ExerciseSineZ: Introduction

The Discrete Fourier Transform (DFT) analysis for the $n = 100$ observations in **Scargle.dat** is performed in **ExerciseScargle**. The DFT periodogram is shown in Figure 1

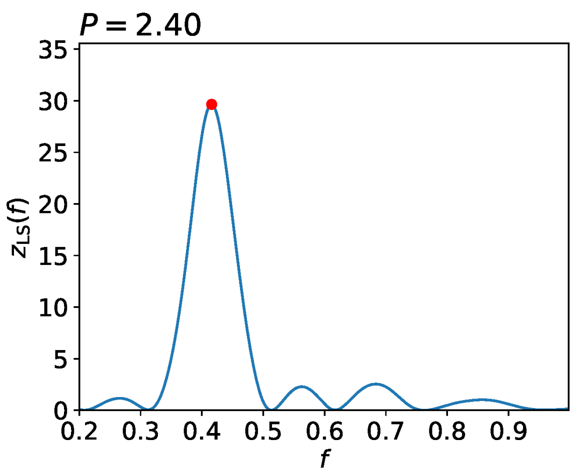

Figure 1: DFT periodogram $z_{\rm LS}(f)$ for $n = 100$ observations in **Scargle.dat**.

The highest $z_{LS}(f)$ peak is at $P = 2.40 \equiv f = 0.41$.

Let us assume that the correct model for the **Scargle.dat** data is

$$g(t, \bar{\beta}) = \beta_1 + \beta_2 \cos(2\pi\beta_4 t) + \beta_3 \sin(2\pi\beta_4 t), \quad (1)$$

where the frequency $\beta_4 = f$ is unknown. This model is **non-linear**, and therefore the solution for the free parameters $\bar{\beta} = [\beta_1, \beta_2, \beta_3, \beta_4]$ is not **unambiguous**. If the frequency f is fixed to any tested constant numerical value, the model

$$g(t, \bar{\beta}) = \beta_1 + \beta_2 \cos(2\pi f t) + \beta_3 \sin(2\pi f t), \quad (2)$$

becomes **linear**, and the solution for the free parameters $\bar{\beta} = [\beta_1, \beta_2, \beta_3]$ is **unambiguous**. For example, the numerical values for the tested frequencies f can be produced using the following subroutine

```
# ------------------------------------------------------------------------
# - Tested frequencies
# ------------------------------------------------------------------------
def Frequencies(T,PMIN,PMAX,OFAC):
    FMIN=1./PMAX
    FMAX=1./PMIN
    DELTAT=np.max(T)-np.min(T)
    f0=1./DELTAT
    FSTEP=f0/OFAC
    m=np.floor((FMAX-FMIN)/FSTEP)+1
    F=FMIN+np.arange(m)*FSTEP
    return F
# ------------------------------------------------------------------------
```

where the tested frequencies are evenly spaced between $f_{\min} = 1/P_{\max}$ and $f_{\max} = 1/P_{\min}$. The same subroutine produces the tested frequencies that are used to compute the DFT periodogram $z_{\rm LS}(f)$ in Figure 1.

The next problem is **how** to introduce the changing tested frequency $f =$ **F** into the **python** model $g(t, \bar{\beta}) =$ **MODEL(T,BETA)**. This is achieved with following two subroutines copied as such from DCM code **dcm.py**.

```
# ------------------------------------------------------------------------
# - Writing tested frequency or frequencies to ALLF.dat
# ------------------------------------------------------------------------
def WriteALLF(ALLF):
    code1=open('ALLF.dat',"w")
    if (np.size(ALLF) < 1.5):
        code1.write("%15.15e\n" %(ALLF))
    else:
        for i in range(np.size(ALLF)):
            code1.write("%15.15e\n" %(ALLF[i]))
    code1.close()
    return
# ------------------------------------------------------------------------
# - Reading tested frequency or frequencies from ALLF.dat
# ------------------------------------------------------------------------
def ReadALLF():
    ALLF=-1.
    file = open('ALLF.dat', 'r')
    i=0
    rivi = file.readline()
    while (len(rivi) > 0):
        rivi=rivi.rstrip('\n')
        ALLF=np.append(ALLF,np.float(rivi))
        rivi = file.readline()
    ALLF=ALLF[1:]
    return ALLF
# ------------------------------------------------------------------------
```

Now any tested frequency $f = \mathbf{F}$ can be introduced into model using

```
# ------------------------------------------------------------------------
# - Model g(t)
# ------------------------------------------------------------------------
def Model(T,BETA):
    F=ReadALLF()
    ny=2.*np.pi*F*T
    G=BETA[0]+BETA[1]*np.cos(ny)+BETA[2]*np.sin(ny)
    return G
# ------------------------------------------------------------------------
```

For any tested frequency f, the DCM test statistic is

$$z(f) = \sqrt{\chi^2/n}, \tag{3}$$

where

$$\chi^2 = \sum_{i=1}^{n} \frac{\epsilon_i^2}{\sigma_i^2} \tag{4}$$

and the residuals are

$$\epsilon_i = y(t_i) - g(t_i) = y_i - g_i. \tag{5}$$

For example, this $z(f) = \mathbf{Z}$ periodogram can be computed using a subroutine that begins with the following **python** commands

```
# ------------------------------------------------------------------------
# - Periodogram z(f) = R/n
# ------------------------------------------------------------------------
def periodogram(PMIN,PMAX,OFAC,T,Y,EY):
    F=Frequencies(T,PMIN,PMAX,OFAC)          # frequencies
    Z=0.*F                                   # empty periodogram
    for i in range(np.size(F)):
        WriteALLF(F[i])
```

Note that the last lines of this subroutine are missing, because your task in this exercise is write the correct **python** code for the solution of this DCM periodogram $z(f) = \mathbf{Z}$.

ExerciseSineZ: Problem

1. Copy the data file **Scargle.dat** from homepage.
2. Edit your **python** program named **ExerciseSineZ.py**, which computes the DCM periodogram $z(f)$ (Eq. 3) for the data in **Scargle.dat**. Your periodogram should resemble the one shown in Figure 2.
3. Use the following format in your periodogram subroutine **F,Z,FBEST,ZBEST=periodogram(PMIN,PMAX,OFAC,T,Y,EY)** where the input is P_{min} = **PMIN=1.0**, P_{max} = **PMAX=5.0**, OFAC= **OFAC=40**, t_i= **T**, y_i = **Y** and σ_i = **EY**. The output should be f = **F**, $z(f)$ = **Z**, best period frequency f_{best} = **FBEST** and $z(f_{\text{best}})$ = **ZBEST**. Mark this best period into your plot **ExerciseSineZ.eps** (Figure 2: red circle).
4. Send your program **ExerciseSineZ.py** and your plot **ExerciseSineZ.eps** to the asssistant.

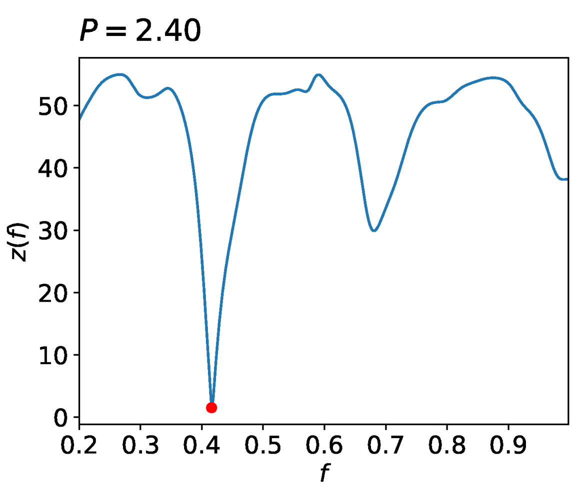

Figure 2: DCM periodogram $z(f)$ for $n = 100$ observations in **Scargle.dat**.

Tips

- Use exactly the same command lines as in subroutines

Frequencies(...)
WriteALLF(...)
ReadALLF(...)
MODEL(...)
periodogram(...)

- Good names for your missing subroutines would be

T,Y,EY=Readfile(...)
Funct(...)
BETA=LSF(...)

and some plotting subroutine, like **plotperiodogram(...)**