VI-Pandas

December 7, 2014

1 VI-Pandas

1.0.1 Index

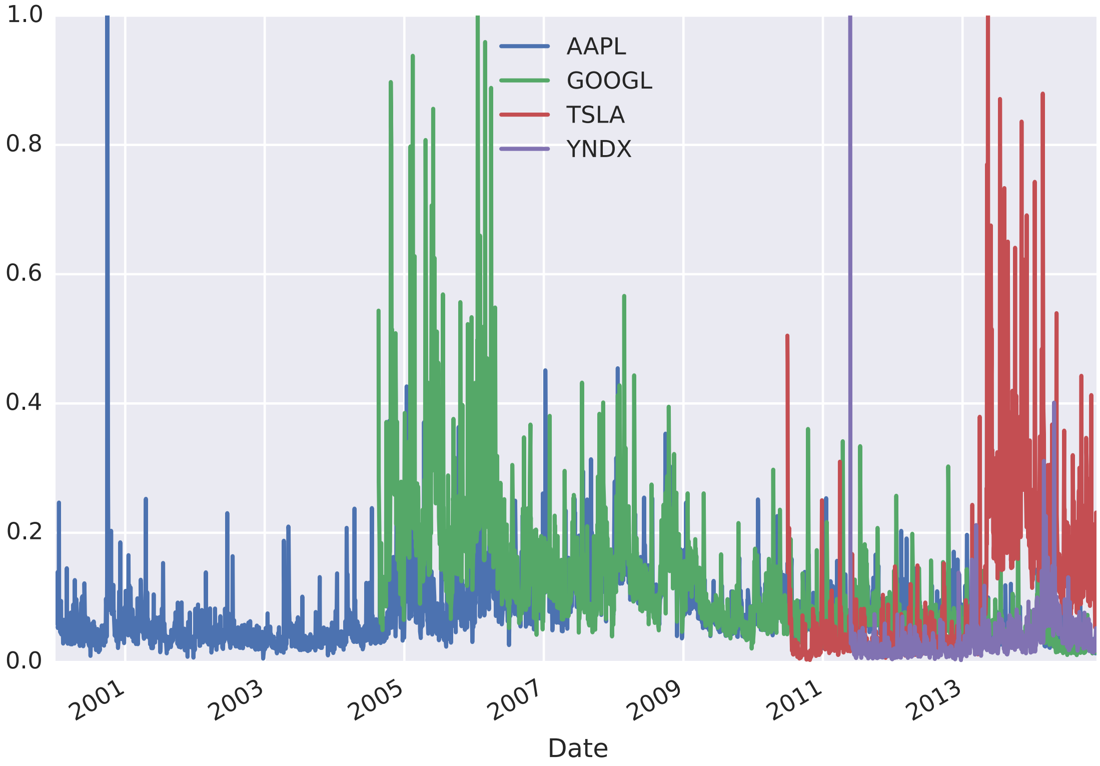
From their website "pandas is an open source, BSD-licensed library providing high-performance, easy-to-use data structures and data analysis tools for the Python programming language."

```
In [2]: from IPython.display import VimeoVideo

        VimeoVideo('59324550',width=900,height=768)

Out[2]: <IPython.lib.display.VimeoVideo at 0x7fb44199b610>
```

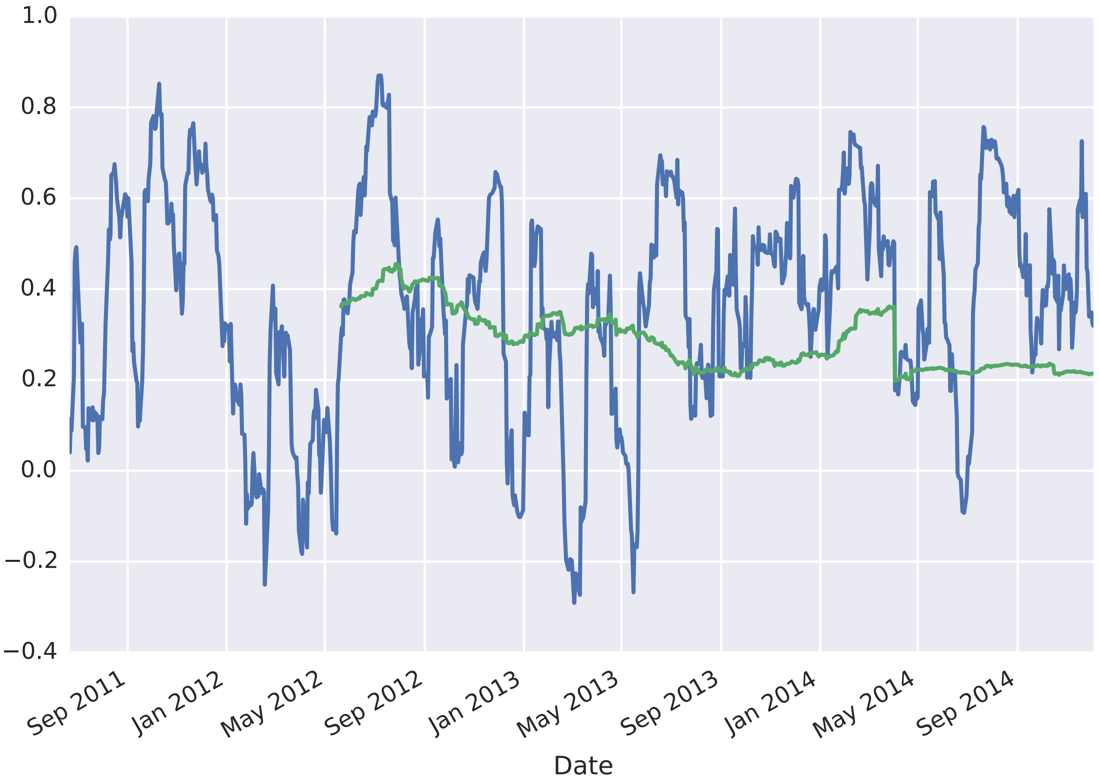
2 Time Series Analysis

```
In [308]: import os

          from pandas import *
          import pandas as pd
          import pandas.io.data as web
          import datetime
          import matplotlib
          import matplotlib.pyplot as plt
          import statsmodels as sm
          import seaborn
          seaborn.set()

          pd.__version__

Out[308]: '0.15.0'

In [309]: startdate = datetime.datetime(2000,1,1)
          enddate = datetime.datetime.today()
          df = web.DataReader(['AAPL','GOOGL','TSLA','YNDX'],'yahoo',start=startdate,end=enddate)

In [310]: normvol = df.Volume/df.Volume.max()

In [311]: normvol.plot()

Out[311]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2754675350>
```

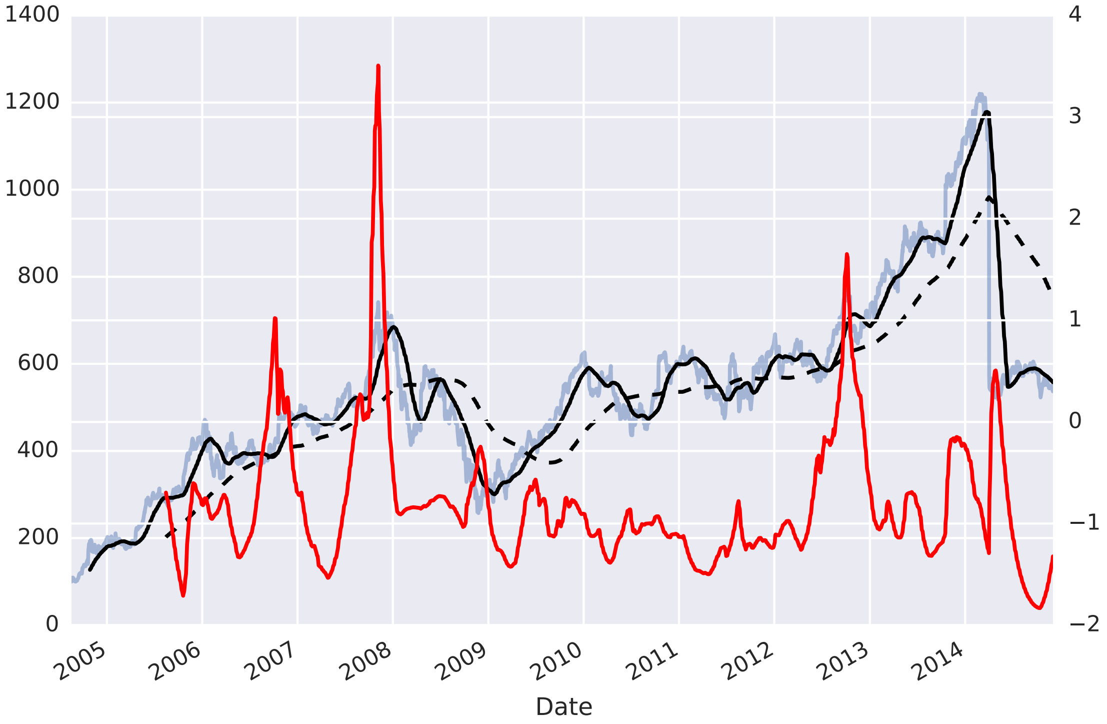
```
/home/jpsilva/anaconda/lib/python2.7/site-packages/matplotlib/font_manager.py:1279: UserWarning: findfor
  (prop.get_family(), self.defaultFamily[fontext]))
```



```
In [312]: goog = df.Close.GOOGL.dropna()
          yndx = df.Close.YNDX.dropna()

In [313]: googret = goog.pct_change()
          yndxret = yndx.pct_change()

In [314]: pd.rolling_corr(googret,yndxret,20).dropna().plot()
          pd.rolling_corr(googret,yndxret,250).dropna().plot()

Out[314]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2760d7cad0>
```



```
In [316]: from matplotlib.pyplot import *

In [318]: goog.plot(alpha=0.45)
          pd.rolling_mean(goog,50).plot(color='k')
          pd.rolling_mean(goog,250).plot(color='k',linestyle='--')
          ax = twinx()
          pd.rolling_kurt(goog,250).plot(color='r')

Out[318]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2760a91b10>
```



3 Energy Markets

```
In [227]: %matplotlib inline
          matplotlib.rcParams['figure.dpi'] = 300
```

Let's list all the data contained in the folder *data/sicherung_eex_daten/energiespot*

```
In [228]: data_dir = './data/sicherung_eex_daten/energiespot/'
          for filename in os.listdir(data_dir):
              print filename
```

```
energy_spot_historie_2010.xls
energy_spot_historie_2005.xls
energy_spot_historie_2003.xls
energy_intraday_history_2009.xls
energy_spot_historie_2012.xls
energy_spot_historie_2008.xls
energy_intraday_history_2007.xls
swiss_power_spot_market_2011.xls
energy_intraday_history_2006.xls
energy_spot_historie_2006.xls
swiss_power_spot_market_2008.xls
energy_intraday_history_2010.xls
energy_spot_historie_end_20020731_xetra.xls
energy_spot_historie_2004.xls
swiss_power_spot_market_2009.xls
energy_intraday_history_2012.xls
```

```
energy_intraday_history_2011 - Konflikt.xls
swiss_power_spot_market_2007.xls
Phelix_Quarterly.xls
energy_spot_historie_2011.xls
energy_spot_historie_2012 - Konflikt.xls
swiss_power_spot_market_2012.xls
energy_spot_historie_2002.xls
swiss_power_spot_market_2006.xls
energy_spot_historie_2007.xls
swiss_power_spot_market_2010.xls
energy_intraday_history_2011.xls
energy_intraday_history_2008.xls
energy_spot_historie_2009.xls
```

We now read the *xls* file which contains intraday data from 2012 for energy prices. We use the read_excel method from *pandas* to read xls files

```
In [229]: df = pd.read_excel(data_dir+'energy_intraday_history_2012.xls',sheetname='Intraday-Spot')
```

```
In [230]: df.head()
```

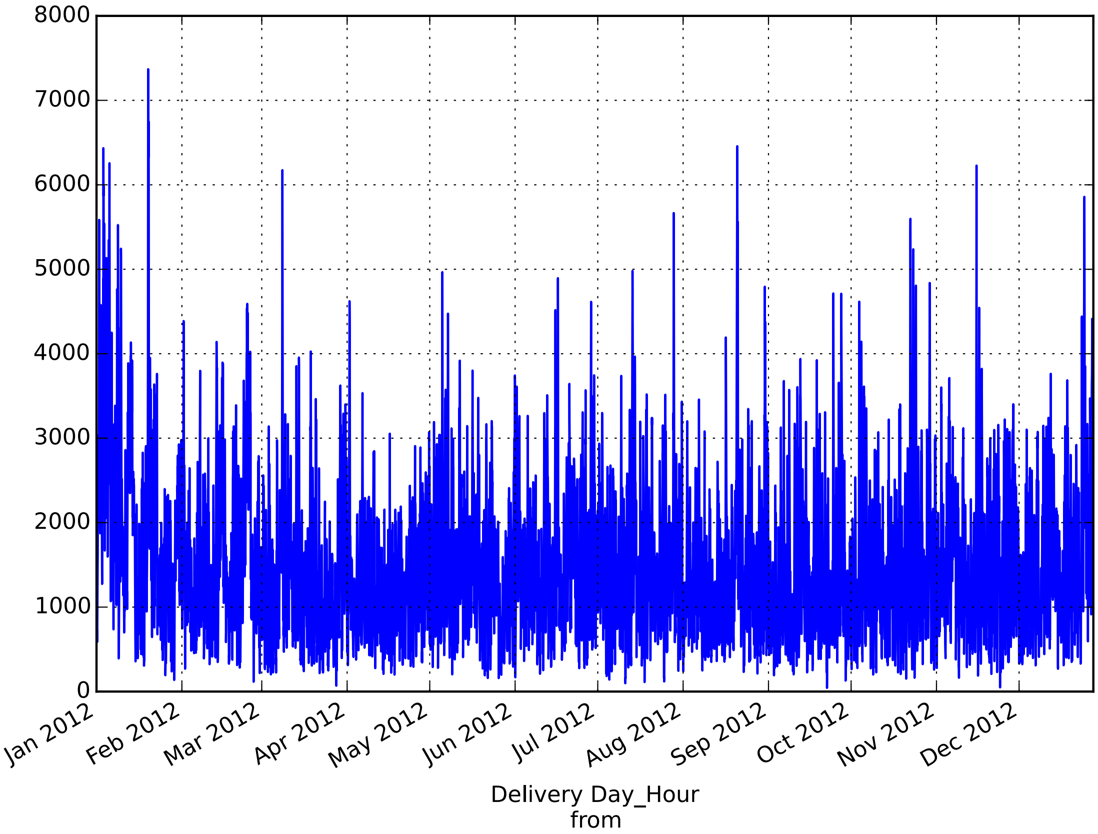
```
Out[230]:   EPEX Spot Intraday-Strom-Handel / EPEX Spot Intraday-Energy-Trading  \
          0                                        Delivery Day
          1                                 2012-12-27 00:00:00
          2                                 2012-12-27 00:00:00
          3                                 2012-12-27 00:00:00
          4                                 2012-12-27 00:00:00

             Unnamed: 1 Unnamed: 2  Unnamed: 3        Unnamed: 4      Unnamed: 5  \
          0  Hour\nfrom    Hour\nto  Volume\nMW  Volume (OTC)\nMW  Low Price\nEUR
          1       23:00      00:00       968.5               NaN               1
          2       22:00      23:00      1640.2               NaN               1
          3       21:00      22:00      1072.3               NaN               1
          4       20:00      21:00      1011.3               NaN               1

                  Unnamed: 6       Unnamed: 7          Unnamed: 8
          0  High Price\nEUR  Last Price\nEUR  Average Price\nEUR
          1               35               12               21.11
          2               45               25               30.16
          3             42.5               11               27.41
          4               43               26               35.96
```

```
In [231]: df = pd.read_excel(data_dir+'energy_intraday_history_2012.xls',sheetname='Intraday-Spot',head
```

```
In [232]: df.head()
```

```
Out[232]:              Hour\nfrom Hour\nto  Volume\nMW  Volume (OTC)\nMW  \
          Delivery Day
          2012-12-27        23:00    00:00       968.5               NaN
          2012-12-27        22:00    23:00      1640.2               NaN
          2012-12-27        21:00    22:00      1072.3               NaN
          2012-12-27        20:00    21:00      1011.3               NaN
          2012-12-27        19:00    20:00      2207.2               NaN

                        Low Price\nEUR  High Price\nEUR  Last Price\nEUR  \
          Delivery Day
          2012-12-27                 1             35.0             12.0
          2012-12-27                 1             45.0             25.0
          2012-12-27                 1             42.5             11.0
          2012-12-27                 1             43.0             26.0
          2012-12-27                 1             56.0             40.5

                       Average Price\nEUR
          Delivery Day
          2012-12-27                 21.11
          2012-12-27                 30.16
          2012-12-27                 27.41
          2012-12-27                 35.96
          2012-12-27                 42.25
```

```
In [233]: df.columns

Out[233]: Index([u’Hour\nfrom’, u’Hour\nto’, u’Volume\nMW’, u’Volume (OTC)\nMW’, u’Low Price\nEUR’, u’Hi

In [234]: df.columns = [column.replace(’ ’,’’).replace(’\n’,’’) for column in df.columns]

In [235]: df.columns

Out[235]: Index([u’Hourfrom’, u’Hourto’, u’VolumeMW’, u’Volume(OTC)MW’, u’LowPriceEUR’, u’HighPriceEUR’

In [236]: df = pd.read_excel(data_dir+’energy_intraday_history_2012.xls’,sheetname=’Intraday-Spot’,\
                             header=1, parse_dates = [[’Delivery Day’,’Hour\nfrom’]],index_col=0)

In [237]: try:
              del df[’Hour\nto’]
          except:
              pass
          df.columns = [column.replace(’ ’,’’).replace(’\n’,’’) for column in df.columns]

In [238]: df.head()

Out[238]:                          VolumeMW  Volume(OTC)MW  LowPriceEUR  HighPriceEUR  \
          Delivery Day_Hour\nfrom
          2012-12-27 23:00:00         968.5            NaN            1          35.0
          2012-12-27 22:00:00        1640.2            NaN            1          45.0
          2012-12-27 21:00:00        1072.3            NaN            1          42.5
          2012-12-27 20:00:00        1011.3            NaN            1          43.0
          2012-12-27 19:00:00        2207.2            NaN            1          56.0

                                   LastPriceEUR  AveragePriceEUR
          Delivery Day_Hour\nfrom
          2012-12-27 23:00:00              12.0            21.11
          2012-12-27 22:00:00              25.0            30.16
          2012-12-27 21:00:00              11.0            27.41
          2012-12-27 20:00:00              26.0            35.96
          2012-12-27 19:00:00              40.5            42.25

In [239]: df.VolumeMW.plot()

Out[239]: <matplotlib.axes._subplots.AxesSubplot at 0x7f27621c9550>
```



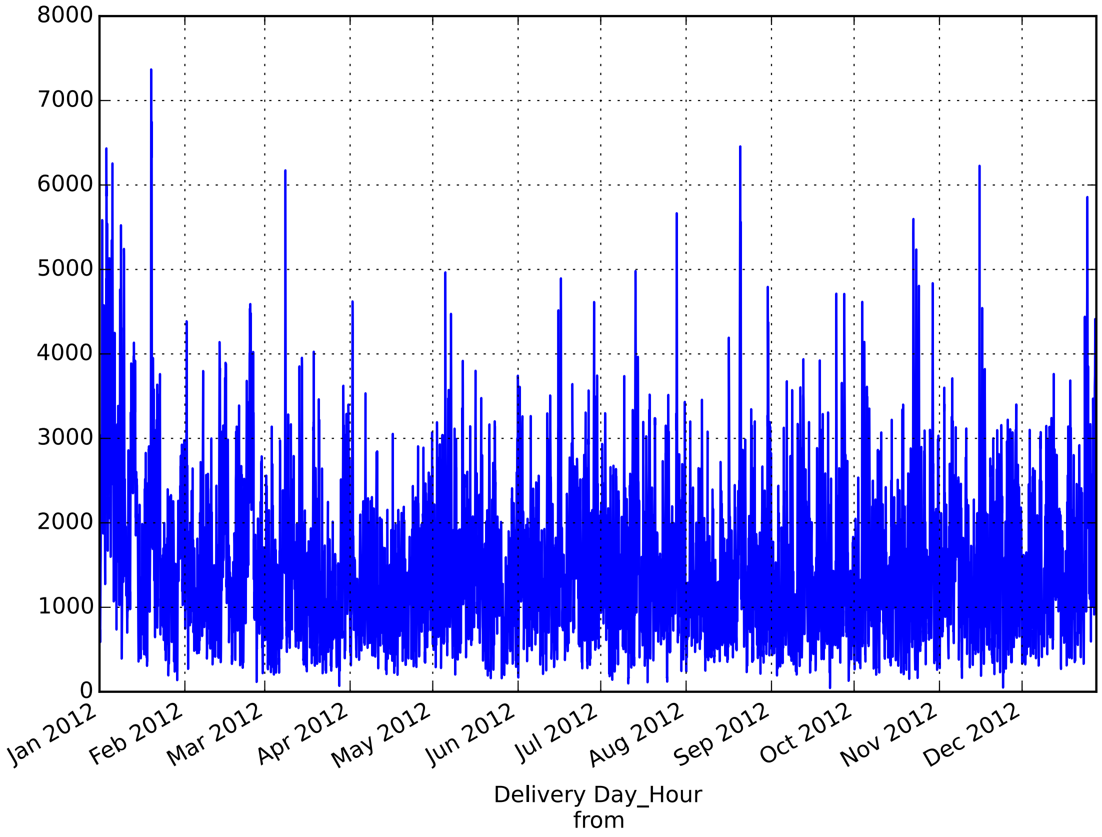
```
In [240]: df.index.get_duplicates()

Out[240]: <class 'pandas.tseries.index.DatetimeIndex'>
          [2012-10-28 02:00:00]
          Length: 1, Freq: None, Timezone: None

In [241]: df.ix[df.index.get_duplicates()]

Out[241]:                          VolumeMW  Volume(OTC)MW  LowPriceEUR  HighPriceEUR  \
          Delivery Day_Hour\nfrom
          2012-10-28 02:00:00           625            150           25            40
          2012-10-28 02:00:00           752            150           18            36

                                   LastPriceEUR  AveragePriceEUR
          Delivery Day_Hour\nfrom
          2012-10-28 02:00:00                40            30.30
          2012-10-28 02:00:00                33            26.67

In [242]: dfgby = df.groupby(df.index).first()
          dfgby.ix['2012-10-28 02:00']

Out[242]: VolumeMW          625.0
          Volume(OTC)MW     150.0
          LowPriceEUR        25.0
          HighPriceEUR       40.0
          LastPriceEUR        40.0
          AveragePriceEUR     30.3
          Name: 2012-10-28 02:00:00, dtype: float64
```

```
In [243]: def wavg(group):
              w = group['VolumeMW']*group['AveragePriceEUR']
              d = group
              return (d*w).sum()/w.sum()

          grouped = df.groupby(df.index).apply(wavg)
          grouped.ix['2012-10-28 02:00']
```

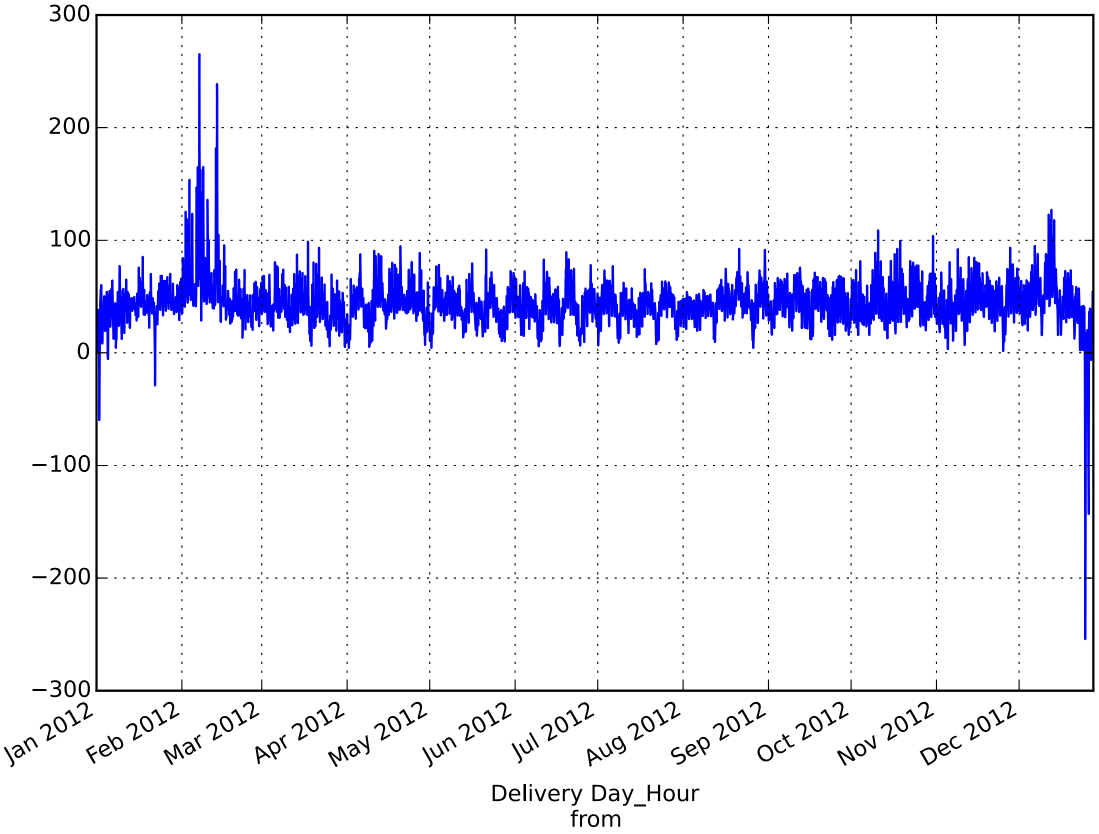
```
Out[243]: VolumeMW           690.321198
          Volume(OTC)MW      150.000000
          LowPriceEUR         21.399619
          HighPriceEUR        37.942639
          LastPriceEUR        36.399619
          AveragePriceEUR     28.432945
          Name: 2012-10-28 02:00:00, dtype: float64
```

```
In [244]: df = grouped
          df.VolumeMW.plot()
```

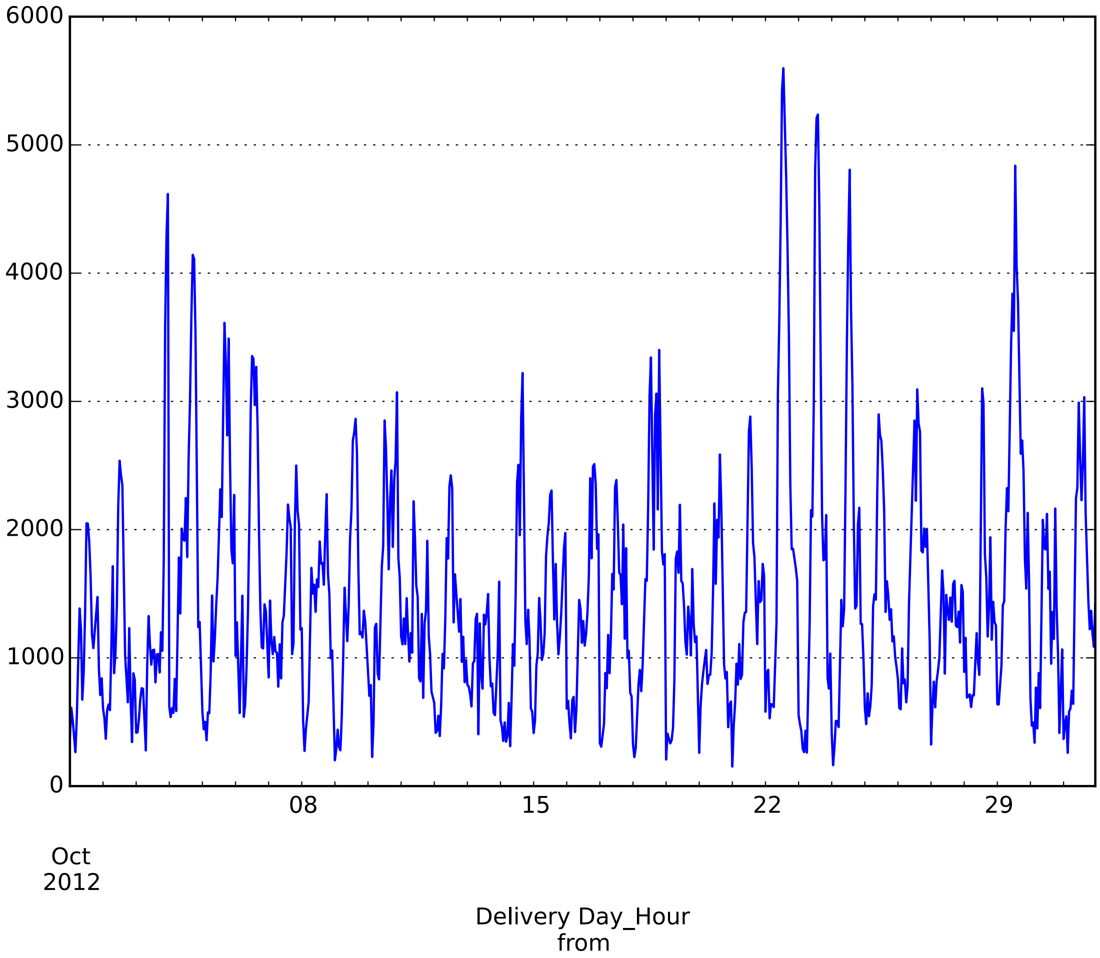
```
Out[244]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2762f19310>
```



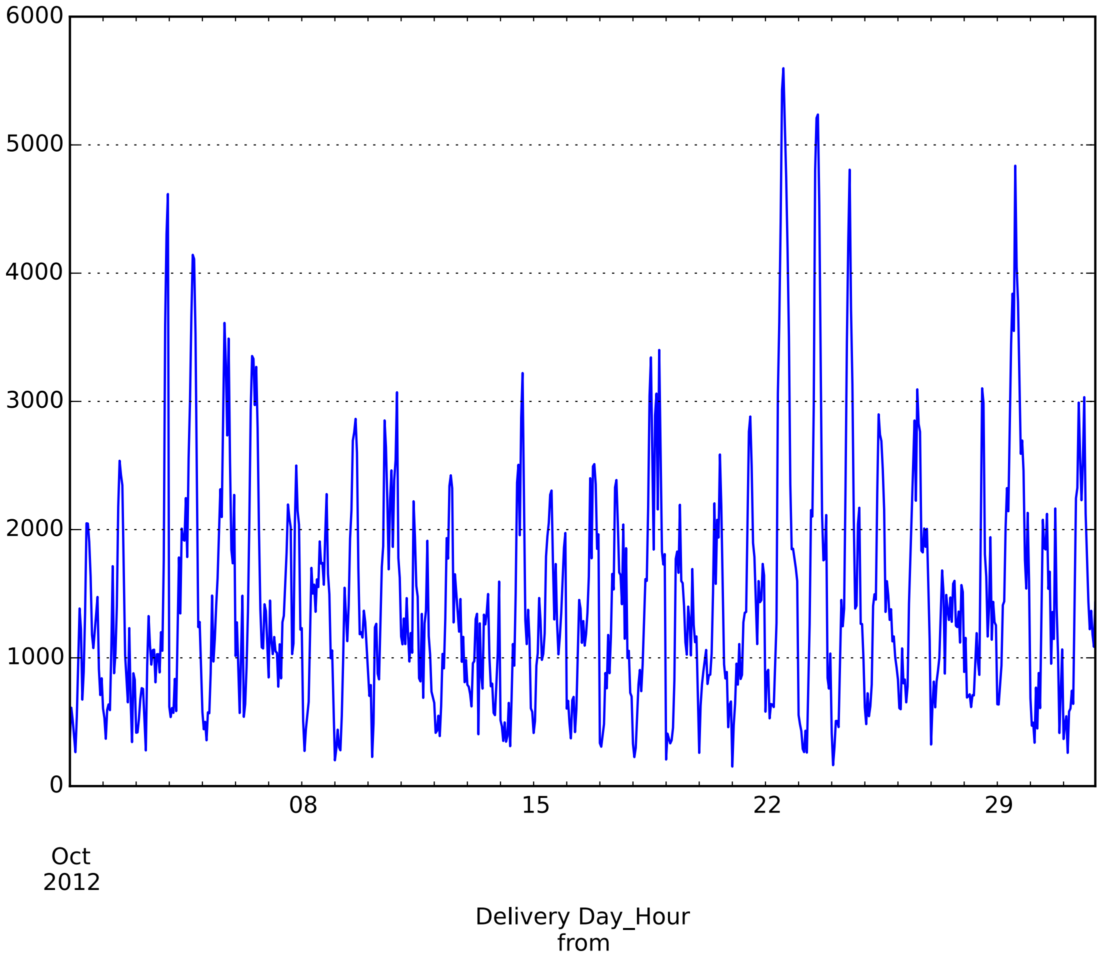
```
In [245]: df.AveragePriceEUR.plot()

Out[245]: <matplotlib.axes._subplots.AxesSubplot at 0x7f27618721d0>
```



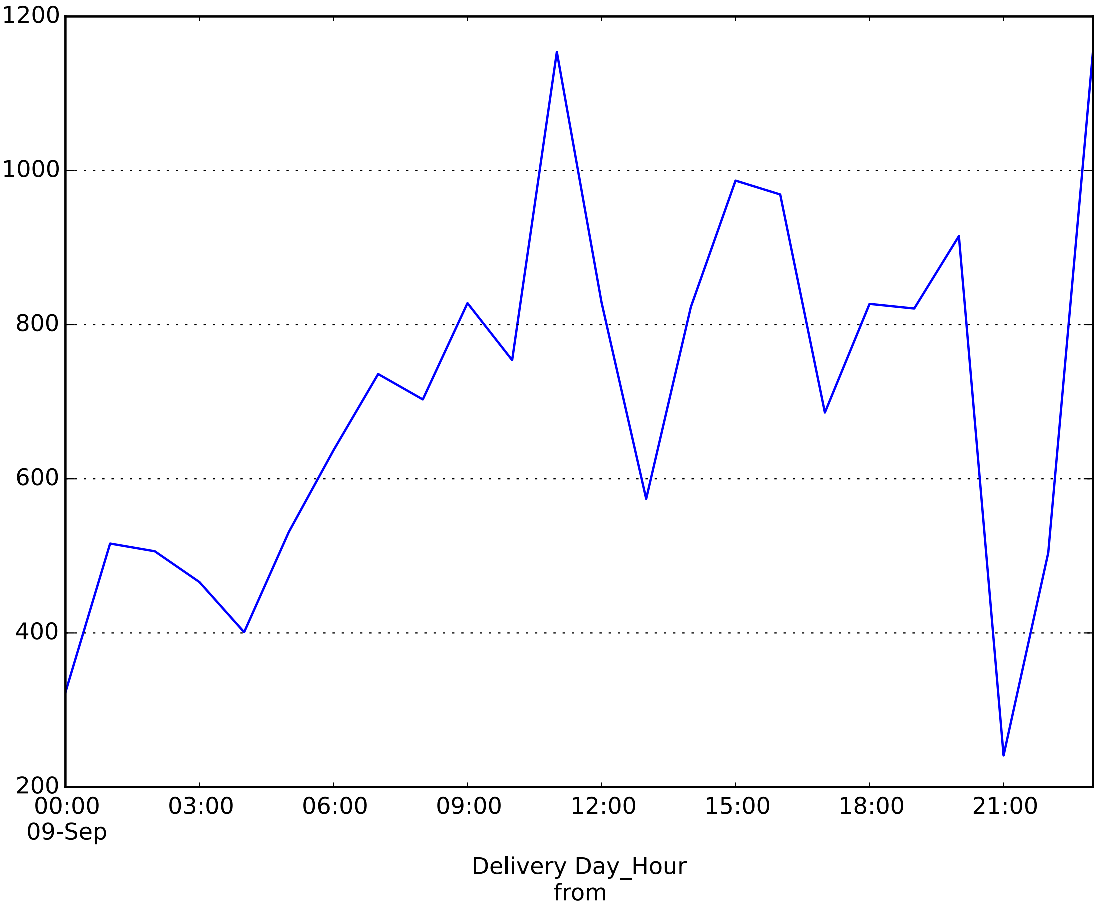
```
In [246]: ts = df.VolumeMW

In [247]: ts['10/2012'].plot()

Out[247]: <matplotlib.axes._subplots.AxesSubplot at 0x7f27681926d0>
```



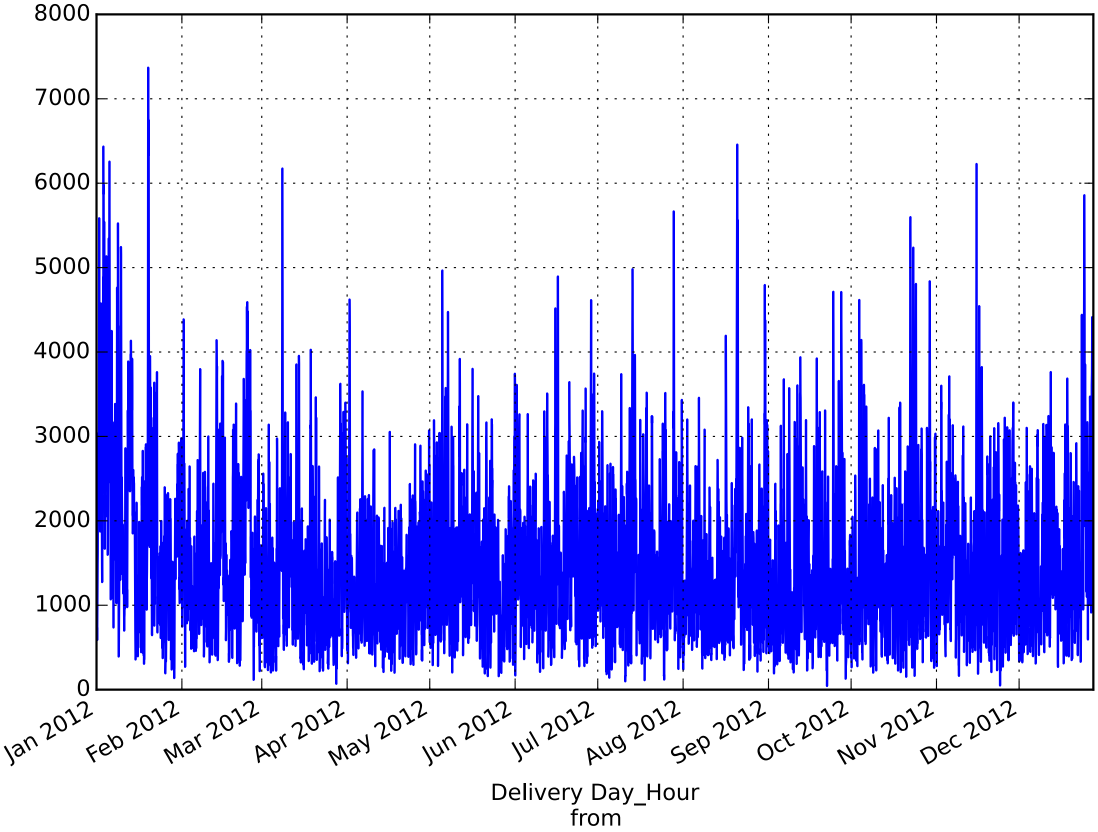
```
In [248]: ts['10-2012'].plot()

Out[248]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2761fe5350>
```



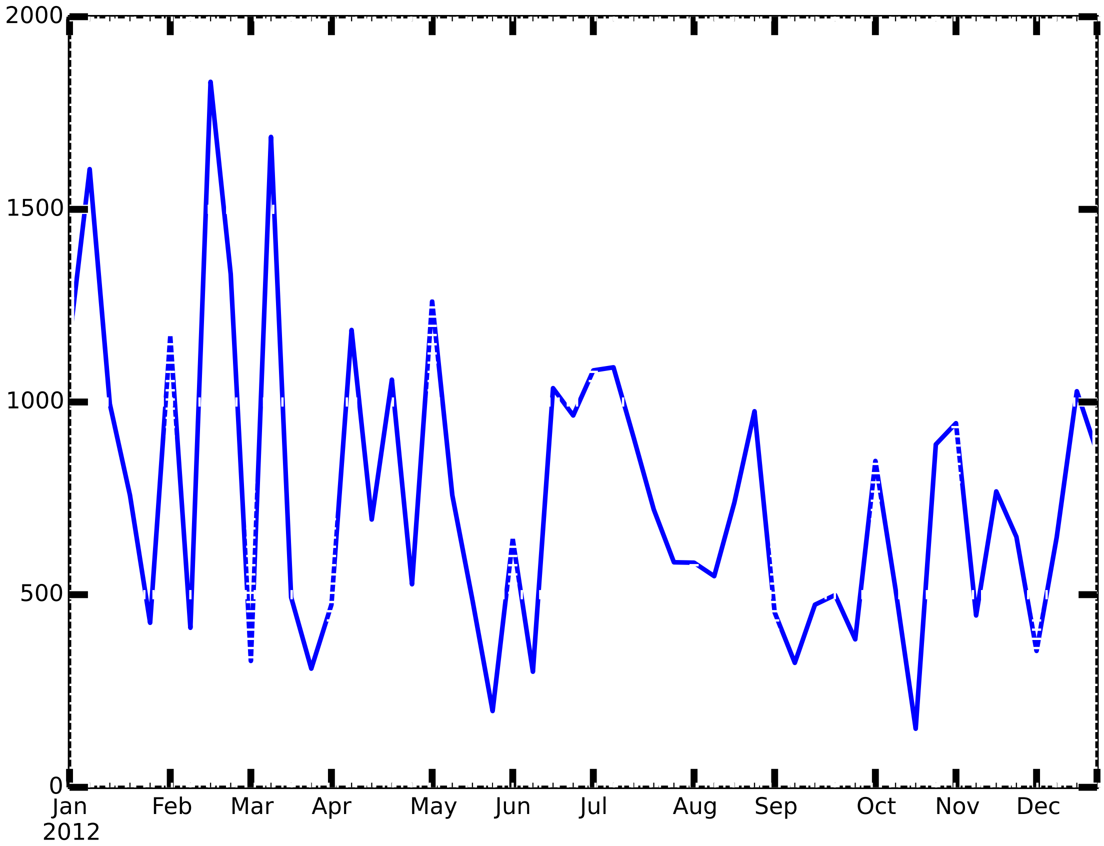
```
In [249]: ts['09-09-2012'].plot()

Out[249]: <matplotlib.axes._subplots.AxesSubplot at 0x7f27618ce150>
```



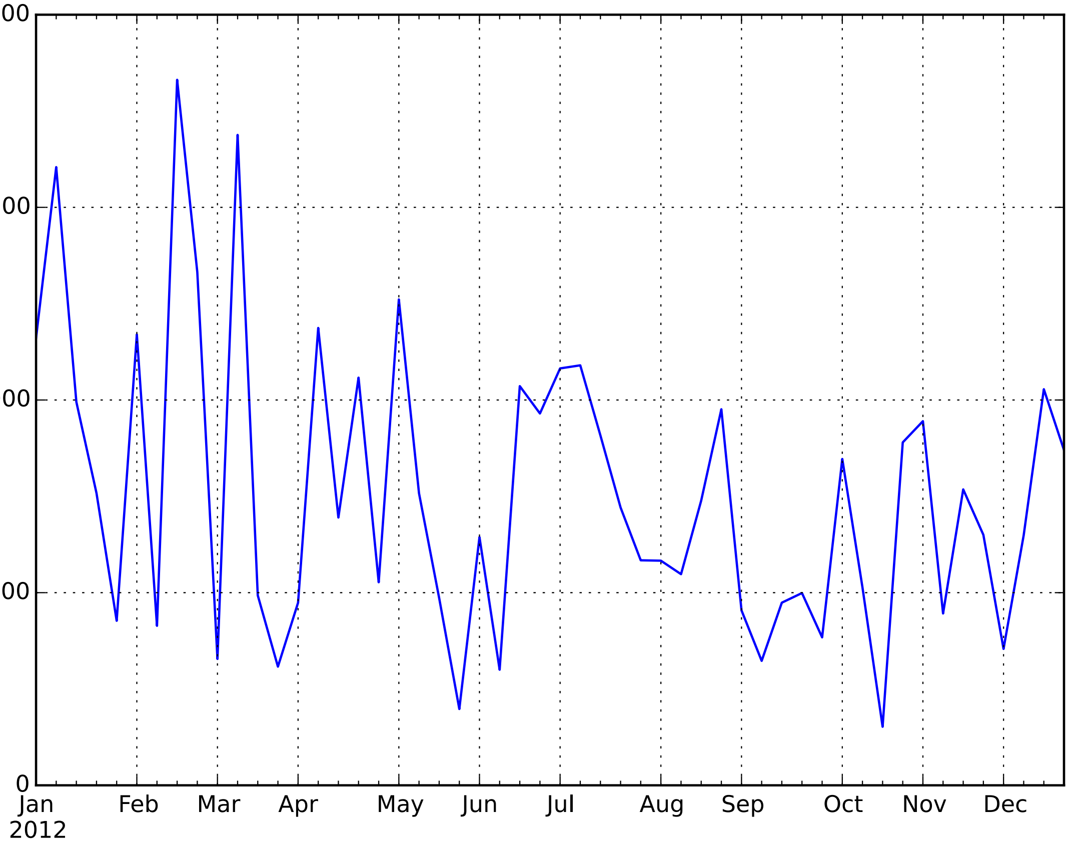
```
In [250]: df.sort_index(inplace=True)

In [251]: df.VolumeMW.plot()

Out[251]: <matplotlib.axes._subplots.AxesSubplot at 0x7f276218c890>
```



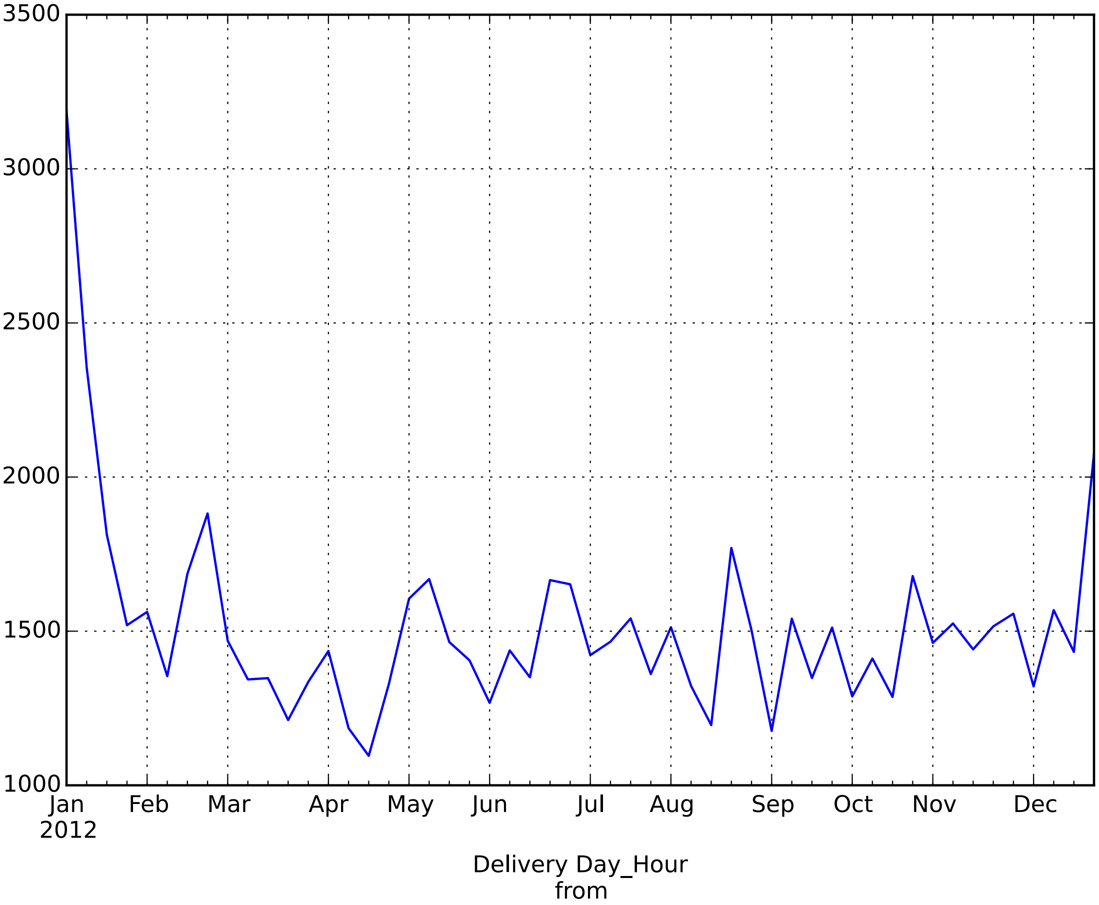
```
In [252]: ts = df.VolumeMW
          with plt.xkcd():
              ts.asfreq(freq='W').plot()
```



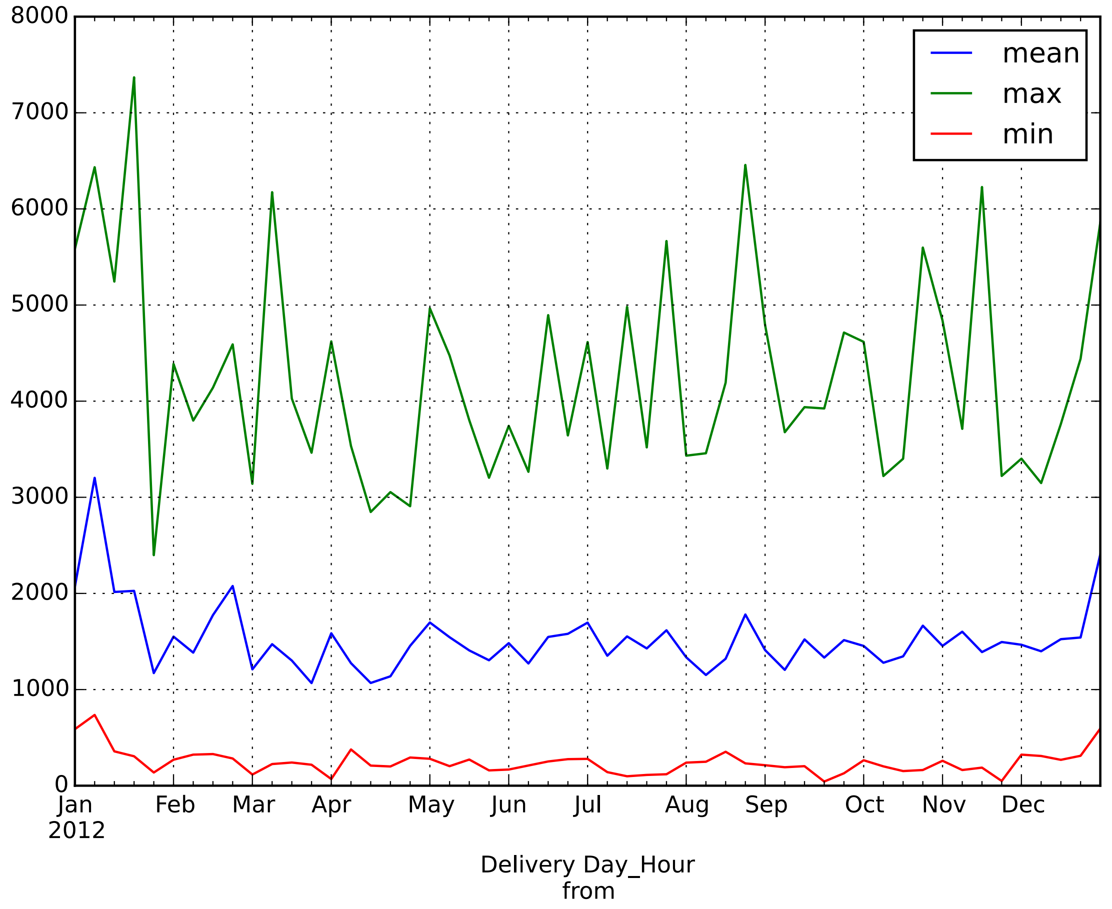
```
In [253]: ts.asfreq(freq='W').plot()

Out[253]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2762a793d0>
```



```
In [254]: ts.describe()

Out[254]: count    8687.000000
          mean     1510.502834
          std       911.039358
          min        43.000000
          25%       840.000000
          50%      1325.000000
          75%      1960.000000
          max      7369.000000
          Name: VolumeMW, dtype: float64

In [255]: ts.resample('W-FRI').plot()

Out[255]: <matplotlib.axes._subplots.AxesSubplot at 0x7f27626b8a50>
```



```
In [256]: ts.resample('W-SUN',how=['mean','max','min']).plot()

Out[256]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2761aade50>
```



```
In [257]: resampled = ts.resample('30t')
          resampled

Out[257]: Delivery Day_Hour\nfrom
          2012-01-01 00:00:00       1161.0
          2012-01-01 00:30:00          NaN
          2012-01-01 01:00:00        791.0
          2012-01-01 01:30:00          NaN
          2012-01-01 02:00:00        911.0
          2012-01-01 02:30:00          NaN
          2012-01-01 03:00:00        666.0
          2012-01-01 03:30:00          NaN
          2012-01-01 04:00:00        694.0
          2012-01-01 04:30:00          NaN
          2012-01-01 05:00:00        730.0
          2012-01-01 05:30:00          NaN
          2012-01-01 06:00:00        587.9
          2012-01-01 06:30:00          NaN
          2012-01-01 07:00:00       1077.7
          ...
          2012-12-27 16:00:00       4034.7
          2012-12-27 16:30:00          NaN
```

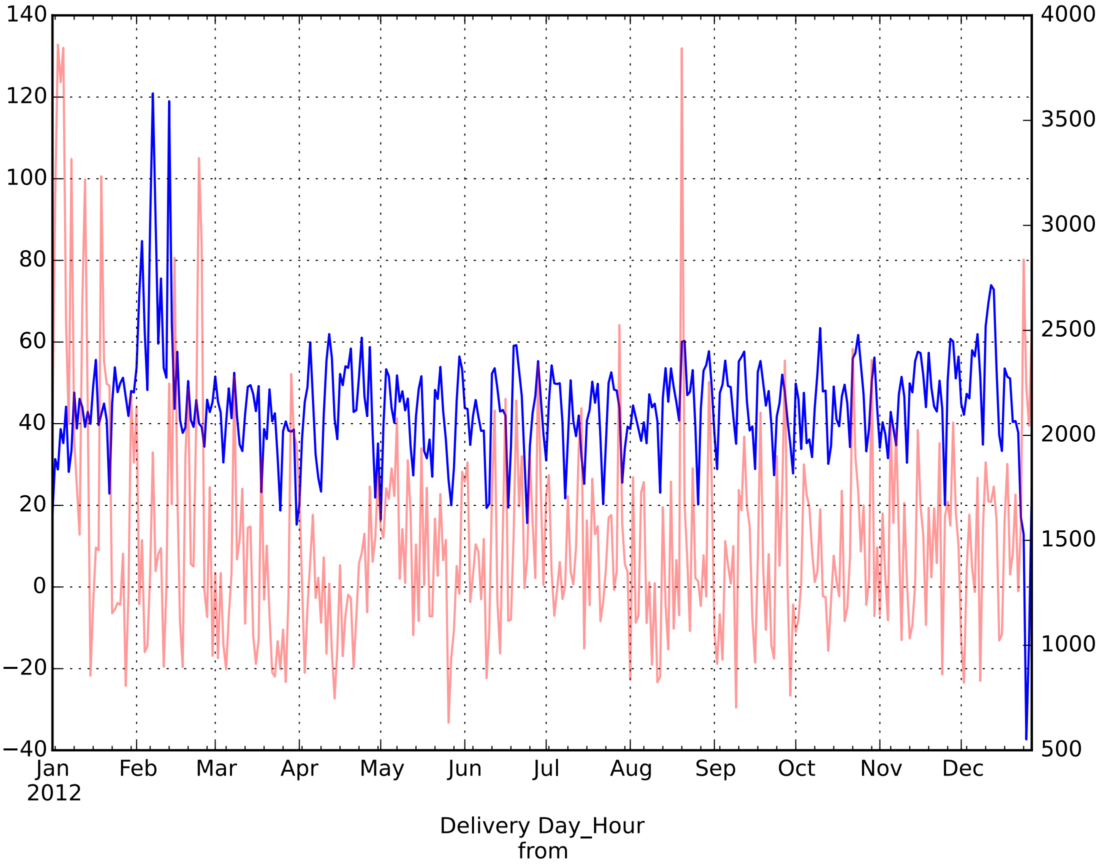
```
2012-12-27 17:00:00        3861.6
2012-12-27 17:30:00           NaN
2012-12-27 18:00:00        4029.4
2012-12-27 18:30:00           NaN
2012-12-27 19:00:00        2207.2
2012-12-27 19:30:00           NaN
2012-12-27 20:00:00        1011.3
2012-12-27 20:30:00           NaN
2012-12-27 21:00:00        1072.3
2012-12-27 21:30:00           NaN
2012-12-27 22:00:00        1640.2
2012-12-27 22:30:00           NaN
2012-12-27 23:00:00         968.5
Freq: 30T, Name: VolumeMW, Length: 17375
```

```
In [258]: resampled.interpolate()
```

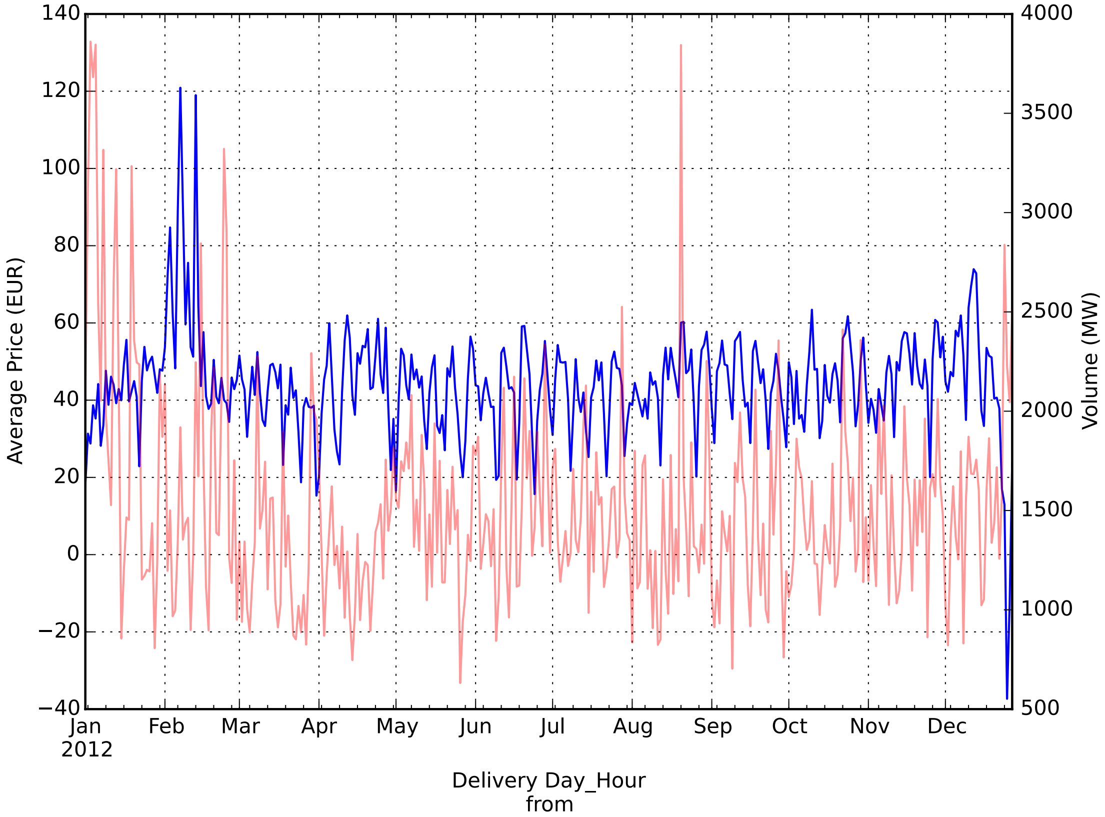
```
Out[258]: Delivery Day_Hour\nfrom
          2012-01-01 00:00:00        1161.00
          2012-01-01 00:30:00         976.00
          2012-01-01 01:00:00         791.00
          2012-01-01 01:30:00         851.00
          2012-01-01 02:00:00         911.00
          2012-01-01 02:30:00         788.50
          2012-01-01 03:00:00         666.00
          2012-01-01 03:30:00         680.00
          2012-01-01 04:00:00         694.00
          2012-01-01 04:30:00         712.00
          2012-01-01 05:00:00         730.00
          2012-01-01 05:30:00         658.95
          2012-01-01 06:00:00         587.90
          2012-01-01 06:30:00         832.80
          2012-01-01 07:00:00        1077.70
          ...
          2012-12-27 16:00:00        4034.70
          2012-12-27 16:30:00        3948.15
          2012-12-27 17:00:00        3861.60
          2012-12-27 17:30:00        3945.50
          2012-12-27 18:00:00        4029.40
          2012-12-27 18:30:00        3118.30
          2012-12-27 19:00:00        2207.20
          2012-12-27 19:30:00        1609.25
          2012-12-27 20:00:00        1011.30
          2012-12-27 20:30:00        1041.80
          2012-12-27 21:00:00        1072.30
          2012-12-27 21:30:00        1356.25
          2012-12-27 22:00:00        1640.20
          2012-12-27 22:30:00        1304.35
          2012-12-27 23:00:00         968.50
          Freq: 30T, Name: VolumeMW, Length: 17375
```

```
In [259]: df.resample('D').AveragePriceEUR.plot(style='b')
          df.resample('D').VolumeMW.plot(secondary_y=True, style='r', alpha = 0.4)
```

```
Out[259]: <matplotlib.axes._subplots.AxesSubplot at 0x7f27617e1cd0>
```



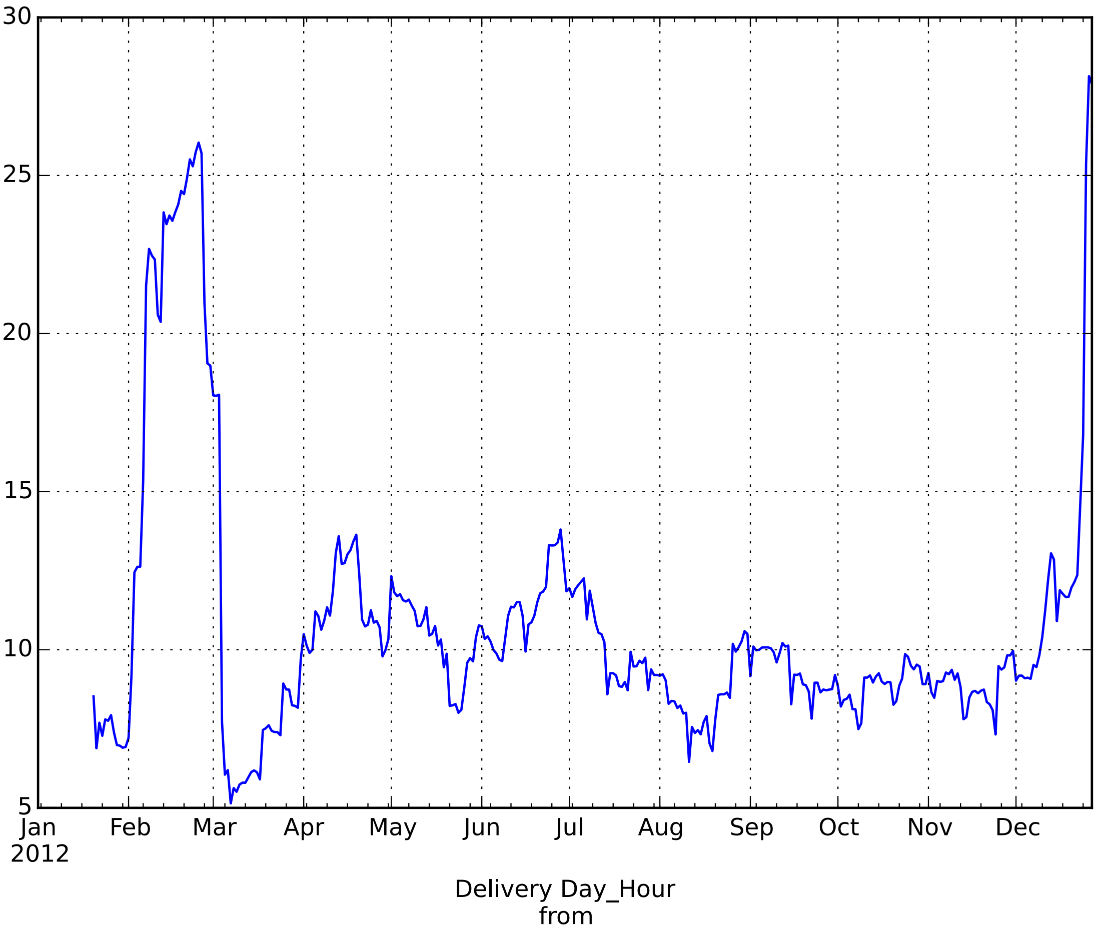
```
In [260]: ax1 = df.resample('D').AveragePriceEUR.plot(style='b')
          ax2 = df.resample('D').VolumeMW.plot(secondary_y=True, style='r', alpha = 0.4)
          ax1.set_ylabel('Average Price (EUR)')
          ax2.set_ylabel('Volume (MW)')

Out[260]: <matplotlib.text.Text at 0x7f276125c310>
```



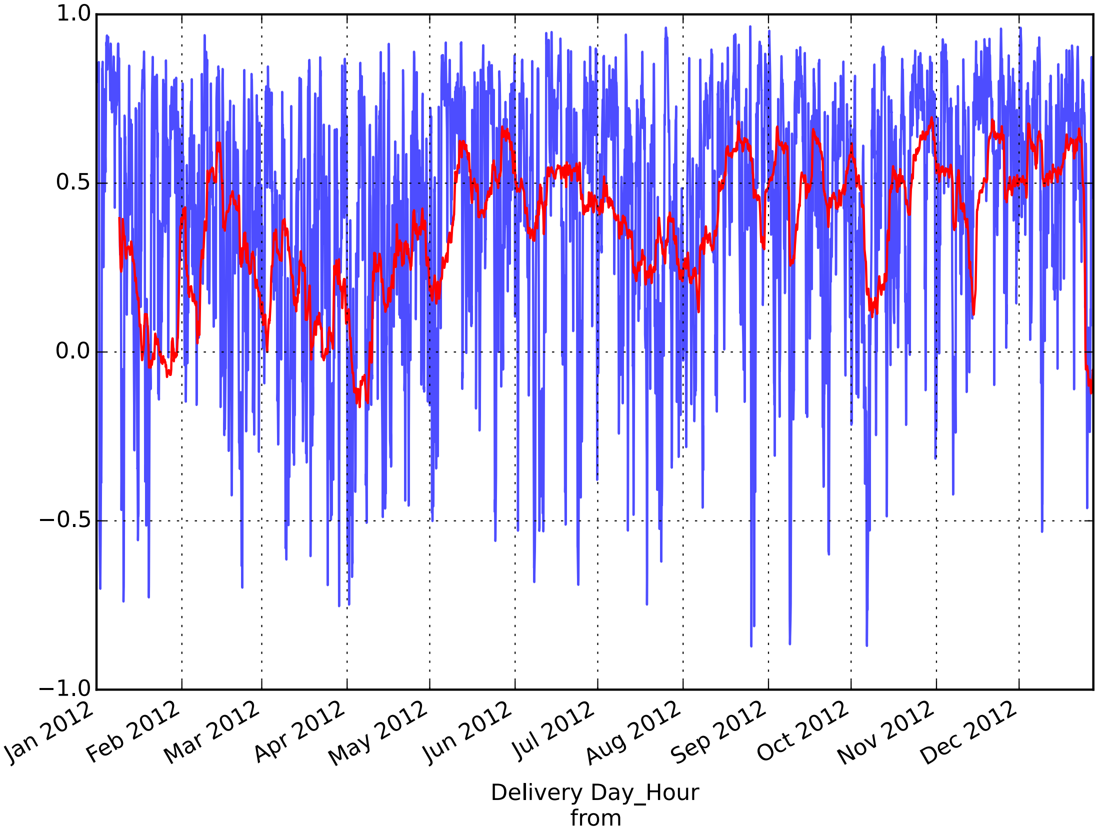
Rolling windows

```
In [261]: rolling_std(df.AveragePriceEUR.resample('1D'), window = 20).plot()
```

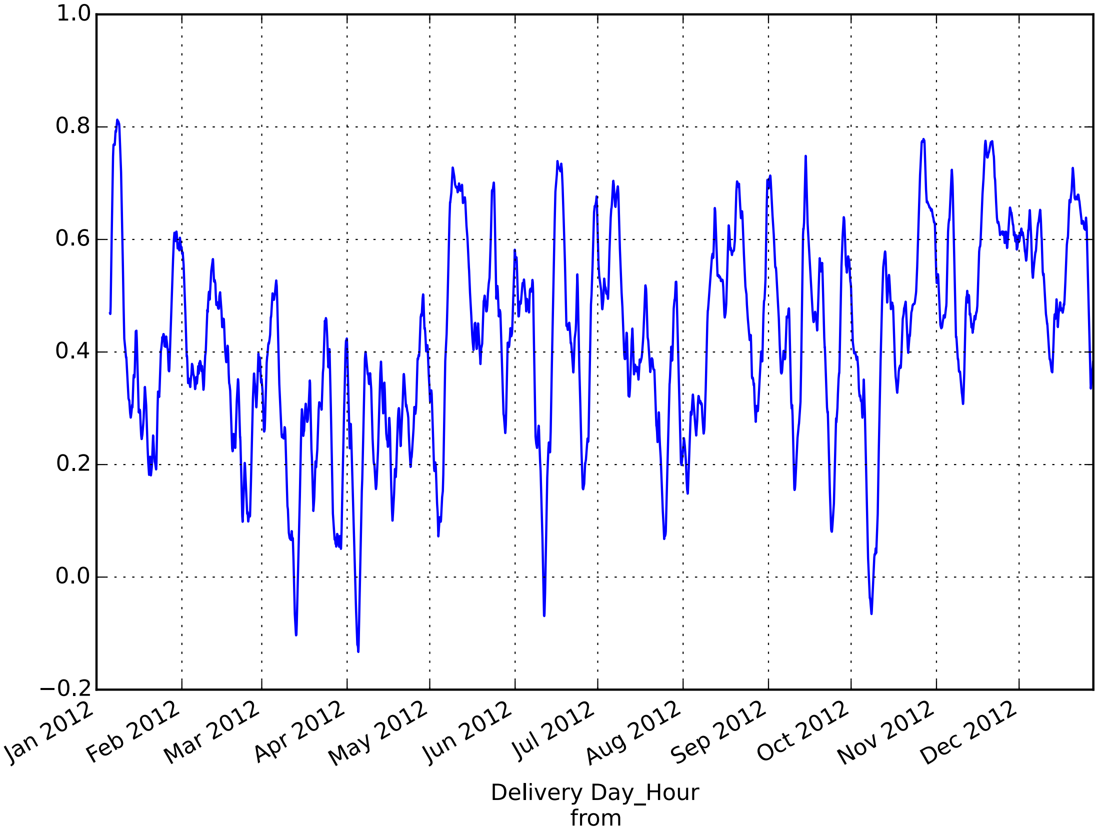
```
Out[261]: <matplotlib.axes._subplots.AxesSubplot at 0x7f276115fcd0>
```



```
In [262]: corr_vol_avgp20 = rolling_corr(df.VolumeMW, df.AveragePriceEUR, window=20)
          corr_vol_avgp200 = rolling_corr(df.VolumeMW, df.AveragePriceEUR, window=200)
          corr_vol_avgp20.plot(alpha=0.7,label='20day correlation')
          corr_vol_avgp200.plot(style='red',label='200day correlation')

Out[262]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2761066550>
```



```
In [263]: rolling_mean(corr_vol_avgp20,window=100).plot()

Out[263]: <matplotlib.axes._subplots.AxesSubplot at 0x7f2761078110>
```



Merging Series

```
In [264]: !ls $data_dir
```

```
energy_intraday_history_2006.xls             energy_spot_historie_2004.xls                energy_spot_histo
energy_intraday_history_2007.xls             energy_spot_historie_2005.xls                Phelix_Quarterly
energy_intraday_history_2008.xls             energy_spot_historie_2006.xls                swiss_power_spot_
energy_intraday_history_2009.xls             energy_spot_historie_2007.xls                swiss_power_spot_
energy_intraday_history_2010.xls             energy_spot_historie_2008.xls                swiss_power_spot_
energy_intraday_history_2011 - Konflikt.xls  energy_spot_historie_2009.xls                swiss_power_spot_
energy_intraday_history_2011.xls             energy_spot_historie_2010.xls                swiss_power_spot_
energy_intraday_history_2012.xls             energy_spot_historie_2011.xls                swiss_power_spot_
energy_spot_historie_2002.xls                     energy_spot_historie_2012 - Konflikt.xls  swiss_power_sp
energy_spot_historie_2003.xls                     energy_spot_historie_2012.xls
```

```
In [265]: dseries = {}
          for filename in os.listdir(data_dir):
              if 'Konflikt' not in filename and 'energy_intraday' in filename:
                  dseries[filename.split('_')[-1][:4]] = pd.read_excel(data_dir+filename,sheetname='Int
                                                                       header=1, parse_dates = [['Deliv
```

```
In [266]: df = concat(dseries.values())
```

```
In [267]: list(df.index.get_duplicates())
```

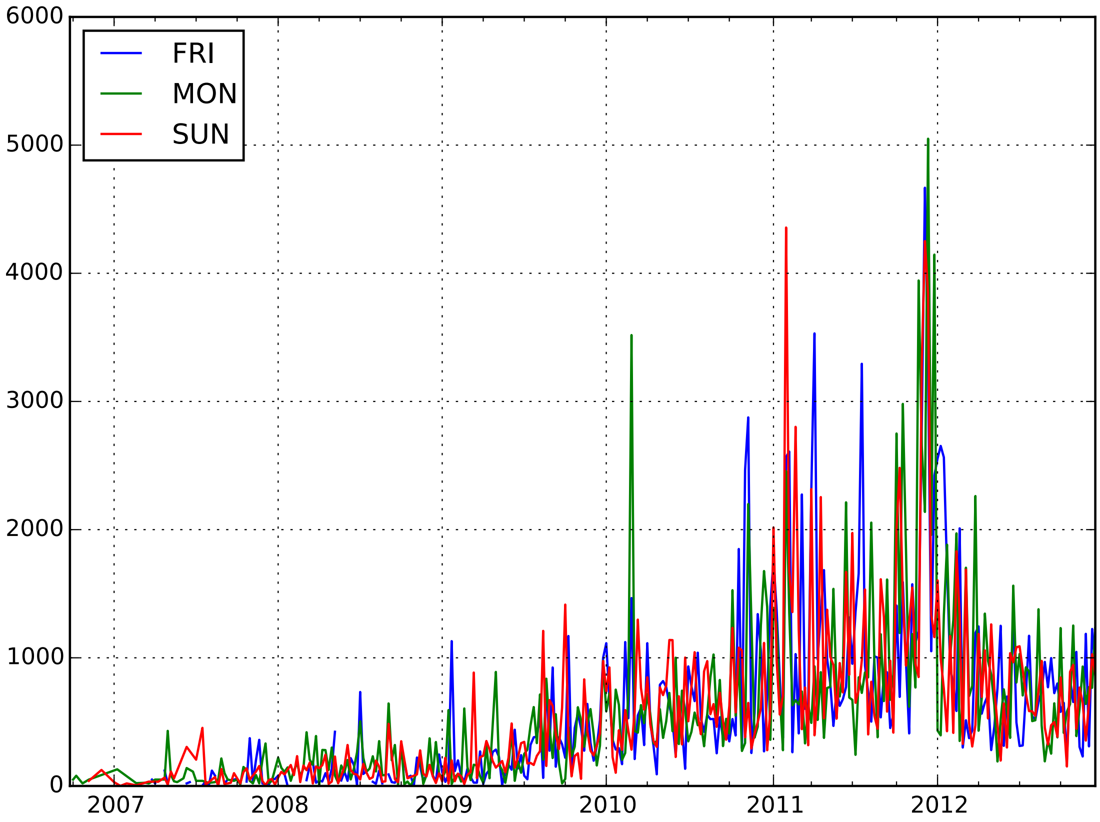
```
Out[267]: [Timestamp('2006-10-29 02:00:00'),
           Timestamp('2007-10-28 02:00:00'),
           Timestamp('2008-10-26 02:00:00'),
           Timestamp('2009-10-25 02:00:00'),
           Timestamp('2010-10-31 02:00:00'),
           Timestamp('2011-10-30 02:00:00'),
           Timestamp('2012-10-28 02:00:00')]
```

```
In [268]: df = df.groupby(df.index).first()
          df.columns = [column.replace(' ','').replace('\n','') for column in df.columns]
```

```
In [269]: from matplotlib.pyplot import *
```

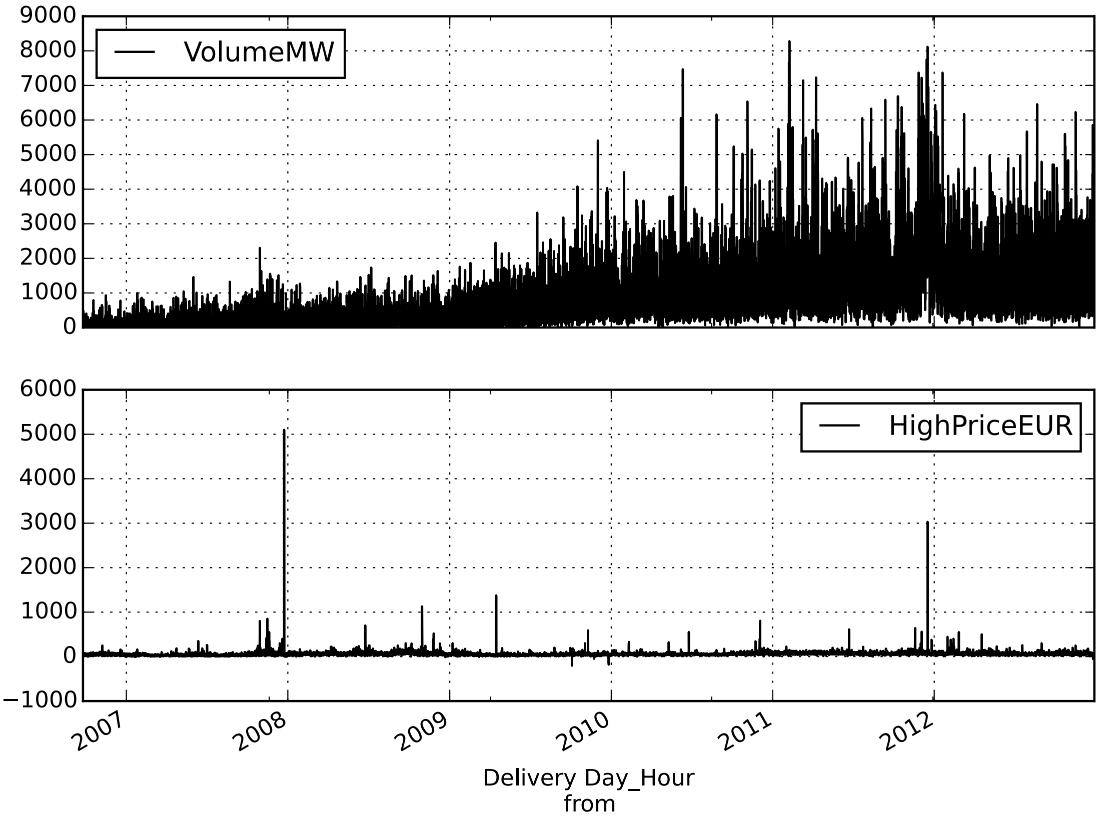
```
In [270]: df.VolumeMW.asfreq('W-FRI').plot()
          df.VolumeMW.asfreq('W-MON').plot()
          df.VolumeMW.asfreq('W-SUN').plot()
          legend(['FRI','MON','SUN'],loc='best')
```

```
Out[270]: <matplotlib.legend.Legend at 0x7f2760872e90>
```



```
In [271]: df.ix[:,['VolumeMW','HighPriceEUR']].plot(subplots=True)
```

```
Out[271]: array([<matplotlib.axes._subplots.AxesSubplot object at 0x7f2760802190>,
                 <matplotlib.axes._subplots.AxesSubplot object at 0x7f2760748650>], dtype=object)
```



3.0.2 Regression

Meteorological historical data for Düsseldorf

```
In [272]: !head './data/kl_10400_00_akt_txt.txt'
```

```
KL01192200001010000101701101861102201101924  691  251 444  38 1  511  641  271  424  41 6  60 6  24 6 7
KL01192200001020000102371102331102251102324  791  271 524   2 1  391  731  741  654  36 6  59 6  63 6 7
KL01192200001030000101981101771101581101784  911  691 224  67 1  781  871  711  774  61 6  70 6  54 6 8
KL01192200001040000101301100891100971101054  921  661 264  58 1  681  771  831  784  63 6  72 6  64 6 9
KL01192200001050000101651101581101421101554  861   41 824 -22 1   71  801  611  524   2 6  55 6  47 6 5
KL01192200001060000101221101091101351101224 1041  431 614  25 1  501  971  781  764  40 6  74 6  70 6 7
KL01192200001070000101771101731101611101704  871  471 404  42 1  551  871  611  664  49 6  64 6  43 6 8
KL01192200001080000101341101361101531101414  791  541 254  45 1  591  681  681  664  39 6  61 6  62 6 6
KL01192200001090000101701102051102541102104  681   51 634   4 1  261  431  161  254  23 6  40 6  12 6 7
KL01192200001100000102991103171103261103144  241   41 204  10 1   81  121  121  114   6 6  10 6   9 6 6
```

```
In [273]: from IPython.display import display, HTML
          HTML('http://www.dwd.de/bvbw/generator/DWDWWW/Content/Oeffentlichkeit/KU/KU2/KU21/klimadaten/g
```

```
Out[273]: <IPython.core.display.HTML at 0x7f2761083710>
```

```
In [274]: table_description = pd.read_html('http://www.dwd.de/bvbw/generator/DWDWWW/Content/Oeffentlichk
```

```
In [275]: table_description.head()
```

```
Out[275]:    KL    KE KE.1 Kennung fuer das Datenkollektiv Unnamed: 4     Unnamed: 5  \
          0  KL  STAT   ST                  Stationsnummer       CODE  STATIONSLISTE
          1  KL    JA   JA                            Jahr        NaN            NaN
          2  KL    MO   MO                           Monat        NaN            NaN
          3  KL    TA   TA                             Tag        NaN            NaN
          4  KL   NaN  NaN        numerisches Leerfeld (0)        NaN            NaN

            Standardformat: Klimadaten aus Klimaroutine des DWD (3 Termine: 07,14,21 MOZ, ab 01.01.1987
          0                                                NaN
          1                                                NaN
          2                                                NaN
          3                                                NaN
          4                                                NaN

             X(2)   1 siehe KE_IND  Unnamed: 10
          0  9(5)   3  00001-99999          NaN
          1  9(4)   8    1800-2100          NaN
          2  9(2)  12        01-12          NaN
          3  9(2)  14        01-31          NaN
          4  9(4)  16         0000          NaN
```

```
In [276]: widths = table_description.iloc[:,8].diff().values
          widths
```

```
Out[276]: array([ nan,   5.,   4.,   2.,   2.,   4.,   5.,   1.,   5.,   1.,   5.,
                   1.,   5.,   1.,   4.,   1.,   4.,   1.,   3.,   1.,   4.,   1.,
                   1.,   4.,   1.,   4.,   1.,   4.,   1.,   4.,   1.,   4.,   1.,
                   1.,   4.,   1.,   1.,   4.,   1.,   1.,   3.,   1.,   3.,   1.,
                   3.,   1.,   3.,   1.,   3.,   1.,   3.,   1.,   3.,   1.,   3.,
                   1.,   3.,   1.,   3.,   1.,   3.,   1.,   2.,   2.,   1.,   2.,
                   2.,   1.,   2.,   2.,   1.,   3.,   1.,   2.,   1.,   2.,   1.,
                   2.,   1.,   2.,   1.,   2.,   1.,   2.,   1.,   2.,   1.,   2.,
                   1.,   2.,   1.,   3.,   1.,   3.,   1.,   1.,   2.,   1.,   2.,
                   1.,   2.,   1.,   2.,   1.,   2.,   1.,   2.,   1.,   2.,   1.,
                   2.,   1.,   2.,   1.,   4.,   1.,   1.,   4.,   1.,   1.,   4.,
                   1.,   1.,   4.,   1.,   1.,   3.,   1.,   1.,   3.,   1.,   1.,
                   3.,   1.,   4.,   1.,   5.,   1.,   5.])
```

```
In [277]: col_names = table_description.ix[:,3]
          widths=[2,5,4,2,2,4,5,1,5,1,5,1,5,1,4,1,4,1,3,1,4,1,1,4,1,4,1,4,1,4,1,4,1,1,4,1,1,4,1,1,3,1,3
          df_temp = pd.read_fwf('./data/kl_10400_00_akt_txt.txt',widths=widths)
```

```
In [278]: df_temp.head()
```

```
Out[278]:    KL  01192  2000  01  01.1  0000  10170  1  10186  1.1  ...    95  4.9  92  \
          0  KL   1192  2000   1     2     0  10237  1  10233    1  ...    85    4  87
          1  KL   1192  2000   1     3     0  10198  1  10177    1  ...    78    4  78
          2  KL   1192  2000   1     4     0  10130  1  10089    1  ...    76    4  87
          3  KL   1192  2000   1     5     0  10165  1  10158    1  ...    81    4  80
          4  KL   1192  2000   1     6     0  10122  1  10109    1  ...    90    4  83

             4.10  86.1  1.9  95.1  1.10  95.2  1.11
          0     4    96    1    80     1    85     1
          1     4    78    1    79     1    78     1
          2     4    93    1    93     1    76     1
```

```
3      4    92    1    68     1    80     1
4      4    85    1    73     1    89     1

[5 rows x 62 columns]
```

```
In [279]: df_temp = pd.read_fwf('./data/kl_10400_00_akt_txt.txt',widths=widths,header=None,parse_dates
          df_temp.ix[:5,[14,16,29]]
```

```
Out[279]:             14  16  29
          2_3_4
          2000-01-01  69  25  42
          2000-01-02  79  27  65
          2000-01-03  91  69  77
          2000-01-04  92  66  78
          2000-01-05  86   4  52
```

```
In [280]: df_temp = df_temp[[14,16,29]].apply(lambda x: x/10.)
          df_temp.head()
```

```
Out[280]:              14   16   29
          2_3_4
          2000-01-01  6.9  2.5  4.2
          2000-01-02  7.9  2.7  6.5
          2000-01-03  9.1  6.9  7.7
          2000-01-04  9.2  6.6  7.8
          2000-01-05  8.6  0.4  5.2
```

```
In [281]: df_temp.columns = ['HighTemp','LowTemp','MeanTemp']
          df_temp.head()
```

```
Out[281]:             HighTemp  LowTemp  MeanTemp
          2_3_4
          2000-01-01       6.9      2.5       4.2
          2000-01-02       7.9      2.7       6.5
          2000-01-03       9.1      6.9       7.7
          2000-01-04       9.2      6.6       7.8
          2000-01-05       8.6      0.4       5.2
```

```
In [282]: df_temp.index.name = 'Date'
          df_temp.head()
```

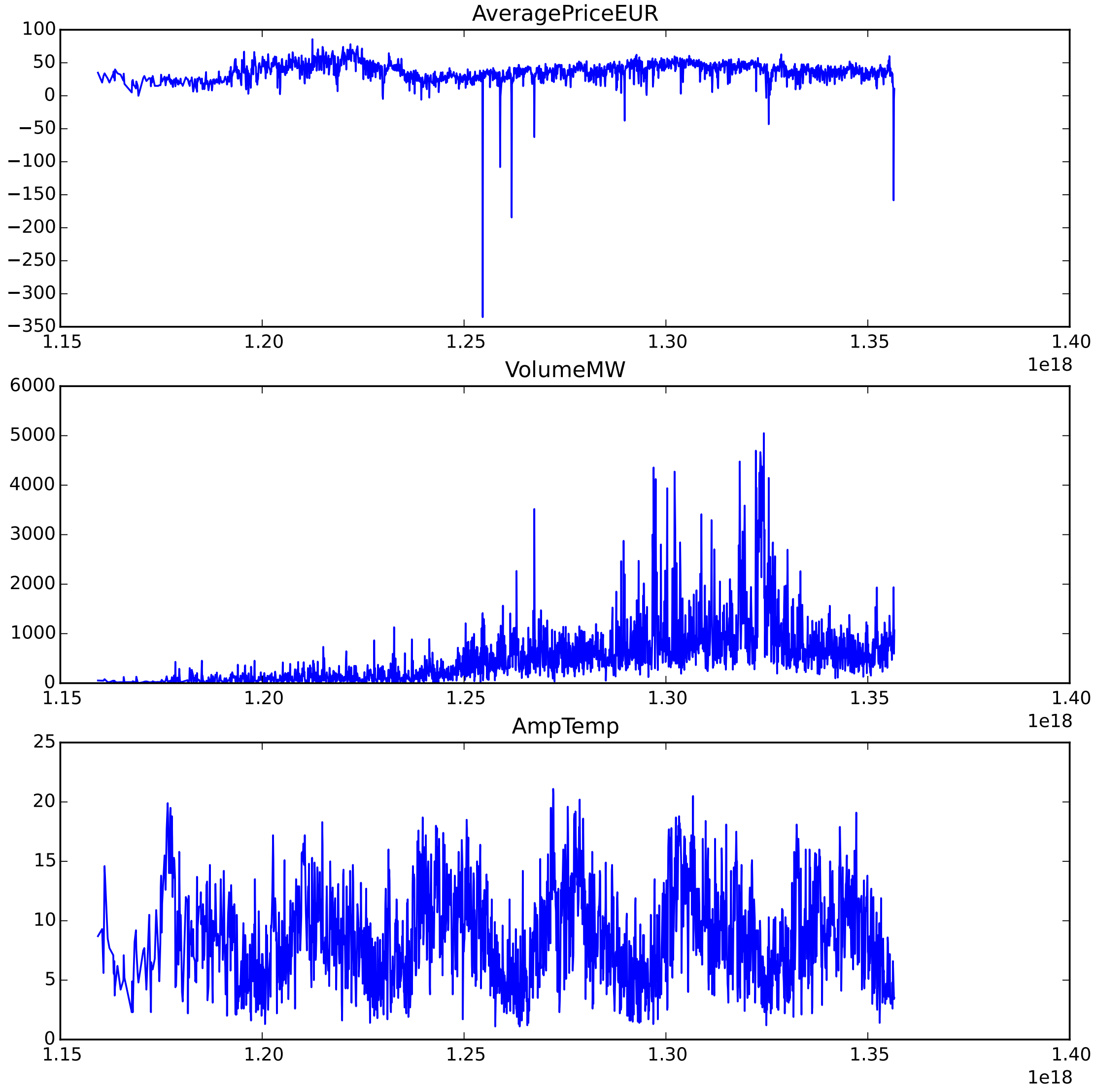
```
Out[282]:             HighTemp  LowTemp  MeanTemp
          Date
          2000-01-01       6.9      2.5       4.2
          2000-01-02       7.9      2.7       6.5
          2000-01-03       9.1      6.9       7.7
          2000-01-04       9.2      6.6       7.8
          2000-01-05       8.6      0.4       5.2
```

```
In [283]: df2 = df.join(df_temp, how='left')
```

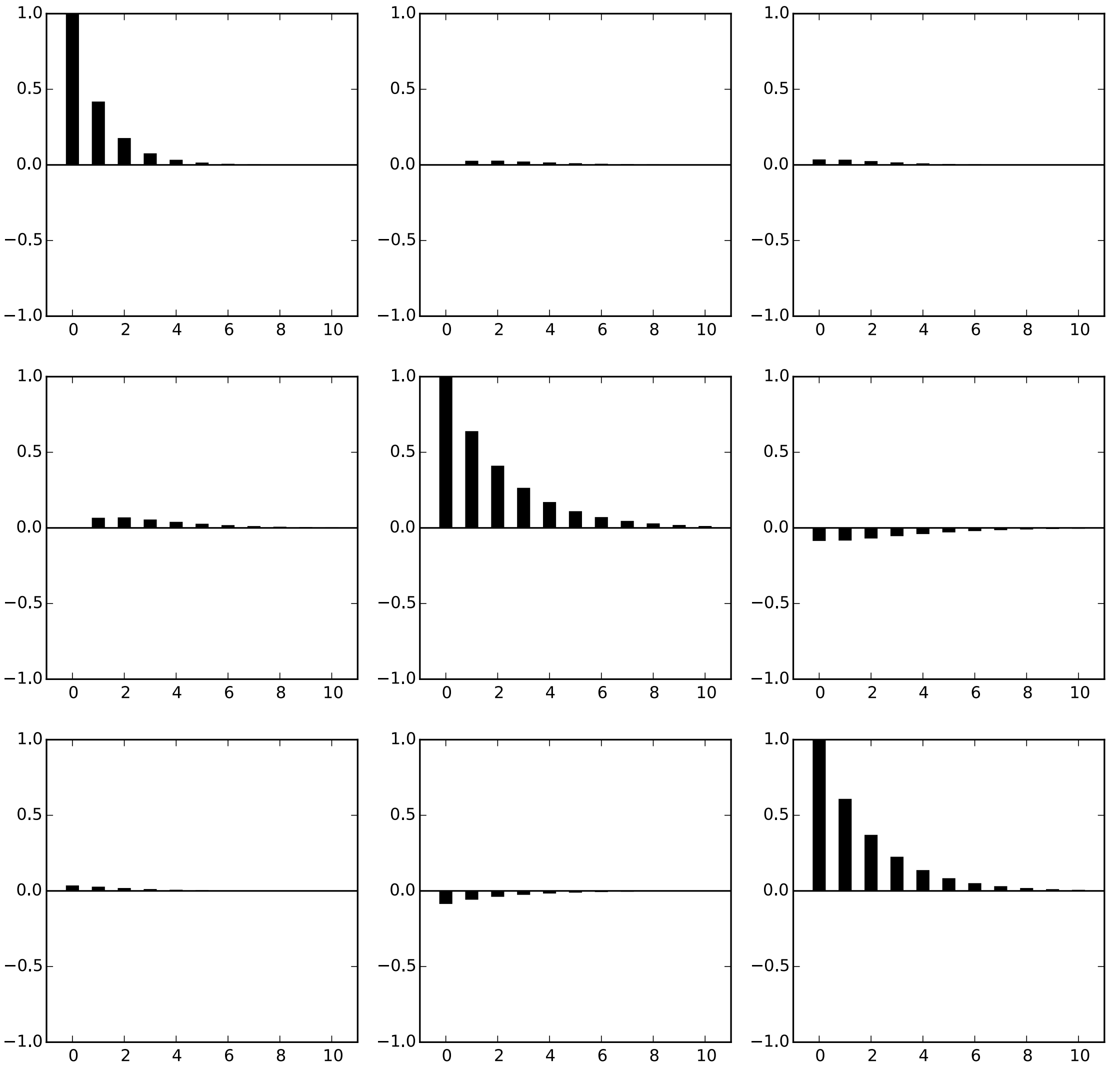
```
In [284]: from statsmodels.tsa.api import *
```

```
In [285]: df2['AmpTemp'] = df2.HighTemp-df2.LowTemp
          data = df2[['AveragePriceEUR','VolumeMW','AmpTemp']].asfreq('D')
          model = VAR(data,missing='drop')      # NaN will produce LinalgError, hence the missing='drop'
```

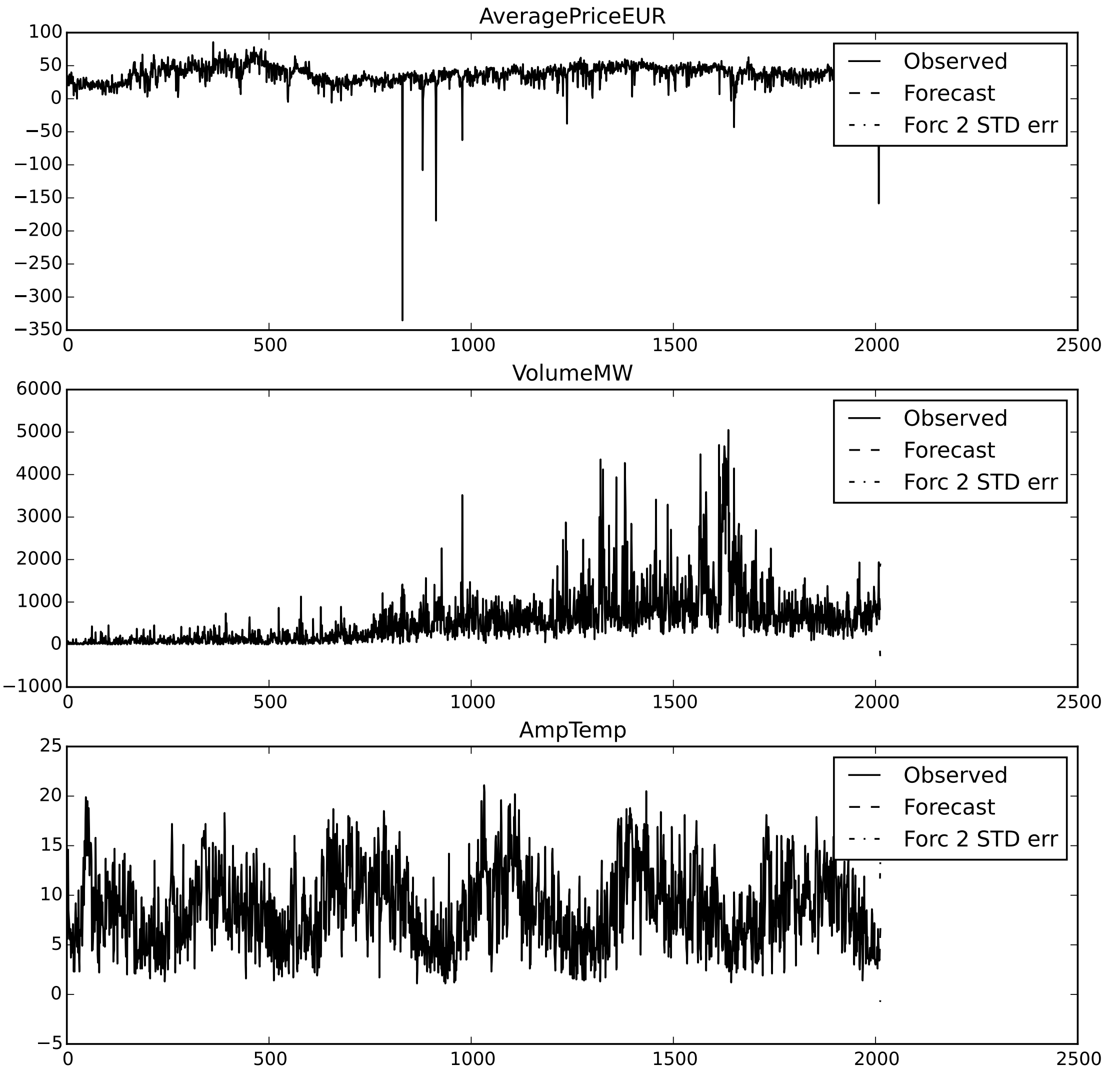
```
In [286]: results= model.fit()
          results.plot()
```



```
In [287]: results.plot_acorr()
```



```
In [288]: lag_order = results.k_ar
          results.forecast(data.values[-lag_order:],2)

Out[288]: array([[  25.47955327,  844.24536267,    5.46105252],
                 [  31.77881602,  731.28635887,    6.682358  ]])

In [289]: results.plot_forecast(2)
          legend(loc='best')

Out[289]: <matplotlib.legend.Legend at 0x7f274ffa3a10>
```



```
In [290]: model = pd.ols(y=df2.AveragePriceEUR, x = df2[['AmpTemp','VolumeMW']])
```

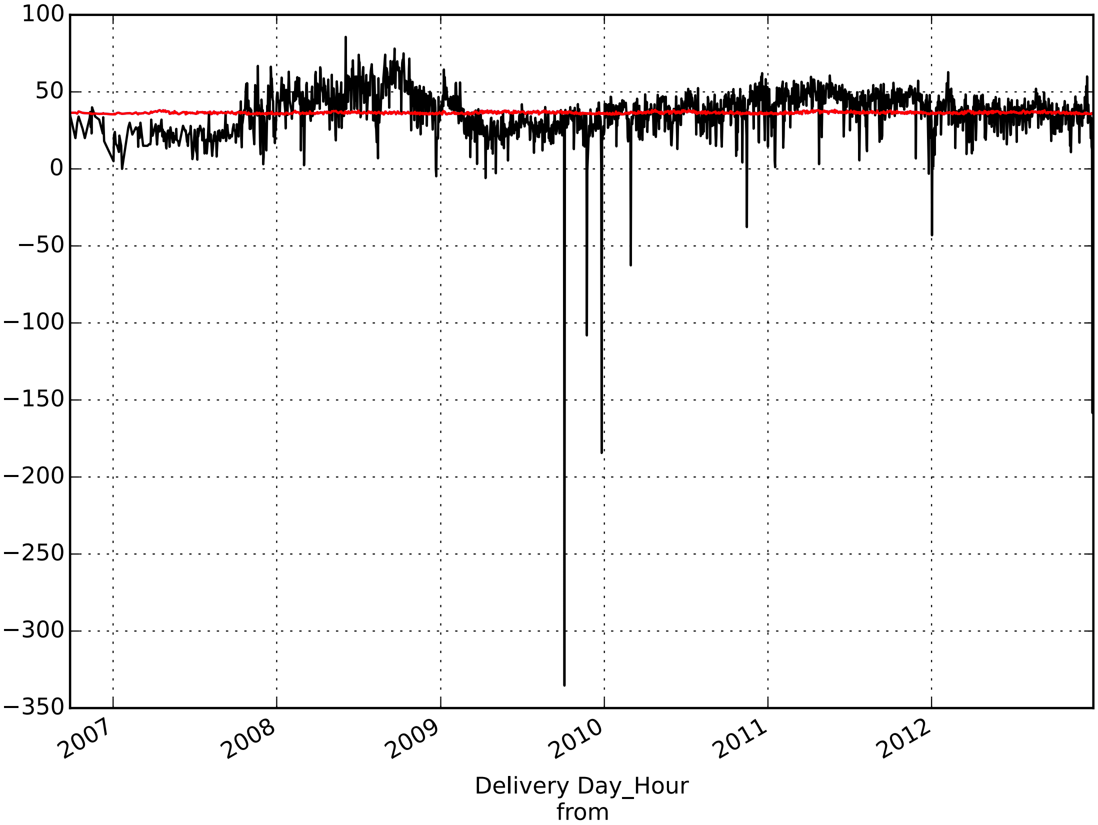
```
In [291]: print(model)
```

```
-------------------------Summary of Regression Analysis-------------------------

Formula: Y ~ <AmpTemp> + <VolumeMW> + <intercept>

Number of Observations:         2011
Number of Degrees of Freedom:   3

R-squared:         0.0013
Adj R-squared:     0.0003

Rmse:             17.0065

F-stat (2, 2008):     1.3320, p-value:     0.2642
```

```
Degrees of Freedom: model 2, resid 2008

-----------------------Summary of Estimated Coefficients------------------------
      Variable       Coef    Std Err     t-stat    p-value    CI 2.5%   CI 97.5%
--------------------------------------------------------------------------------
       AmpTemp     0.1553     0.0954       1.63     0.1039    -0.0318     0.3423
      VolumeMW     0.0002     0.0006       0.27     0.7891    -0.0010     0.0013
     intercept    35.0410     0.9885      35.45     0.0000    33.1035    36.9784
---------------------------------End of Summary---------------------------------
```

```
In [292]: model.y_fitted.plot()
          model.y.plot(style='k')
          model.y_predict.plot(style='red')
```

```
Out[292]: <matplotlib.axes._subplots.AxesSubplot at 0x7f275416e710>
```



```
In [3]: %reload_ext version_information

        %version_information numpy, scipy, matplotlib, pandas, statsmodels
```

```
Out[3]:
```

Software	Version
Python	2.7.8 —Anaconda 2.1.0 (64-bit)— (default, Aug 21 2014, 18:22:21) [GCC 4.4.7 20120313 (Red Hat 4.4.7-
IPython	2.3.0
OS	posix [linux2]
numpy	1.9.1
scipy	0.14.0
matplotlib	1.4.2
pandas	0.15.0
statsmodels	0.5.0
Thu Nov 13 10:39:56 2014 CET	

The full notebook can be downloaded here, or viewed statically on nbviewer

```
In [148]: df = pd.read_html('lista.html')[6]
          df.head()
```

```
Out[148]:   Stations-Kennziffer  Klima-Kennung ICAO-Kennung    Stationsname  \
          0               10501           2205          NaN          Aachen
          1               10505           2206          NaN  Aachen-Orsbach
          2               10291           3058          NaN      Angermünde
          3               10091           3005          NaN          Arkona
          4               10852           4128         EDMA        Augsburg

             Stationshöhe in Metern geogr. Breite geogr. Länge  \
          0                     202        50° 47'      06° 05'
          1                     231        50° 47'      06° 01'
          2                      54        53° 01'      13° 59'
          3                      42        54° 40'      13° 26'
          4                     462        48° 25'      10° 56'

            Automat für Lufttemperatur\n\nseit:\n  Beginn Klimareihe
          0                            01.07.1993               1891
          1                                   NaN               2011
          2                                   NaN               1947
          3                                   NaN               1947
          4                            10.11.1996               1947
```

```
In [1]: from IPython.core.display import HTML
        def css_styling():
            styles = open("./styles/custom.css", "r").read()
            return HTML(styles)
        css_styling()
```

```
Out[1]: <IPython.core.display.HTML at 0x7fc1784fd7d0>
```

Back to top

```
In [ ]:
```