Advanced Topics in Algebra – Lab#4

dr Michał Jarema

Topic: Fourier transform (2h)

Task 1 – Fourier series example

Define a vector `t` with numbers from 0 to 5 by 0.01
Tabulate values of functions: $f_n(t)=1/n*\sin(2*pi*n*t)$ for n from 1 to 20.

Plot functions f_1, f_1+f_2, $f_1+f_2+f_3$, …

Task 2

Generate data for two periods of triangular wave. Plot it. Calculate FFT and plot the complex module vs. frequency. Use the template from the NOTES part.

Do it also for square wave and random noise signal (`rnorm()`).

Task 2a (*) For noise signal: nullify the high frequency part of Fourier transform (keep the transform data symmetric!) Make the inverse transform and compare with original signal.

Task 3

Use the default dataset `AirPassengers`

```
AP <- AirPassengers
print(AP)
plot(AP)
class(AP)
```

It is of the type `ts` which means "time series" [7]

```
start(AP)
end(AP)
frequency(AP)   #frequency per year
```

Time series is very useful in R,[8] but sometimes requires advanced functions (colnames() etc. do not work).
You can always extract the raw data from it:

```
raw.data <- AP[1:length(AP)]
class(raw.data)
```

Plot the Fourier transform and interpret the position of the peak. Repeat after subtracting the mean value from the data.
(*)Calculate `log10` of the original data and then detrend it. Make Fourier transform of it and interpret the position of the peak.

Task 4 - The Parseval's theorem states that

$$\sum_{n=0}^{N-1} |x[n]|^2 = \frac{1}{N} \sum_{k=0}^{N-1} |X[k]|^2$$

where $X[k]$ is the DFT of $x[n]$, both of length N.

Loosely speaking, it says that the sum (or integral) of the square of a function is equal to the sum of the square of its transform. In physical terms: the total energy of a signal in time domain equals the total energy of its frequency spectrum. Note that the normalization factor 1/N may be different (eg. it may be equal 1) depending on the normalization done in FFT.

Check if it works for your data and its Fourier transform.

Task 5*

Use Fourier transform to detect seasonality in some real data which you found. You may need to normalize and detrend the data, and deal with missing values.

Some interesting datasets can be found here:
https://www.kaggle.com/datasets
https://catalog.data.gov/dataset

NOTES to list #3

Interesting explanations of Fourier transform[9] [10]

Example with R code – use this as your template [11]

```
plot.frequency.spectrum <-
function(X.k, xlimits=c(0,length(X.k)/2)) {
  #horizontal axis from zero to Nyquist frequency

  #we plot complex modules of data vs. index numbers
  plot.data  <- cbind(0:(length(X.k)-1), Mod(X.k))

  # TODO: why this scaling is necessary?
  plot.data[2:length(X.k),2] <- 2*plot.data[2:length(X.k),2]

  plot(plot.data, t="h", lwd=2, main="",
       xlab="Frequency (number of cycles in the time range)",
       ylab="Strength",
       xlim=xlimits, ylim=c(0,max(Mod(plot.data[,2]))))
}

tt <- seq(from=0, to=10, by=0.01)   #time in sec
freq <- 2.5     #whole cycles per second
mydata <- sin(2*pi*freq * tt)

ft.mydata <- fft(mydata)

plot.frequency.spectrum(ft.mydata)
```

[7] https://datascienceplus.com/time-series-analysis-in-r-part-1-the-time-series-object/
[8] https://datascienceplus.com/time-series-analysis-in-r-part-2-time-series-transformations/
[9] https://betterexplained.com/articles/an-interactive-guide-to-the-fourier-transform/
[10] https://anomaly.io/detect-seasonality-using-fourier-transform-r/
[11] http://www.di.fc.ul.pt/~jpn/r/fourier/fourier.html

Module of complex numbers:

```
Mod()        #note uppercase name!
```

Removing linear trend from data

```
?lm        #linear model
t.var <- 0:(length(mydata)-1)
trend <- lm(mydata ~  t.var)
detrended.mydata <- trend$residuals
plot(mydata,t="l"); abline(trend);
```

Importing data from an online source [12]

without saving to the hard disk. You need an URL link to a text file (does not work for binary or zipped file!)

```
X <-
read.csv(url("http://www.stats.ox.ac.uk/pub/d
atasets/csb/ch11b.dat"))

head(X)
```

[12] https://www.r-bloggers.com/getting-data-from-an-online-source/