MeteoCH.output

March 2, 2023

1 Meteoschweiz

1.1 Current meteorological observations

```
[1]: # 'Soft' reset: Only clears your namespace, leaving history intact.
     %reset -sf
     import pandas as pd
     from datetime import datetime
     import matplotlib.cbook
```

1.2 Available weather stations

```
[2]: url = 'https://data.geo.admin.ch'
     path = 'ch.meteoschweiz.klima/nbcn-tageswerte'
     wsurl = url + '/' + path + '/' + 'liste-download-nbcn-d.csv'
     ws = pd.read_csv(wsurl, sep=";", header=0, encoding = "ISO-8859-1").dropna()
     ws.drop(['URL Previous years (verified data)', 'URL Current year'], axis=1)
```

```
[2]:              Station station/location        WIGOS-ID  Data since  \
     0             Altdorf               ALT  0-20000-0-06672  01.01.1864
     1             Andermatt             ANT  0-20000-0-06695  01.01.1864
     2       Basel / Binningen           BAS  0-20000-0-06601  01.01.1755
     3       Bern / Zollikofen           BER  0-20000-0-06631  01.01.1864
     4       La Chaux-de-Fonds           CDF  0-20000-0-06612  01.01.1900
     5          Château-d'Oex            CHD  0-20000-0-06627  01.01.1879
     6             Chaumont              CHM  0-20000-0-06608  01.01.1864
     7             Davos                 DAV  0-20000-0-06784  01.01.1864
     8             Elm                   ELM  0-20000-0-06682  01.02.1878
     9             Engelberg             ENG  0-20000-0-06655  01.01.1864
     10            Grächen               GRC  0-20000-0-06728  01.01.1864
     11        Grimsel Hospiz            GRH  0-20000-0-06744  01.01.1932
     12  Col du Grand St-Bernard         GSB  0-20000-0-06717  01.01.1818
     13       Genève / Cointrin          GVE  0-20000-0-06700  01.01.1753
     14            Jungfraujoch          JUN  0-20000-0-06730  01.01.1933
     15            Lugano                LUG  0-20000-0-06770  01.01.1864
     16            Luzern                LUZ  0-20000-0-06650  01.01.1864
     17            Meiringen             MER  0-20000-0-06637  01.07.1889
```

```
18                Neuchâtel          NEU  0-20000-0-06604  01.01.1864
19          Locarno / Monti          OTL  0-20000-0-06760  01.12.1882
20                  Payerne          PAY  0-20000-0-06610  01.08.1964
21                Bad Ragaz          RAG  0-20000-0-06686  01.06.1870
22                   Säntis          SAE  0-20000-0-06680  01.01.1864
23                  Samedan          SAM  0-20000-0-06792  01.01.1864
24            S. Bernardino          SBE  0-20000-0-06783  01.01.1864
25              Segl-Maria          SIA  0-20000-0-06779  01.12.1863
26                     Sion          SIO  0-20000-0-06720  01.01.1864
27        Zürich / Fluntern          SMA  0-20000-0-06660  01.01.1864
28               St. Gallen          STG  0-20000-0-06681  01.01.1864

     Station height m. a. sea level  CoordinatesE  CoordinatesN   Latitude  \
0                            438.0     2690181.0     1193564.0  46.887069
1                           1438.0     2687445.0     1165044.0  46.630914
2                            316.0     2610909.0     1265612.0  47.541142
3                            553.0     2601934.0     1204410.0  46.990744
4                           1017.0     2550919.0     1214862.0  47.082947
5                           1028.0     2577040.0     1147655.0  46.479819
6                           1136.0     2565060.0     1211007.0  47.049169
7                           1594.0     2783519.0     1187459.0  46.812969
8                            958.0     2732266.0     1198425.0  46.923747
9                           1036.0     2674162.0     1186069.0  46.821639
10                          1605.0     2630738.0     1116062.0  46.195314
11                          1980.0     2668583.0     1158215.0  46.571689
12                          2472.0     2579191.0     1079754.0  45.869092
13                           411.0     2498904.0     1122632.0  46.247519
14                          3571.0     2641939.0     1155287.0  46.547556
15                           273.0     2717874.0     1095883.0  46.004217
16                           454.0     2665545.0     1209850.0  47.036439
17                           589.0     2655844.0     1175930.0  46.732222
18                           485.0     2563087.0     1205560.0  47.000067
19                           367.0     2704167.0     1114316.0  46.172256
20                           490.0     2562131.0     1184612.0  46.811581
21                           497.0     2756911.0     1209351.0  47.016631
22                          2501.0     2744188.0     1234920.0  47.249447
23                          1709.0     2787251.0     1155685.0  46.526247
24                          1639.0     2734116.0     1147294.0  46.463542
25                          1804.0     2778576.0     1144976.0  46.432331
26                           482.0     2591633.0     1118584.0  46.218650
27                           556.0     2685118.0     1248066.0  47.377925
28                           776.0     2747866.0     1254588.0  47.425475

   Longitude               Climate region Canton
0   8.621894    Central Alpine north slope     UR
1   8.580553    Central Alpine north slope     UR
2   7.583525                 Eastern Jura     BL
```

```
3    7.464061            Central plateau    BE
4    6.792314              Western Jura     NE
5    7.139656   Western Alpine north slope  VD
6    6.978825              Western Jura     NE
7    9.843558  Northern and central Grisons GR
8    9.175350    Eastern Alpine north slope GL
9    8.410514    Central Alpine north slope OW
10   7.836822                   Valais      VS
11   8.333256   Western Alpine north slope  BE
12   7.170683            Alpine south side  VS
13   6.127742              Western plateau  GE
14   7.985444   Western Alpine north slope  VS
15   8.960322            Alpine south side  TI
16   8.301022            Central plateau    LU
17   8.169247   Western Alpine north slope  BE
18   6.953297              Western plateau  NE
19   8.787494            Alpine south side  TI
20   6.942469              Western plateau  VD
21   9.502594  Northern and central Grisons SG
22   9.343469    Eastern Alpine north slope AI
23   9.879469                 Engadine      GR
24   9.184700            Alpine south side  GR
25   9.762325                 Engadine      GR
26   7.330203                   Valais      VS
27   8.565742        North-eastern plateau  ZH
28   9.398528        North-eastern plateau  SG
```

1.3   Select one weather station

```
[3]: # Define the default parameters and tag the cell accordingly
     wsno = -1 # default -1 selects the last index, 2 sets BAS weather station
     #
     # Calling syntax from shell:
     #
     # time for i in {0..28}; do \
     #   papermill MeteoCH.ipynb \
     #   MeteoCH.output.ipynb \                # ...or /dev/null
     #   -p wsno $i; done
     #
     # The time command at the beginning of the call may be omitted.
```

```
[4]: # Parameters
     wsno = 7
```

```
[5]: wstation = ws['Station'].tolist()[wsno]
     print(wsno)
     ws[ws.Station==wstation]
```

```
label = ws[ws.Station==wstation]['station/location'].to_string()[::-1][0:3][::
    ↪-1]
print(f"The label of weather station {wstation} is {label}.")
```

7
The label of weather station Davos is DAV.

1.4 Read online observations from selected weather station

```
[6]: maxrows = 400 # displayed number of past days
     filenm = "nbcn-daily_"
     ext="csv"
     currurl = url + "/" + path + "/" + filenm + label + "_current." + ext
     prevurl = url + "/" + path + "/" + filenm + label + "_previous." + ext
     cf = pd.read_csv(currurl, sep=";", index_col='date',converters={'date':pd.
       ↪to_datetime}).drop(['station/location'], axis=1) #, engine='pyarrow')
     for col in  cf.columns:
         cf[col] = pd.to_numeric(cf[col], errors='coerce')
     pf = pd.read_csv(prevurl, sep=";", index_col='date',converters={'date':pd.
       ↪to_datetime}).drop(['station/location'], axis=1) #, engine='pyarrow')
     for col in  pf.columns:
         pf[col] = pd.to_numeric(pf[col], errors='coerce')
     df = pd.concat([pf, cf], axis=0).tail(maxrows)
```

1.5 Compute summary statistics

```
[7]: df.describe()
```

```
[7]:           gre000d0     hto000d0  nto000d0   prestad0     rre150d0    sre000d0  \
     count  400.00000  400.000000       0.0  400.00000  400.000000  400.00000
     mean   161.78750   20.332500       NaN  841.47975    2.143500  315.73000
     std     93.40379   30.891547       NaN    7.02271    4.877708  216.34652
     min      8.00000    0.000000       NaN  811.30000    0.000000    0.00000
     25%     85.00000    0.000000       NaN  838.05000    0.000000  116.50000
     50%    141.50000    0.000000       NaN  842.65000    0.000000  321.50000
     75%    230.50000   29.000000       NaN  846.50000    1.825000  494.25000
     max    362.00000  104.000000       NaN  854.60000   37.100000  698.00000

              tre200d0    tre200dn    tre200dx    ure200d0
     count  400.000000  400.000000  400.000000  400.000000
     mean     4.660500   -0.183250   10.295250   70.623000
     std      7.602176    7.397225    8.427093   13.286446
     min    -12.600000  -17.100000   -9.100000   37.500000
     25%     -1.525000   -5.525000    3.450000   61.000000
     50%      4.000000   -0.150000   10.250000   71.650000
     75%     11.700000    6.125000   17.550000   81.025000
     max     19.700000   14.200000   27.800000   95.000000
```

4

```python
[8]: (rows, cols) = df.shape
     print(f"{rows} observations from {min(df.index)} to {max(df.index)}.")
```

```
400 observations from 2022-01-25 00:00:00 to 2023-02-28 00:00:00.
```

1.6 Description of observed parameters

```python
[9]: from urllib.request import urlopen
     from io import BytesIO
     from zipfile import ZipFile

     zip_url = url + "/" + path + "/" + "data.zip"
     plist = []  # parameter
     ulist = []  # unit
     dlist = []  # description

     with urlopen(zip_url) as f:
         with BytesIO(f.read()) as b, ZipFile(b) as myzipfile:
             rf = myzipfile.open('1_how-to-download-nbcn-d.txt')
             blines = rf.readlines()
             rf.close()
             for i in range(14, 25):
                 line = blines[i].decode('unicode-escape').rstrip('\r\n')
                 plist.append(line[0:21].strip())
                 ulist.append(line[21:38].strip())
                 dlist.append(line[38:].strip('\n'))

     # list of lists instead of list of tuples
     ##zipped = zip(plist[1:], ulist[1:], dlist[1:])
     list_of_lists = [list(tup) for tup in zip(plist[1:], ulist[1:], dlist[1:])]
     cols = [plist[0], ulist[0], dlist[0]]

     par = pd.DataFrame(list_of_lists, columns = cols)
     print(par)
```

```
   Parameter Einheit                                     Beschreibung
0   gre000d0    W/m²                         Globalstrahlung; Tagesmittel
1   hto000d0      cm           Gesamtschneehöhe; Morgenmessung von 6 UTC
2   nto000d0       %                         Gesamtbewölkung; Tagesmittel
3   prestad0     hPa          Luftdruck auf Stationshöhe (QFE); Tagesmittel
4   rre150d0      mm      Niederschlag; Tagessumme 6 UTC - 6 UTC Folgetag
5   sre000d0     min                      Sonnenscheindauer; Tagessumme
6   tre200d0      °C             Lufttemperatur 2 m über Boden; Tagesmittel
7   tre200dn      °C            Lufttemperatur 2 m über Boden; Tagesminimum
8   tre200dx      °C            Lufttemperatur 2 m über Boden; Tagesmaximum
9   ure200d0       %  Relative Luftfeuchtigkeit 2 m über Boden; Tage…
```

1.7 Scatter plot air temperature

```
[10]: import matplotlib.pyplot as plt
      plt.style.use('_mpl-gallery')
      fswidth = 10
      fsheight = 5
```

```
[11]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

      axs.plot(df.index, df.tre200d0)
      axs.plot(df.index, df.tre200dn, color='0.8')
      axs.plot(df.index, df.tre200dx, color='0.8')
      axs.grid(visible='visible', which='major', color='0.8', linestyle='-')
      #axs.grid(which='major', color='0.8', linestyle='-')

      plt.xlabel('')
      plt.ylabel('Air temperature [°C]')
      plt.title('Air temperature 2 m above ground; daily mean\n' + wstation)
      plt.xticks(rotation=30)

      plt.show()
```
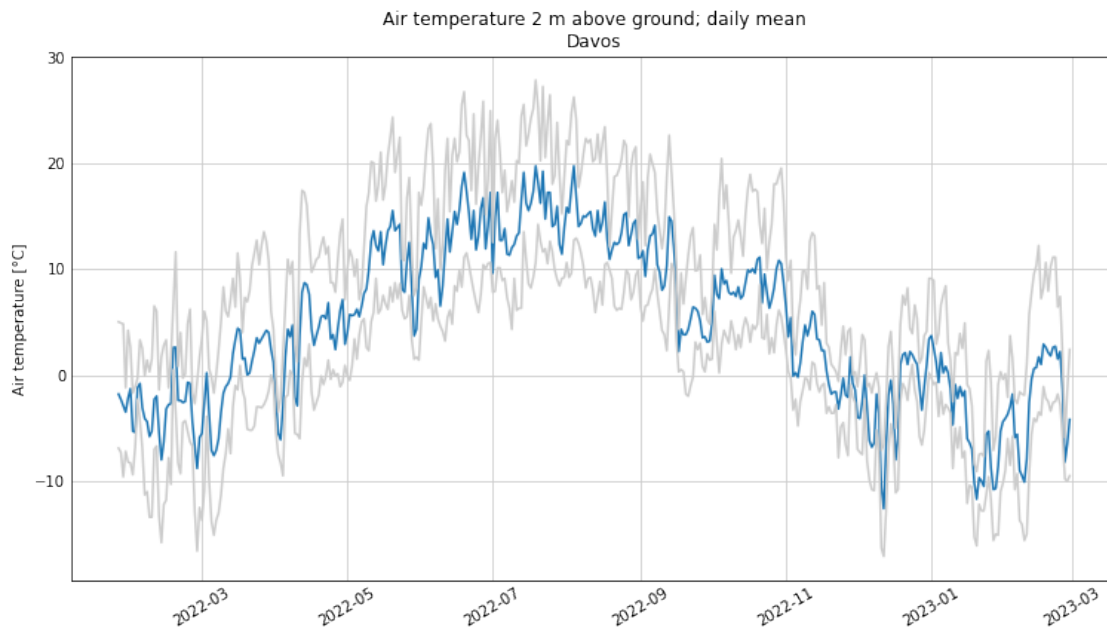
1.8 Scatter plot global radiation

```
[12]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

      axs.plot(df.index, df.gre000d0)
      axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

      plt.xlabel('')
      plt.ylabel('Global radiation [W/m²]')
      plt.title('Global radiation; daily mean\n' + wstation)
      plt.xticks(rotation=30)

      plt.show()
```
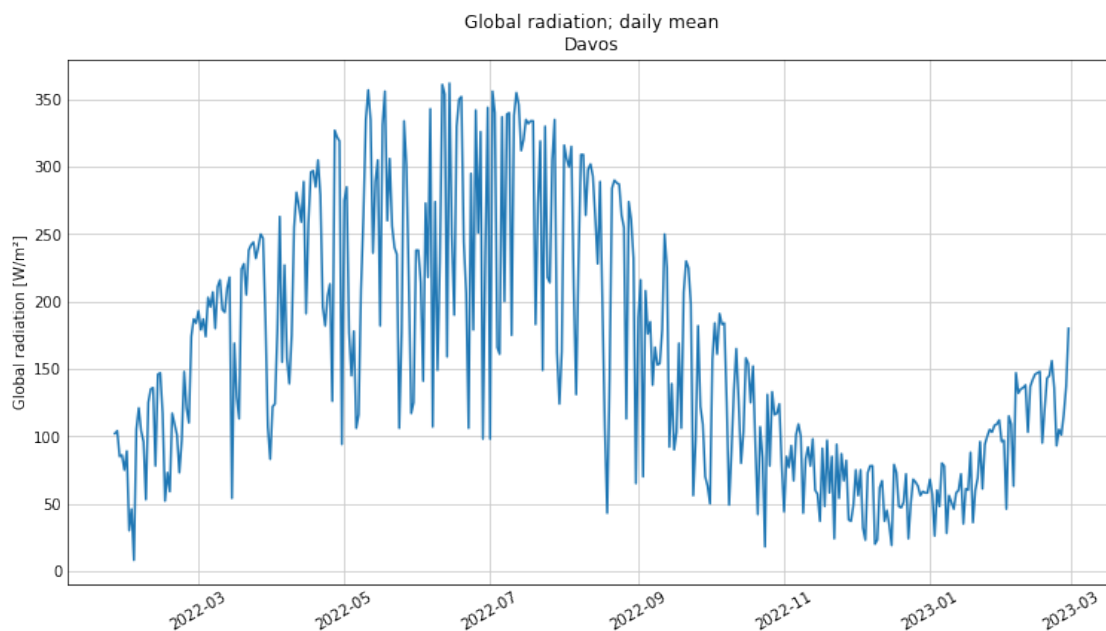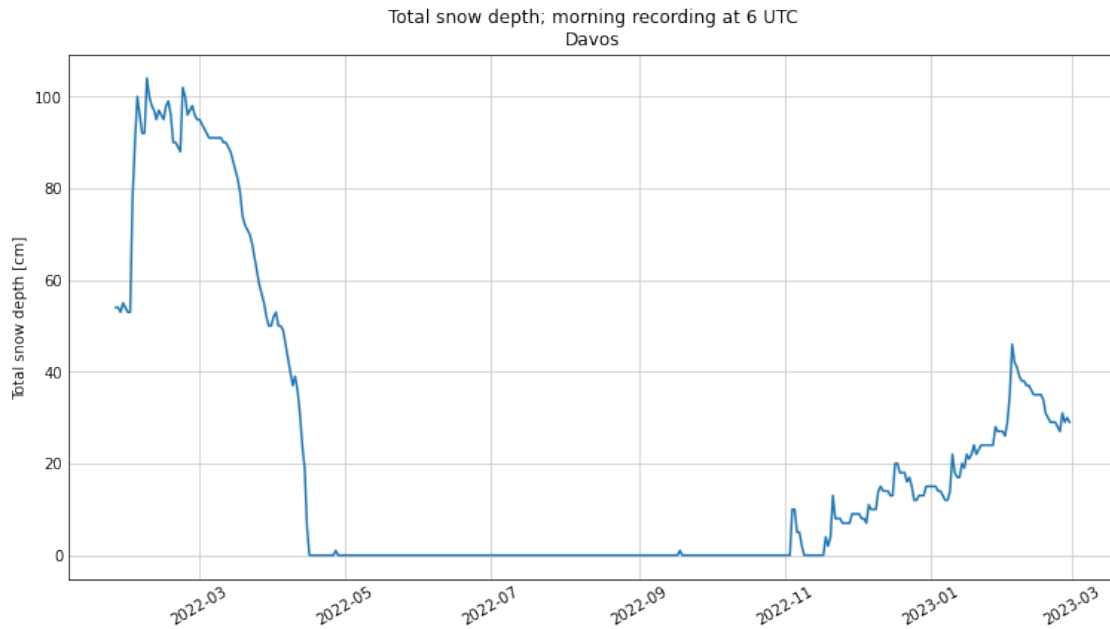

1.9 Scatter plot total snow depth

```
[13]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

      axs.plot(df.index, df.hto000d0)
      axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

      plt.xlabel('')
      plt.ylabel('Total snow depth [cm]')
      plt.title('Total snow depth; morning recording at 6 UTC\n' + wstation)
      plt.xticks(rotation=30)
```
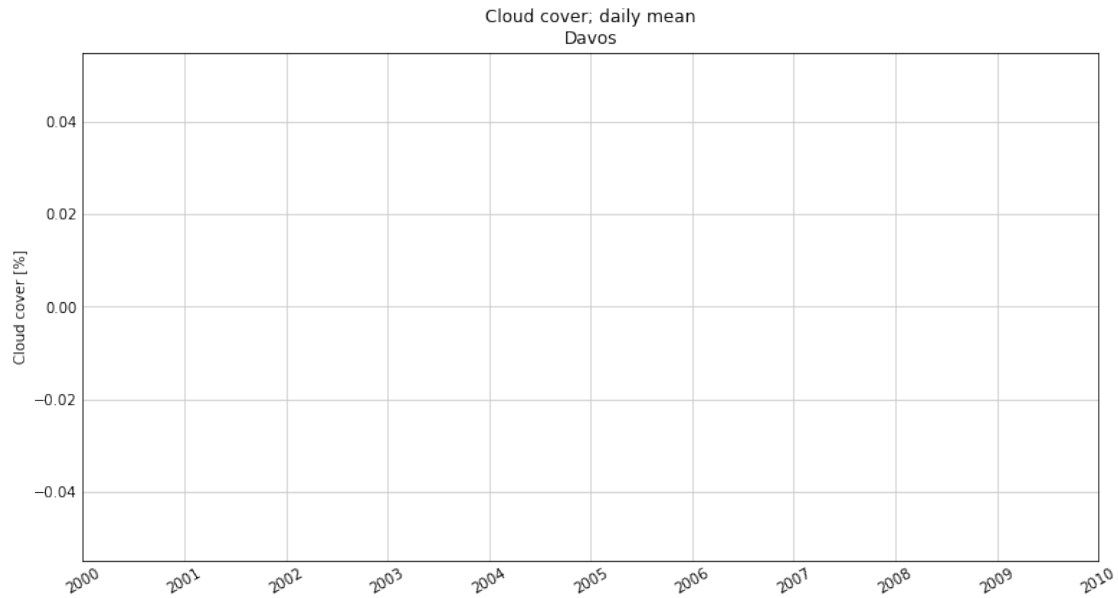
```
plt.show()
```


Total snow depth; morning recording at 6 UTC
Davos

1.10   Scatter plot cloud cover

```
[14]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

axs.plot(df.index, df.nto000d0)
axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

plt.xlabel('')
plt.ylabel('Cloud cover [%]')
plt.title('Cloud cover; daily mean\n' + wstation)
plt.xticks(rotation=30)

plt.show()
```
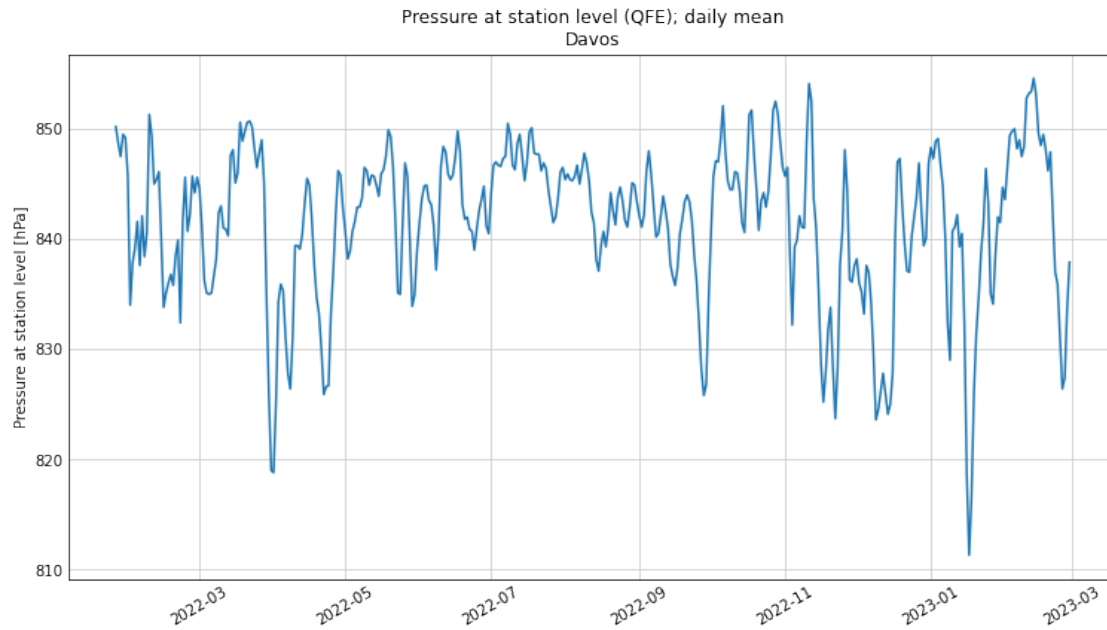
Cloud cover; daily mean
Davos

1.11 Scatter plot pressure at station level

```
[15]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

axs.plot(df.index, df.prestad0)
axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

plt.xlabel('')
plt.ylabel('Pressure at station level [hPa]')
plt.title('Pressure at station level (QFE); daily mean\n' + wstation)
plt.xticks(rotation=30)

plt.show()
```

Pressure at station level (QFE); daily mean
Davos

[16]: ```
Scatter plot cloud cover
```

[17]: ```
fig, axs = plt.subplots(figsize=(fswidth, fsheight))

axs.plot(df.index, df.rre150d0)
axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

plt.xlabel('')
plt.ylabel('Cloud cover [mm]')
plt.title('Precipitation; daily total 6 UTC\n' + wstation)
plt.xticks(rotation=30)

plt.show()
```

Precipitation; daily total 6 UTC
Davos
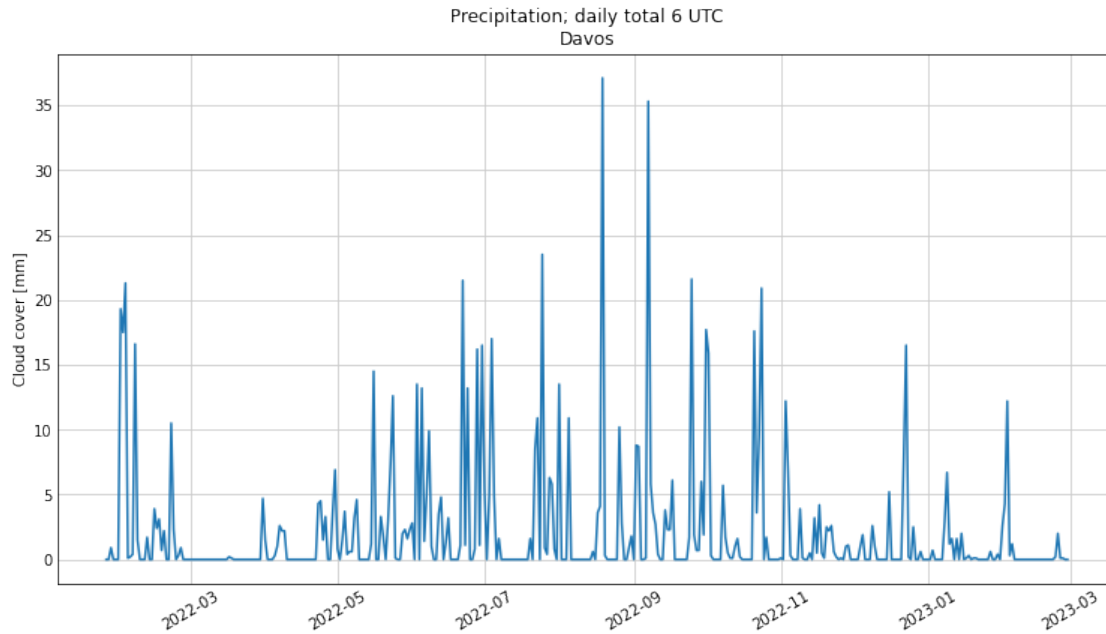
1.12 Scatter plot sunshine duration

```
[18]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

axs.plot(df.index, df.sre000d0)
axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

plt.xlabel('')
plt.ylabel('Sunshine duration [min]')
plt.title('Sunshine duration; daily total\n' + wstation)
plt.xticks(rotation=30)

plt.show()
```

Sunshine duration; daily total
Davos
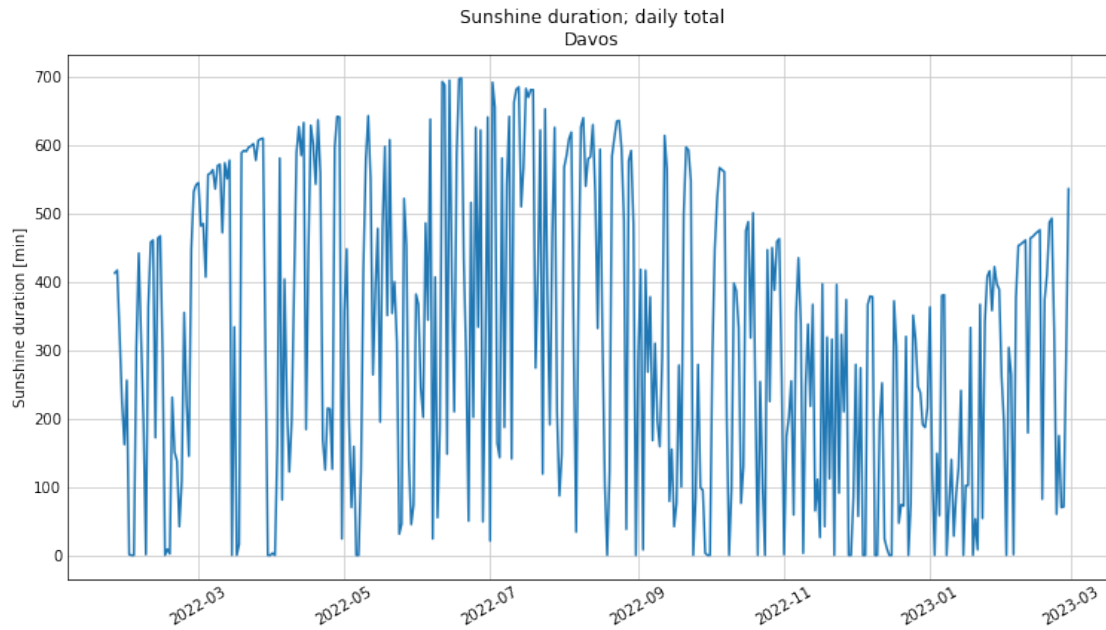
1.13 Scatter plot relative air humidity

```
[19]: fig, axs = plt.subplots(figsize=(fswidth, fsheight))

      axs.plot(df.index, df.ure200d0)
      axs.grid(visible='visible', which='major', color='0.8', linestyle='-')

      plt.xlabel('')
      plt.ylabel('Relative air humidity [%]')
      plt.title('Relative air humidity; 2 m above ground; daily\n' + wstation)
      plt.xticks(rotation=30)

      plt.show()
```

Relative air humidity; 2 m above ground; daily
Davos
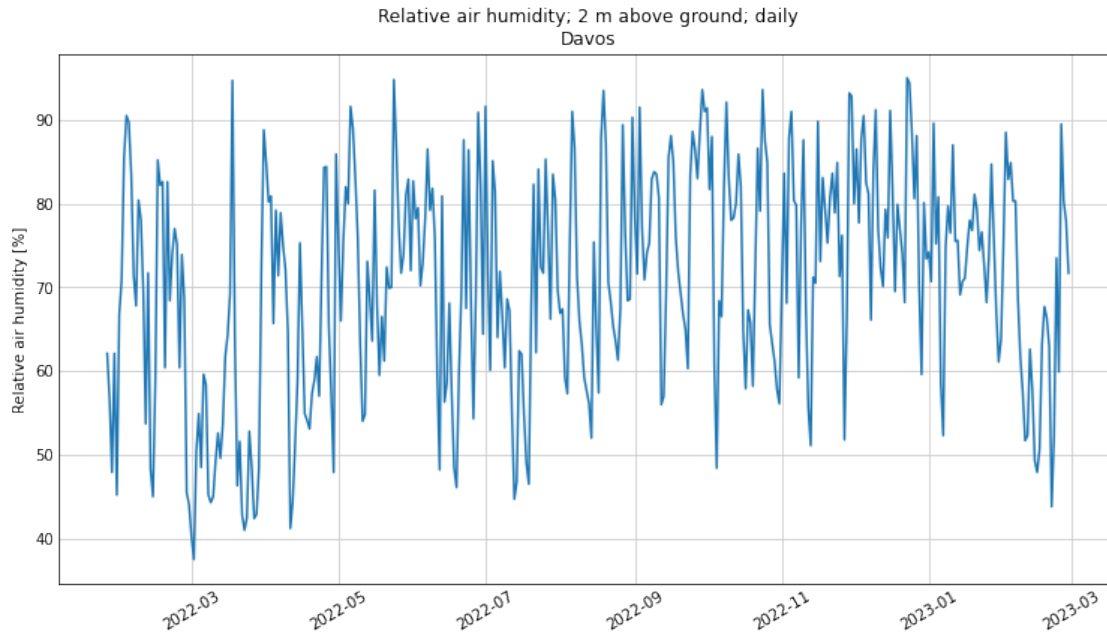
1.14 Export as HTML Report

```python
import os
#import ipynbname
#nb_fname = ipynbname.name()
nb_fname = 'MeteoCH' # hard-coded: import ipynbname raises an exception...
#nb_path = ipynbname.path()
#print(f"{nb_fname=}")
#print(f"{nb_path=}")

out_fname = nb_fname + ".output"
static_format = 'pdf'   # pdf or html, etc.
os.system(f'jupyter nbconvert --to {static_format} {out_fname}.ipynb')
os.system(f'mv {out_fname}.{static_format} {label}.{static_format}')
```

[ ]: