Subject Model Simulation

Maciej Jankowski

30 April 2019

Agenda

1. Introduction
2. Tools
3. Modelling
4. Calculations
5. Tasks
6. Q&A

Motivation

Testing sessions are long and place-dependent - they require time, money and space.

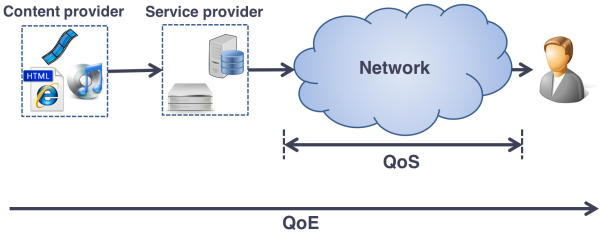
QoS & QoE

Objective:

- reliability
- accessibility
- transmission rates

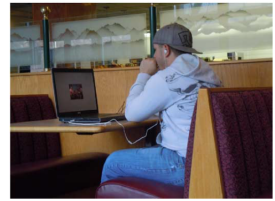
Subjective:

- aesthetics
- effectiveness
- ability to meet expectations



Subjective Tests

- **Processed Video Sequence** (PVS) – (degraded) video sample
- **Absolute Category Rating** (ACR) – 5-level scale: from "Bad" to "Excellent"
- **Mean Opinion Score** (MOS) – mean value of ACR grades for every PVS



Python

```
import time
from itertools import combinations
import numpy as np
import scipy.stats as stats
from matplotlib import pyplot as plt
import pandas as pd
```

NumPy

fast, elegant, vectorized code resembling mathematical notation

- multidimensional data structure – **ndarray**
- optimized vector & matrix operations – **np.dot()**
- basic statistical operations – **np.mean()**
- linear algebra
- Fourier transform

SciPy.stats

module containing probability distributions & statistical functions

- continuous distributions – **stats.expon.rvs(size, loc, scale)**
- discrete distributions
- summary & frequency statistics
- correlation functions
- statistical tests – **stats.ttest_ind(), stats.mannwhitneyu()**

matplotlib.pyplot

- plots – **plt.plot()**
- histograms – **plt.hist()**
- bar charts
- power spectra
- **plt.xlabel(), plt.ylabel(), plt.legend()**

Pandas

Python Data Analysis Library

- .csv – **pd.read_csv()**
- .xlsx – **pd.read_excel()**
- **dataframe**
- .head(), .tail(), .drop()
- df[df.col1 $<$ 17].head()

Distributions

- $\psi_j \sim U(1, 5)$
  psi = stats.uniform.rvs(size=PVS, loc=1, scale=4)
- $\Delta_i \sim N(0, 0.25)$
  delta = stats.norm.rvs(size=subjects, loc=0, scale=0.25)
- $\sigma_j \sim \Gamma(N = 30, \lambda = \frac{0.7}{30})$
  sigma = stats.gamma.rvs(size=PVS, a=30, loc=0, scale=0.7/30)

Discretization

$$Disc(x) = \begin{cases} 1 & \text{if } x \leq 1 \\ [x] & \text{if } 1 < x < 5 \\ 5 & \text{if } x \geq 5 \end{cases}$$

Matrix of scores

$$U_{ji} = Disc(N(\psi_j + \Delta_i, \sigma_j))$$

- ψ_j – true quality value of j^{th} PVS
- Δ_i – opinion bias of i^{th} subject
- σ_j – standard deviation
- $i = 1, 2, \ldots, I$ and I is an even number
- $j = 1, 2, \ldots, J$

MOS

means for each PVS in two groups of $\frac{I}{2}$ subjects

$$\mu_{j1} = \frac{2}{I} \sum_{i=1}^{\frac{I}{2}} U_{ji} \quad \text{and} \quad \mu_{j2} = \frac{2}{I} \sum_{i=\frac{I}{2}+1}^{I} U_{ji}$$

t-test

$$(\text{t\_stat},\ \text{p}) = \text{stats.ttest\_ind}(\mu_1,\ \mu_2)$$

- input: pair of MOS **values** (normal distribution)
- output: t statistic & p-value
- assumes equal variance by default (σ_j is constant with respect to subjects)
- checks if means are equal
- performed J times

U-test

$$(\text{U\_stat, p}) = \text{stats.mannwhitneyu}(U_a, U_b)$$

- input: pair of MOS **vectors**
- output: U statistic & p-value
- checks the number of statistically unique PVSs
- performed $\frac{J(J-1)}{2}$ times
- passing indices instead of combinations each time

Warmup

1. Plot a sine wave (dots) and an exponential (line) in one plot window.
2. Prepare a histogram of a uniform distribution.
3. Generate a 100×100 array A with numbers from 1 to 10000, count primes and replace them with 42.
4. Calculate standard deviation of rows and columns in A.
5. Implement a Disc() function, generate a random list and discretize every element, plot a histogram.

The ones from the document

1. Other ways to calculate mean value
   Compare different methods to calculate mean value of a given vector. Is there a way to swiftly calculate **means for two groups** at once (kind of) without using any loops?
2. Distributions & Plotting
   Generate random variables of your choice with **SciPy.stats** module and draw plots/histograms using **matplotlib.pyplot**.
3. Single Gamma vs. multiple exponentials
   Compare times for 10^3 and 10^5 $X \sim \Gamma(N = 30, \lambda = 0.7)$ and sum of 30 $R_n \sim exp(\lambda = 0.7)$.
4. Titanic
   Import the Titanic dataset (an .xlsx file) using the **Pandas** module. Check the header and delete useless columns. Conduct simple data analysis e.g. survival rate based on age.

Useful commands

- np.mean(), np.dot()
- stats.expon.rvs(size, loc, scale)
- plt.plot(), plt.hist(), plt.show()
- dataframe_name.drop(columns=['your_column_name_here'])

What questions do you have?