CS2900 Assessed Coursework 3

This assignment must be submitted by 1:00 pm 4 May 2020.

Feedback will be provided within ten working days of the submission deadline.

Learning outcomes assessed

This assignment covers the first topic of this course. In particular, the learning outcomes assessed are :-

1. how to perform least squares using SVD,
2. how to compute a pseudo-inverse of a matrix.

Instructions

You will need to submit this coursework via Moodle. Log onto the course page. The submission link is labelled `Assignment 3` under the fifth topic. You will submit a single zip file. The zip file will have two directories `Q1` and `Q2`. In `Q1` there will be the files `leastSquares.py` and `leastSquares.txt` as discussed in first question. In `Q2` there will be the files `pseudoinverse.py` and `pseudoinverse.txt` as discussed in the second question. No other files or directories should be in the zip file.

NOTE: All the work you submit should be solely your own work. Coursework submissions are routinely checked for this. Any assessment offence will be investigated subject to the College regulations as detailed in `https://intranet.royalholloway.ac.uk/staff/assets/docs/pdf/aqpo/2019/regulations-on-academic-misconduct-2018-19.pdf`

Answer all of the following questions:

1 Question 1 Least Squares 50 Marks

In the labs you used SVD to carry out least squares analysis to perform regression on a set of two dimensional data. In this question you will do the same for three dimensional data. Specifically the data set to perform regression on is a three dimensional data set, which we'll label $(\underline{x}, \underline{y}, \underline{z})$. The vectors $\underline{x}$, $\underline{y}$ and $\underline{z}$ represent the vectors of the initial data.

The procedure is similar to what you did in the lab.

In this case you will perform the following regression

$$z = a\,x + b\,y + c \ .$$

We can rewrite this as a vector equation using all the data provided.

$$\underline{z} = \mathbf{A}\,\underline{p} \ ,$$

where

$$\mathbf{A} = \begin{pmatrix} & & 1 \\ \underline{x} & \underline{y} & \vdots \\ & & 1 \end{pmatrix}$$

and

$$\underline{p} = \begin{pmatrix} a \\ b \\ c \end{pmatrix}$$

We can determine a best estimate for $\underline{p}$ by computing the pseudo-inverse of $\mathbf{A}$.

1. Create a Python module called `leastSquares.py`. In this module write a function called `findCoefficients` that accepts one argument.

   - `data` - a NumPy array with three columns and an arbitrary number of rows of type float.

   `findCoefficients` will return the coefficients `a`, `b`, `c` described above by implementing the above approach. Use `numpy.linalg.pinv` to compute the pseudo-inverse.

   This code will be tested using the test code `testLeastSquares.py` that is also available on the moodle site.

Tip: the first column of a NumPy array `data` can be selected using `data[:,0]`. The second column can be selected using `data[:,1]` and so on.

2. Test `leastSquares.py` using the data set `ass3Data.csv` (available on the moodle site) and print your values for `a, b` and `c` into a file called `leastSquares.txt`.

2 Question 2 Pseudo-inverse 50 Marks

You have used the pseudo-inverse as defined using SVD. Previously you have used the function `numpy.linalg.pinv` to compute this. In this question you will write a function to compute the pseudo-inverse directly with SVD. If a matrix $\mathbf{A}$ is rewritten using SVD as

$$\mathbf{A} = \mathbf{U}\,\mathbf{D}\,\mathbf{V}^{\intercal} \ .$$

then the pseudo-inverse $\mathbf{A}^{+}$ can be expressed as

$$\mathbf{A}^{+} = \mathbf{V}\,\mathbf{D}^{\mathbf{inv}}\,\mathbf{U}^{\intercal} \ ,$$

where

$$D^{inv}_{ij} = \begin{cases} 1/D_{ii} & \text{iff } i = j \\ 0 & \text{otherwise} \end{cases}$$

furthermore if D_{ii} is less than some cutoff than we set the equivalent element of $\mathbf{D}^{\mathbf{inv}}$ to zero as well.

On a component by component basis then

$$A^{+}_{ij} = \sum_{k} V_{ik}\, D^{inv}_{kk}\, U^{\intercal}_{kj} \ .$$

So the algorithm for computing the pseudo-inverse is then

```
procedure COMPUTE($\mathbf{A}$, cutoff)
    $\mathbf{U}, \mathbf{D}, \mathbf{V}^{\intercal} \leftarrow SVD(\mathbf{A})$
    for $i = 0; i <$ diagonal size of $\mathbf{D}; i++$ do
        if $D_{ii} <$ cutoff then $D^{inv}_{ii} = 0$
        else$D^{inv}_{ii} = 1/D_{ii}$
        end if
    end for
    for each relevant value of $i$, $j$ do
        $A^{+}_{ij} = 0$
        for $k = 0; k <$ non-zero entries of $\mathbf{D}^{\mathbf{inv}}; k++$ do
            $A^{+}_{ij} \leftarrow A^{+}_{ij} + V_{ik}\, D^{inv}_{kk}\, U^{\intercal}_{kj}$
        end for
    end for
    Return $\mathbf{A}^{+}$
end procedure
```

1. Create a Python module called `pseudoinverse.py`. In this module write a function called `compute` that accepts two arguments.

    - `A` - NumPy array (size $N \times M$, N rows, M columns) of type float.
    - `cutoff` - float.

    `compute` will return the pseudo-inverse of `A` by implementing the above algorithm. Use `numpy.linalg.svd` to perform the SVD of `A`.

    This code will be tested using the test code `testPseudoinverse.py` that is also available on the moodle site.

2. Test `pseudoinverse.py` using to compute the pseudoinverse of the following matrix with a cutoff of $1e-6$ (10^{-6}).

$$\begin{pmatrix} 1.0 & 2.0 \\ -1.0 & 3.0 \\ 3.0 & 1.0 \\ 4.0 & 2.0 \end{pmatrix}$$

Print out the pseudoinverse matrix for this into a file called `pseudoinverse.txt`
.

Marking criteria

- Full marks will be given for replies that answer the questions and requirements in this document.
- Follow instructions on the filenames and directories to be submitted. Files that do not conform to the naming conventions will be ignored. Correspondingly not following the specifications for function names and arguments described above for both questions will lose marks.
- In question 1 make use of `numpy.linalg.pinv`. Do **not** use this function in `pseudoinverse.py` in question 2 or you will get zero marks for that question.
- Marks will be lost if the modules described in the questions above generate an error when called by their equivalent test scripts.
- Your scripts must be compliant with Python 3.