Lab
1

Traffic Characterization

Purpose of this lab:

In this lab you will study and compare the properties of different types of network traffic. The traffic types considered are:

- Generated Poisson traffic;
- Traffic from compressed video sources; and
- Measured traffic from an Ethernet segment.

Software Tools:

- This lab uses Python with pandas and matlibplot libraries

What to turn in:

- On Crowdmark: A report with your answers to the questions in this lab, including the plots.
- On Quercus: Your modified code files.

January 2026

© Jörg Liebeherr, 2007-2026. All rights reserved. Permission to use all or portions of this material for educational purposes is granted, as long as use of this material is acknowledged in all derivative works.

Table of Content

LAB 1 3

PART 1. POISSON TRAFFIC 4

Exercise 1-a. Scaling Poisson traffic arrivals.......... 5

Exercise 1-b. Compound Poisson arrival process.......... 6

PART 2. COMPRESSED VIDEO TRAFFIC 7

Exercise 2-a. Download a VBR video trace 8

Exercise 2-b. Determine statistical properties of the video trace.......... 8

Exercise 2-c. Scaling video traffic 9

PART 3. AGGREGATE TRAFFIC ON AN ETHERNET NETWORK 11

Exercise 3-a. Download an Ethernet traffic trace.......... 11

Exercise 3-b Determine statistical properties of the Ethernet trace.......... 11

Exercise 3-c Scaled depiction of Ethernet traffic 12

FEEDBACK FORM FOR LAB 1 13

Lab 1

Quality of plots in lab report
This lab asks you to produce plots for a lab report. Please ensure that the graphs are of good quality. All plots must be properly labelled. This includes that the units on the axes of all graphs are displayed, and that each plot has a header line that describes the content of the graph.

Units and abbreviations
The lab instructions use the following conventions:

s	= second
ms	= millisecond
µs	= microsecond
kb, kB, kbps	= kilobit, kilobyte, kilobit per second
Mb, MB, Mbps	= Megabit, Megabyte, Megabit per second

Arrival functions:
In this lab (and in the lecture) we will sometimes describe the arrivals of traffic by a function A.

- We write *A(t)* to denote the arrivals up to time *t*. We write $A(t_1, t_2)$ to denote the arrivals in the time interval $[t_1, t_2]$.
- Sometimes arrivals occur only at discrete times n= 0, 1, 2, 3,…. In this case, we write *A(n)* to denote the arrivals until *n*.

The lab assumes that you have some experience with programming in Python. The tasks in Lab 1 involve:

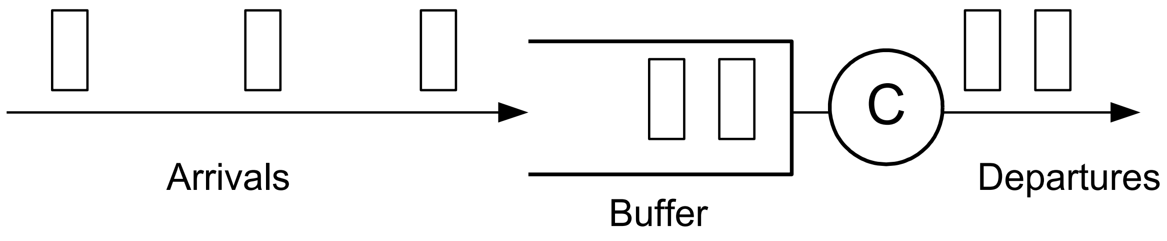
- Reading and processing a data file with multiple columns;
- Use of statistical functions, such as mean value, variance, covariance, etc;
- Generation of graphs with the matlibplot library.

Python code and data files: In addition to files with traffic trace data, we provide reference code for the Python programs. The files are available on Quercus. Use the Python code as starting points for your lab work.

- **Part 1:** Sample code: lab1_part1.py
  Traffic traces: poisson1.data, poisson2.data, poisson3.data
- **Part 2:** Sample code: lab1_part2.py
  Traffic traces: movietrace.data
- **Part 3:** Sample code: lab1_part3.py
  Traffic traces: BC-pAug89.TL.Z

Part 1. Poisson Traffic

Suppose we want to predict the delay and throughput at a buffer of a network link that operates at link capacity *C* Mbps. The situation is illustrated in the figure, where packets are represented as rectangular boxes.



To conduct an analysis of this link, we need to make assumptions on the traffic that arrives to the link. The traffic description must specify the spacing between packets and the size of packets that arrive to the buffer.

A very popular model to represent the number of packet arrivals in a time interval is the Poisson Process. In a **Poisson process** with rate λ, the number of (arrival) events in a time interval $(t, t+\tau]$, denoted by $N(t+\tau) - N(t)$, is given by

$$P[N(t+\tau)-N(t) = k] = \frac{(\lambda\tau)^k}{k!}e^{-\lambda\tau} \qquad , k = 0, 1, \ldots,$$

A Poisson process has some interesting and useful properties:

- In a Poisson process, the time between two (arrival) events follows an exponential distribution, i.e., $$P[\text{Time between two events} \leq X] = 1 - e^{-\lambda X}$$
- If we add the arrivals of two Poisson processes with rate λ and μ, we obtain a Poisson process with rate $\lambda + \mu$.

The following exercises ask you to generate Poisson traffic and study its properties.

Exercise 1-a. Scaling Poisson traffic arrivals

Here, you create sample paths of different lengths and compare them in scaled plots.

Consider a packet arrival pattern of a flow, where all packets have a constant size of 100 Bytes and where packet arrivals follow a Poisson process. The average traffic rate of the flow is 1 Mbps. The file *poisson1.data* contains a traffic trace of the data. The file has three columns:

Column 1: Packet number
Column 2: Timestamp (in microseconds)
Column 3: Packet size (in bytes).

Step 1: Verify the measured mean bit rate of the flow with the target rate, by calculating the mean and the variance of the times between consecutive arrival events. Compare the values with the theoretically expected values.

Step 2: Create three graphs that plot the data generated by the trace, viewed at different time scales. Each graph has 100 data points.

- **Plot 1:** Generate a vector with 100 elements. Each element stores the number of bytes from the Poisson trace that arrive in a time period of 1 second.
    - 1st element: # bytes arriving in time period [0, 1 s];
    - 2nd element: # bytes arriving in time period [1, 2 s];
    - …
- **Plot 2:** Generate a vector with 100 elements. Each element stores the number of bytes from the Poisson trace that arrive in a time period of 100 milliseconds, beginning at a randomly selected start time.
    - Pick a random starting point, e.g., time 30 s.
    - 1st element: # bytes arriving in time period [30, 30.1 s];
    - 2nd element: # bytes arriving in time period [30.1, 30.2 s];
    - ….
- **Plot 3:** Generate a vector with 100 elements. Each element stores the number of bytes from the Poisson trace that arrive in a time period of 10 milliseconds, beginning at a randomly selected start time.
    - Pick a random starting point, e.g., time 50.2 s.
    - 1st element: # bytes arriving in time period [50.2, 50.21 s];
    - 2nd element: # bytes arriving in time period [50.21, 50.22 s];
    - ….

Plot the content of the vectors in three separate graphs, with the time intervals on the x-axis, and the number of bytes on the y-axis. The data points should be depicted as vertical bars.

Step 3: Provide a brief comparison of the graphs.

Exercise 1-b. Compound Poisson arrival process

Now consider a packet arrival pattern of a flow, which is a combination of the previously discussed flows. Here, the packet arrival events follow a Poisson process with rate λ = 1250 packets/s, and the packet size has an exponential distribution with average size $1/\mu = 100$ Bytes. The file *poisson3.data* contains a traffic trace of the data.

Step 1: Verify the measured mean bit rate of the flow with the target rate, by calculating the mean and the variance of the times between consecutive arrival events. Compare the values with the theoretically expected values.

Step 2: Repeat the tasks in Exercise 1-a with the above arrival flows.

Step 3: Provide a discussion where you compare the results with those of Exercise 1-a.

Lab Report

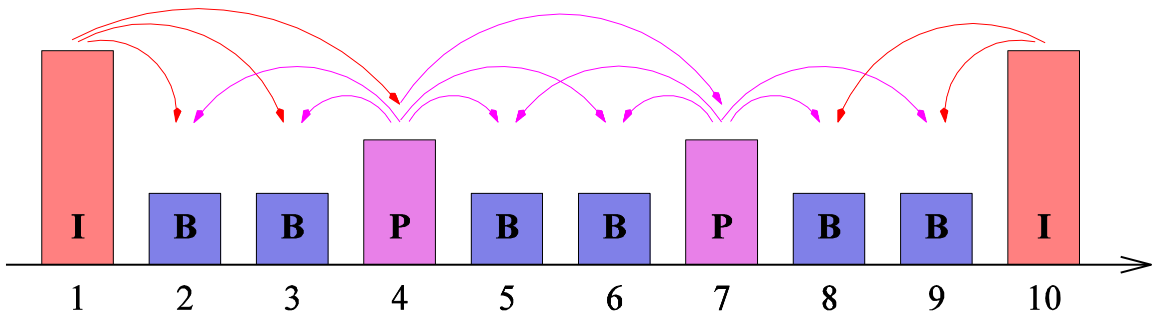
- Include the plots, a description of the plots, and your observations and discussions.

Part 2. Compressed Video Traffic

In this part of the lab, you will explore the properties of a traffic flow that consists of compressed digital video traffic.

Digital video consists of a sequence of video frames that are displayed at a rate of typically 30 frames per seconds. Assuming a frame size of 720 x 480 pixels (a standard format used for digital video) and a depth of 24 bits per pixel, this results in a data rate 240 Mbps per video stream. Using modern compression methods, such as MPEG and H.264, the data rate can be drastically reduced to a few Mbps.

Most video compression algorithms generates three types of frames: Intra-coded (I), Inter-coded or predictive (P), and bi-directional (B). I frames are coded as still images not unlike JPEG; P frames encode the differences from the most recent I or P frame; and B are interpolations of the next and previous I or P frames. An MPEG encoder generates these frames in a repeating pattern, called the Group-of-Picture (GOP) pattern. With 30 frames per second, the elapsed time between two frames is approximately 33 ms. A typical GOP pattern of IBBPBBPBB, where the dependencies between frames is indicated by arrows, is shown in the following figure:



As indicated in the figure, frame types have different sizes, with I frames being the largest and B frames the smallest. Note that a B frame is constructed from two reference frames of type I or P (e.g., Frame 3 above depends on Frame 1 and Frame 4). Thus, to encode or decode a B frame the transmitter or receiver must have available the next I or P frame. To account for this need, frames are transmitted in a different sequence than they are displayed. In the above example, the display and transmission sequence is as follows:

Display sequence:	1	2	3	4	5	6	7	8	9	10
Transmit sequence:	1	4	2	3	7	5	6	10	8	9

Due to the different sizes of I, P, and B frames, the data rate of an encoded video stream changes from frame to frame. For this reason, traffic that carries compressed video such as MPEG or H.264 is referred to a Variable Bit Rate (VBR) traffic.

Exercise 2-a. Download a VBR video trace

Your first task is to download a file that contains a trace of a VBR video source. The file contains the frame sizes obtained from a compression of a picture movie. The size of the file is 2.6 MB.

Content: Silence of the Lambs (~ 30 min)

Source Format:	DVD
Frame size and rate:	CIF 352x288 @30 fps (frames per second)
Compression algorithm:	H.264/AVC (Full)
Quantizer (compression factor):	10
GoP Pattern:	I B B B P B B B P B B B P B B B
Number of Frames:	approximately 54000

Step 1: Download the trace file from the following URL:
http://www.comm.utoronto.ca/~jorg/teaching/ece466/labs/lab1/movietrace.data

Step 2: Explore the content of the file. The file contains one line for each frame and has multiple columns. The first four columns are relevant for us:

- Column 1: Sequence number of the frame (in transmit sequence)
- Column 2: Display time of the frame
- Column 3: Frame type
- Column 4: Frame size in bytes
- Columns 5 – 7: not used

Exercise 2-b. Determine statistical properties of the video trace

Step 1: Compute the following properties of the video trace.

- Number of frames and total number of bytes;
- Size of the smallest frame, size of the largest frame, and mean frame size;
- Size of the smallest, largest and mean I, P, and B frame;
- Mean bit rate (Computed as the mean frame size divided by the frame duration);
- Peak bit rate (Computed as the max. frame size divided by the frame duration);
- Ratio of the peak rate and the average rate. This peak-to-average rate ratio is often used as an indicator how bursty a traffic flow is. A flow with a peak-to-average ratio of 10 or higher is highly variable.

Pandas helper functions
Use the panda functions *len(), mean(), max(),* and *min()* to calculate the length, mean, maximum, and minimum.

Step 2: Generate a set of graphs that show the properties of the video trace:

a. Generate a graph that shows the frame size as a function of the frame sequence number. (Use the sequence in which the frames are listed in the file, i.e., the transmit sequence.)

b. Generate a histogram graph that shows the distribution of I frames, P frames, and B frames. (x-axis is the frame size, y-axis is the relative frequency).

Exercise 2-c. Scaling video traffic

As in Part 1 with the Poisson traces, you are asked to generate and compare scaled plots of the video trace.

Step 1: Create three plots of the video traffic trace viewed at different scales. Each graph has 100 data points.

- **Plot 1:** Generate a vector with 100 elements. Starting with Frame 1, each element stores the amount of data from 500 frames of the video sequence.
    - 1st element: #bytes from the first 500 frames (frames 1, 2, …, 500).
    - 2nd element: #bytes from the frames 501,502, …, 1000.
    - …
- **Plot 2:** Generate a vector with 100 elements. Starting with a randomly selected frame, each element stores the amount of data from 50 frames of the video sequence.
    - Pick a random starting point, e.g., frame 3000.
    - 1st element: #bytes from frames 3001, 3002, … 3050.
    - 2nd element: #bytes from frames 3051, 3052, … 3100.
    - ….
- **Plot 3:** Generate a vector with 100 elements. Starting with a randomly selected frame, each element stores the amount of data from 5 frames of the video sequence.
    - Pick a random starting point, e.g., frame 5010.
    - 1st element: #bytes from frames 5010, 5011, … 5015.
    - 2nd element: #bytes from frames 5016, 5017, … 5018.
    - ….

Plot the content of the vectors in three separate graphs, with the frame number on the x-axis, and the number of bytes on the y-axis. The data points should be depicted as a bar chart, where data points are represented as vertical bars.

Bar charts
You can generate bar charts using the matlibplot plot functions *fill_between* or *step.*

Step 2: Describe your observations of the graphs and compare them to the scaled versions of the Poisson plots from Part 1.

Lab Report

- Include your answers to the exercises. Include the plots, a description of the plots, and your observations and discussions.

Part 3. Aggregate traffic on an Ethernet network

Until the early 1990s, it was common to represent traffic in a packet network by a Poisson process. Measurement studies performed in late 1980s showed that this is not a good assumption. One of the most influential such studies, done by Will Leland and Dan Wilson in 1989, measured the total traffic in an Ethernet network at the Bellcore Morristown Research and Engineering facility. The researchers collected traffic traces that each contains a million packet arrivals seen on an Ethernet network.

These traces, the most extensive measurement study at the time, were studied extensively, and led to the conclusion that aggregate traffic (=traffic from many traffic sources) is not "Poisson-like". In fact, it was convincingly argued that aggregate packet traffic is "self-similar", meaning, that it looks roughly the same if viewed at different scales. This discovery of the self-similarity of Ethernet traffic had important implications on the design of networks and packet switches.

Exercise 3-a. Download an Ethernet traffic trace

Download one of the Bellcore traffic traces. The file contains one line for each captured Ethernet packets, with two columns. The first column is a timestamp (in seconds), the second column is the size of the captured Ethernet packets (in Bytes).

Content: August 1989 trace from Bellcore

Start date: Aug 29, 1989, 11:25am

Data is recorded for about 3142.82 seconds (until 1,000,000 packets had been captured).

Step 1: Download the total trace from

http://www.comm.utoronto.ca/~jorg/teaching/ece466/labs/BC-pAug89.TL.Z

This compressed file has a size of about 5 MB.

Exercise 3-b Determine statistical properties of the Ethernet trace

Step 1: Analyze the trace of the Ethernet traffic:

- Compute the number of captured packets and total number of bytes;
- Compute the mean bit rate of the entire trace;
- Compute the peak bit rate of the trace (defined as the rate that maximizes the ratio "packet size"/"time since last packet").
- Compute the ratio of the peak rate and the average rate. Compare this value to the peak-to-average rate ratio from the video trace in Part 2.

Step 2: Generate a set of graphs that show properties of the Ethernet traffic trace:

1. Generate a graph that depicts the packet size as a function of time.
2. Generate a histogram graph that shows the distribution of packet sizes (x-axis shows the packet size, y-axis shows the relative frequency).

Exercise 3-c Scaled depiction of Ethernet traffic

This repeats the exercise from Exercise 2-b for Ethernet traffic where you generate and compare scaled plots.

Step 1: Continue with the 100 second excerpt of Ethernet traffic from the previous exercise.
Create three plots of the Ethernet traffic trace viewed at different scales. Each graph has 100 data points.

- **Plot 1:** Generate a vector with 100 elements, where each element stores the amount of data captured in a 1 s interval.
  - 1st element: #bytes from the first second of traffic, i.e., interval [0, 1 s].
  - 2nd element: #bytes from the interval [1, 2 s].
  - …

- **Plot 2:** Generate a vector with 100 elements, where each element stores the traffic from 100 ms worth of captured traffic. Select a random starting time.
  - Pick a random starting point, e.g., t = 20 s.
  - 1st element: #bytes from the interval [20, 20.1 s].
  - 2nd element: #bytes from the interval [20.1, 20.2 s].
  - …

- **Plot 3:** Generate a vector with 100 elements, where each element stores the traffic from 10 ms worth of captured traffic. Select a random starting time.
  - Pick a random starting point, e.g., t = 90 s.
  - 1st element: #bytes from the interval [90, 90.01 s].
  - 2nd element: #bytes from the interval [90.01, 90.02 s].
  - …

Plot the content of the vectors in three separate graphs, with time the x-axis, and the number of bytes on the y-axis. The plot should be presented as a bar chart.

Step 2: Describe your observations of the graphs, and compare them to the plots from the Poisson traffic and the video trace.

Lab Report

- Include your answers to the exercises. Include the plots, a description of the plots, and your observations and discussions.

Feedback Form for Lab 1

The lab has been revised for the W2026 semester. We appreciate any feedback on any issues that you may have encountered, e.g.,

- Instructions that are in need of clarification;
- Issues with provided Python code;
- …

If you have feedback, please add it to your lab report as an appendix.

Thank you!