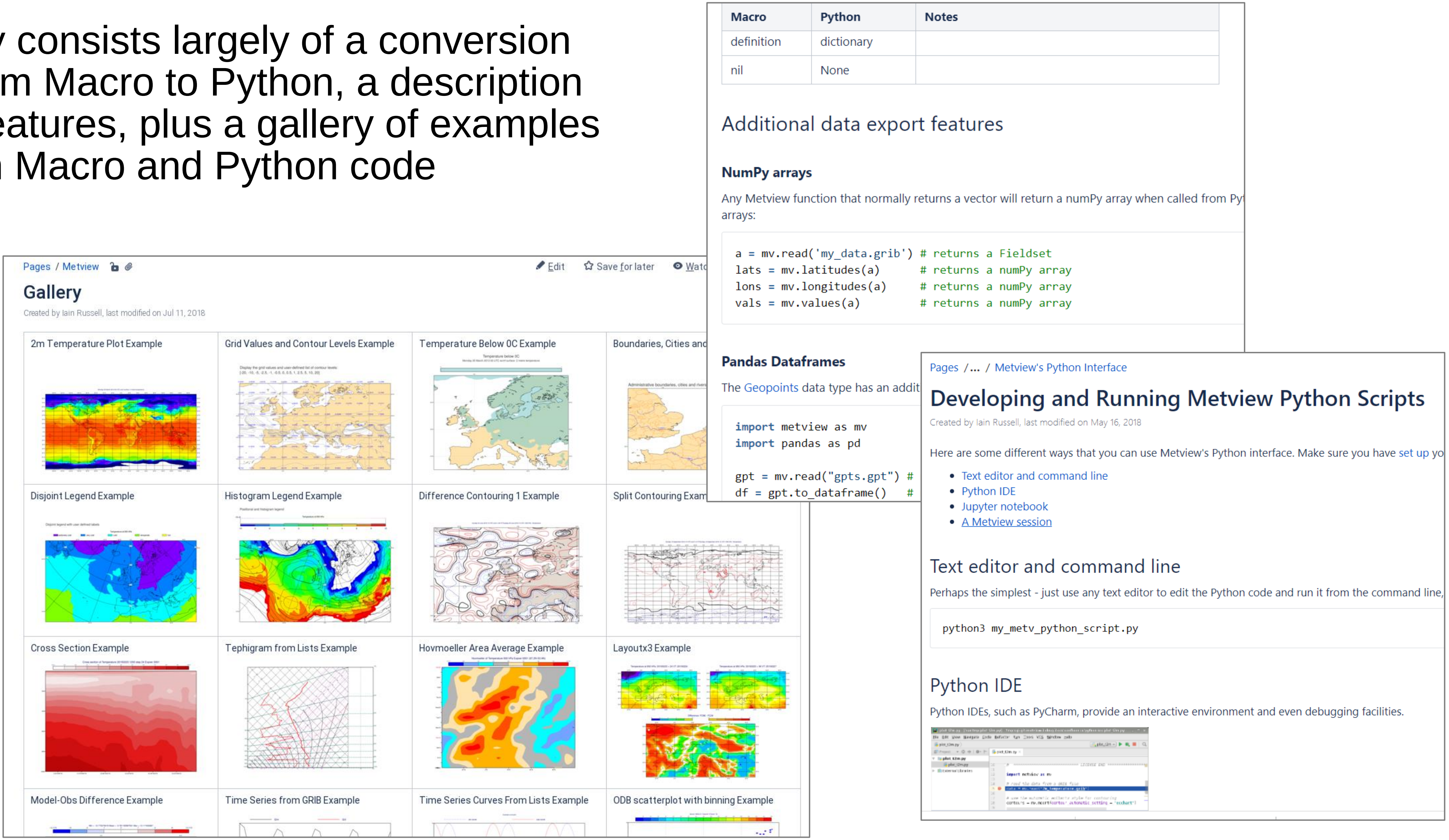
- But… Metview has a service-oriented architecture – how are services run and managed from Python?
- A suitable version of Metview must be in the PATH
  - The Python code starts a lightweight Metview session and links to the shared library
  - This allows the Macro functions to invoke other Metview modules (e.g. Cross Section, uPlot) and return their results
  - Not necessary if run from a Metview session!
- A little extra magic required here and there, e.g. to pass plots to Jupyter notebooks
- `import metview`
- Faster import - we noticed that importing some modules was quite slow (e.g. IPython for detecting the Jupyter environment), so we only import when actually needed

Tricky things

- Indexing
  - Macro uses 1 as its base index, Python uses 0
  - There is now extra code into Metview’s binaries and the Python layer to handle this
    - Macro: `first_field = my_fields[1]`
    - Python: `first_field = my_fields[0]`
    - Macro: `print(search(’Where is this’, ’this’))     # 3`
    - Python: `print(mv.search(’Where is this’, ’this’)) # 2`
- Keeping version independence
  - Users should be able to update either the Metview binaries or its Python layer without updating the other
  - Try to keep the interface functions as generic as possible
  - New data types in Metview will require a little code in the Python layer
  - We currently don’t force a particular Metview version (apart from >= 5.0), but a change in the interface functions (as opposed to additions) could prompt it

Documentation

- Currently consists largely of a conversion guide from Macro to Python, a description of new features, plus a gallery of examples with both Macro and Python code



Macro	Python	Notes
definition	dictionary	
nil	None	

Additional data export features

NumPy arrays

Any Metview function that normally returns a vector will return a numPy array when called from Py
arrays:

```
a = mv.read('my_data.grib') # returns a Fieldset
lats = mv.latitudes(a)      # returns a numPy array
lons = mv.longitudes(a)     # returns a numPy array
vals = mv.values(a)         # returns a numPy array
```

Pandas Dataframes

The Geopoints data type has an addit

```
import metview as mv
import pandas as pd

gpt = mv.read("gpts.gpt") #
df = gpt.to_dataframe()   #
```

Pages / Metview

Gallery

Created by Iain Russell, last modified on Jul 11, 2018

2m Temperature Plot Example	Grid Values and Contour Levels Example	Temperature Below 0C Example	Boundaries, Cities and
Disjoint Legend Example	Histogram Legend Example	Difference Contouring 1 Example	Split Contouring Exam
Cross Section Example	Tephigram from Lists Example	Hovmoeller Area Average Example	Layoutx3 Example
Model-Obs Difference Example	Time Series from GRIB Example	Time Series Curves From Lists Example	ODB scatterplot with binning Example

Pages /... / Metview's Python Interface

Developing and Running Metview Python Scripts

Created by Iain Russell, last modified on May 16, 2018

Here are some different ways that you can use Metview's Python interface. Make sure you have set up yo

- Text editor and command line
- Python IDE
- Jupyter notebook
- A Metview session

Text editor and command line

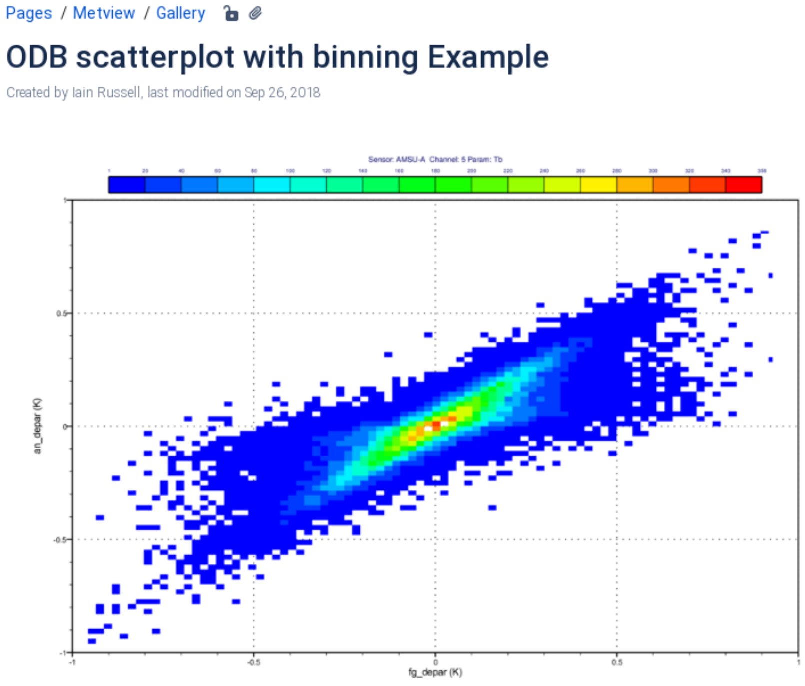
Perhaps the simplest - just use any text editor to edit the Python code and run it from the command line,

```
python3 my_metv_python_script.py
```

Python IDE

Python IDEs, such as PyCharm, provide an interactive environment and even debugging facilities.

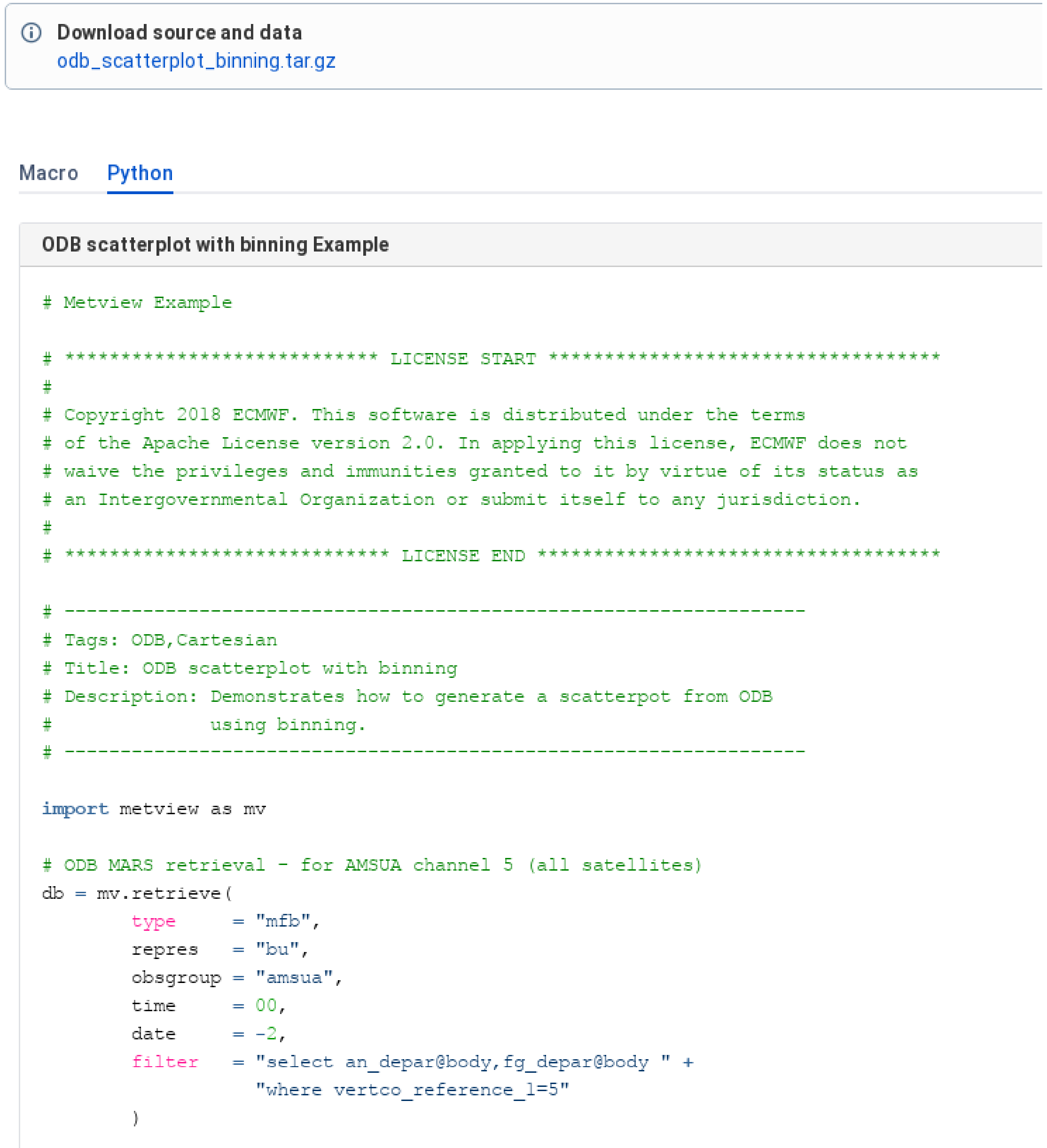
Gallery example



Pages / Metview / Gallery

ODB scatterplot with binning Example

Created by Iain Russell, last modified on Sep 26, 2018



Download source and data
odb_scatterplot_binning.tar.gz

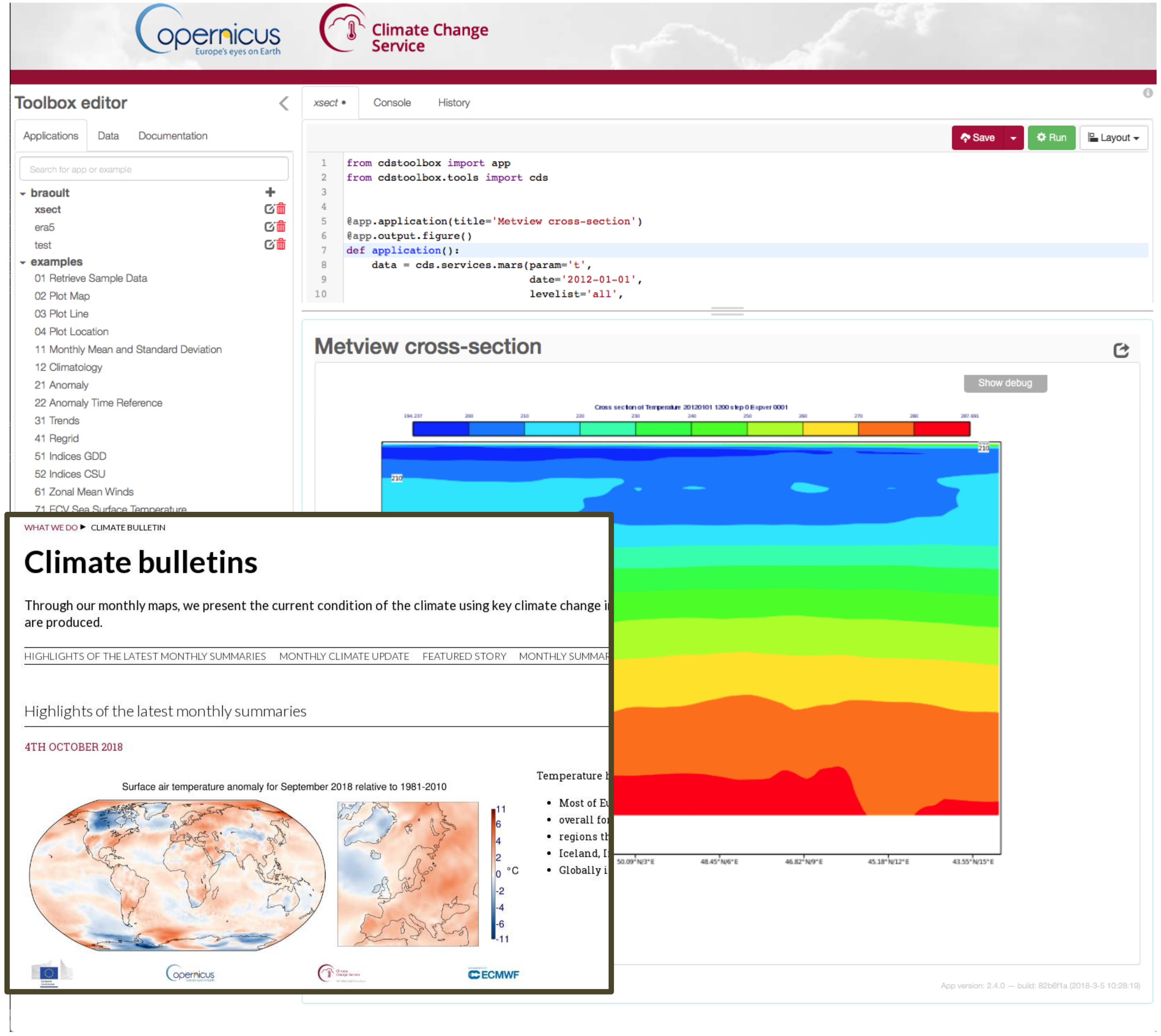
ODB scatterplot with binning Example

```python
# Metview Example

# **************************** LICENSE START ***********************************
#
# Copyright 2018 ECMWF. This software is distributed under the terms
# of the Apache License version 2.0. In applying this license, ECMWF does not
# waive the privileges and immunities granted to it by virtue of its status as
# an Intergovernmental Organization or submit itself to any jurisdiction.
#
# ***************************** LICENSE END ************************************

# ------------------------------------------------------------------
# Tags: ODB,Cartesian
# Title: ODB scatterplot with binning
# Description: Demonstrates how to generate a scatterpot from ODB
#              using binning.
# ------------------------------------------------------------------

import metview as mv

# ODB MARS retrieval - for AMSUA channel 5 (all satellites)
db = mv.retrieve(
        type     = "mfb",
        repres   = "bu",
        obsgroup = "amsua",
        time     = 00,
        date     = -2,
        filter   = "select an_depar@body,fg_depar@body " +
                   "where vertco_reference_1=5"
        )
```

Reaching out

- Metview, with its Python interface, is in the Copernicus CDS Toolbox

- We are also exploring ways to harmonise verification efforts across ECMWF with Metview supplying the core functionality (MARS, fieldsets, observations, …) - part of a drive for having fewer packages to maintain



```python
from cdstoolbox import app
from cdstoolbox.tools import cds


@app.application(title='Metview cross-section')
@app.output.figure()
def application():
    data = cds.services.mars(param='t',
                             date='2012-01-01',
                             levelist='all',
```

Metview cross-section

Cross section of Temperature 20120101 1200 step 0 Expver 0001

WHAT WE DO ▶ CLIMATE BULLETIN

Climate bulletins

Through our monthly maps, we present the current condition of the climate using key climate change i
are produced.

HIGHLIGHTS OF THE LATEST MONTHLY SUMMARIES   MONTHLY CLIMATE UPDATE   FEATURED STORY   MONTHLY SUMMAR

Highlights of the latest monthly summaries

4TH OCTOBER 2018

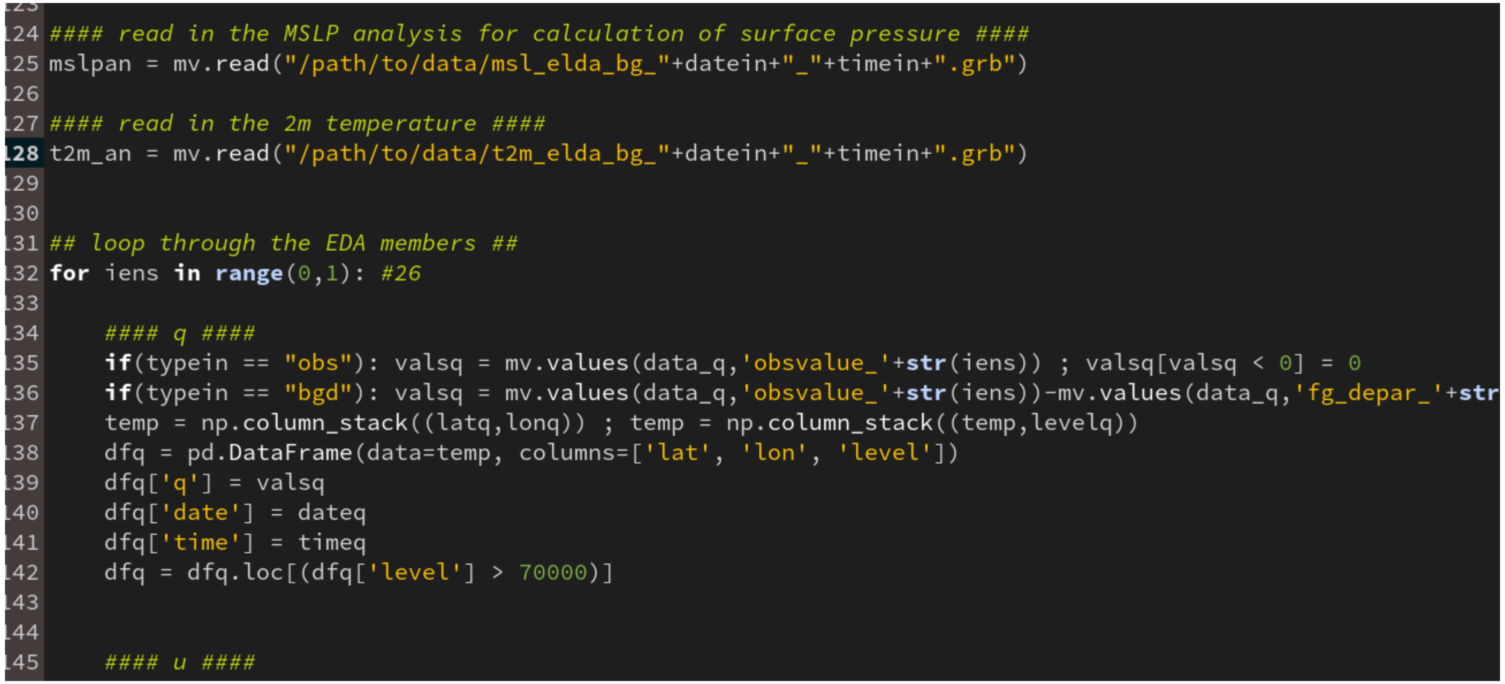
Surface air temperature anomaly for September 2018 relative to 1981-2010

Temperature h

- Most of Eu
- overall fo
- regions th
- Iceland, I
- Globally i

Feedback

- Feedback has so far been very positive
- We have some enthusiastic users: “It combines all the power of Metview with all the power of Python!” (internal user to Iain, Oct 2018)
- As is often the case, we only hear from users if they encounter a problem, but log files suggest quite a lot of activity



```
#### read in the MSLP analysis for calculation of surface pressure ####
mslpan = mv.read("/path/to/data/msl_elda_bg_"+datein+"_"+timein+".grb")

#### read in the 2m temperature ####
t2m_an = mv.read("/path/to/data/t2m_elda_bg_"+datein+"_"+timein+".grb")


## loop through the EDA members ##
for iens in range(0,1): #26

    #### q ####
    if(typein == "obs"): valsq = mv.values(data_q,'obsvalue_'+str(iens)) ; valsq[valsq < 0] = 0
    if(typein == "bgd"): valsq = mv.values(data_q,'obsvalue_'+str(iens))-mv.values(data_q,'fg_depar_'+str
    temp = np.column_stack((latq,lonq)) ; temp = np.column_stack((temp,levelq))
    dfq = pd.DataFrame(data=temp, columns=['lat', 'lon', 'level'])
    dfq['q'] = valsq
    dfq['date'] = dateq
    dfq['time'] = timeq
    dfq = dfq.loc[(dfq['level'] > 70000)]


    #### u ####
```

Future

- Release beta version in October 2018
  - Advertise more widely to get more feedback
- Release version 1.0.0 – end of 2018?

- Provide tools for automatic translation from Macro to Python
- Improve information available for IDEs (e.g. function descriptions)

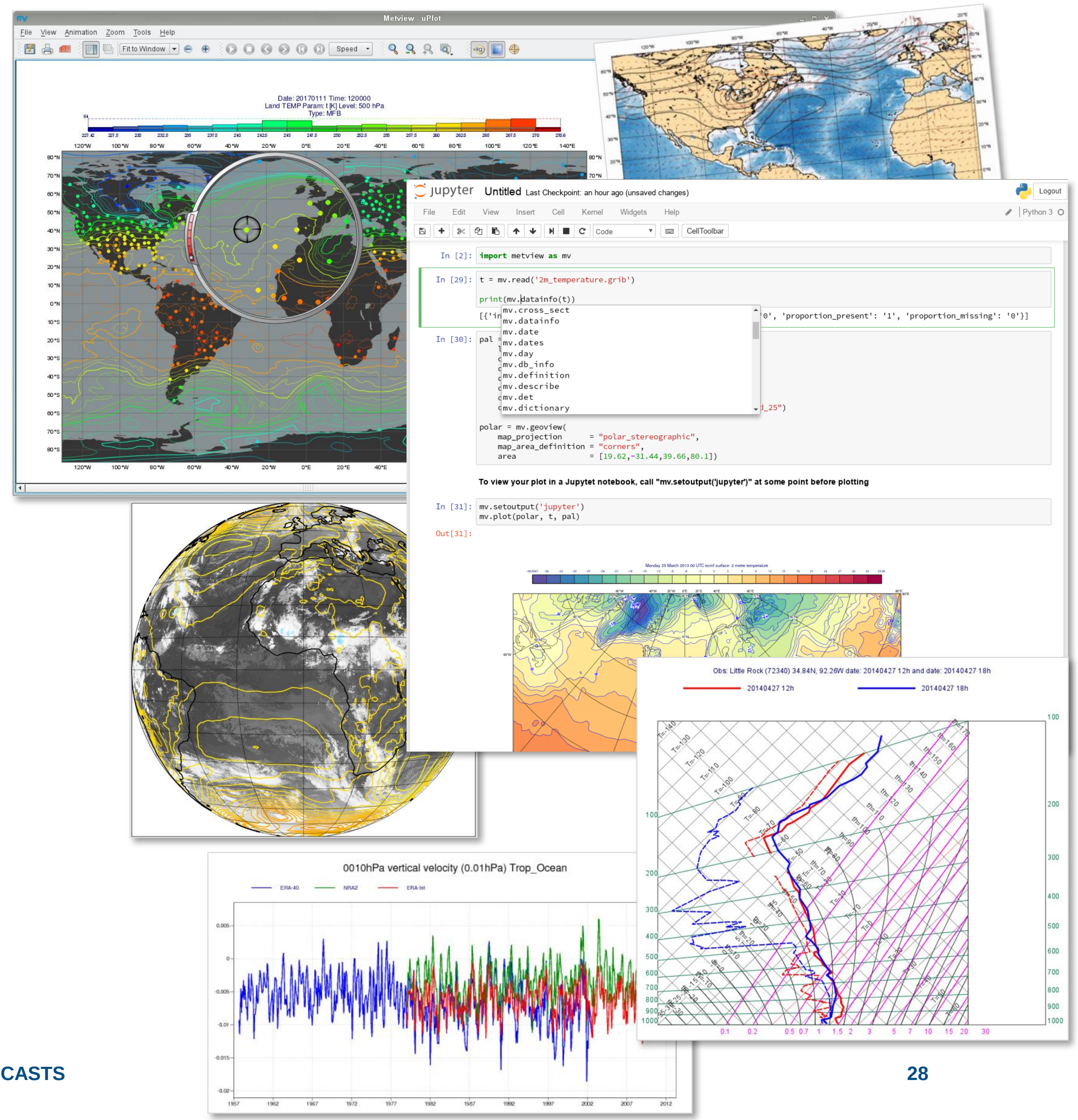
- Investigate **conda** for packaging Metview's binaries and the Python layer together

- Use Metview's Python interface as a base layer for a new verification toolkit (some work already underway)
- Work more closely with the CDS Toolbox and other ECMWF Python frameworks

For more information…

- Email us:
  - Developers: metview@ecmwf.int
  - Support: software.support@ecmwf.int
- Visit our web pages:
  - http://confluence.ecmwf.int/metview
- Download (Metview source, binaries)
- Documentation and tutorials available
- Metview articles in ECMWF newsletters
- e-Learning material
- Download Metview's Python interface:
  - pip install metview
  - https://github.com/ecmwf/metview-python

Questions?


Metview - uPlot
File View Animation Zoom Tools Help
Fit to Window
Speed
Date: 20170111 Time: 120000
Land TEMP Param: t [K] Level: 500 hPa
Type: MFB
jupyter Untitled Last Checkpoint: an hour ago (unsaved changes)
Logout
File Edit View Insert Cell Kernel Widgets Help
Python 3
Code
CellToolbar
Monday 25 March 2013 00 UTC ecmf surface 2 metre temperature
Obs: Little Rock (72340) 34.84N, 92.26W date: 20140427 12h and date: 20140427 18h
20140427 12h
20140427 18h
0010hPa vertical velocity (0.01hPa) Trop_Ocean
ERA-40
NRA2
ERA-Int

```
In [2]: import metview as mv

In [29]: t = mv.read('2m_temperature.grib')

         print(mv.datainfo(t))
         mv.cross_sect
         mv.datainfo
         mv.date
         mv.dates
         mv.day
         mv.db_info
         mv.definition
         mv.describe
         mv.det
         mv.dictionary
         [{'in                    '0', 'proportion_present': '1', 'proportion_missing': '0'}]

In [30]: pal =
                                  d_25")

         polar = mv.geoview(
             map_projection      = "polar_stereographic",
             map_area_definition = "corners",
             area                = [19.62,-31.44,39.66,80.1])
```

To view your plot in a Jupytet notebook, call "mv.setoutput('jupyter')" at some point before plotting

```
In [31]: mv.setoutput('jupyter')
         mv.plot(polar, t, pal)

Out[31]:
```