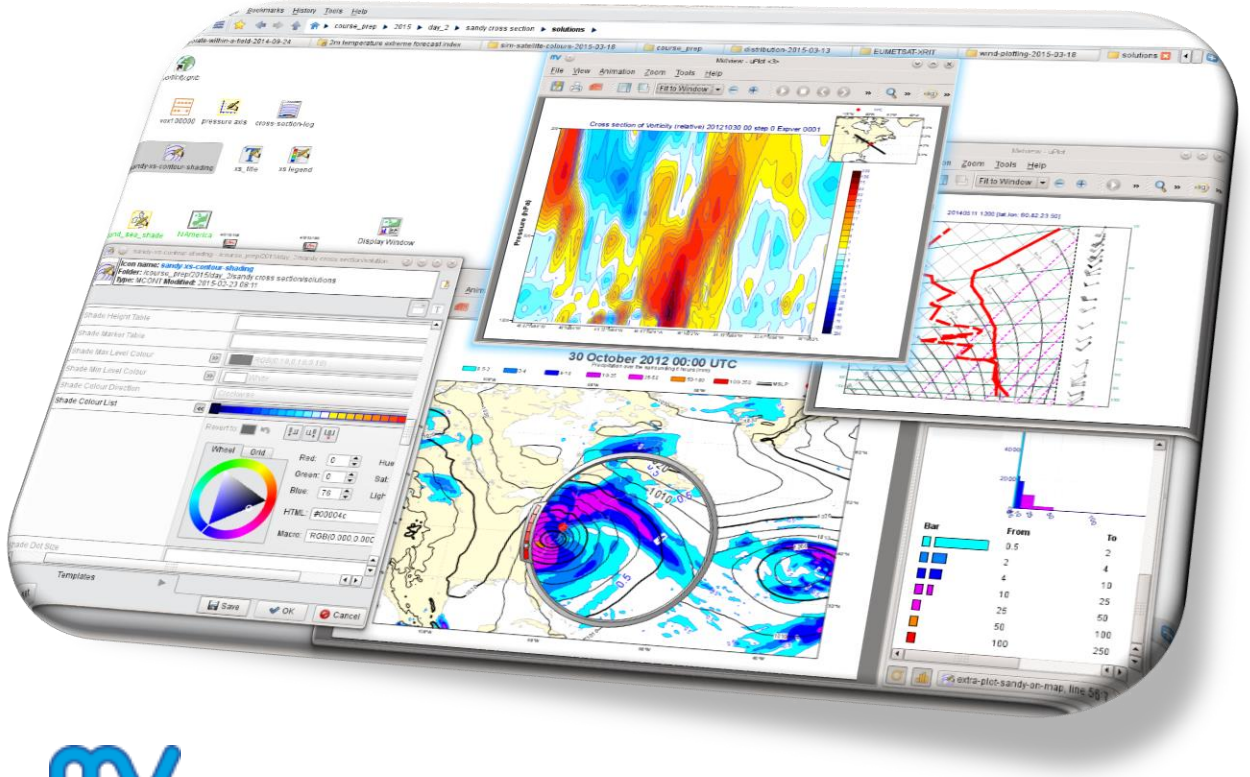
Metview and Python - what they can do for each other

Workshop on Python for Earth System Sciences, ECMWF

Iain Russell,
Fernando Ii,
Sándor Kertész,
Stephan Siemen

Development Section, ECMWF

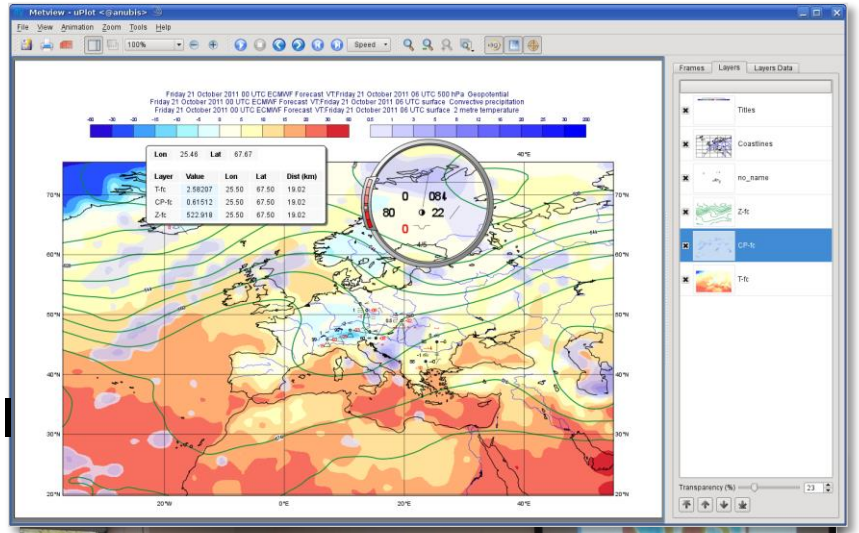


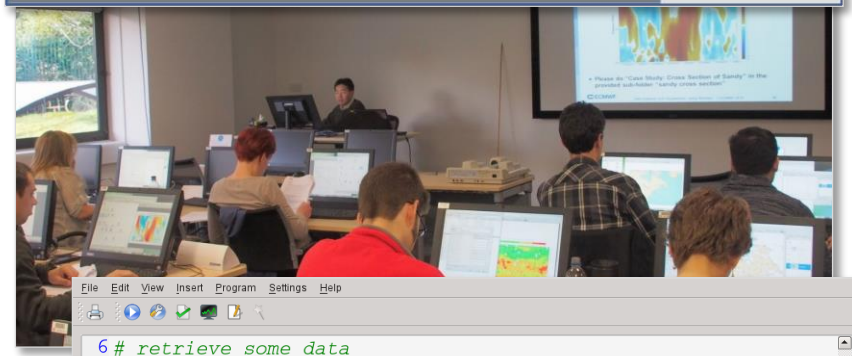
What is Metview?

- Workstation software for meteorological researchers and operational analysts; also for teaching (e.g. OpenIFS workshops)
- Handles GRIB, BUFR, NetCDF, ODB, Geopoints, CSV, ASCII
- Can access MARS, either locally or through the Web API

High-level interface, built on core ECMWF technologies:
MARS, ecCodes, Magics, ODB, Emoslib
(future: MIR)

- Interactive (icons) or batch usage (Macro language)
- UNIX, Open Source under Apache Licence 2.0
- Metview is a co-operation project with INPE (Brazil)

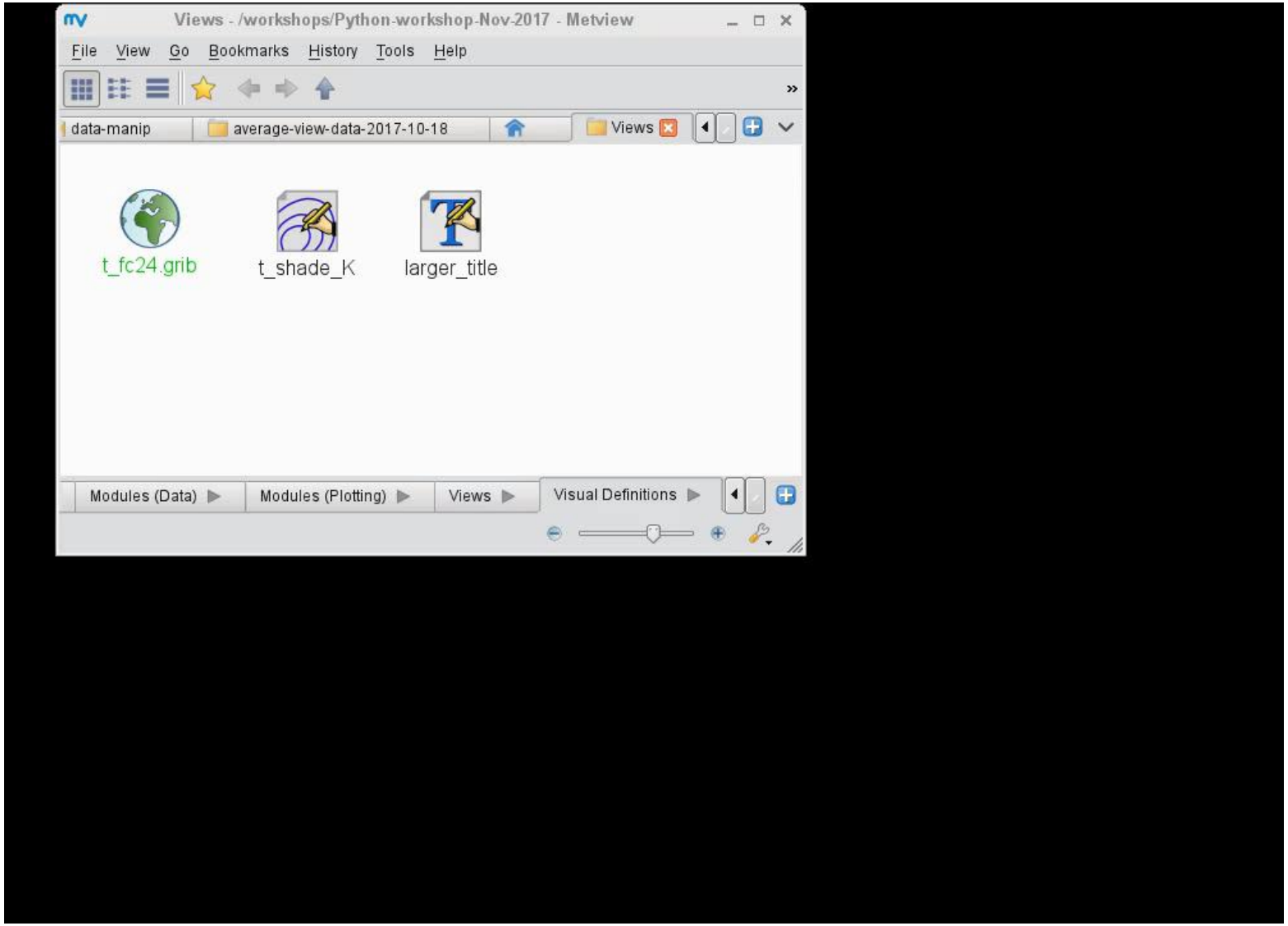




```
# retrieve some data

f1 = retrieve (date : -1, levels : 1000, grid : [1.5, 1.5])
f2 = retrieve (date : -2, levels : 1000, grid : [1.5, 1.5])


# perform some calculations for comparison

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
```

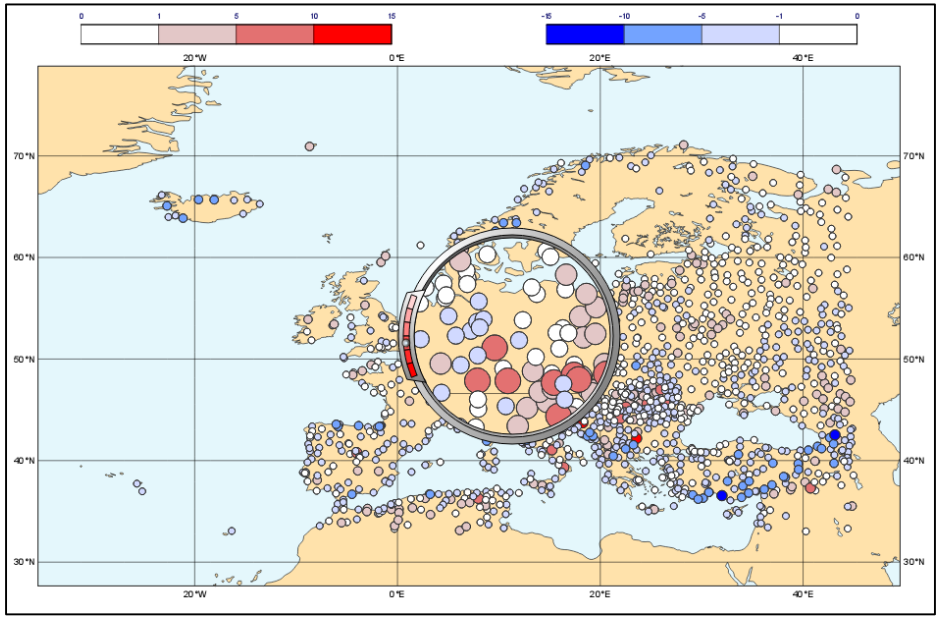
Metview Recorded Demonstration



High-level data processing with Metview Macro

Forecast – observation difference:

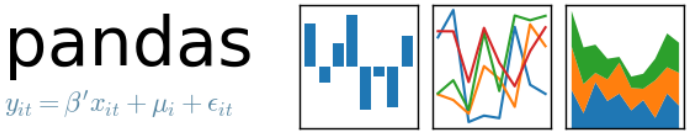
```
forecast = retrieve (…) # GRIB from MARS
obs      = retrieve (…) # BUFR from MARS
t2m_obs  = obsfilter(data:      obs,
                     parameter: 012004,
                     output:    'geopoints')
plot(forecast – t2m_obs)
```



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



- Enable Metview to work seamlessly within the Python eco-system
  - Bring Metview's data processing and interactive data inspection tools into Python sessions
  - Use existing solutions where possible (e.g. for multi-dimensional data arrays, data models)

- Should be able to interact with the Copernicus Climate Data Store Toolbox (in development)

Current Status (1)

- Working prototype (developed with B-Open)
- Python code very similar to Macro!
- Still need to finalise packaging and name

Macro

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

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

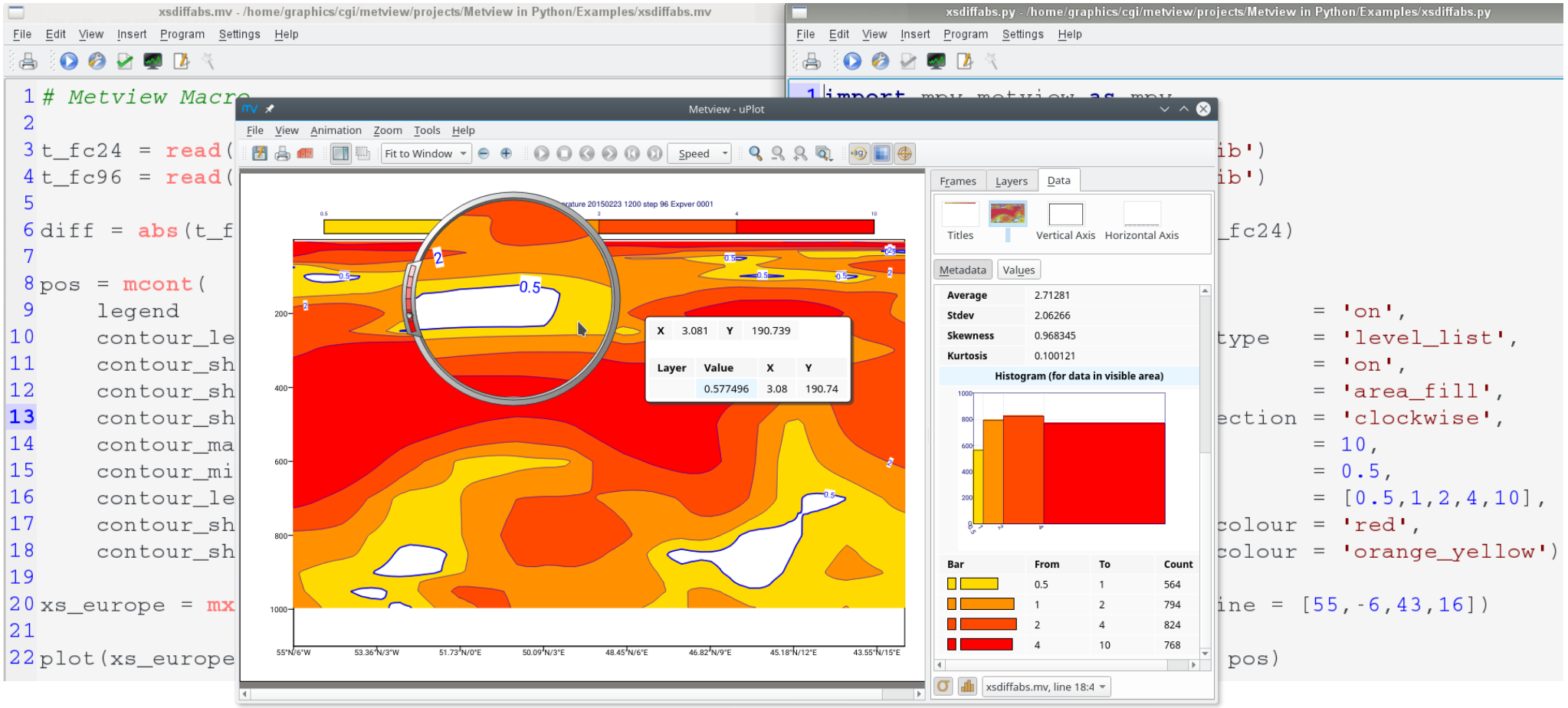
xsdiffabs.py - /home/graphics/cgi/metview/projects/Metview in Python/Examples/xsdiffabs.py

File Edit View Insert Program Settings Help

```
import mpy.metview as mpy

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

absdiff = mpy.abs(t_fc96 - t_fc24)

pos = mpy.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 = mpy.mxsectview(line = [55,-6,43,16])

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

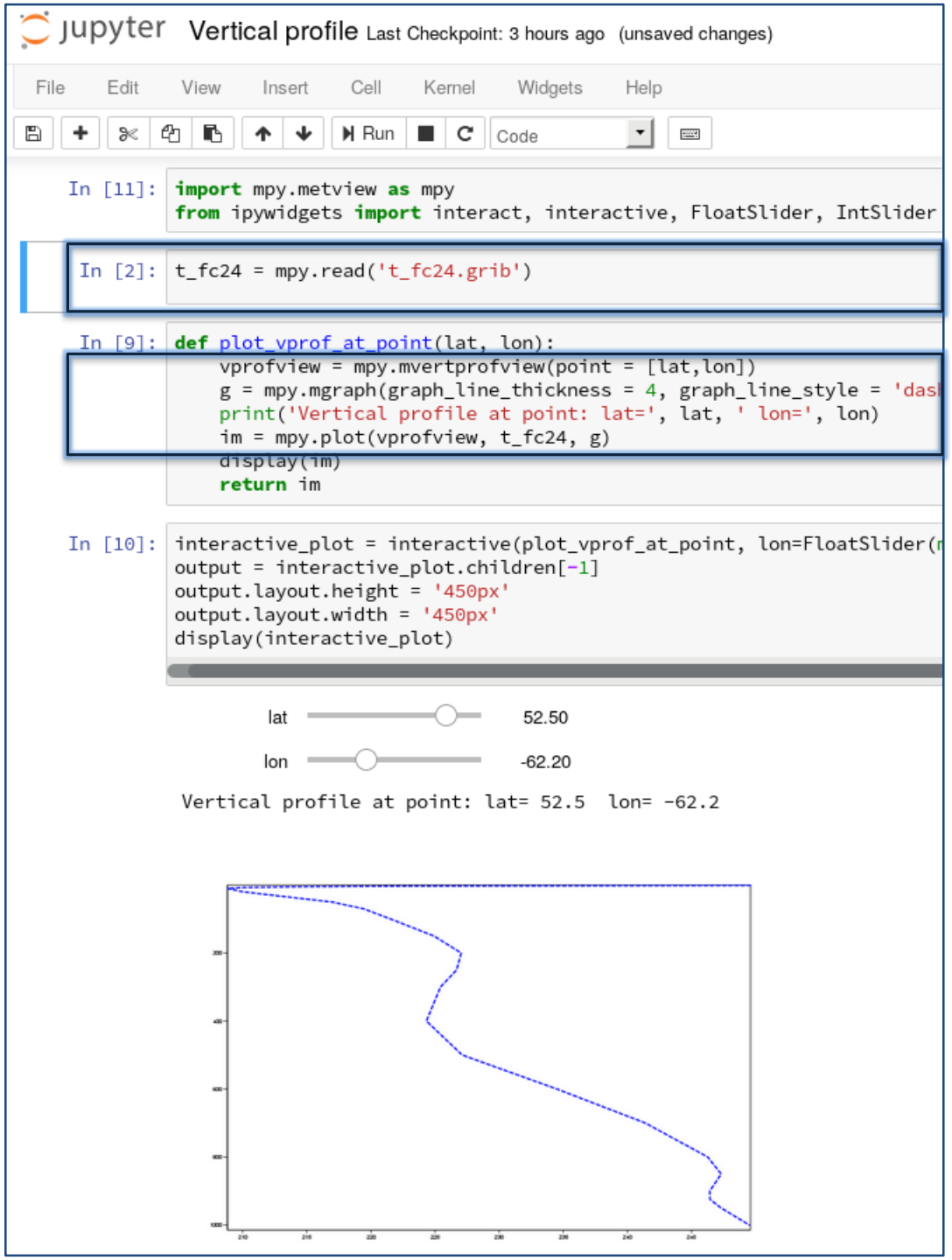
Current Status (1)

- Working prototype (developed with B-Open)
- Python code very similar to Macro!
- Still need to finalise packaging and name



Current status (2)

- Can run purely in batch (write data or plots into files), invoke Metview's interactive windows and can plot to Jupyter notebooks
- Requires "Python-ready" version of Metview to be installed, plus the Python package
  - Not widely installed at ECMWF yet
  - Metview 5.0 (beta to be released this December) will be the minimum version that can connect to Python
- Thin interface between Metview and Python
  - The Python layer should be fairly independent of the underlying Metview version



Metview and Pandas

Working prototype example

```
In [7]: import mpy.metview as mpy
        import numpy as np
        from scipy import stats

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

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

        diff = t2m_grib - t2m_gpt

        df = diff.to_dataframe()

        outliers = np.abs(stats.zscore(df)) > 1.5
        print('Num outliers: ', outliers.sum())

        Num outliers:  7
```

Metview

Scipy / numpy

Metview and xarray

Work in progress, but this is what we want

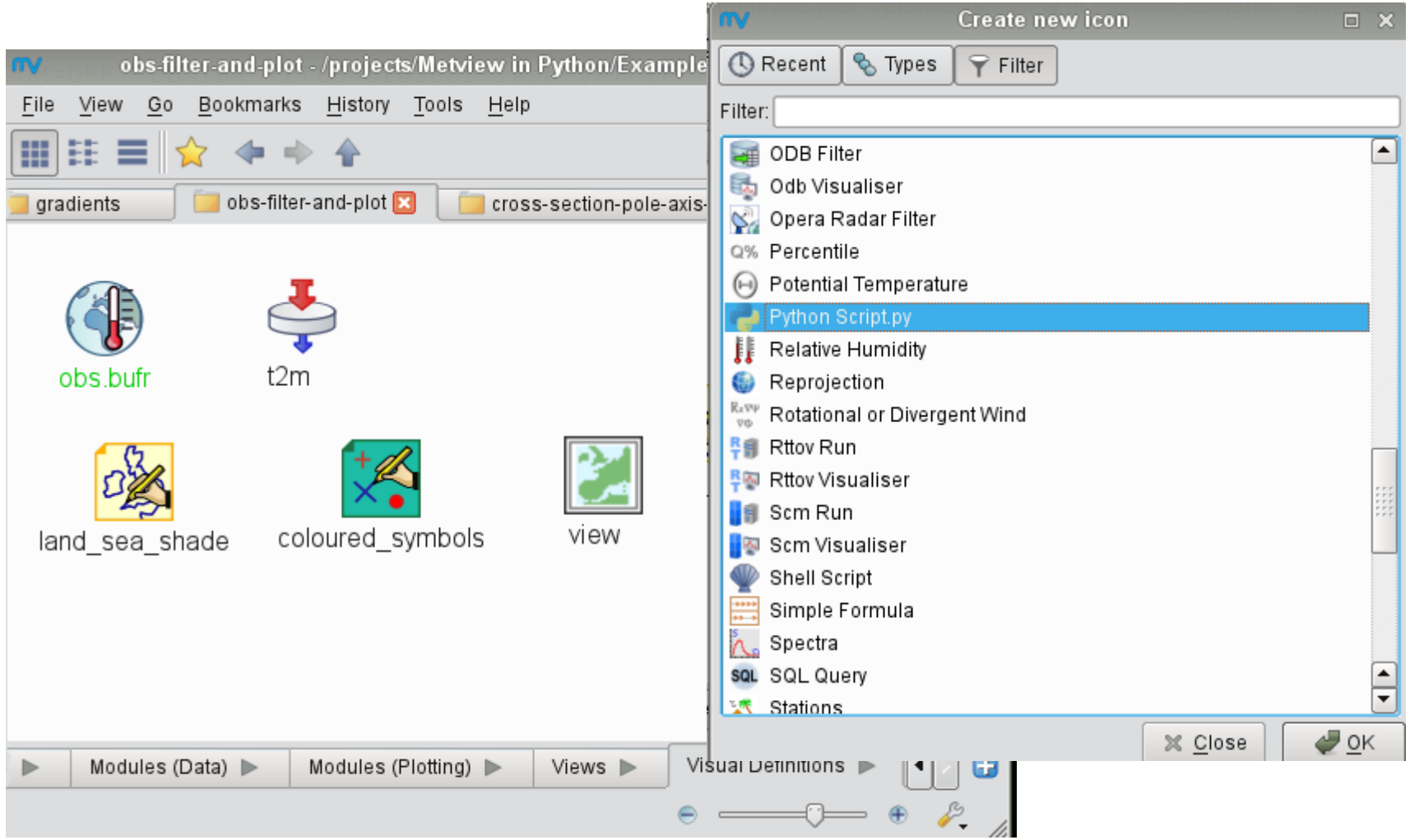
```
1  import mpy.metview as mpy
2  import xarray as xr
3
4  # use Metview to retrieve data from MARS
5  # ensemble members for various vertical levels
6  ens = mpy.retrieve(
7      type      = "pf",
8      stream    = "ef",
9      levelist  = [1000,850,500,300,100],
10     param     = "t",
11     date      = -2,
12     number    = "all",
13     grid      = [2,2])
14
15 # use xarray to compute the ensemble mean for each level
16 ens_xr = ens.as_xarray()
17 ens_mean_xr = ens_xr.mean(dim='number')
18
19 # use Metview to compute a vertical cross section of the result
20 xs_data = mpy.mcross_sect(
21         line = [59.9,-180,-13.5,158.08],
22         data = ens_mean_xr)
23
24 # plot, and write the data to file
25 mpy.plot(xs_data)
26 mpy.write('xs.nc', xs_data)
```

Metview

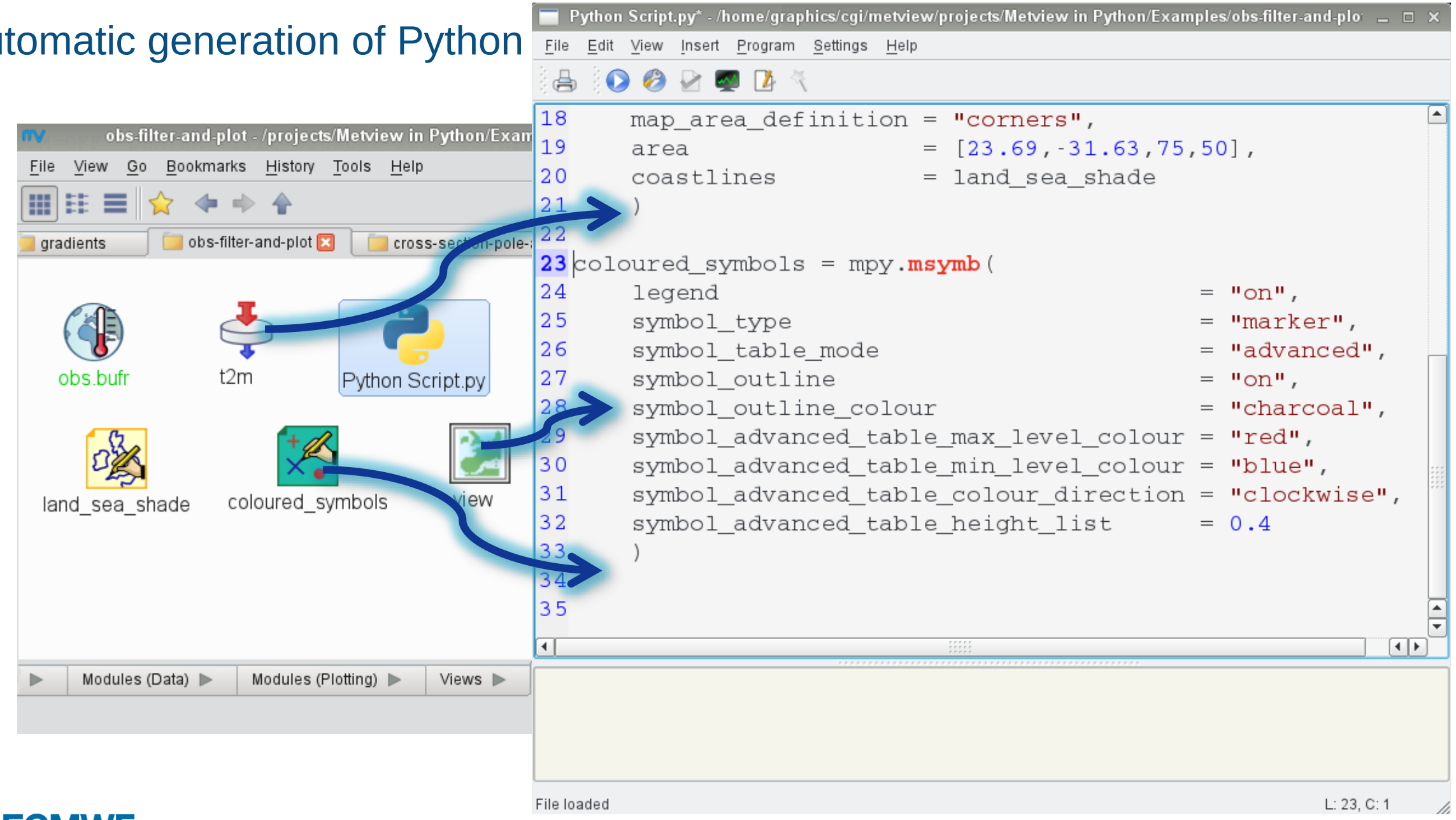
xarray

Metview

Automatic generation of Python code from Metview



Automatic generation of Python



Python Script.py* - /home/graphics/cgi/metview/projects/Metview in Python/Examples/ob

File Edit View Insert Program Settings Help

```
18     map_area_definition = "corners",
19     area                = [23.69,-31.63,75,50],
20     coastlines          = land_sea_shade
21     )
22
23 coloured_symbols = mpy.msymb(
24     legend                                  = "
25     symbol_type                             = "
26     symbol_table_mode                       = "
27     symbol_outline                          = "
28     symbol_outline_colour                   = "
29     symbol_advanced_table_max_level_colour  = "red",
30     symbol_advanced_table_min_level_colour  = "blue",
31     symbol_advanced_table_colour_direction  = "clockwise",
32     symbol_advanced_table_height_list       = 0.4
33     )
34
35 plot(view, t2m, coloured_symbols)
```

File loaded

L: 35, C: 34

Metview - uPlot

File View Animation Zoom Tools Help

Fit to Window | Speed

Frames | Layers | Data

Titles | land_sea_shade | t2m | land_sea

Metadata | Values

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

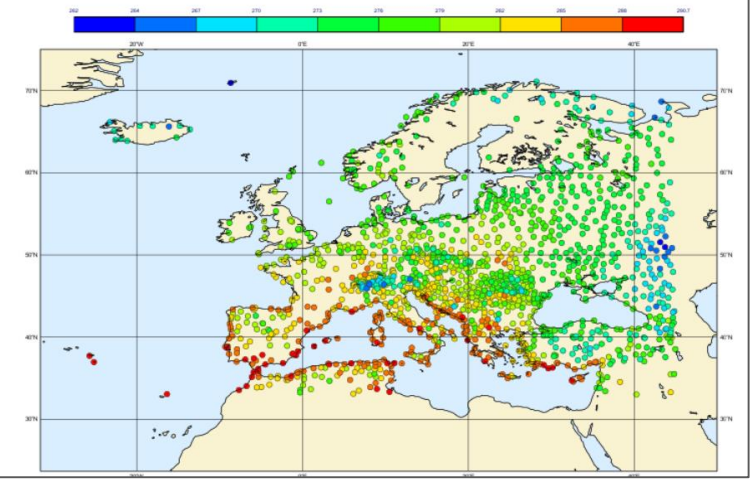
Points	1627
Minimum	262
Maximum	290.7
Average	278.714
Stdev	4.99917
Skewness	-0.149101
Kurtosis	-0.194182

Histogram (for data in visibl

coloured_symbols

Run | Code

```
mpy.plot(view, t2m, coloured_symbols)
Out[4]:
```



Timeline

- Dec 2017: internal alpha release
  - Installed at ECMWF, beta xarray GRIB driver, small gallery of examples and basic documentation
- March 2018: internal (ECMWF) beta release
  - Ready for further user testing, netCDF/GRIB harmonisation strategy
- June 2018: external beta release
  - Performance, feature completeness, test suite, documentation
- December 2018: production release

For more information…

- Email us:
  - metview@ecmwf.int
- Visit our web pages:
  - http://software.ecmwf.int/metview
- Download (Metview source, binaries, virtual machine)
- Documentation and tutorials available
- Metview articles in ECMWF newsletters
- e-Learning material