TECHNISCHE UNIVERSITÄT WIEN
Vienna University of Technology

institute of telecommunications

Exercise 3: Time Series Analysis

Network Security 1 - Summer Semester 2019 (VU 389.159)

Communication Networks Group at the Institute of Telecommunications
T. Zseby, F. Iglesias, F. Meghdouri, M. Bachl, E. Zeraliu

In this exercise we are going to focus on the analysis of time series. Time series analysis is common in network security to perform the online monitoring of networks. By using such approach it is possible to promptly detect events that cause an important impact on the whole network, like, for instance, DDoS attacks. The time series perspective is also relevant to discover trends, repeating patterns, and characterize how traffic evolves.

Specifically, we are going to study real data belonging to the Internet Background Radiation (aka darkspace). Data has been already preprocessed as shown in previous exercises. You will find the corresponding files with time series in CSV format in your work-folder: [`/home/teamX/working_directory/`].

1 The Internet Background Radiation

The Internet background noise, Internet background radiation or simply the darkspace consists of traffic packets going through the Internet but addressing IP destinations that do not exist. Therefore, all this traffic is undesired and corresponds to missconfigurations, attacks and attack preparations. Studying the darkspace has a considerable value from a security perspective as it gives a reflection of the main types of illegitimate traffic that pollutes the Internet.

The UCSD IP Darkspace Monitor (aka Network Telescope) monitors a large (/8) network of an empty IP Version 4 (IPv4) address space. The monitor records all darkspace traffic destined to this (/8) network. UCSD displays a continuous update of the data observed by the monitor and an initial classification of the data at `http://www.caida.org/data/realtime/telescope/`. This monitor is managed by the Center for Applied Internet Data Analysis (CAIDA), which "conducts network research and builds research infrastructure to support large-scale data collection, curation, and data distribution to the scientific research community". You can visit the CAIDA website for further information in: `https://www.caida.org`.

We will analyze data captured between 2008 and 2018 by the UCSD IP Darkspace Monitor and already aggregated. Note that timestamps in the provided files use the **epoch time format**, i.e. the number of seconds elapsed since January 1, 1970 midnight (UTC).

2 Analyzing the Darkspace Evolution

The file `global_last10years.csv` contains a matrix with samples related to darkspace aggregated data for approximately 10 years. Every row in the file corresponds to a single day, and the information for each specific day is given by 5 comma-separated values (columns), which are:

- Timestamp
- Number of bytes per hour (daily average), henceforth: #bytes/hour (daily ave.)
- Number of packets per hour (daily average), henceforth: #pkts/hour (daily avg.)
- Number of unique IP sources per hour (daily average), henceforth: #uIPs/hour (daily avg.)
- Number of unique IP destinations per hour (daily average), henceforth: #uIPd/hour (daily avg.)

Use MATLAB for solving the exercises and work from the `/home/teamX/working_directory/` folder. Save the required commands in MATLAB *m* files following a clear nomenclature, e.g. `teamX_ex3_1` for the exercise 3.1. **Make sure that you understand the meaning and the goal of every code line and example step before continuing to the next one**.

Listing 1 shows some first steps to prepare the environment, load the data and execute a simple operation (lines are commented after the `%` symbol).[1]

Listing 1: MATLAB initial settings

```
1 > format long % displays values in long format
2 > more off % disables paging of the output in the
      command window
3 > set (gca, 'fontname', 'Helvetica', 'fontsize', 20);
      % sets plot font and size
4 > dataset = csvread ('global_last10years.csv',1,0); %
      loads csv data
5 > ts_packets = dataset(:,3); % takes packets_per_day
      from dataset
```

[1] Broad information about every MATLAB instruction is available on the Internet; we encourage students to check support web pages for a better understanding of the diverse instruction options and functionalities. For instance, suitable tutorials and introductions to MATLAB can be found in [?] and [?].

We are ready to plot the time series of #pkts/hour (daily avg.). Before we plot the data we should transform timestamps into a more convenient format to read the date/time information. To do this we use the function `datenum`, which converts the epoch time into the **datenum format**. The datenum format (internal format used in MATLAB) stores the time as number of days elapsed since Jan 1, 0000 (day 1).

The function `datenum` requires the date/time be entered as separated year, month, day, minutes and seconds fields (Y, M, D, MN, S). To transform epoch time into datenum time we set the values for Y, M, D, H and MN to Jan 1, 1970 midnight (epoch format reference or 0-value) and then add the epoch time as S (seconds). Finally, we use `stem` to plot a stem graph, which represents each value by a vertical line. Listing 2 shows the commands to perform the described task.

Listing 2: Plotting number of packets per hour in a time period in 2012

```
> timestamps = datenum (1970, 1, 1, 0, 0,
    dataset(:,1)); % transforms epoch into datenum
    format
> stem(timestamps, ts_packets/10^6, 'marker', 'none')
    % plots stem graphic
> datetick('x', 'mmm/yy'); % sets x-axis display
    format
> xlabel('days of observed time span') % sets x-axis
    label
> ylabel('#packets [millions]') % sets y-axis label
> title('number of packets per hour (daily average)')
    % set plot title
> grid on % enables grid lines
> set(gca,'layer','top'); % places grid lines on the
    top
> xlim([min(timestamps) max(timestamps)])
```

[rep-25] The steps in Listing 1 and 2 are useful to create a plot that shows the #pkts/hour (daily avg.) seen in the darkspace for the last 10 years. Use the listings as a guidance and prepare the following figures:

1. #pkts/hour (daily avg.), i.e., the given example.
2. #bytes/hour (daily avg.), `dataset(:,2)`.
3. #uIPs/hour (daily avg.), `dataset(:,4)`.
4. #uIPd/hour (daily avg.), `dataset(:,5)`.

The plots obtained in the previous exercise show the evolution of the darkspace throughout time.

Based on such plots, answer the following questions:

- **[rep-26]** Comment briefly (bullet list) at least 3 things that, in your opinion, stand out from the plots.
- **[rep-27]** Write in the report the correlation for the following 3 cases: (a) #pkts/hour and #uIPd/hour, (b) #pkts/hour and #bytes/hour, and (c) #uIPs/hour and #uIPd/hour. Use the MATLAB function: `corr` with both options 'Spearman' and 'Pearson' (show three decimals). Are signals correlated for the three cases?
- **[rep-28]** What could be the reason why the drop in the number of unique IP sources after Jan/16 does not cause a proportional drop in the other signals?
- **[rep-29]** What is bigger on average: the number of sources sending packets to the darkspace or the number of darkspace addresses receiving packets? What is the ratio? Explain your calculation and show the used commands. Does such ratio make sense for you? What does it mean?
- **[rep-30]** What did you use in the previous exercise: mean or median? What is better? Does it matter?
- **[rep-31]** Find the moment when the main peak in #uIPs/hour (daily avg.) appears. When was it exactly? How long did it last? Write the specific date/s and the used commands (with short explanations if necessary).

Have a look at the Appendix to get some tips about preparing data graphs. Also note that the MATLAB functions `max` and `min` return not only the maximum and minimum values, but also the index to find such values in their respective arrays. Finally, note that with the `datestr` function we translate the epoch time into a more readable time format.

3 Analyzing a Specific Period

In the following exercises we analyze a shorter period of the darkspace. In your working folder you will find a .csv file corresponding the a darkspace aggregated data per hour. This file is specific for your team and shows the following format: `teamX_monthly.csv`. It contains data that is not daily averaged (i.e., samples directly correspond to hours instead of daily averages). Look at the header of the .csv file to see the how the data is formatted (columns are sorted differently for each team). Your data, hourly aggregated, covers approximately one month of traffic (so $30 \times 24 = 720$ samples approx.).

[rep-32] Tables with global statistics.

a) Create a table with some basics statistics for every signal in the `teamX_monthly.csv` file. Column values should be: total sum, mean, median and standard deviation[2]. Row values: number of packets per hour (#pkts/hour), number of bytes per hour (#bytes/hour), number of unique IP sources per hour (#uIPs/hour) and number of unique IP destinations per hour (#uIPd/hour). Give values in millions with three decimals.

b) Repeat the process and do a second table for **exactly** the same period (i.e., the month and year

[2] Be careful with the capture gaps, i.e., samples with 0-values. They can distort calculations. You can use the following trick: transform 0s in NaNs values, e.g, `a(a==0)=NaN`, and later use the MATLAB functions: `nanmean`, `nanmedian` and `nanstd`.

assigned to your group) but extracted from the `global_last10years.csv` file, i.e., the equivalent daily averaged values.

[rep-33] Do values in [rep-32.a] and [rep-32.b] tables coincide? If not, why?

[rep-34] Create a figure with 8 plots that show complete statistical information of the data used in the previous tables (a first-top row with 4 plots for the hourly aggregated data, and a second-bottom row of plots for the daily averaged data). Use the MATLAB functions: `subplot` and `boxplot`. Search for *box plot* on the Internet, and make sure that you understand why box plots are used and which information they display.

[rep-35] what are the main differences between the box plots corresponding to the hourly data and the daily averaged data?

In your working folder you will also find a .csv file about the darkspace aggregated data per hour of the three most recurring protocols. Again, this file is specific for your team and shows the following format: `teamX_protocol.csv`. Columns refer to the aggregated protocols and three measurements per protocol: #pkts/hour, #uIPs/hour and #uIPd/hour.

Open `teamX_protocol.csv` with a proper editor and identify the three protocol numbers appearing in the first line. **[rep-36]** Write in the report the protocol numbers, write also their common names and give a brief description (one/two sentences) for each of them.

[rep-37] What is the proportion of network traffic that does not belong to any of these three most used protocols (call it "others")? Give a percentage with two decimals for (a) packets, (b) number of unique IP sources, and (c) number of unique IP destinations.
Note that in order to calculate statistics for "others" you have to use data from `teamX_monthly.csv`, which stores total values without separating by protocol (have a look on the MATLAB function: `bsxfun` to handle matrix vs array operations).

[rep-38] Do previous results make sense? Did you obtain negative values? Why? And why not for the case of packets?.

[rep-39] Create a figure with three scatter plots where the data about the three main protocols are displayed together. They should be:

- Plot 1: x-axis: #uIPs; y-axis: #uIPd.
- Plot 2: x-axis: #uIPs; y-axis: #pkts.
- Plot 3: x-axis: #uIPd; y-axis: #pkts.

Use different colors for every protocol and add a legend to identify the protocols in the scatter plots. Have a look on the MATLAB functions: `subplot` and `scatter`.

[rep-40] What can you infer about the traffic from the three main protocols based on the previous plots?

4 Analyzing Temporal Patterns

To better understand the behaviour of the darkspace as a whole, we are going to have a look at some of the aggregated signals from temporal and frequency perspectives. We will only analyze a few signals, but we recommend students to further use the introduced methodologies and explore also other time series. In other words, try to figure out by yourself the meaning of the different shapes, anomalies and patters that you discover in the provided material (it would not be strange that you disclose phenomena that affected the whole Internet and passed unperceived for experts).

To analyze temporal patterns we use the frequency spectrum of the time series. Our time series is a discrete signal and contains an aggregated value per hour (number of packets, number of unique IP sources, etc.). For transforming the discrete finite time signal we use the Discrete Fourier Transform (DFT). The DFT transforms a time series of N data points $x_0 ... x_n ... x_{N-1}$ into a set of N complex numbers $X_0 ... X_k ... X_{N-1}$. Each complex number X_k represents a sinusoidal signal. The DFT is described by the following formula:

$$X_n = \sum_{k=0}^{N-1} x_n e^{-i 2\pi k \frac{n}{N}}$$

Our time series has one data point per hour and contains data from M days (let us assume that our period covers 30 days, so $M = 30$). Therefore, in our example we have $N = 24 \times 30 = 720$ data points, which means $N = 720$ complex numbers in the frequency domain. In the exercise you will have to examine your data and find out your particular N.

A fast way to calculate the DFT is the Fast Fourier Transform (FFT) algorithm. MATLAB provides the function `fft(x)` for calculating the FFT from a discrete time signal with values stored in the vector x. To derive the amplitude for the k^{th} signal we calculate the absolute value of the complex number X_k using the function `abs()`. In Listing 3 you can find an example of how to apply the FFT for the time series corresponding to the aggregated number of TCP packets (`p6_pkts`). Make sure that you understand Listing 3 before starting the exercises.

Listing 3: Calculating the FFT for the packet count per hour

```
> N=length(p6_pkts); % gives vector length
> p6_pkts(p6_pkts==0)=median(p6_pkts); % in case of
    gaps (0-values), replace with the median
> pkt_fft=fft(p6_pkts); % calculates the fft
> pkt_amp=abs(pkt_fft); % returns absolute values
> k=(0:N-1); % creates an array from 0 to N-1
> stem(k(2:(floor(N/2)+1)),pkt_amp(2:(floor(N/2)+1)),
    'marker', 'none') % plots stem graph
> xlim([1 floor(N/2)]);
> xlabel('k')
> ylabel('Amplitude [millions of pkts]')
> title('Amp. Spectrum for #pkts') %displays title
> [max_amp max_k]=sort(pkt_amp(2:(floor(N/2)+1)),
    'descend'); % finds maximum value and index
```

The spectrum of the signal should show the index k on the x-axis[2] and one vertical line for each of the sinusoidal signals

[2] **Note:** The MATLAB index starts at 1, whereas k starts at 0. So, to get the data from $k = 1$ to $k = N/2$ we need to use the indices $ind = k + 1 = 2$ to $ind = k + 1 = (N/2) + 1$.

at the corresponding k. The y-axis shows the amplitude of the signal. The first coefficient X_0 for $k = 0$ describes the offset of the signal. To make the other frequencies more visible we do not include the offset in the plot of the frequency spectrum. We only need to look at the first $N/2$ coefficients because the spectrum repeats itself.

Each k corresponds to the number of cycles for the sinusoidal signal within the whole duration of the signal (720 hours in this example). The associated frequency is $f_k = \frac{k}{720} \times \frac{cycles}{hour}$ and the associated period of that signal is $p_k = 1/f_k = \frac{720}{k} \times \frac{hours}{cycle}$. The sinusoidal signal at $k = 1$ has 1 cycle over the whole duration $f_1 = \frac{1}{720} \times \frac{cycles}{hour}$ and is the fundamental frequency of the signal. Its period is 720 hours:

$$p_1 = 1/f_1 = \frac{720}{1} \times \frac{hours}{cycle} \tag{1}$$

If we want to detect temporal patterns in the time series we have to check peaks in the FFT signal. With the `sort` function in Listing 3 we rearrange the FFT in order to get the maximum values of the FFT in the lower positions of the `max_amp` array, whereas the `max_k` array contains the corresponding indices. Therefore, `max_amp(1)` shows the value of the main peak in the FFT and `max_k(1)` the value of its k.

[rep-41] Prepare a figure showing two plots: one for the time series of the TCP #pkts/hour, and the second plot for the time series of the TCP #uIPs/hour. Use the data from the `teamX_protocol.csv` file.

[rep-42] Prepare a figure showing two plots: one for the FFT of the TCP #pkts/hour, and the second plot for the FFT of TCP #uIPs/hour. Use Listing 3 as a template.

[rep-43] Do the signals show any periodicity? Explain your answer in the report and show the following values for both signals (TCP #pkts/hour and TCP #uIPs/hour):

- FFT maximum value.
- k corresponding to the FFT maximum value.
- Period (in hours) of the k corresponding to the FFT maximum value.

Additionally, comment on other possible periodicity and patterns in case that your plots show more than one periodical pattern.

[rep-44] Prepare a figure with two plots. The first plot must show an average day (24 hours) of the TCP #pkts/hour of your month under analysis. The second plot must show an average day (24 hours) of the TCP #uIPs/hour of your month under analysis. For the average use the mean (red), and the median (green). Show also the standard deviation (blue). For this exercise you will probably need the MATLAB functions: `reshape` and `errorbar`.

[rep-45] Did you find anything else in the previous plots that is worth commenting on?