Coursework 2
Mathematics for Machine Learning (CO-496)

This coursework has both writing and coding components. The python code you submit must compile on a standard CSG Linux installation.

You are not permitted to use any symbolic manipulation libraries (e.g. `sympy`) or automatic differentiation tools (e.g. `tensorflow`) for your submitted code (though, of course, you may find these useful for checking your answers). Your code will be checked for imports. Note that <u>if you use python you should not need to import anything other than `numpy`</u> for the submitted code for this assignment.

The writing assignment requires plots, which you can create using any method of your choice. You should not submit the code used to create these plots.

No aspect of your submission may be hand-drawn. You are strongly encouraged to use LaTeXto create the written component.

In summary, you are required to submit a zip-file named `cw2.zip` containing the following:

- A file `write_up.pdf` for your written answers.
- A file `coding_answers.py` which implements all the methods for the coding exercises.

1 Differentiation

In this question, we define the following constants:

$$\mathbf{B} = \begin{pmatrix} 4 & -2 \\ -2 & 4 \end{pmatrix}, \quad \mathbf{a} = \begin{pmatrix} 0 \\ 1 \end{pmatrix} \text{ and } \quad \mathbf{b} = \begin{pmatrix} -2 \\ 1 \end{pmatrix}.$$

We define also the following functions, which are all $\mathbb{R}^2 \to \mathbb{R}$

$$f_1(\mathbf{x}) = \mathbf{x}^T\mathbf{B}\mathbf{x} - \mathbf{x}^T\mathbf{x} + \mathbf{a}^T\mathbf{x} - \mathbf{b}^T\mathbf{x}$$
$$f_2(\mathbf{x}) = \cos\left((\mathbf{x}-\mathbf{b})^T(\mathbf{x}-\mathbf{b})\right) + (\mathbf{x}-\mathbf{a})^T\mathbf{B}(\mathbf{x}-\mathbf{a})$$
$$f_3(\mathbf{x}) = 1 - \left(\exp\left(-(\mathbf{x}-\mathbf{a})^T(\mathbf{x}-\mathbf{a})\right) + \exp\left(-(\mathbf{x}-\mathbf{b})^T\mathbf{B}(\mathbf{x}-\mathbf{b})\right) - \tfrac{1}{10}\log\left|\tfrac{1}{100}\mathbf{I} + \mathbf{x}\mathbf{x}^T\right|\right)$$

a) **[3 marks]** Write $f_1(\mathbf{x})$ in the completed square form $(\boldsymbol{x}-\boldsymbol{c})^T\boldsymbol{C}(\boldsymbol{x}-\boldsymbol{c}) + c_0$, i.e., determine $\boldsymbol{C}, \boldsymbol{c}, c_0$.

b) **[2 marks]** Explain how you can tell that f_1 has a minimum point. State the minimum value of f_1 and find the input which achieves this minimum.

c) **[6 marks]** Write three python functions `grad_f1(x)`, `grad_f2(x)` and `grad_f3(x)` that return the gradient for each of the functions above.
All functions must accept numpy $(2,)$ *array inputs and return numpy* $(2,)$ *outputs.*

d) **[4 marks]** Use your gradients to implement a gradient descent algorithm with 50 iterations to find a local minimum for both f_2 and f_3. Show the steps of your algorithm on a contour plot of the function. Start from the point $(0.3, 0)$ and state the step size you used. Produce separate contour plots for the two functions, using first component of $\mathbf{x}$ on the x axis and the second on the y.

e) **[5 marks]** For the two functions f_2 and f_3:

- Discuss the qualitative differences you observe when performing gradient descent with step sizes varying between 0.01 and 1, again starting the point $(0.3, 0)$.
- Briefly describe also what happens in the two cases with grossly mis-specified step-sizes (i.e. greater than 1).