AULA 6

Input/output de dados ascii.

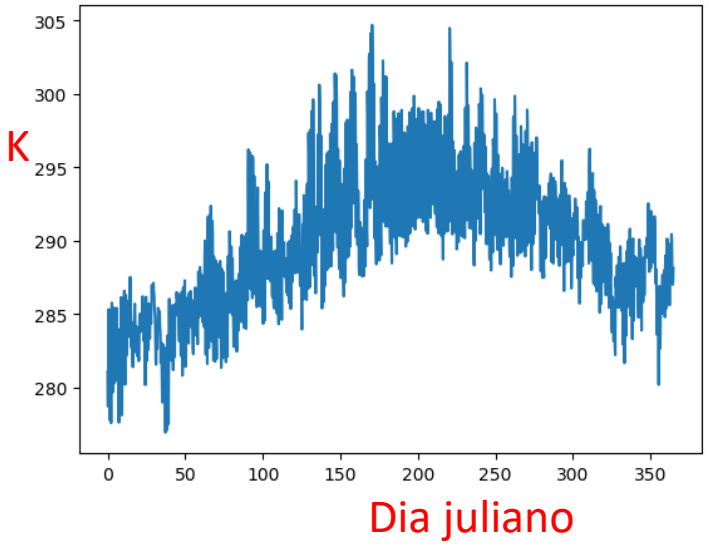
Interpolação 2D. griddata

Leitura de dados em numpy

Exemplo:

Temperatura analisada na região de Lisboa em 2015

Dados de 6 em 6 horas

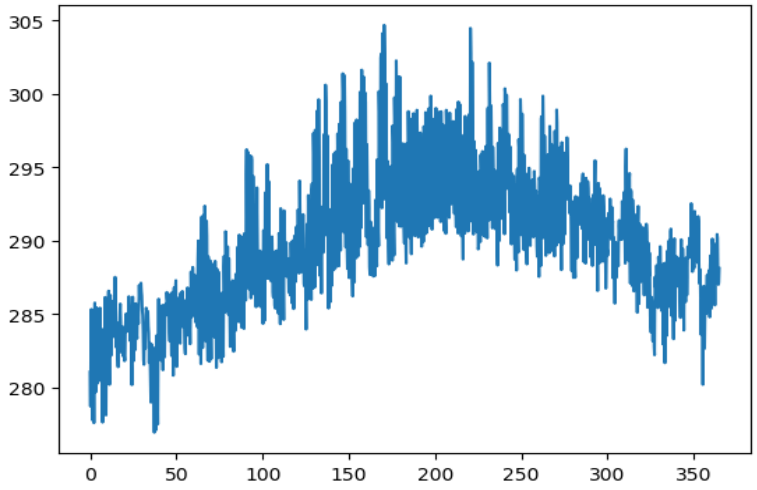


			Horas	Temperatura
2015	1	1 0	0	281.08
2015	1	1 6	6	278.73
2015	1	1 12	12	283.69
2015	1	1 18	18	285.34
2015	1	2 0	24	281.53
2015	1	2 6	30	277.81
2015	1	2 12	36	282.82
2015	1	2 18	42	285.29
2015	1	3 0	48	281.27
2015	1	3 6	54	277.60
2015	1	3 12	60	283.13
2015	1	3 18	66	285.78
2015	1	4 0	72	281.50
2015	1	4 6	78	279.70
2015	1	4 12	84	283.73
2015	1	4 18	90	285.46
2015	1	5 0	96	281.69
2015	1	5 6	102	280.29
2015	1	5 12	108	282.93
2015	1	5 18	114	283.26
2015	1	6 0	120	281.99
2015	1	6 6	126	282.10
2015	1	6 12	132	285.17
2015	1	6 18	138	285.43

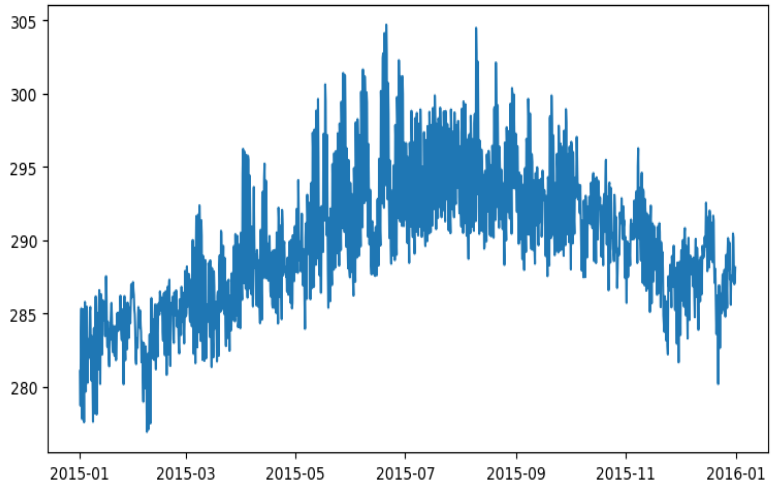
Contagem do tempo

A contagem do tempo é mais complicada do que parece à primeira vista.
Os meses têm tamanho variável. Os anos têm número de dias variável. Hora legal depende do local. Fica para depois…

Tempo corrido



Tempo calendário

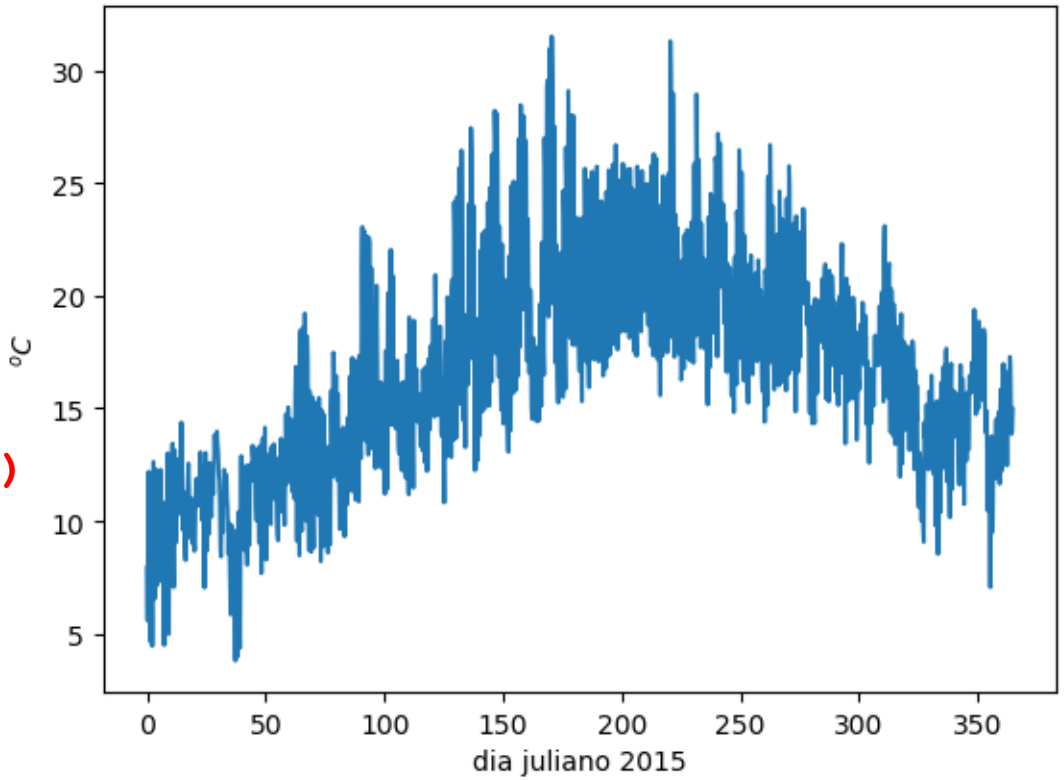


Leitura simples em numpy (ascii): em bloco

```
import numpy as np
import matplotlib.pyplot as plt
dados=np.loadtxt('Tlis.txt')
horas=dados[:,4]
T=dados[:,5]
plt.close('all')
plt.plot(horas/24,T-273.15)
plt.xlabel('dia juliano 2015')
plt.ylabel(r'$^oC $')
```

2015	1	1	0	0	281.08
2015	1	1	6	6	278.73
2015	1	1	12	12	283.69
2015	1	1	18	18	285.34
2015	1	2	0	24	281.53
2015	1	2	6	30	277.81
2015	1	2	12	36	282.82
2015	1	2	18	42	285.29
2015	1	3	0	48	281.27
2015	1	3	6	54	277.60
2015	1	3	12	60	283.13
2015	1	3	18	66	285.78
2015	1	4	0	72	281.50
2015	1	4	6	78	279.70
2015	1	4	12	84	283.73
2015	1	4	18	90	285.46
2015	1	5	0	96	281.69
2015	1	5	6	102	280.29
2015	1	5	12	108	282.93
2015	1	5	18	114	283.26
2015	1	6	0	120	281.99
2015	1	6	6	126	282.10
2015	1	6	12	132	285.17
2015	1	6	18	138	285.43

```
plt.ylabel(r'$^oC $')
```

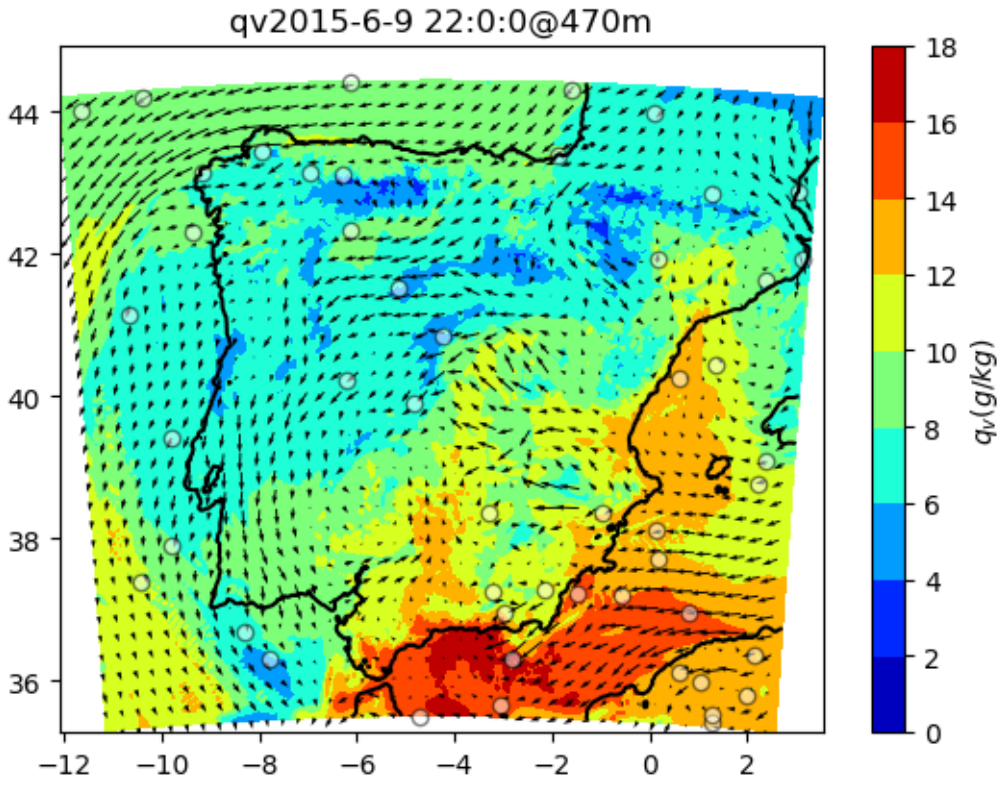


Caso 2D ou nD

Neste caso os dados não são ordenáveis.

Os métodos são por isso, em geral, mais complexos.

Dados 2D: Humidade específica (g/kg)



50 pontos

-0.8336	42.3440	4.1490
-11.4589	39.6431	7.7394
-1.9566	39.5920	10.7605
-3.2849	44.1691	8.4349
-10.4623	37.1318	10.6476
0.4476	38.8952	12.4602
-7.1906	43.3876	8.4591
3.0062	41.3846	8.7632
0.5424	37.2976	13.6643
-0.6014	40.6630	7.4095
-6.7003	38.0017	9.8708
-1.4290	42.0902	5.8712
-8.9528	41.5527	6.3083
-5.9784	39.3632	5.7522
2.5176	37.3554	12.3149
-9.2159	36.7341	7.4718
-1.2014	36.2284	15.2414
-10.7569	43.7935	9.8617
-0.1344	41.8624	6.7847
-3.1041	40.0421	9.5298
-7.5967	39.6263	6.6690
-9.9726	39.9410	6.3462
-5.1361	38.9402	8.1630
-8.9420	35.6134	11.5646

Interpolação 2D simples

N observações (xOBS,yOBS,OBS), cálculo do valor num ponto genérico (x, y, v):

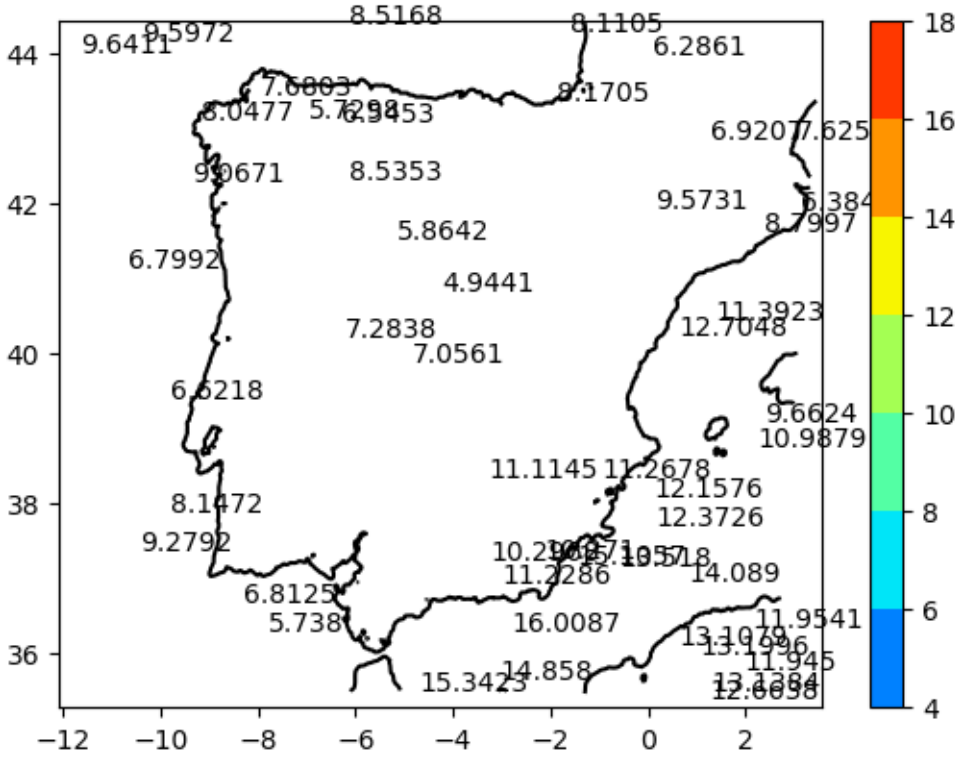
$$v(x,y) = \sum_{k=1}^{N} w_k OBS_k$$

$$w_k = \frac{d_k}{\sum_{k=1}^{N} d_k}$$

$$d_k = \frac{1}{\left((x - x_k)^2 + (y - y_k)^2\right)^m}$$

m=1: inverso do quadrado da distância

Verificação

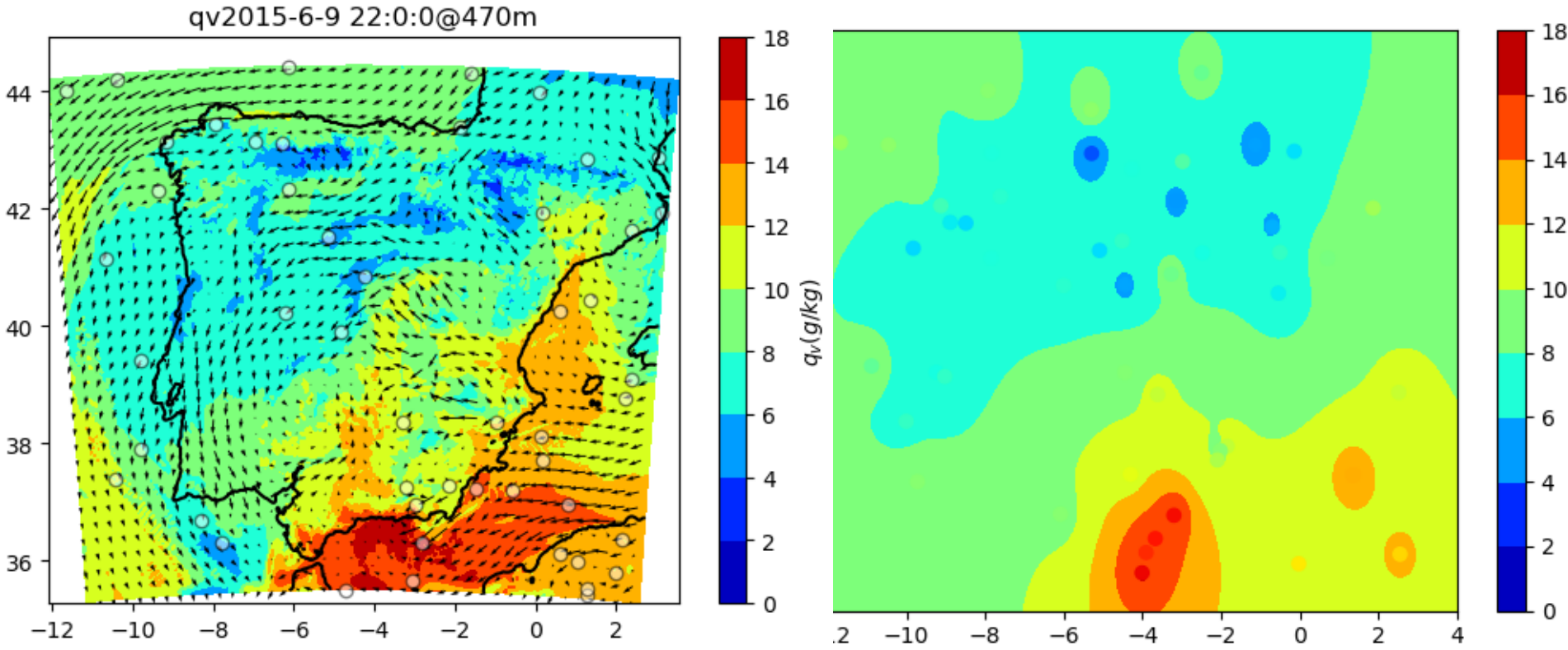



```
plt.figure()
dados=np.loadtxt('qvsample.txt')
long=dados[:,0];lati=dados[:,1];qv=dados[:,2]
for k in range(len(qv)):
    plt.text(long[k],lati[k],str(qv[k]))
```

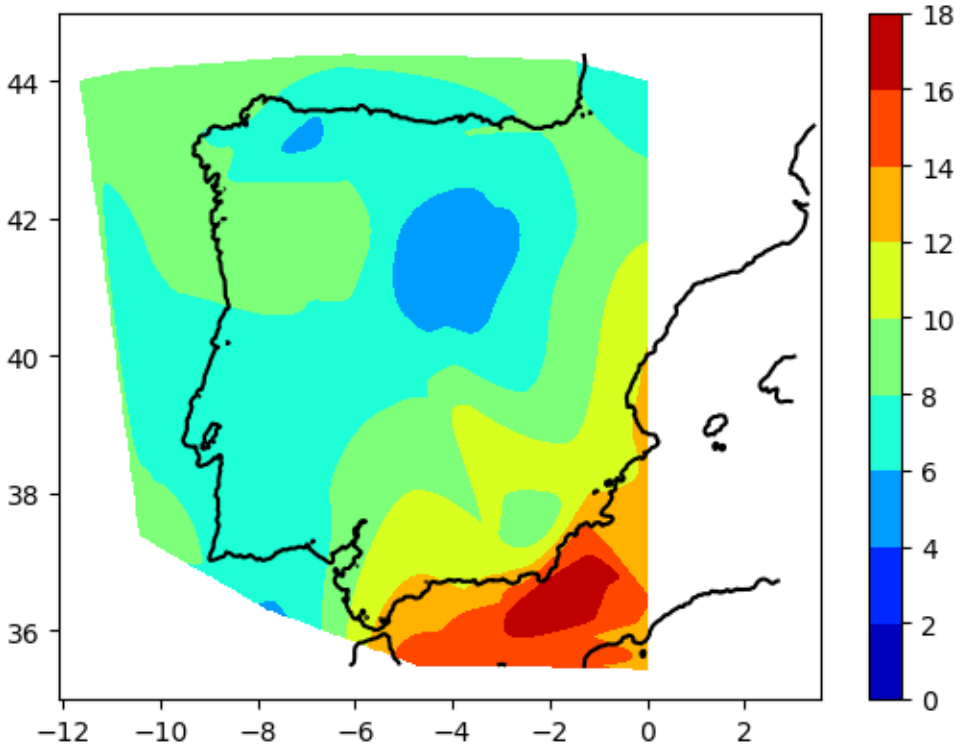
Inverso do quadrado da distância

```
import numpy as np
import matplotlib.pyplot as plt
OBS=np.loadtxt('qvIB.txt')
lonOBS=OBS[:,0];latOBS=OBS[:,1];qvOBS=OBS[:,2]
plt.close('all')
power=1.
lon1d=np.linspace(-12,4,121)
lat1d=np.linspace(35,45,151)
nlon=len(lon1d);nlat=len(lat1d)
qvINT=np.zeros((nlat,nlon),dtype=float)
for klon in range(nlon):
    for klat in range(nlat):
        pesos=1./((lon1d[klon]-lonOBS[:])**2+(lat1d[klat]-latOBS[:])**2)**power
        pesoTot=np.sum(pesos)
        pesos=pesos/pesoTot
        qvINT[klat,klon]=np.sum(pesos*qvOBS)
#lon,lat=np.meshgrid(lon1d,lat1d)
lev=np.linspace(0,18,10)
map=plt.contourf(lon1d,lat1d,qvINT,cmap='jet',levels=lev,vmin=0,vmax=18)
plt.colorbar(map)
map2=plt.scatter(lonOBS,latOBS,c=qvOBS,cmap='jet',vmin=0,vmax=18)
```

$$\frac{1}{d^2}, 121 \times 151$$



Interpolação 2D




```
import numpy as np
import matplotlib.pyplot as plt
from scipy.interpolate \
     import griddata
plt.figure()
dados=np.loadtxt('qvsample.txt')
long=dados[:,0];
lati=dados[:,1];qv=dados[:,2]
lonReg=np.linspace(-12,0,401)
latReg=np.linspace(35,45,401)
lonMesh,latMesh=np.meshgrid(lonReg,latReg)
lev=np.linspace(0,18,10)
qvI=griddata((long,lati),qv,(lonMesh,latMesh),method='cubic')
map=plt.contourf(lonReg,latReg,qvI,cmap='jet',levels=lev)
plt.colorbar(map)
plt.contour(lon,lat,var[:,:,9],colors='black',levels=[10])
```

`lonMesh,latMesh=np.meshgrid(lonReg,latReg)`

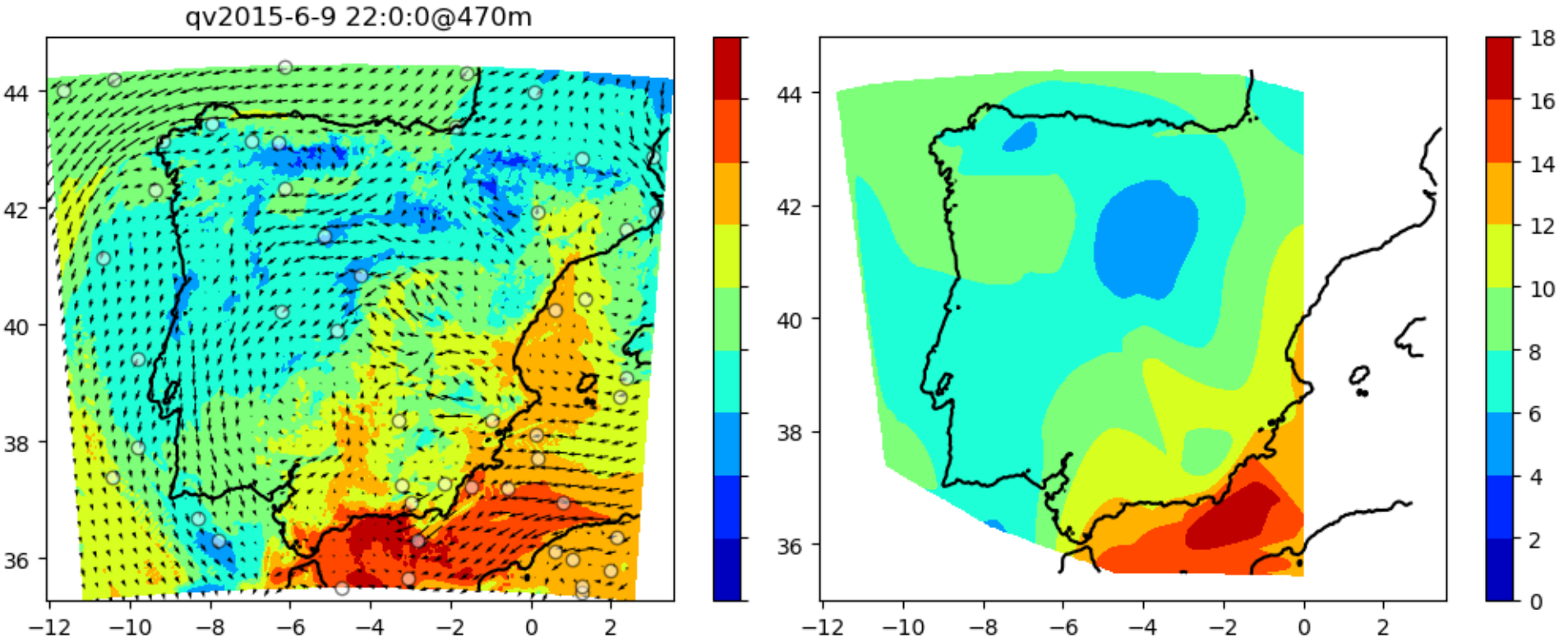
Converte os vectores lonReg[0:nx], latReg[0:ny] em matizes da mesma forma lonMesh[0:ny,0:nx], latMesh[0:ny,0:nx]

por replicação na direção perpendicular

Exemplo:

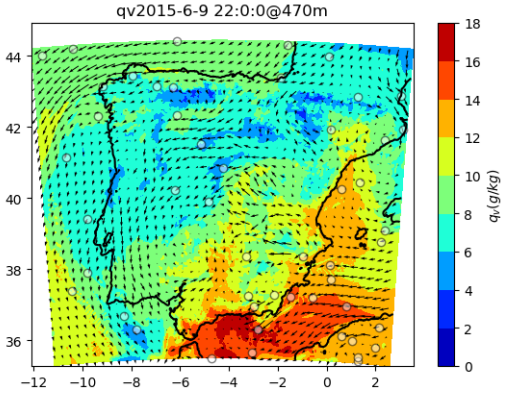
$$lonMESH = \begin{bmatrix} lon_0 & \dots & lon_{nx} \\ & \dots & \\ lon_0 & \dots & lon_{nx} \end{bmatrix} \text{ latMESH=} \begin{bmatrix} lat_0 & lat_0 \\ \dots & \vdots \quad \dots \\ lat_{ny} & lat_{ny} \end{bmatrix}$$

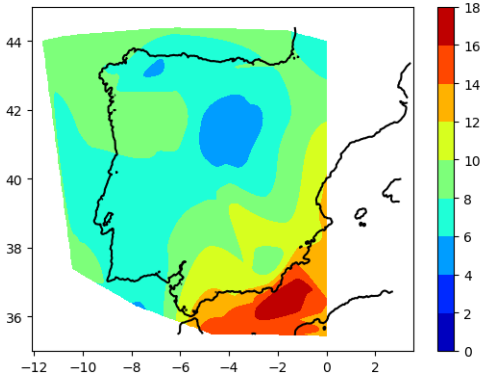
50 pontos (cubic)



griddata(method=)

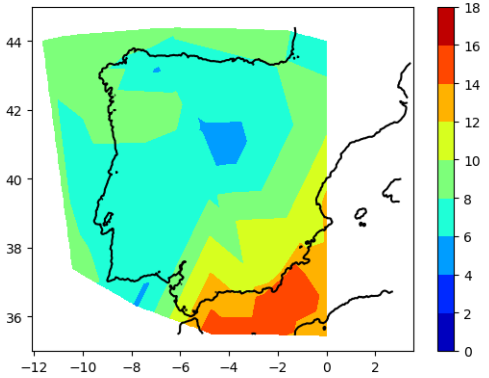
Original
417x333

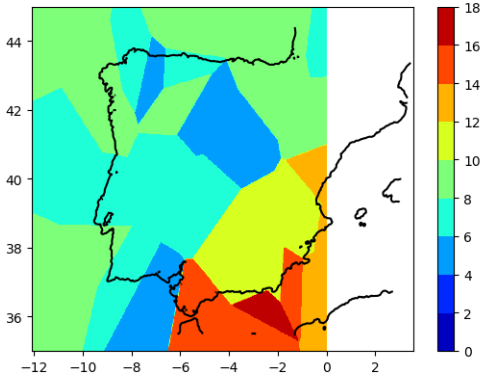




cubic

linear

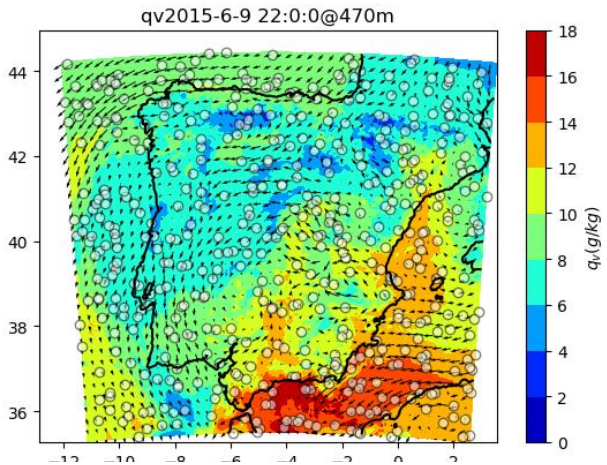


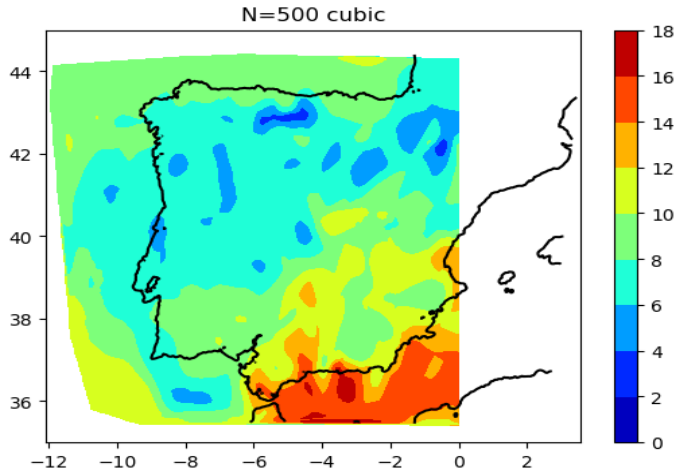


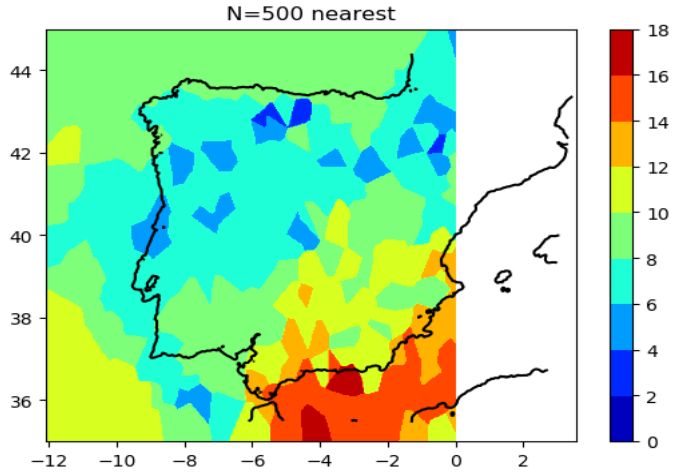
nearest

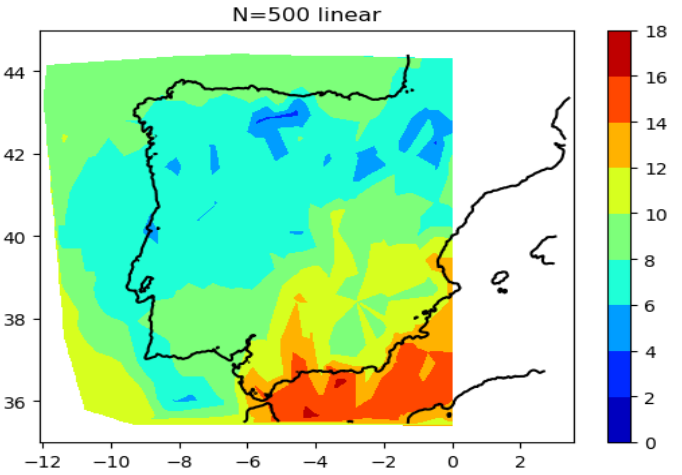
N=500

Original
417x333









Escrita de ficheiro ascii, amostrando var[:,:] aleatoriamente

```
nA=500
xis=np.zeros((nA));yps=np.zeros((nA))
zed=np.zeros((nA))
for k in range(nA):
    ix=np.int((np.random.rand())*nx)
    iy=np.int((np.random.rand())*ny)
    xis[k]=lon[ix,iy]
    yps[k]=lat[ix,iy]
    zed[k]=var[ix,iy]
Dados=np.transpose(np.array([xis,yps,zed]))
np.savetxt('qvsample.txt',Dados,\
           fmt='%8.4f %8.4f %8.4f')
```

`np.random.rand(n)`

gera um vetor de n números aleatórios em [0,1]

`np.random.rand()*n`

gera um número aleatório no interval [0,n]