Machine Learning

Summer Semester 2018, Homework 1

Prof. Dr. J. Peters, D. Tanneberg, B. Belousov

TECHNISCHE UNIVERSITÄT DARMSTADT

Total points: 85 + 15 bonus
Due date: Wednesday, 02 May 2018 (before the lecture)

Alessia Di Blasi, 2949457 Tommaso Padovan, 2859275

Problem 1.1 Linear Algebra Refresher [20 Points]

a) Matrix Properties [5 Points]

A colleague of yours suggests matrix addition and multiplication are similar to scalars, thus commutative, distributive and associative properties can be applied. Is the statement correct? Prove it analytically or give counterexamples (for both operations) considering three matrices A, B, C of size $n \times n$.

- Addition
  - Commutative
    Given $A = [a_{ij}]$ and $B = [b_{ij}]$ $\forall i, j = 1, 2, ...n$,
    $A + B = [a_{ij} + b_{ij}]$,
    $B + A = [b_{ij} + a_{ij}]$.
    Since the sum of two scalars is commutative than the sum of two matrices is commutative.
  - Associative Given $A = [a_{ij}]$, $B = [b_{ij}]$ and $C = [c_{ij}]$ $\forall i, j = 1, 2, ...n$,
    $A + B + C = [a_{ij} + b_{ij} + c_{ij}]$ $(A + B) + C = [\left(a_{ij} + b_{ij}\right) + c_{ij}]$ Since the sum of two scalars is associative than the sum of two matrices is associative.
- Multiplication
  - Commutative
    This property is not verified.
    For instance, given

$$A = \begin{pmatrix} 1 & 2 \\ 13 & 4 \end{pmatrix},$$

$$B = \begin{pmatrix} 1 & 1 \\ 2 & 3 \end{pmatrix}$$

$$\Rightarrow A * B = \begin{pmatrix} 5 & 7 \\ 11 & 15 \end{pmatrix} \;\&\; B * A = \begin{pmatrix} 4 & 6 \\ 11 & 16 \end{pmatrix}$$

  - Distributive
    Given $A = [a_{ij}]$, $B = [b_{ij}]$ and $C = [c_{ij}]$ $\forall i, j = 1, 2, ...n$.
    I have to prove that $A * (B + C) = A * B + A * C$
    $A * (B + C) = [\sum_{k=1}^{n} a_{ik} * \left(b_{kj} + c_{kj}\right) = \sum_{k=1}^{n} a_{ik} b_{kj} + \sum_{k=1}^{n} a_{ik} c_{kj}] = A * B + A * C$

– Associative
Given $A=[a_{ij}]$, $B=[b_{ij}]$ and $C=[c_{ij}]$ $\forall i,j=1,2,...n$,
I have to prove that $(A*B)*C=A*(B*C)$.
$(A*B)*C=[r_{ij}]$ where $r_{ij}=\sum_{k=1}^{n}\left(\sum_{l=1}^{n}a_{il}*b_{lk}\right)c_{kj}=\sum_{k=1}^{n}\sum_{l=1}^{n}\left(a_{il}*b_{lk}\right)*c_{kj}$
$A*(B*C)=[s_{ij}]$ where $s_{ij}=\sum_{k=1}^{n}a_{ik}\left(\sum_{l=1}^{n}b_{kl}*c_{lj}\right)=\sum_{k=1}^{n}\sum_{l=1}^{n}a_{il}*\left(b_{lk}*c_{kj}\right)$.
Since the multiplication of two scalars is associative than the multiplication of two matrices is associative.

b) Matrix Inversion [6 Points]

Given the following matrix

$$A=\begin{pmatrix}1&2&3\\1&2&4\\1&4&5\end{pmatrix}$$

analytically compute its inverse A^{-1} and illustrate the steps.
If we change the matrix in

$$A=\begin{pmatrix}1&2&3\\1&2&4\\1&2&5\end{pmatrix}$$

is it still invertible? Why?

1. II - I and III - I
$$\left[\begin{array}{ccc|ccc}1&2&3&1&0&0\\1&2&4&0&1&0\\1&4&5&0&0&1\end{array}\right]=\left[\begin{array}{ccc|ccc}1&2&3&1&0&0\\0&0&1&-1&1&0\\0&2&2&-1&0&1\end{array}\right]$$

2. III divided by 2 and change of II with III
$$\left[\begin{array}{ccc|ccc}1&2&3&1&0&0\\0&0&1&-1&1&0\\0&2&2&-1&0&1\end{array}\right]=\left[\begin{array}{ccc|ccc}1&2&3&1&0&0\\0&1&1&-\frac{1}{2}&0&\frac{1}{2}\\0&0&1&-1&1&0\end{array}\right]$$

3. II - III and I - 3*III
$$\left[\begin{array}{ccc|ccc}1&2&3&1&0&0\\0&1&1&-\frac{1}{2}&0&\frac{1}{2}\\0&0&1&-1&1&0\end{array}\right]=\left[\begin{array}{ccc|ccc}1&2&0&4&-3&0\\0&1&0&\frac{1}{2}&-1&\frac{1}{2}\\0&0&1&-1&1&0\end{array}\right]$$

4. I- 2* II
$$\left[\begin{array}{ccc|ccc}1&2&0&4&-3&0\\0&1&0&\frac{1}{2}&-1&\frac{1}{2}\\0&0&1&-1&1&0\end{array}\right]=\left[\begin{array}{ccc|ccc}1&0&0&3&-1&-1\\0&1&0&\frac{1}{2}&-1&\frac{1}{2}\\0&0&1&-1&1&0\end{array}\right]$$

The second matrix is not invertible because the first and the second columns are linearly depends, so it is not a full rank matrix so it is singular.

c) Matrix Pseudoinverse [3 Points]

Write the definition of the right and left Moore-Penrose pseudoinverse of a generic matrix $A \in \mathbb{R}^{n \times m}$.
Given $A \in \mathbb{R}^{2 \times 3}$, which one does exist? Write down the equation for computing it, specifying the dimensionality of the matrices in the intermediate steps.

For a generic matrix $A \in \mathbb{R}^{n \times m}$, a pseudoinverse of A is defined as a matrix $A^+ \in \mathbb{R}^{m \times n}$ satisfying all the four Moore-Penrose conditions:

1. $AA^+A = A$
2. $A^+AA^+ = A^+$
3. $\left(AA^+\right)^T = AA^+$
4. $\left(A^+A\right)^T = A^+A$

When A has linearly independent columns (it means that matrix A^TA is invertible), A^+ can be computed as:
$A^+ = \left(A^TA\right)^{-1}A^T$
This is the left Moore-Penrose pseudoinverse, since $A^+A = I$.
When A has linearly independent rows (matrix AA^T is invertible), A^+ can be computed as:
$A^+ = A^T\left(AA^T\right)^{-1}$
This is the right Moore-Penrose pseudoinverse, since $AA^+ = I$.

Given $A = [a_{ij}]^{2\times3}$ only the right inverse exists: $A^+_{\text{right}} = A^T\left(AA^T\right)^{-1}$
By components it is computed as:

$AA^T = [a_{ij}]^{2\times3}[a_{ji}]^{3\times2} = [b_{ij}]^{2\times2}$

$\left(AA^T\right)^{-1} = [b_{ij}]^{-1} = \frac{1}{det\left(AA^T\right)}[c_{ij}]^{2\times2}$

$A^T\left(AA^T\right)^{-1} = \frac{1}{det\left(AA^T\right)}[a_{ji}]^{3\times2}[c_{ij}]^{2\times2} = [d_{ij}]^{3\times2} = A^+_{\text{right}}$

d) Eigenvectors & Eigenvalues [6 Points]

What are eigenvectors and eigenvalues of a matrix A? Briefly explain why they are important in Machine Learning.

Given a matrix A s.t. $\exists$ v $\in \mathbb{R}^n$ and $\lambda \in \mathbb{R}$ if $Ax = \lambda x$ then v is called eigenvector of A and λ is the corresponding eigenvalue.
Eigenvalues and Eigenvectors are important in Machine Learning because they form a basis and allow simple compunting since product between matrix and vectors becomes a product between a scalar and the vector.

Alessia Di Blasi, 2949457 Tommaso Padovan, 2859275

Problem 1.2 Statistics Refresher [25 Points]

a) Expectation and Variance [8 Points]

Let Ω be a finite set and $P : \Omega \to \mathbb{R}$ a probability measure that (by definition) satisfies $P(\omega) \geq 0$ for all $\omega \in \Omega$ and $\sum_{\omega \in \Omega} P(\omega) = 1$. Let $f : \Omega \to \mathbb{R}$ be an arbitrary function on Ω.

1) Write the definition of expectation and variance of f and discuss if they are linear operators.

2) You are given a set of three dices $\{A, B, C\}$. The following table describes the outcome of six rollouts for these dices, where each column shows the outcome of the respective dice. (Note: assume the dices are standard six-sided dices with values between 1-6)

A	4	4	2	4	1	1
B	3	6	3	3	4	3
C	5	5	2	1	1	1

Estimate the expectation and the variance for each dice using unbiased estimators. (Show your computations).

3) According to the data, which of them is the "most rigged"? Why?

1) Expected value is defined as:

$$E(f) = \sum_{\omega \in \Omega} f(\omega) P(\omega)$$

Variance is defined as:

$$var(f) = \sum_{\omega \in \Omega} \Big(f(\omega) - E(f)\Big)^2 P(\omega)$$

$E[\alpha f(x) + \beta g(x)] = \alpha E[f(x)] + \beta E[g(x)]$ thanks to the Expectation operator properties. So it is a linear operator.
$Var[\alpha f(x) + \beta g(x)] = \alpha^2 Var[f(x)] + \beta^2 E[g(x)] + 2\alpha\beta Cov[f(x), g(x)]$. So it is not a linear operator

2)

- Dice A:

$$E(A) = \frac{1}{6}(4+4+2+4+1+1) = \frac{16}{6}$$

$$var(A) = \frac{3}{5}\left(4 - \frac{16}{6}\right)^2 + \frac{1}{5}\left(2 - \frac{16}{6}\right)^2 + \frac{2}{5}\left(1 - \frac{16}{6}\right)^2 = \frac{34}{15}$$

- Dice B:

$$E(A) = \frac{1}{6}(3+6+3+3+4+3) = \frac{11}{3}$$

$$var(A) = \frac{4}{5}\left(3 - \frac{11}{3}\right)^2 + \frac{1}{5}\left(6 - \frac{11}{3}\right)^2 + \frac{1}{5}\left(4 - \frac{11}{3}\right)^2 = \frac{22}{15}$$

- Dice C:

$$E(A) = \frac{1}{6}(5+5+2+1+1+1) = \frac{5}{2}$$

$$var(A) = \frac{2}{5}\left(5 - \frac{5}{2}\right)^2 + \frac{1}{5}\left(2 - \frac{5}{2}\right)^2 + \frac{3}{5}\left(1 - \frac{5}{2}\right)^2 = \frac{39}{10}$$

3) The most rigged dice is the dice with higher observed standard deviation of the result frequencies with respect to the one of a balanced dice.
A balanced dice is expected to output any number with probability $1/6$.

- A

$$\sqrt{\left(\frac{2}{6}-\frac{1}{6}\right)^2+\left(\frac{1}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{3}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2}=$$

$$=\sqrt{\frac{2}{9}}=\frac{\sqrt{2}}{3}\approx 0.4714$$

- B

$$\sqrt{\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{4}{6}-\frac{1}{6}\right)^2+\left(\frac{1}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{1}{6}-\frac{1}{6}\right)^2}=$$

$$=\sqrt{\frac{1}{3}}=\frac{1}{\sqrt{3}}\approx 0.5774$$

- C

$$\sqrt{\left(\frac{3}{6}-\frac{1}{6}\right)^2+\left(\frac{1}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2+\left(\frac{2}{6}-\frac{1}{6}\right)^2+\left(\frac{0}{6}-\frac{1}{6}\right)^2}=$$

$$=\sqrt{\frac{2}{9}}=\frac{\sqrt{2}}{3}\approx 0.4714$$

Hence the dice B is the most rigged since he has the biggest standard deviation.

b) It is a Cold World [7 Points]

Consider the following three statements:
a) A person with a cold has backpain 25% of the time.
b) 4% of the world population has a cold.
c) 15% of those who do not have a cold, still have backpain.

1) Identify random variables from the statements above and define a unique symbol for each of them.
2) Define the domain of each random variable.
3) Represent the three statements above with your random variables.
4) If you suffer from backpain, what are the chances that you suffer from a cold? (Show all the intermediate steps.)

1) Let C be a Bernoulli random variable that identify if a person has or not cold.
Let B be a Bernoulli random variable that identify if a person has or not backpain.
2) The domain of bot C and B is $\{0,1\}$ since they are Bernoulli.
3)

a) $P\left(B=1|C=1\right)=25\%$
b) $P\left(C=1\right)=4\%$
c) $P\left(B=1|C=0\right)=15\%$

4) We want to obtain the following probability: $P\left(C=1|B=1\right)$
Applying Bayes theorem:

$$P\left(C=1|B=1\right)=\frac{P\left(B=1|C=1\right)P\left(C=1\right)}{P\left(B=1\right)}$$

Both $P\left(B=1|C=1\right)$ and $P\left(C=1\right)$ are known.
Now we need to calculate only $P\left(B=1\right)$:

$$P\left(B=1\right)=P\left(B=1|C=1\right)\cdot P\left(C=1\right)+P\left(B=1|C=0\right)\cdot P\left(C=0\right)=$$

$$=\frac{25}{100}\cdot\frac{4}{100}+\frac{15}{100}\cdot\frac{96}{100}=0.154$$

Hence:

$$P\left(C=1|B=1\right)=\frac{P\left(B=1|C=1\right)P\left(C=1\right)}{P\left(B=1\right)}=\frac{0.25\cdot 0.04}{0.154}=\frac{5}{77}$$

c) Journey to THX1138 [10 Points]

After the success of the Rosetta mission, ESA decided to send a spaceship to rendezvous with the comet THX1138. This spacecraft consists of four independent subsystems A, B, C, D. Each subsystem has a probability of failing during the journey equal to $1/3$.
1) What is the probability of the spacecraft S to be in working condition (i.e., all subsystems are operational at the same time) at the rendezvous?
2) Given that the spacecraft S is not operating properly, compute analytically the probability that only subsystem A has failed.
3) Instead of computing the probability analytically, do a simple simulation experiment and compare the result to the previous solution. Include a snippet of your code.
4) An improved spacecraft version has been designed. The new spacecraft fails if the critical subsystem A fails, or any two subsystems of the remaining B, C, D fail. What is the probability that only subsystem A has failed, given that the spacecraft S is failing?

1)Since the random variables A, B, C, D are i.i.d. the probability of the disjunction is obtained simply by multiplying the single probabilities, hence:

$$P(A=1 \cap B=1 \cap C=1 \cap D=1) = P(A=1) \cdot P(B=1) \cdot P(C=1) \cdot P(D=1) = \left(\frac{2}{3}\right)^4$$

2)

$$P\left(A=0 \cap B,C,D=1 \mid S=0\right) = \frac{P\left(S=0|A=0 \cap B=1 \cap C=1 \cap D=1\right) \cdot P(A=1 \cap B=1 \cap C=1 \cap D=1)}{P(S=0)} =$$

$$= \frac{1 \cdot \left(\frac{2}{3}\right)^3 \cdot \frac{1}{3}}{1-\left(\frac{2}{3}\right)^4} = \frac{8}{65}$$

3)

```
import numpy as np

TRIES = 100000000
PROB = 1. / 3

a = np.random.binomial(1, PROB, TRIES)
b = np.random.binomial(1, PROB, TRIES)
c = np.random.binomial(1, PROB, TRIES)
d = np.random.binomial(1, PROB, TRIES)

success = 0
for i in range(TRIES):
    if a[i] == 0 and b[i] == 1 and c[i] == 1 and d[i] == 1:
        success += 1

print "%d / %d" % (success, TRIES)
print (0. + success) / TRIES
```

The analytical result is $8/81 \approx 0.09876543209$
The result of the simple simulation above (in one specific computation) was 0.0986829, and they match up to the fourth decimal digit.

4) We know that

$$P(S=0) \;=\; P\big(S=0 \mid A=1\big)\cdot P(A=1)+P\big(S=0 \mid A=0\big)\cdot P(A=0) \;=\; \left(\binom{3}{2}\frac{2}{3}\left(\frac{1}{3}\right)^2+\left(\frac{1}{3}\right)^3\right)\cdot\frac{2}{3}+1\cdot\frac{1}{3}=\frac{41}{81}$$

Hence

$$P\big(A=0\cap B=1\cap C=1\cap D=1 \mid S=0\big) \;=$$

$$=\frac{P\big(S=0 \mid A=0\cap B=1\cap C=1\cap D=1\big)\cdot P\,(\,A=0\cap B=1\cap C=1\cap D=1)}{P(S=0)}=$$

$$=\frac{1\cdot\frac{1}{3}\cdot\left(\frac{2}{3}\right)^3}{\frac{41}{81}}=\frac{8}{41}\approx 0.19512$$

Problem 1.3 Optimization and Information Theory [40 Points + 15 Bonus]

a) Entropy [5 Points]

You work for a telecommunication company that uses a system to transmit four different symbols S_1, S_2, S_3, S_4 through time. In the current system, each symbol has a probability to occur according to the following table

	S_1	S_2	S_3	S_4
p_i	0.03	0.62	0.26	0.09

Compute the entropy of the system and write the minimum number of bits requires for transmission.

$H(p) = -\sum_{i=1}^{4} p_i \cdot log_2 p_i = 1.397298$
Minimum number of bits = $log_2 4 = 2$

b) Constrained Optimization [25 Points]

After an upgrade of the system, your boss asks you to change the probabilities of transmission in order to maximize the entropy. However, the new system has the following constraint

$$4 = \sum_{i=1}^{4} 2p_i i.$$

1) Formulate it as a constrained optimization problem. Do you need to include additional constrains beside the one above?
2) Write down the Lagrangian of the problem. Use one Lagrangian multiplier per constraint.
3) Compute the partial derivatives of the Lagrangian above for each multiplier and the objective variable. Is it easy to solve it analytically?
4) Formulate the dual function of this constrained optimization problem. Solve it analytically.
5) Name one technique for numerically solve these problems and briefly describe it.

1) $\max_p J(p) = -\sum_{i=1}^{4} p_i \log_2 p_i$

s.t. $\begin{cases} f_1(p) = \sum_{i=1}^{4} p_i i - 2 = 0 \\ f_2(p) = \sum_{i=1}^{4} p_i - 1 = 0 \end{cases}$

2) $L(p,\lambda) = -\sum_{i=1}^{4} p_i \log_2 p_i + \lambda_1 (p_1 + 2p_2 + 3p_3 + 4p_4 - 2) + \lambda_2 (p_1 + p_2 + p_3 + p_4 - 1)$

3) $\frac{\partial L}{\partial p_1} = -\log_2 p_1 - \log_2 e + \lambda_1 + \lambda_2$

$\frac{\partial L}{\partial p_2} = -\log_2 p_2 - \log_2 e + 2\lambda_1 + \lambda_2$

$\frac{\partial L}{\partial p_3} = -\log_2 p_3 - \log_2 e + 3\lambda_1 + \lambda_2$

$\frac{\partial L}{\partial p_4} = -\log_2 p_4 - \log_2 e + 4\lambda_1 + \lambda_2$

$\frac{\partial L}{\partial \lambda_1} = p_1 + 2p_2 + 3p_3 + 4p_4 - 2$

$\frac{\partial L}{\partial \lambda_2} = p_1 + p_2 + p_3 + p_4 - 1$

It is NOT easy to solve it analytically.

4) Let the derivatives be equal to 0, we are able to write p_i depending on λ_1 and λ_2 $\frac{\partial L}{\partial p_1} = 0 \Rightarrow p_1 = \frac{1}{e} 2^{\lambda_1 + \lambda_2}$

$\frac{\partial L}{\partial p_2} = 0 \Rightarrow p_2 = \frac{1}{e} 2^{2\lambda_1 + \lambda_2}$

$\frac{\partial L}{\partial p_3} = 0 \Rightarrow p_3 = \frac{1}{e} 2^{3\lambda_1 + \lambda_2}$

$\frac{\partial L}{\partial p_4} = 0 \Rightarrow p_4 = \frac{1}{e} 2^{4\lambda_1 + \lambda_2}$

Then:

$L(\theta,\lambda) = -\frac{1}{e} 2^{\lambda_1+\lambda_2} \log_2 \left(\frac{1}{e} 2^{\lambda_1+\lambda_2}\right) - \frac{1}{e} 2^{2\lambda_1+\lambda_2} \log_2 \left(\frac{1}{e} 2^{2\lambda_1+\lambda_2}\right) - \frac{1}{e} 2^{3\lambda_1+\lambda_2} \log_2 \left(\frac{1}{e} 2^{3\lambda_1+\lambda_2}\right) - \frac{1}{e} 2^{4\lambda_1+\lambda_2} \log_2 \left(\frac{1}{e} 2^{4\lambda_1+\lambda_2}\right) +$

$\lambda_1 \left(\frac{1}{e} 2^{\lambda_1+\lambda_2} + \frac{1}{e} 2 \cdot 2^{2\lambda_1+\lambda_2} + \frac{1}{e} 3 \cdot 2^{3\lambda_1+\lambda_2} + \frac{1}{e} 4 \cdot 2^{4\lambda_1+\lambda_2} - 2\right) + \lambda_2 \left(\frac{1}{e} 2^{\lambda_1+\lambda_2} + \frac{1}{e} 2^{2\lambda_1+\lambda_2} + \frac{1}{e} 2^{3\lambda_1+\lambda_2} + \frac{1}{e} 2^{4\lambda_1+\lambda_2} - 1\right) =$

$= -\frac{1}{e} 2^{\lambda_1+\lambda_2} \left[\log_2 \frac{1}{e} + \lambda_1 + \lambda_2 - \lambda_1 - \lambda_2\right] - \frac{1}{e} 2^{2\lambda_1+\lambda_2} \left[\log_2 \frac{1}{e} + 2\lambda_1 + \lambda_2 - 2\lambda_1 - \lambda_2\right] +$

$- \frac{1}{e} 2^{3\lambda_1+\lambda_2} \left[\log_2 \frac{1}{e} + 3\lambda_1 + \lambda_2 - 3\lambda_1 - \lambda_2\right] - \frac{1}{e} 2^{4\lambda_1+\lambda_2} \left[\log_2 \frac{1}{e} + 4\lambda_1 + \lambda_2 - 4\lambda_1 - \lambda_2\right] - 2\lambda_1 - \lambda_2 =$

$= \frac{1}{e} log_2^e 2^{\lambda_2} \left[2^{\lambda_1} + 2^{2\lambda_1} + 2^{3\lambda_1} + 2^{4\lambda_1}\right] - 2\lambda_1 - \lambda_2$

$\frac{\partial L}{\partial \lambda_1} = \frac{log_2^e}{e} 2^{\lambda_2} \left[2^{\lambda_1} \ln(2) + 2 \cdot 2^{2\lambda_1} \ln(2) + 3 \cdot 2^{3\lambda_1} \ln(2) + 4 \cdot 2^{4\lambda_1} \ln(2)\right] - 2$

$\frac{\partial L}{\partial \lambda_2} = \frac{log_2^e}{e} 2^{\lambda_2} \ln(2) \left[2^{\lambda_1} + 2^{2\lambda_1} + 2^{3\lambda_1} + 2^{4\lambda_1}\right] - 1$

Letting the two derivatives be equal to 0, we obtain the following system:

$$\begin{cases} \frac{1}{e} 2^{\lambda_2} \left[2^{\lambda_1} + 2 \cdot 2^{2\lambda_1} + 3 \cdot 2^{3\lambda_1} + 4 \cdot 2^{4\lambda_1}\right] = 2 \\ \frac{1}{e} 2^{\lambda_2} \left[2^{\lambda_1} + 2^{2\lambda_1} + 2^{3\lambda_1} + 2^{4\lambda_1}\right] = 1 \end{cases} \quad (2)$$

Dividing the first equation by the second one and letting $x = 2^{\lambda_1}$, we obtain:

$\frac{x+2x^2+3x^3+4x^4}{x+x^2+x^3+x^4} = 2 \Rightarrow \frac{1+2x+3x^2+4x^3}{1+x+x^2+x^3} = 2 \Rightarrow 2x^3 + x^2 - 1 = 0 \Rightarrow x \approx 0.6573 \Rightarrow 2^{\lambda_1} \approx 0.6573 \Rightarrow \lambda_1 = -0.6054$

$2^{\lambda_2} = e\frac{1}{2^{\lambda_1}+2^{2\lambda_1}+2^{3\lambda_1}+2^{4\lambda_1}} \approx 1.7425 \Rightarrow \lambda_2 \approx 0.8012$

$p_1 = \frac{1}{e}(0.6573)(1.7425) = 0.4213$

$p_2 = \frac{1}{e}(0.6573)^2(1.7425) = 0.277$

$p_3 = \frac{1}{e}(0.6573)^3(1.7425) = 0.182$

$p_4 = \frac{1}{e}(0.6573)^4(1.7425) = 0.12$

5) We can use the steepest descent method: it is a first-order iterative optimization algorithm for finding a local minimum of a function. It uses steps that are proportional to the opposite of the gradient of the function at the current point. However if steps are proportional to the gradient, the method returns a local maximum of that function (gradient ascent).

c) Numerical Optimization [10 Points]

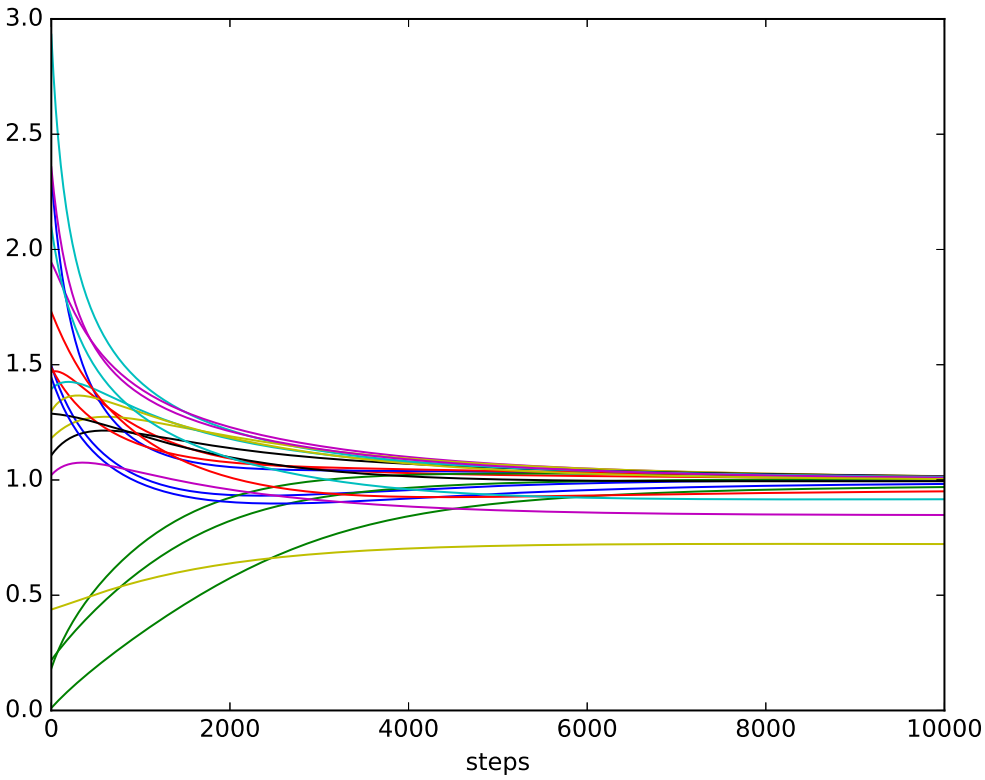
Rosenbrock's function (to be minimized) is defined as

$$f(\boldsymbol{x}) = \sum_{i=1}^{n-1} \left[100\left(x_{i+1} - x_i^2\right)^2 + \left(x_i - 1\right)^2 \right].$$

Write in Python a simple gradient descent algorithm and simulate it for 10,000 steps on Rosenbrock's function with $n = 20$. Attach a snippet of your algorithm, discuss the effects of the learning rate and attach a plot of your learning curve with your best learning rate.

```
import numpy as np
import matplotlib.pyplot as plt
import math
import random

ITERATIONS = 10000
LEARNING_RATE = 1e-6

histX = []
x = np.array([random.uniform(0.0,3.0) for i in range(20)])


#Gradient of Rosenbrock function
def rosenbrockGradient(x):
    xm = x[1:-1]
    xm_m1 = x[:-2]
    xm_p1 = x[2:]
    gradient = np.zeros_like(x)
    gradient[1:-1] = 200*(xm-xm_m1**2) - 400*(xm_p1 - xm**2)*xm - 2*(1-xm)
    gradient[0] = -400*x[0]*(x[1]-x[0]**2) - 2*(1-x[0])
    gradient[-1] = 200*(x[-1]-x[-2]**2)
    return gradient



for i in range(1, ITERATIONS):
    lastX = x
    x = x - LEARNING_RATE * rosenbrockGradient(x)
    histX.append(x)



plt.plot(histX[:])
plt.xlabel("steps")
plt.show()
```

The learning rate determines how fast or slow we will move towards the optimal weights. If the learning rate is very large we will skip the optimal solution. If it is too small we will need too many iterations to converge to the best values. In our case the best learning rate is 10^{-6}. The plot rapresents the parameters curve that tend to the minimum value.


3.0
2.5
2.0
1.5
1.0
0.5
0.0
0
2000
4000
6000
8000
10000
steps

d) Natural Gradient [10 Bonus Points]

Let $\theta \in \mathbb{R}^n$ be a parameter vector and $J : \mathbb{R}^n \to \mathbb{R}$ a cost function. The negative gradient $-\nabla J(\theta)$ is sometimes called the steepest descent direction. But is it really? To be able to claim that it is the steepest descent direction, we should compare it to other descent directions and pinpoint what is so unique about the negative gradient direction.

Covariant gradient. A fair way to compare descent directions is to make a small step of fixed length, say ε, in every direction $\Delta\theta$ and check which direction leads to the greatest decrease in $J(\theta)$. Since we assume that the step size is small, we can evaluate the decrease in $J(\theta)$ using its first-order Taylor approximation

$$J(\theta + \Delta\theta) - J(\theta) \approx \nabla J(\theta)^T \Delta\theta.$$

To make precise what we mean by 'small' step size, we need to introduce a norm (or a distance) in the space of parameters θ. A good choice, that among other advantages captures the intuition that some parameters may influence the objective function more than others, is the generic quadratic norm

$$\|\Delta\theta\|^2 = \frac{1}{2}\Delta\theta^T F(\theta)\Delta\theta$$

with a positive-definite matrix $F(\theta)$; note that in general F may depend on θ.

1) Find the direction $\Delta\theta$ that yields the largest decrease in the linear approximation of $J(\theta)$ for a fixed step size ε. Does this direction coincide with $-\nabla J(\theta)$? The direction that you found is known as the negative covariant gradient direction.

Natural gradient. In statistical models, parameter vector θ often contains parameters of a probability density function $p(x;\theta)$ (for example, mean and covariance of a Gaussian density); thus, the cost function J depends on θ indirectly through $p(x;\theta)$. This two-level structure gives a strong hint as to what matrix F to pick for measuring the distance in the parameter space in the most 'natural' way. Namely, one can carry over the notion of 'distance' between probability distributions $p(x;\theta+\Delta\theta)$ and $p(x;\theta)$ (which is known from information theory to be well captured by the Kullback-Leibler divergence) to the distance between the corresponding parameter vectors $\theta+\Delta\theta$ and θ.

2) Obtain the quadratic Taylor approximation of the KL divergence from $p(x;\theta)$ to $p(x;\theta+\Delta\theta)$ in the form

$$KL\big(p(x;\theta+\Delta\theta)\,||\,p(x;\theta)\big) \approx \frac{1}{2}\Delta\theta^T F(\theta)\Delta\theta.$$

Covariant gradient with the matrix $F(\theta)$ that you found is known as the natural gradient.

1) $\underset{\Delta\theta}{\operatorname{argmin}}\ \nabla J(\theta)^T \Delta\theta$ s.t. $\frac{1}{2}\Delta\theta^T F(\theta)\Delta\theta = \epsilon^2$.

Let $F(\theta) = \epsilon^2 D^T D$ and $z = D\Delta\theta$

$\underset{\Delta\theta}{\operatorname{argmin}}\ \Delta\theta^T \nabla J(\theta)$ s.t. $z^T z = 1$

$D^{-1}\ \underset{z}{\operatorname{argmin}} \left(D^{-1}z\right)^T \nabla J(\theta)$ s.t. $z^T z = 1$

$D^{-1}\ \underset{z}{\operatorname{argmin}}\ z^T D^{-T} \nabla J(\theta)$ s.t. $z^T z = 1$

$\propto D^{-1}[-D^{-T}\nabla J(\theta)] \propto -F(\theta)^{-1}\nabla J(\theta)$

2) We know that:

$\log \frac{P(x;\theta+\Delta\theta)}{P(x;\theta)} = \log P(x;\theta+\Delta\theta) - \log P(x;\theta) \approx \frac{\partial}{\partial\theta}\log P(x;\theta)^T \Delta\theta + \frac{1}{2}\Delta\theta^T \frac{\partial^2}{\partial\theta^2}\log P(x;\theta)\Delta\theta.$

Hence:
$D_{KL}\left(P(x;\theta+\Delta\theta)||P(x;\theta)\right) = \sum_x P(x;\theta+\Delta\theta)\log\frac{P(x;\theta+\Delta\theta)}{P(x;\theta)}$

$\simeq \sum_x [P(x;\theta) + \frac{\partial}{\partial\theta}P(x;\theta)^T\Delta\theta + \frac{1}{2}\Delta\theta^T\frac{\partial^2}{\partial\theta^2}P(x;\theta)\Delta\theta]\cdot[\frac{\partial}{\partial\theta}\log P(x;\theta)^T\Delta\theta + \frac{1}{2}\Delta\theta^T\frac{\partial^2}{\partial\theta^2}\log P(x;\theta)\Delta\theta]$

$\simeq \sum_x \left[P(x;\theta)\left(\frac{\frac{\partial}{\partial\theta}P(x;\theta)}{P(x;\theta)}\right)^T\Delta\theta + \frac{1}{2}\Delta\theta^T P(x;\theta)\cdot\frac{P(x;\theta)\frac{\partial^2 P(x;\theta)}{\partial\theta^2} - \left(\frac{\partial P(x;\theta)}{\partial\theta}\right)^T\left(\frac{\partial P(x;\theta)}{\partial\theta}\right)}{P^2(x;\theta)}\cdot\Delta\theta\right] +$

$+\sum_x \Delta\theta^T\left(\frac{\partial}{\partial\theta}P(x;\theta)\right)\left(\frac{\frac{\partial}{\partial\theta}P(x;\theta)}{P(x;\theta)}\right)^T\Delta\theta$

$= \sum_x \left[\left(\frac{\partial}{\partial\theta}P(x;\theta)\right)^T\Delta\theta + \frac{1}{2}\Delta\theta^T\left(\frac{\partial^2 P(x;\theta)}{\partial\theta^2} - P(x;\theta)\left(\frac{\frac{\partial P(x;\theta)}{\partial\theta}}{P(x;\theta)}\right)^T\left(\frac{\frac{\partial P(x;\theta)}{\partial\theta}}{P(x;\theta)}\right)\right)\cdot\Delta\theta\right]$

$+\sum_x \Delta\theta^T\left(\frac{\partial}{\partial\theta}P(x;\theta)\right)\left(\frac{\frac{\partial}{\partial\theta}P(x;\theta)}{P(x;\theta)}\right)^T\Delta\theta$

$= \left(\frac{\partial}{\partial\theta}\sum_x P(x;\theta)\right)^T\Delta\theta + \frac{1}{2}\Delta\theta^T\left(\frac{\partial^2}{\partial\theta^2}\sum_x P(x;\theta)\right)\Delta\theta + \frac{1}{2}\Delta\theta^T\left(\sum_x P(x;\theta)\left(\frac{\partial}{\partial\theta}\log P(x;\theta)\right)\left(\frac{\partial}{\partial\theta}\log P(x;\theta)\right)^T\right)\Delta\theta$

$= \left(\frac{\partial}{\partial\theta}1\right)^T\Delta\theta + \frac{1}{2}\Delta\theta^T\left(\frac{\partial^2}{\partial\theta^2}1\right)\Delta\theta + \frac{1}{2}\Delta\theta^T\left(\sum_x P(x;\theta)\left(\frac{\partial}{\partial\theta}\log P(x;\theta)\right)\left(\frac{\partial}{\partial\theta}\log P(x;\theta)\right)^T\right)\Delta\theta$

$= \frac{1}{2}\Delta\theta^T F(\theta)\Delta\theta$

e) Gradient Descent Variants [5 Bonus Points]

Throughout this class we have seen that gradient descent is one of the most used optimization techniques in Machine Learning. This question asks you to deepen the topic by conducting some research by yourself.

1) There are several variants of gradient descent, namely batch, stochastic and mini-batch. Each variant differs in how much data we use to compute the gradient of the objective function. Discuss the differences among them, pointing out pros and cons of each one.

2) Many gradient descent optimization algorithms use the so-called momentum to improve convergence. What is it? Is it always useful?

1) In Gradient Descent methods, parameters have to be updated according to the following formula:
$\theta = \theta + \alpha\nabla J(\theta)$.
Suppose we have data (x_i, y_i),
$i = 1 : M$

- Batch GD: All data are used to compute the approzimation of $\nabla J(\theta)$:
  $\nabla J(\theta) = \frac{1}{M}\sum_{i=1}^{M}(\widehat{y_i} - y_i)x_i$
- Stochastic GD: Only a sample is used to compute the approzimation of $\nabla J(\theta)$:
  $\nabla J(\theta) = (\widehat{y_i} - y_i)x_i$
- Mini-batch GD: Only N data are used to compute the approzimation of $\nabla J(\theta)$:
  $\nabla J(\theta) = \frac{1}{N}\sum_{i=1}^{N}(\widehat{y_i} - y_i)x_i$

Batch (and Mini-Batch) gradient descent is more computationally efficient than stochastic gradient descent and it has a more stable error gradient. However The more stable error gradient may result in premature convergence of the model to a less optimal set of parameters.
2) Momentum takes into account the update $\Delta\theta$ at each iteration, and determines the next update as a linear combination of the gradient and the previous update:
$\Delta\theta_{n+1} = \alpha\Delta\theta_n - \beta\nabla J(\theta)$ $\theta_{n+1} = \theta_n + \Delta\theta_{n+1}$.