Assignment 4

Q1. Scaling (30 points)

Consider the following optimization problem:

$$\text{minimize } x_1^2 - 0.001x_2^2 + 1000x_3^2$$
$$\text{subject to } x_1^2 + x_2^2 + x_3^2 = 1$$

(a) Find the optimal solution to the problem.

First we find the jacobian of the lagragian of the problem and set it equal to zero:

$$\nabla\mathcal{L}(x,\lambda) = \begin{bmatrix} 2x_1 + 2x_1\lambda \\ -0.002x_2 + 2x_2\lambda \\ 2000x_3 + 2x_3\lambda \\ x_1^2 + x_2^2 + x_3^2 - 1 \end{bmatrix} = 0 = \begin{bmatrix} x_1(1+\lambda) \\ x_2(-0.001+\lambda) \\ x_3(1000+\lambda) \\ x_1^2 + x_2^2 + x_3^2 - 1 \end{bmatrix}$$

By observing the first three equations, and setting the inside of the parenthesis equal to zero it will lead to 3 different local minimum as shown below:

$$\lambda = -1, \quad x_1 = 1, \quad x_2 = 0, \quad x_3 = 0, \quad J = 1$$
$$\lambda = 0.001, \quad x_1 = 0, \quad x_2 = 1, \quad x_3 = 0, \quad J = -0.001$$
$$\lambda = -1000, \quad x_1 = 0, \quad x_2 = 0, \quad x_3 = 1, \quad J = 1000$$

Clearly the optimal solution is **X*** = [0,1,0], J(**X***) = -0.001

The hessian is shown below:

$$H(x,\lambda) = \begin{bmatrix} 2+2\lambda & 0 & 0 & 2x_1 \\ 0 & -0.002+2\lambda & 0 & 2x_2 \\ 0 & 0 & 2000+2\lambda & 2x_3 \\ 2x_1 & 2x_2 & 2x_3 & 0 \end{bmatrix}$$

The eigenvalues at X* are [2, -2, 2.002, 2000.002]
Solving this using newton's method the result (**X***) was achieved in 7 iterations however the X* had slightly errors built in:
X1 =1.38619e-26, X2 = 1, X3 = 4.44748e-47, λ = 0.001, J = -0.001

(b) Find a non-singular transformation x=Ly such that the condition number of the Hessian matrix of f is close to unity.
First lets define X as:

$$\boldsymbol{X} = \begin{bmatrix} ay_1 \\ by_2 \\ cy_3 \end{bmatrix}$$

Transforming the minimizer and the constraints we have:

$$J(y) = a^2y_1^2 - 0.001b^2y_2^2 + 1000c^2y_3^2$$
$$s.t: a^2y_1^2 + b^2y_2^2 + c^2y_3^2 - 1 = 0$$

Solving for the Jacobian and Hessian:

$$\nabla\mathcal{L}(y,\lambda) = \begin{matrix} 2a^2y_1(1+\lambda) \\ 2b^2y_2(-0.001+\lambda) \\ 2c^2y_3(1000+\lambda) \\ a^2y_1^2 + b^2y_2^2 + c^2y_3^3 - 1 \end{matrix}$$

$$H(y,\lambda) = \begin{bmatrix} 2a^2(1+\lambda) & 0 & 0 & 2a^2y_1 \\ 0 & 2b^2(-0.001+\lambda) & 0 & 2b^2y_2 \\ 0 & 0 & 2c^2(1000+\lambda) & 2c^2y_3 \\ 2a^2y_1 & 2b^2y_2 & 2c^2y_3 & 0 \end{bmatrix}$$

clearly 'a' should be set to 1/sqrt(2), 'b' and 'c' are a little more complicated to define. Since the magnitude within the parenthesis in the diagonal will not change independently of 'b' or 'c', what it was done was to have both components with in the parenthesis to be the same distance away from one. Since the units are separated in the 1 thousand, it was taken sqrt(1000) and since b is squared it was taken yet another square root, finally it was divided by sqrt(2) to offset the 2 in front. In summary:

$$\begin{bmatrix} a \\ b \\ c \end{bmatrix} = \begin{bmatrix} 1/sqrt(2) \\ sqrt\big(sqrt(1000)\big)/sqrt(2) \\ sqrt\big(sqrt(0.001)\big)/sqrt(2) \end{bmatrix}$$

The hessian becomes:

$$H(y,\lambda) = \begin{bmatrix} 1+\lambda & 0 & 0 & y_1 \\ 0 & -0.03162 + 31.62\lambda & 0 & 31.62y_2 \\ 0 & 0 & 31.62 + 0.03162\lambda & 0.3162y_3 \\ y_1 & 31.62y_2 & 0.3162y_3 & 0 \end{bmatrix}$$

(c) Quantify the effect of rescaling the problem, either on the number of iterations or function evaluations required to find the solution (convergence), or the quality of the optimal solution itself, or a combination of the two.

Counter to my intuition, rescaling the system did not alter the number of iteration needed to converge. The yield result was the exact same J(X) = -0.001, and the quality of the result remained unchanged and the values of X are remarkably near to those before scaling.

$$Y = \begin{bmatrix} 1.96037e-26 \\ 0.251487 \\ 3.5443e-46 \end{bmatrix} \rightarrow X_{rescaling} = \begin{bmatrix} 1.38619e-26 \\ 1 \\ 4.45672e-47 \end{bmatrix}$$

Q2. Isoperformance (30 points)

The Jacobian matrix at a point x_0 is given as:

$$\nabla J(x_0) = \begin{bmatrix} \frac{\partial J_1}{\partial x_1} & \frac{\partial J_2}{\partial x_1} & \frac{\partial J_3}{\partial x_1} & \frac{\partial J_4}{\partial x_1} \\ \frac{\partial J_1}{\partial x_2} & \frac{\partial J_2}{\partial x_2} & & \frac{\partial J_4}{\partial 2} \\ \vdots & & \ddots & \vdots \\ \frac{\partial J_1}{\partial x_6} & \cdots & \cdots & \frac{\partial J_4}{\partial x_6} \end{bmatrix}_{x_0} = \begin{bmatrix} 1 & 4 & 5 & 6 \\ 0.2 & 0.7 & 0.9 & 0.1 \\ 4 & 6 & 1 & 0 \\ 7 & 8 & 8 & 7 \\ 2.4 & 1.7 & 2.9 & -1.1 \\ 12 & 8 & 7 & 1 \end{bmatrix}$$

(a) What are the performance invariant directions that we can step to from x_o?

By calculating the null space of the Jacobian of J we have the 2 orthonormal vectors of isoperformance. In python it was used the scipy.linalg.null_space library to calculate the null space and it can be seen below. As we have 6 variables and 4 equations, therefore we have 2 DOF, so it is expected two vectors.

	t1	t2
X1	-0.58988	-0.27646
X2	-0.24959	0.910574
X3	0.015811	-0.06546
X4	0.607658	0.193909
X5	0.294589	-0.226
X6	-0.36534	-0.03823

(b) Show an example of a step direction (vector), Δx, such that J(x+Δx) ≈ 10^{-4}.

To calculate Δx, we can use the equation below:

$$\Delta x = \alpha^1 \cdot t^1 + \alpha^2 \cdot t^2$$

Where t^k is the direction in which we are moving and α^k is the step size, the direction, t^k, was defined in the previous question using the null space.
It is possible to re-arrange the equation below:

$$J(x + \Delta x) = J(x_0) + \nabla J * \Delta x + \frac{1}{2}\Delta x' H \Delta x$$

Assuming:

$$\nabla J * \Delta x \cong 0$$

Then:

$$J(x + \Delta x) - J(x_0) = \frac{1}{2}\Delta x' H \Delta x = \frac{1}{2}\Delta x' I \Delta x$$

Assuming the Hessian is properly scaled and approximating it as the identity matrix. It is possible to identify many step sizes α^k such that $\frac{1}{2}\Delta x' I \Delta x$ ≈ 10^{-4}. 2 of those can be seen in the table below:

	t1	t2
ΔX1	-0.0059	-0.00276
ΔX2	-0.0025	0.009106
ΔX3	0.000158	-0.00065
ΔX4	0.006077	0.001939
ΔX5	0.002946	-0.00226
ΔX6	-0.00365	-0.00038

To verify the assumption $\nabla J * \Delta x \cong 0$ these were calculated, and shown in the table below. The values are either zero or below machine precision.

	t1	t2
∇J1 · Δx	0.00E+00	8.67E-19
∇J2 · Δx	0	-2.04E-17
∇J3 · Δx	1.39E-17	-1.52E-17
∇J4 · Δx	-6.51E-18	-8.08E-18

For each of these vectors a different step size, α^k, was determined starting at 1 and dividing by 10 each iteration until $\frac{1}{2}\Delta x' I \Delta x$term was smaller than 10e-4.

	t1	t2
α^k	0.01	0.01