HydroGeoSines

Signal In the Noise Exploration Software for hydrogeological datasets

Code demonstration

First, we import the sines package and create a new instance of a sines model.

In [1]:
```
import hydrogeosines

s = hydrogeosines.model()
```

Next, we import groundwater pressure data, take a look at the first ten values, and then visualise the full dataset.

In addition to the barometric pressure dataset, we can assess one of three groundwater pressure datasets: RN027214, RN039613, or RN039617.

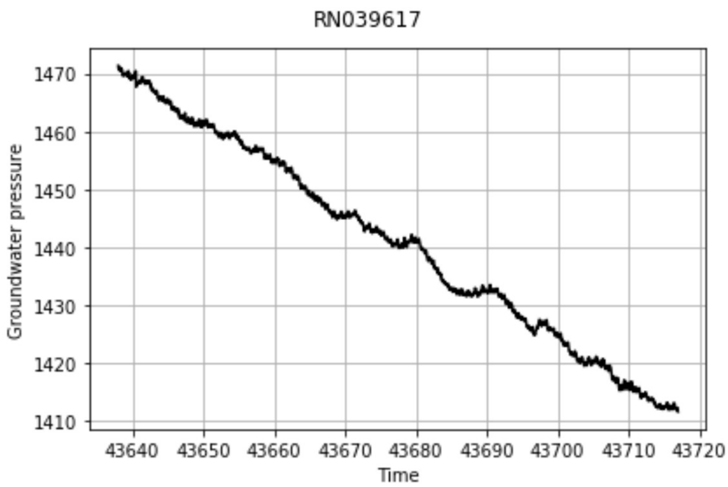
In [2]:
```
s.wd = 'hydrogeosines/test_data/port_keats/'
s.id = 'RN039617'

s.get_GW()

s.print_GW(10)

s.plot_GW(pname=None)
```

```
      time  pressure
 43638.000  1471.346
 43638.040  1470.830
 43638.080  1470.910
 43638.130  1470.803
 43638.170  1471.088
 43638.210  1470.652
 43638.250  1470.440
 43638.290  1470.916
 43638.330  1470.855
 43638.380  1470.922
```



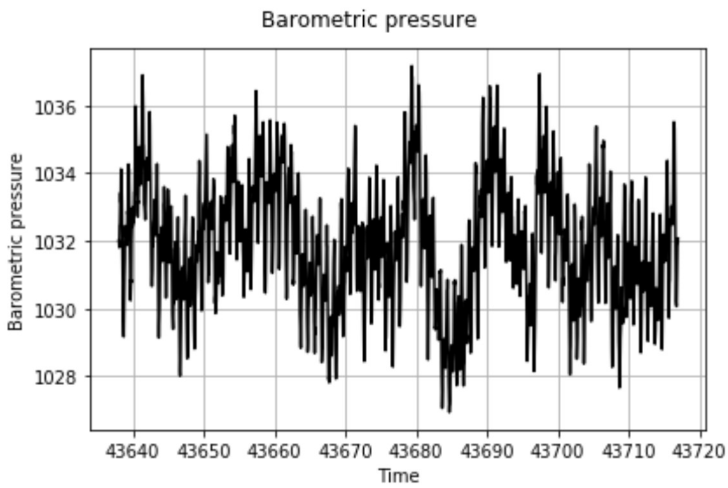
Next, we import groundwater pressure data, take a look at the first ten values, and then visualise the full dataset.

```
In [3]: s.get_BA()

        s.print_BA(10)

        s.plot_BA(pname=None)
```

```
      time  pressure
 43638.000  1032.010
 43638.040  1032.004
 43638.080  1031.804
 43638.130  1032.470
 43638.170  1032.487
 43638.210  1033.031
 43638.250  1033.424
 43638.290  1033.127
 43638.330  1034.121
 43638.380  1033.933
```



Next, we calculate temporal differences between consecutive groundwater and barometric pressure measurements.

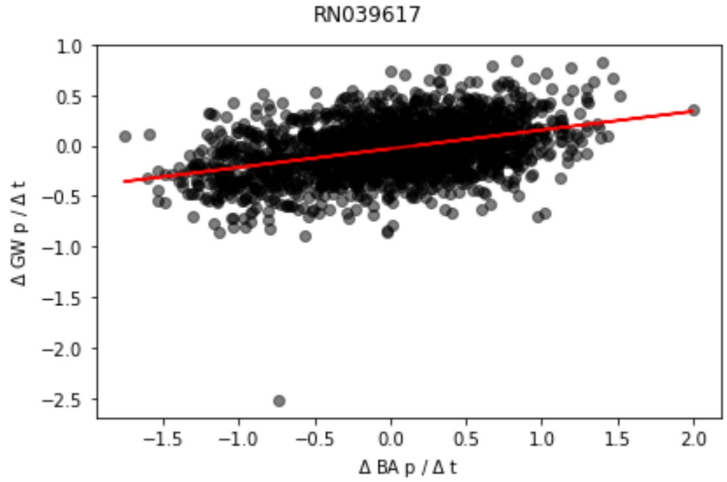
```
In [4]: s.calc_delta_GW()
        s.calc_delta_BA()
```

We then calculate the linear regression of (1) temporal differences of groundwater pressure against (2) temporal differences of barometric pressure. We print the resulting parameters and plot both the data and the estimated linear function.

```
In [5]: s.calc_linear_GW()

        %matplotlib inline
        s.plot_linregress(pname=None)
```

```
Slope = 0.185, intercept = -0.032
```



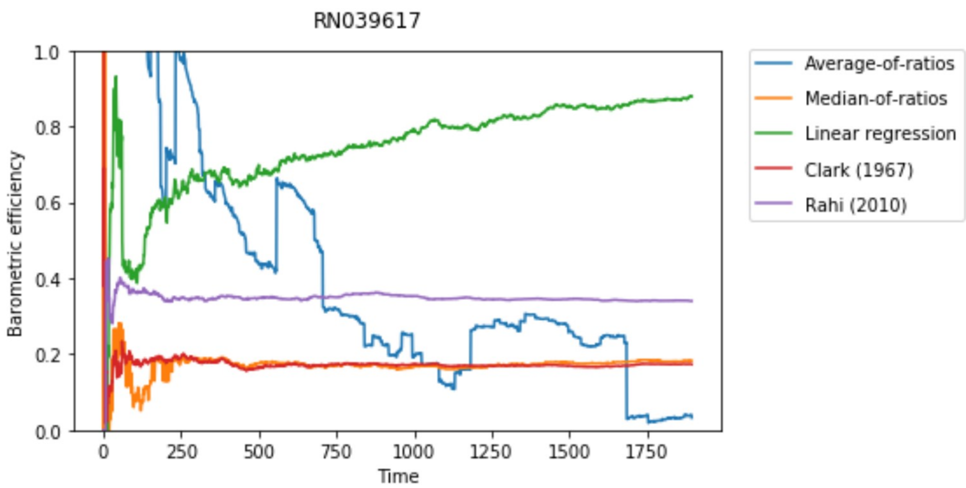
Next we estimate barometric efficiency values using a range of time domain methods; i.e., the average-of-ratios, median-of-ratios, linear regression, Clark (1967), and Rahi (2010) methods.

```
In [6]: s.calc_BE_AoR()
        s.calc_BE_MoR()
        s.calc_BE_LR()
        s.calc_BE_Clark()
        s.calc_BE_Rahi()
```

```
BE (AoR) = 0.033
BE (MoR) = 0.183
BE (LR)  = 0.879
BE (Clk) = 0.173
BE (Rah) = 0.340
```

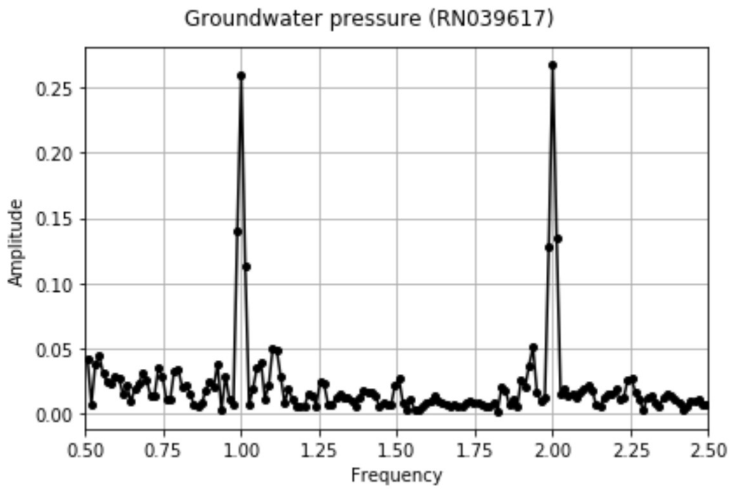
We can also examine the convergence of each of these metrics over the length of the input datasets.

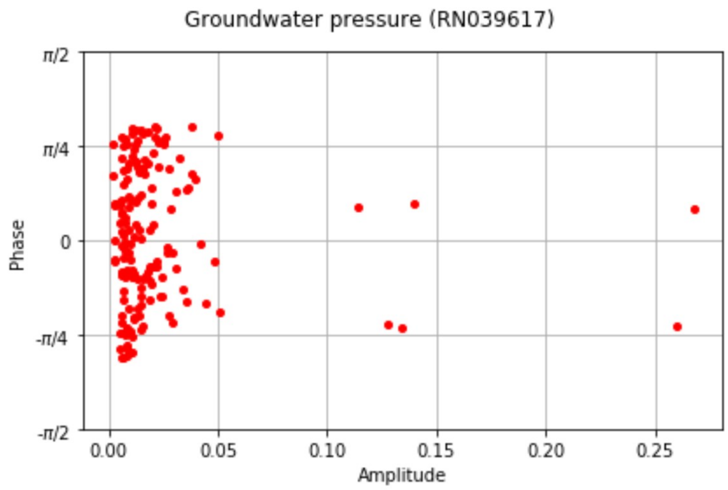
```
In [7]: s.plot_BE_vs_time(pname=None)
```



Next, we calculate the discrete Fourier transform of the groundwater pressure dataset and plot the resulting amplitude spectrum, as well as an amplitude versus phase plot, in order to identify dominant component frequencies.

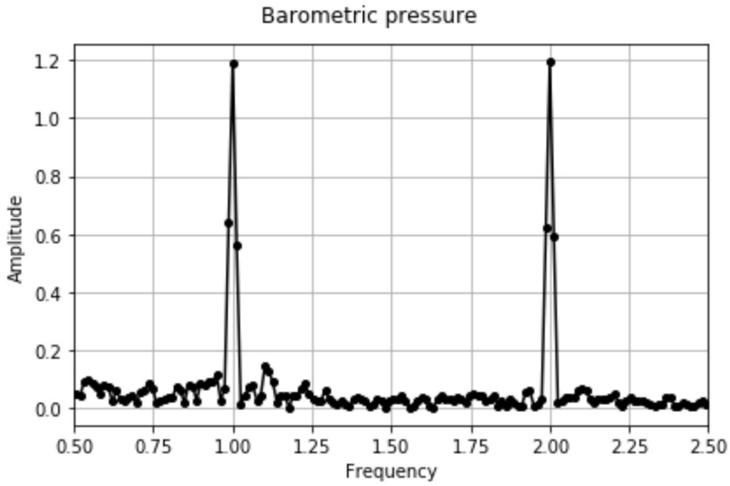
In [8]:
```
s.calc_ft_GW()
%matplotlib inline
s.plot_ft_avf_GW(pname=None)
s.plot_ft_pva_GW(pname=None)
```

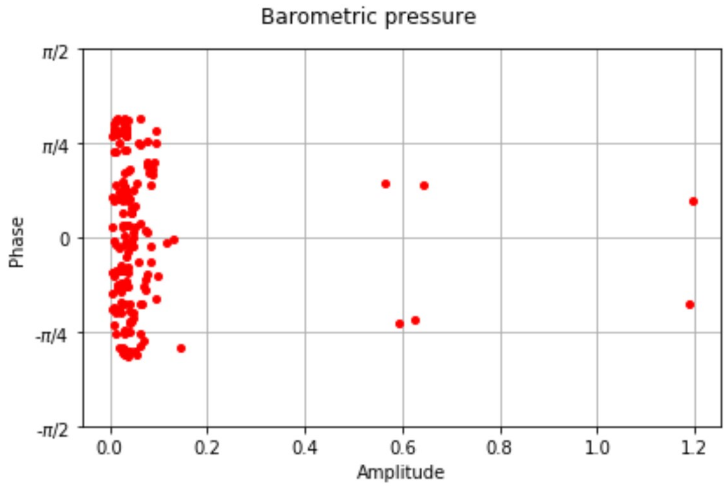




Next, we repeat the analysis for the barometric pressure dataset.

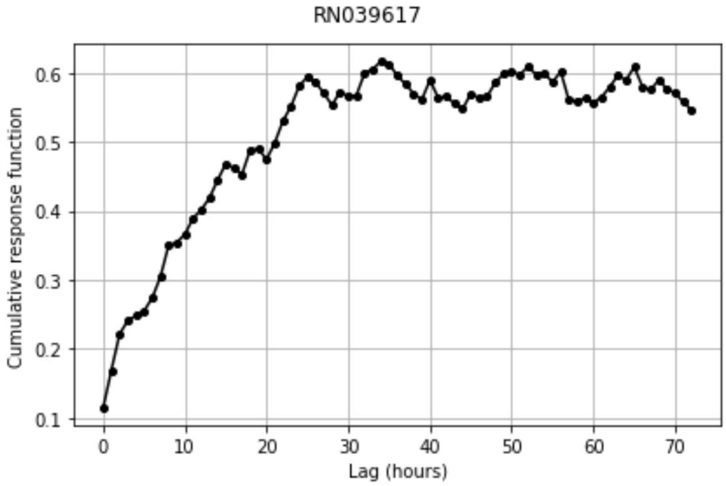
In [9]:
```
s.calc_ft_BA()
%matplotlib inline
s.plot_ft_avf_BA(pname=None)
s.plot_ft_pva_BA(pname=None)
```

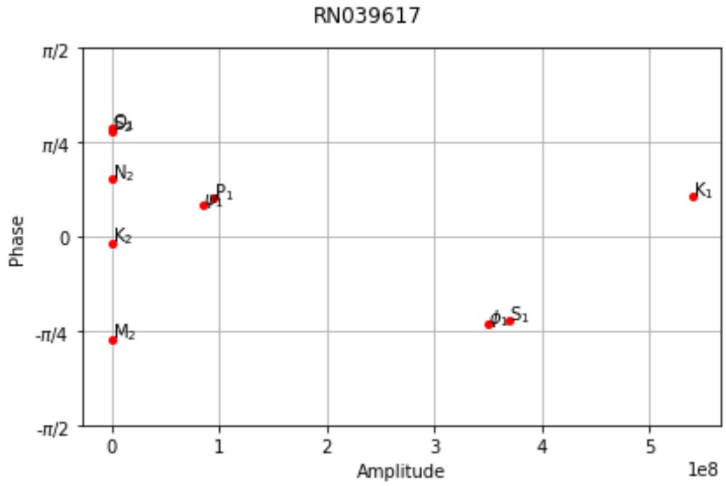




Next, we use regression deconvolution to estimate the amplitudes and phases of specific frequencies of interest, and to estimate a cumulative response function.

In [10]:
```
s.calc_regress_deconv()
s.plot_regress_deconv_crf(pname=None)
s.plot_regress_deconv_pva(pname=None)
```





In [ ]: