# Python and the Analysis of Atmospheric and Oceanic Data Sets

**Jon Sáenz, Juan Zubillaga and Jesús Fernández**

eman ta zabal zazu

**6–7th March 2001**

Postal Address:
Departamento de Física Aplicada II
Universidad del País Vasco
Apdo. 644
48080–Bilbao
SPAIN

jsaenz@wm.lc.ehu.es
juan@zubi.net
chus@wm.lc.ehu.es

Thanks to CICYT Grant# CLI98–0236, UPV Grant# UPV060.310-G16/98 and Euskal Meteorologi Zerbitzua

# Introduction

- Recent concern about the anthropogenic nature of climate change has motivated an increase in the analysis of atmospheric and oceanic data sets.
- The techniques rely heavily on matrix operations and eigenvalue techniques.
- Data are distributed using extremely heterogeneous formats. Access to external libraries or system calls is unavoidable.
- There are different scales of motion (hours $\rightarrow$ years) involved in the physics.
- Data sets are huge.

`pyclimate` (GPL license) has been developed using Python and C to perform these analysis tasks.
`pyclimate`, `Python` and `NumPy` are currently a extremely valuable resource in our data–analysis tasks.

# Handling COARDS–compliant netCDF files

Several attributes and variables needed to define a COARDS–compliant netCDF file:

```
netcdf NCARreanal200 {
dimensions:
  time = UNLIMITED ; // (1 currently)
  lat = 33 ;
  lon = 49 ;
variables:
  double time(time) ;
    time:long_name = "time" ;
    time:units = "hours since 1992-1-10 0:0:0" ;
  float lat(lat) ;
    lat:long_name = "latitude" ;
    lat:units = "degrees_north" ;
  float lon(lon) ;
    lon:long_name = "longitude" ;
    lon:units = "degrees_east" ;
  float hgt(time, lat, lon) ;
    hgt:long_name = "Geopotencial height at 200mb" ;
    hgt:units = "m" ;

// global attributes:
    :title = "NMC/NCAR Reanalysis hgt at 200mb" ;
    :history = "Created by bin2nc.py" ;
}
```

To replicate its structure with pyclimate :

```
from Scientific.IO.NetCDF import *
from pyclimate.ncstruct import
dims=("time","lat","lon")
vars=dims+("hgt",)
onc=nccopystruct("out.nc",NetCDFFile("NCARreanal200.nc"),
  dims,vars,dims[1:])
```

# Access to DCDFLIB.C

DCDFLIB.C is a publicly available library which provides functions to perform direct and inverse computations of parameters related to several statistical distribution functions ($\chi^2$, $t$, $F$,...)

`pyclimate` provides access from `Python` to the functions in the library through an intermediate layer which leaves the original library untouched (see prototype for $\chi^2$):

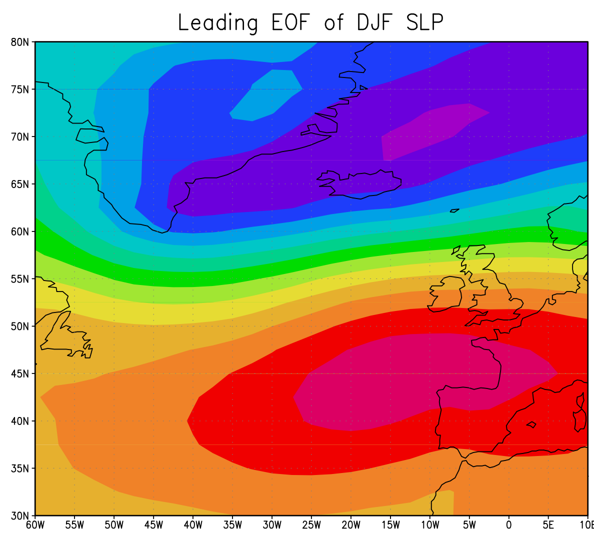
```
void cdfchi(int *which,double *p,
double *q,double *x,double *df,
int *status,double *bound);
```

The functions are called from `Python` using classes:

```
from pyclimate.pydcdflib import *
chi2=CDFChi()
chi2.which=2 # Get the value of x
chi2.p=0.975 # associated with this prob.
chi2.df=25 # Degrees of freedom
pycdfchi(chi2)
print "%9.5f(%1d)"%(chi2.x,chi2.status),
```

# EOF and SVD–based analysis

Consider data sets $X = [x_{ij}]$ $(N \times M)$ and $Y = [y_{ij}]$ $(N \times P)$ with $i = 1 \ldots N$ meaning temporal samples and $j = 1 \ldots M$ and $j = 1 \ldots P$ grid points or observing sites. There are functions to compute the eigenvectors, eigenvalues, temporal expansion coefficients and correlation maps derived from the EOF and SVD analysis.



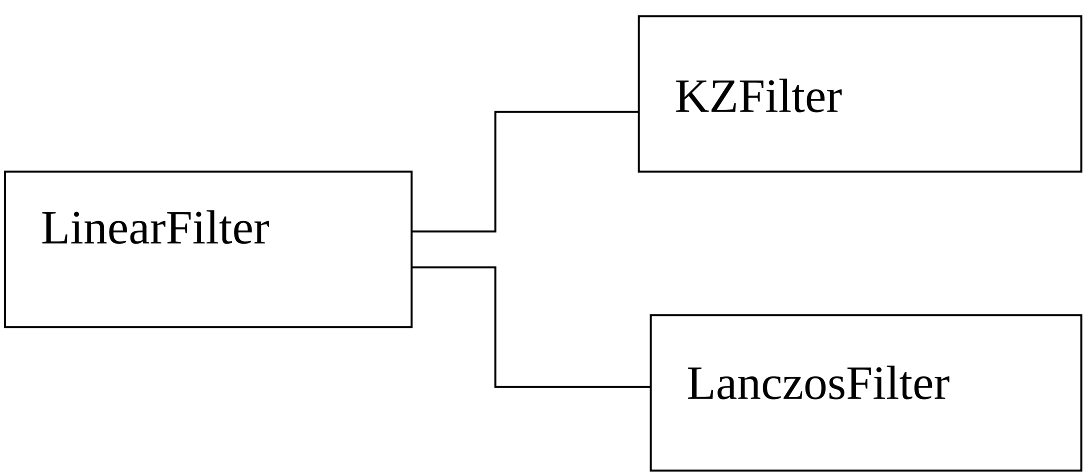

```
from Numeric import *
from Scientific.IO.NetCDF import *
from pyclimate.svdeofs import *

inc = NetCDFFile("ncepslp.djf.nc")
slpdata = inc.variables["djfslp"][:]
theshape = slpdata.shape
slpdata.shape = (theshape[0], theshape[1]*theshape[2]*theshape[3])
pcs, lambdas, eofs = svdeofs(slpdata)
```

# Linear filters for huge data sets

Given an arbitrarily shaped input data set $X_t$, a general linear filter involves the operation:

$$Y_t = \sum_{i=-n}^{n} a_i X_{t+i}.$$

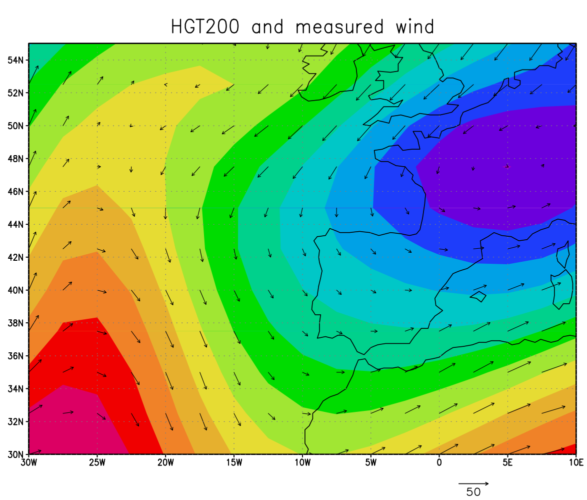
This basic operation has been implemented as a base class which derives different kinds of filters by means of the constructor, which simply computes different values for the $a_k$ coefficients. Thus, the basic filter can be extended easily to other methods simply by redefining these coefficients. This strategy has been used to define Kolmogorov–Zurbenko and Lanczos filters.

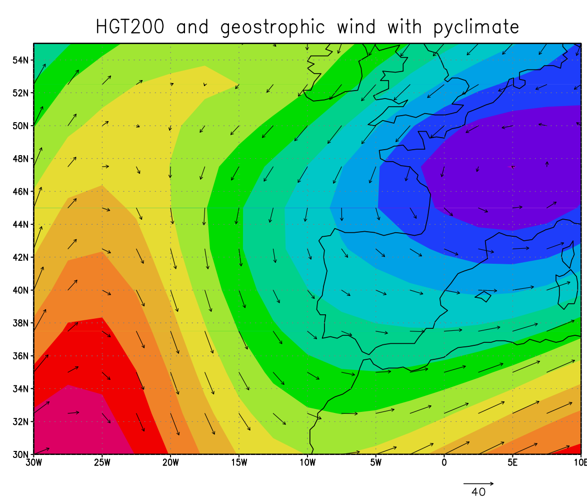


# Differential operators over the sphere

They work over regular longitude$\times$latitude non–periodic grids arranged as in the `COARDS` conventions.

- The horizontal component of the gradient of a scalar (i.e. geostrophic wind)
- The divergence of a two–dimensional vector field (i.e. the balance Evaporation minus Precipitation from the vertically integrated moisture transport)
- The vertical component of the curl of a vector field (i.e. vertical velocity at the bottom of the oceanic Ekman layer due to the wind stress)



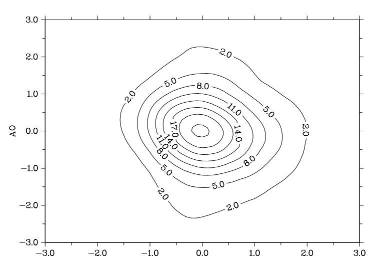
# Kernel–based PDF estimation

`pyclimate` provides functions to perform univariate and multivariate kernel–based Probability Density Function (PDF) estimation.

$$f(x) = \frac{1}{nh} \sum_{1=1}^{n} K\left(\frac{x - X_i}{h}\right)$$

$$f(\mathbf{x}) = \frac{\det S^{-\frac{1}{2}}}{nh^d} \sum_{1=1}^{n} k\left(h^{-2}(\mathbf{x} - \mathbf{X}_i)^T S^{-1}(\mathbf{x} - \mathbf{X}_i)\right)$$

Example: 2-D PDF of El Niño (CTI) and the Arctic Oscillation (AO):




```
from pyclimate.KPDF import *
from Numeric import *
from pyclimate.readdat import *
# Read the experimental data
ctiao=readdat("ctiao.dat")
# Define a grid to evaluate the PDF
X=arange(-5,5.01,0.05)
Y=arange(-4,4.01,.05)
# Get the 2D grid as a linear array of nodes
XY=MPDF2DGrid2Array(X,Y)
# Evaluate the PDF at every point of the grid
mpdf=MPDFEpanechnikov(ctiao,XY,1.5)
```

# Conclusions

- `Python` and `NumPy` are valuable tools to perform data analysis of massive atmospheric and occeanic data sets.
- A `Python` package ( `pyclimate` ) developed entirely using `NumPy` and C with extensions devoted to the analysis of atmospheric and oceanic data has been presented.
- Some of its features (DCDFLIB.C, KPDF or Filters) can also be used for general data analysis.
- `pyclimate` is distributed under the GPL public license, contributions and suggestions are welcome.

**LINKS:**

`http://starship.python.net/crew/jsaenz`

`http://lcdx00.wm.lc.ehu.es/~jsaenz`