Assignment 1

Pratyush Kant

September 13, 2024

1: Convex and Coercive functions (5 points)

Problem 1

Write code to check whether the functions $f1$ and $f2$ are convex or not in an interval of $[-2, 2]$. Report which functions are strictly convex. Both functions are one dimensional. Write a function isConvex(functionname, interval) that takes the mentioned arguments and returns either true or false. To call $f1$, $f2$ you need to pass two arguments,

i. SR.No (int format)

ii. x value. eg: $f(x) = f1(\mathsf{Sr.No}, x)$

(a) Report how you tested the convexity.

(b) Report which function is convex or strongly convex or not a convex along with the values of x^* and $f(x^*)$ and explain how you evaluated. Is x^* unique?

Solution

(a)

Let $f : \mathbb{R} \mapsto \mathbb{R}$ be a function in C^2. We claim that f is convex (strictly convex) if and only if $f''(x) \geq 0$ ($f''(x) > 0$) for all $x \in \mathbb{R}$. We have already seen in class that f is convex (strictly convex) if and only if $f(y) \geq f(x) + f'(x)(y-x)$ ($f(y) > f(x) + f'(x)(y-x)$) for all $x, y \in \mathbb{R}$. If $f \in C^2$ then from the Taylor's theorem, $\exists\ \alpha \in [0, 1]$ such that

$$f(y) = f(x) + f'(x)(y-x) + \frac{f''(x + \alpha(y-x))}{2}(y-x)^2$$

f is convex (strictly convex) $\iff$ $f(y) \geq f(x) + f'(x)(y-x)$ $((f(y) > f(x) + f'(x)(y-x)))$ $\iff$ $f''(z) \geq 0$ ($f''(z) > 0$) for all $z \in [x, y]$.

We will use this second derivative test to check the convexity of the functions $f1$ and $f2$. If $f''(x) > 0$ for all $x \in [-2, 2]$, then the function is strictly convex. If $f''(x) \geq 0$ for all $x \in [-2, 2]$, then the function is convex. If $f''(x) < 0$ at some point $x \in [-2, 2]$, then the function is not convex.

Approximating Second derivative

Let $f : \mathbb{R} \mapsto \mathbb{R}$ be a function in C^2. We can approximate the second derivative of f at a point x by using the following formula:

$$f''(x) \approx \frac{f(x+h) - 2f(x) + f(x-h)}{h^2}$$

where h is a small number. To justify it we can use the Taylor's theorem. Let $f \in C^2$. Then from the Taylor's theorem, $\exists\ \alpha \in [0, 1]$ such that

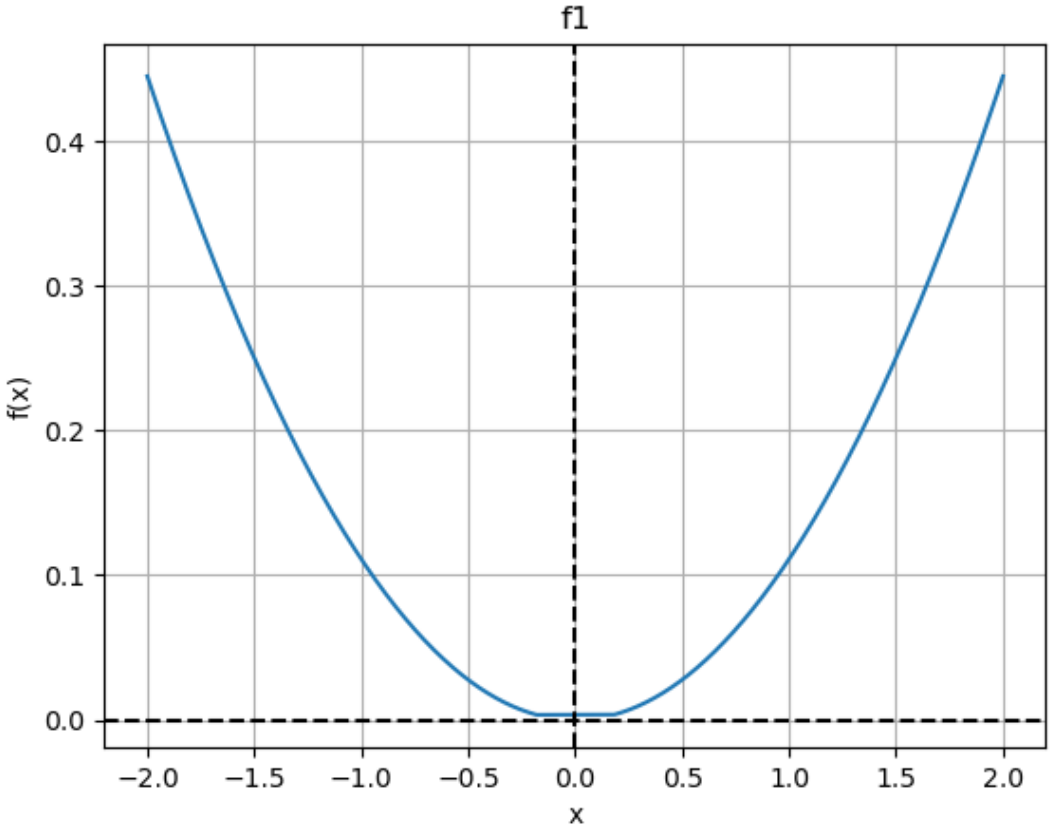

Figure 1: Plot of the function $f1$

$$f(x+h) = f(x) + f'(x)h + \frac{f''(x+\alpha h)}{2}h^2$$

$$f(x-h) = f(x) - f'(x)h + \frac{f''(x-\alpha h)}{2}h^2$$

Adding the above two equations, we get

$$f(x+h) + f(x-h) = 2f(x) + \frac{f''(x+\alpha h)}{2}h^2 + \frac{f''(x-\alpha h)}{2}h^2$$
$$\frac{f(x+h) - 2f(x) + f(x-h)}{h^2} = \frac{f''(x+\alpha h) + f''(x-\alpha h)}{2}$$

As $h \to 0$, $\alpha h \to 0$. So, $\frac{f(x+h)-2f(x)+f(x-h)}{h^2} \to \frac{f''(x)+f''(x)}{2} = f''(x)$.
The plot of the functions $f1$ and $f2$ are shown in Figure 1 and Figure 2 respectively.
Their second derivatives are plotted in Figure 3 and Figure 4 respectively.

(b)

Recall the definition that f is strongly convex in an interval I if $\forall\ x \in I\ f''(x) > m > 0$ for some m. In the function isConvex we are checking the minimum value of the second derivative for f. We have three possibilities:

i. If the minimum value of the second derivative is greater than $m > 0$, then the function is strongly convex.

ii. If the minimum value of the second derivative is equal to 0, then the function is convex but not strongly convex.

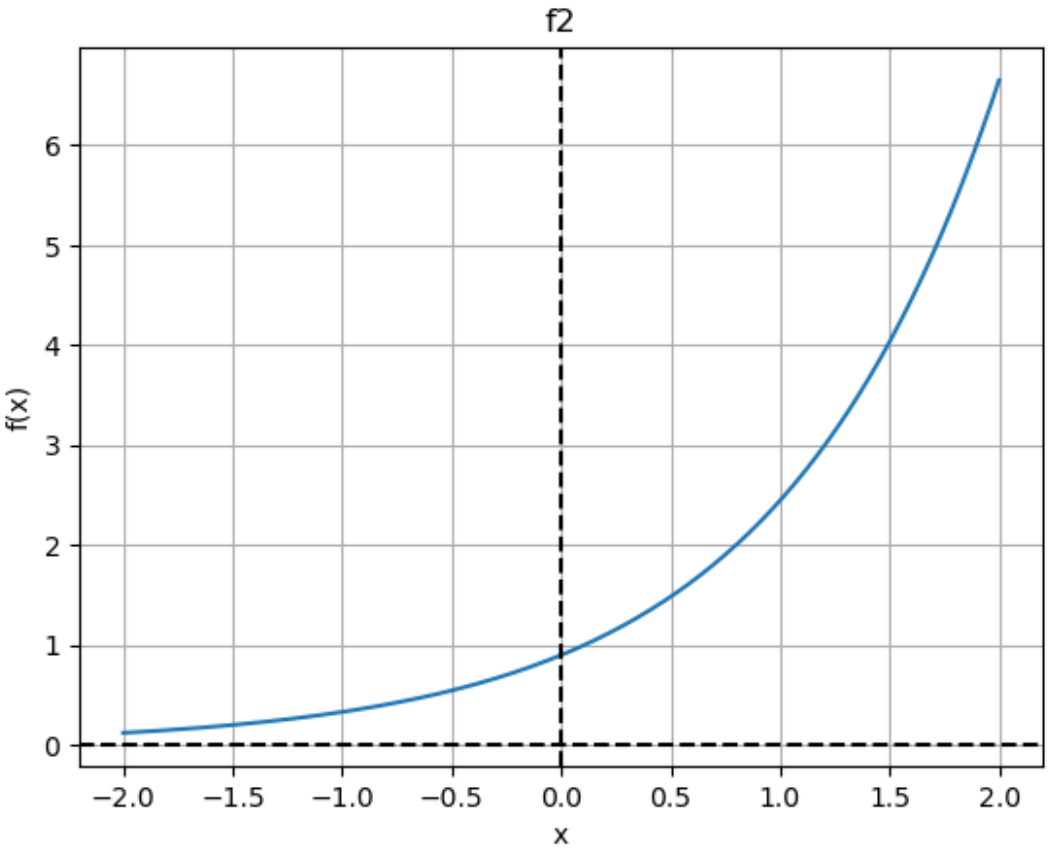

Figure 2: Plot of the function $f1$

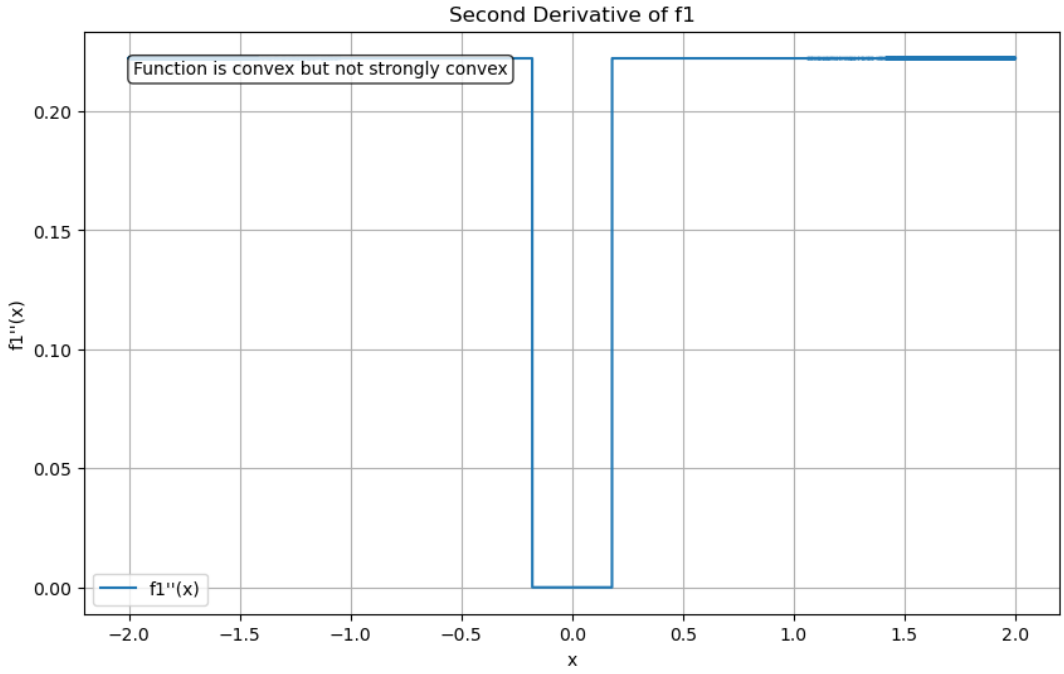

Figure 3: Plot of the second derivative of the function $f1$

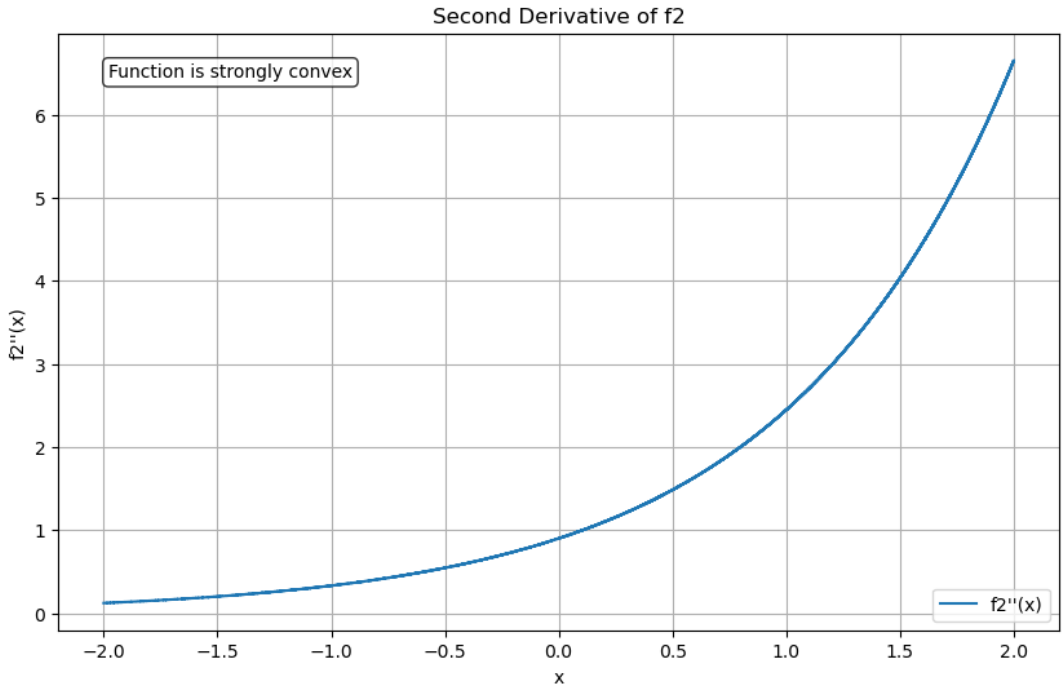

Figure 4: Plot of the second derivative of the function $f2$

iii. If the minimum value of the second derivative is less than 0, then the function is not convex.

We have used the value of $m = 1e - 20$ which gives the observations as follows:

i. The function $f1$ is convex but not strongly convex.

ii. The function $f2$ is strongly convex.

Finding minima of a function

We claim that if f is a convex function then any local minima x^* is a global minima. To see this if f is convex then $\forall\ x, y$

$$f(y) \geq f(x) + \nabla f(x)(y - x)$$

If x^* is a global minima then $\nabla f(x^*) = 0 \implies f(y) \geq f(x^*)$ for all y. Hence, x^* is a global minima. To find the local minima it suffices to check where $f'(x) = 0$ and $f'(x)$ can be approximated as

$$f'(x) \approx \frac{f(x+h) - f(x)}{h}$$

If $f'(x) > 0$ then the minima on $[-2, 2]$ is $f(-2)$ and $x^* = -2$, else it is the point where $f(x^*) = 0$. From this analysis, we conclude the following:

- The function $f1$ has minima in the interval $[-0.02, 0.02]$, $x^* \in [-0.02, 0.02]$ and $f1(x^*) = 0.0036$. Hence, x^* is not unique for $f1$.
- The function $f2$ has global minima at $x^* = -2$ and $f2(x^*) = 0.1218$. Hence, x^* is unique for $f2$.

The plots for first derivatives for $f1$ and $f2$ alongwith their minimas marked in red in the plots of $f1$ and $f2$ are shown in Figure 5 and Figure 6 respectively.

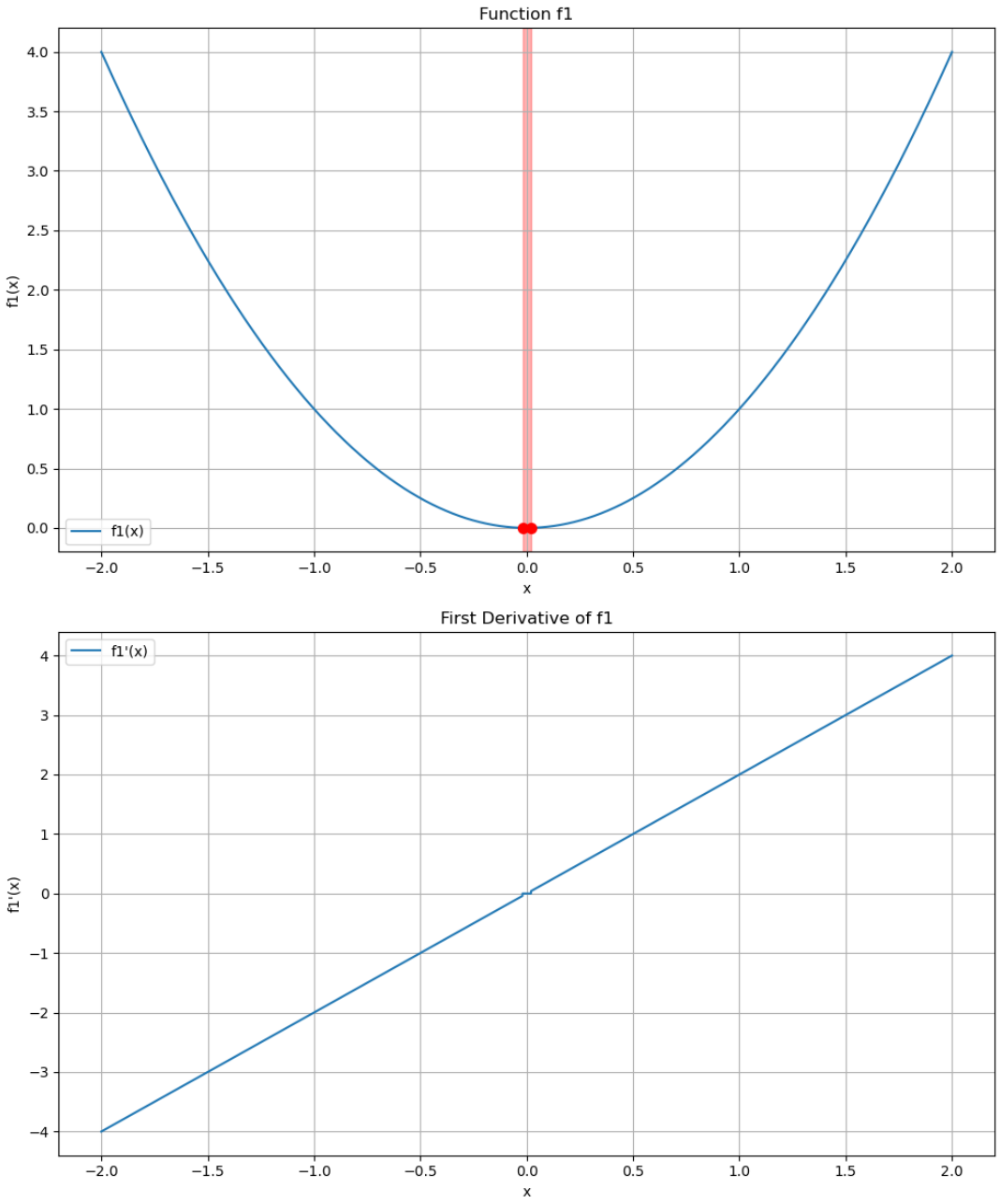

Figure 5: Plot of the first derivative of the function $f1$ and minimas in $f1$

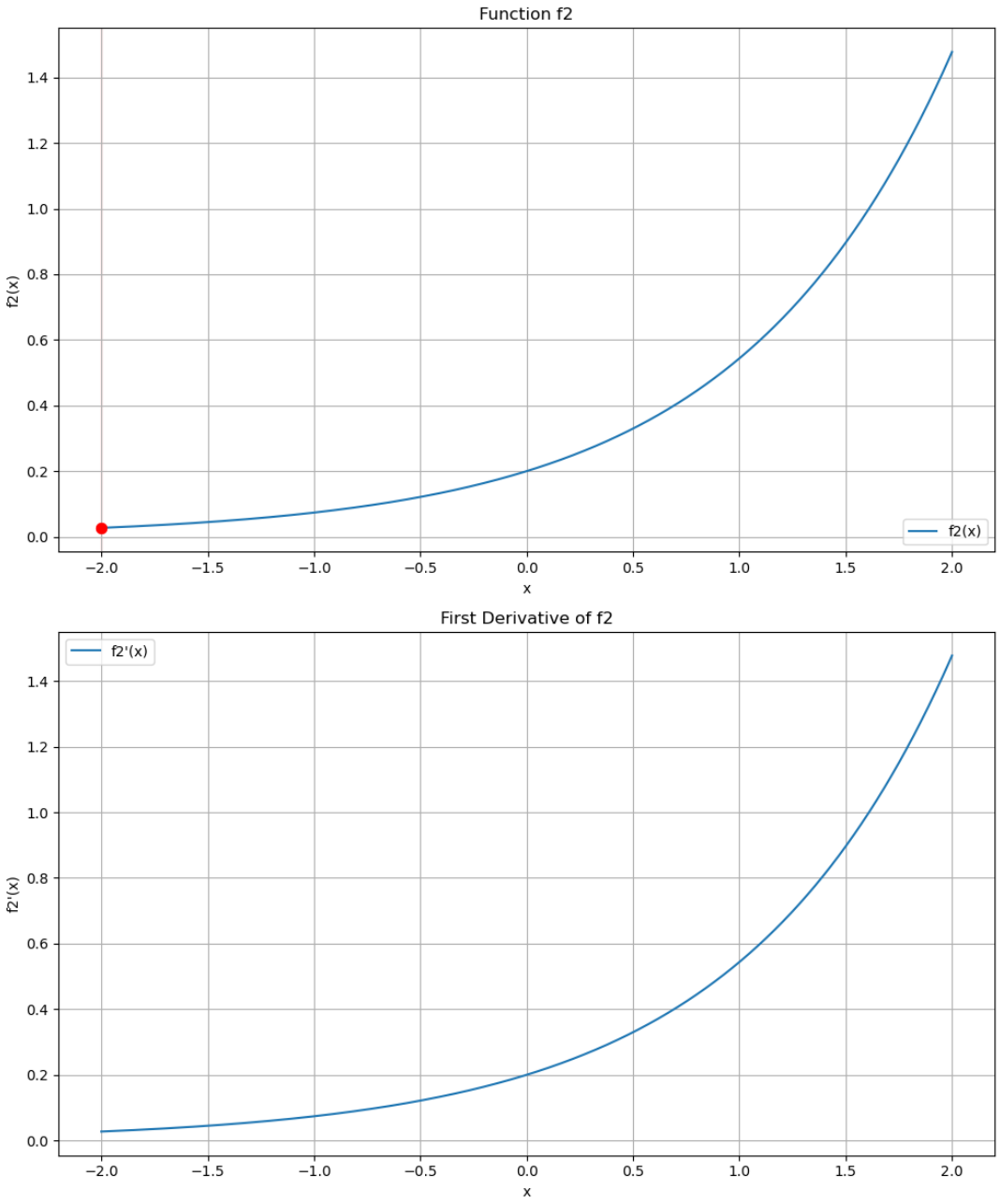

Figure 6: Plot of the first derivative of the function $f2$ and minimas in $f2$

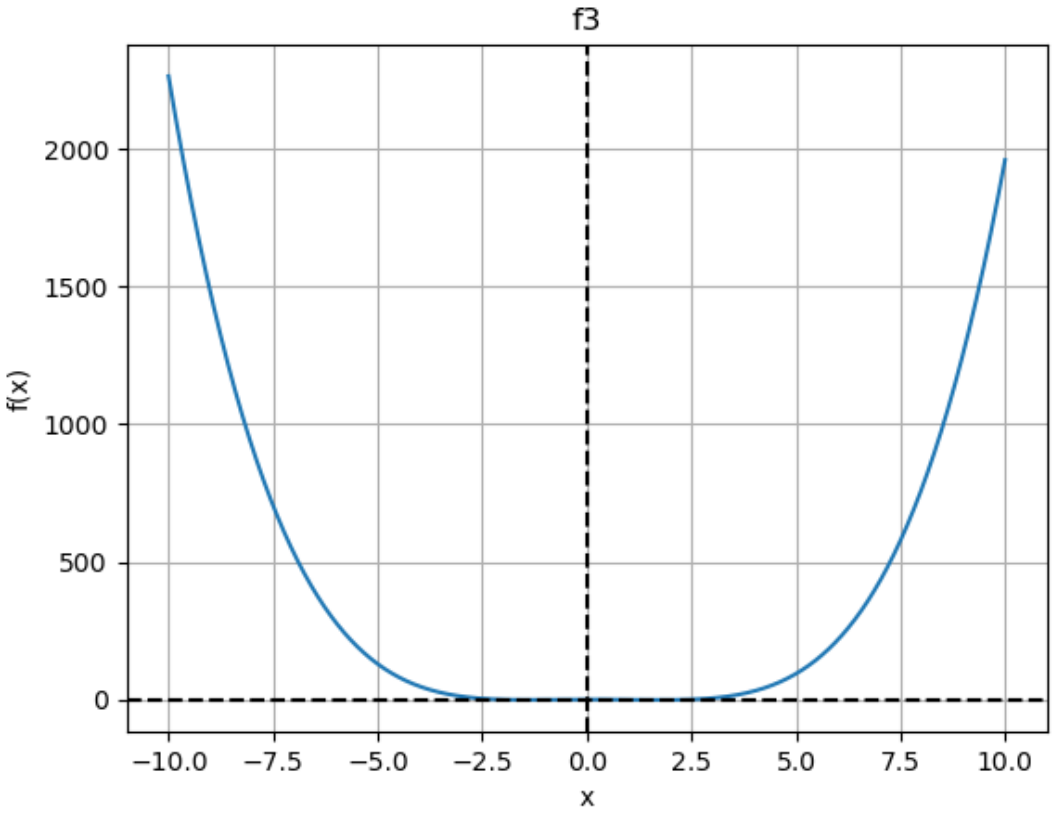

Figure 7: Plot of the function $f3$

Problem 2

Given $f3$, a quartic polynomial, write two functions isCoercive(functionname) that returns whether the function is coercive or not and FindStationaryPoints(functionname) that returns a dictionary with keys as Roots, Minima, LocalMaxima. Call $f3$ like how you called $f1$ or $f2$.

(a) Report how you tested the coercivity.

(b) Report all the stationary points and roots and explain how you evaluated them.

Solution

(a)

A function f is coercive if $\lim_{\|x\|\to\infty} f(x) = \infty$. If the leading coefficient of the polynomial is positive then the function is coercive. Hence, we can check the sign of the leading coefficient of the polynomial to check if the function is coercive or not.

The plot of the function $f3$ is shown in Figure 7.

$$
\begin{aligned}
f(x) &= ax^4 + bx^3 + cx^2 + dx + e \\
f'(x) &= 4ax^3 + 3bx^2 + 2cx + d \\
f''(x) &= 12ax^2 + 6bx + 2c \\
f^3(x) &= 24ax + 6b \\
f^4(x) &= 24a
\end{aligned}
$$

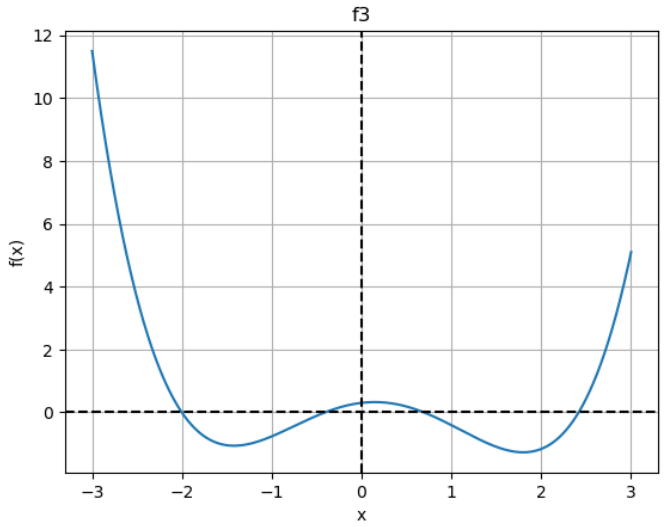

Figure 8: Plot of the function $f3$ in the interval $[-3, 3]$

Hence, it suffices to check $f^4(x)$ which is a constant function. The fourth derivative can be approximated by the following formula:

$$f^4(x) \approx \frac{f(x+2h) - 4f(x+h) + 6f(x) - 4f(x-h) + f(x-2h)}{h^4}$$

To see a justification of the above formula, we can use the Taylor's theorem. Let $f \in C^4$. Then from the Taylor's theorem, $\exists\ \alpha_i \in [0, 1]$ such that

$$\begin{aligned}
f(x+2h) &= f(x) + 2hf'(x) + 2h^2 f''(x) + (2h)^3 \frac{f^3(x)}{3!} + (2h)^4 \frac{f^4(x+\alpha_1 2h)}{4!} \\
f(x+h) &= f(x) + hf'(x) + h^2 \frac{f''(x)}{2} + h^3 \frac{f^3(x)}{3!} + h^4 \frac{f^4(x+\alpha_2 h)}{4!} \\
f(x-h) &= f(x) - hf'(x) + h^2 \frac{f''(x)}{2} - h^3 \frac{f^3(x)}{3!} + h^4 \frac{f^4(x-\alpha_3 h)}{4!} \\
f(x-2h) &= f(x) - 2hf'(x) + 2h^2 f''(x) - (2h)^3 \frac{f^3(x)}{3!} + (2h)^4 \frac{f^4(x-\alpha_4 2h)}{4!}
\end{aligned}$$

Hence, we have

$$\lim_{h \to 0} f(x+2h) - 4f(x+h) + 6f(x) - 4f(x-h) + f(x-2h) = h^4 f^4(x)$$

Since $f^4(x)$ is a constant function, we can check the sign of $f^4(0)$ to check if the function is coercive or not. By making this observation, we conclude that $f3$ is a coercive function.

(b)

The plot of $f3$ in the interval $[-3, 3]$ is shown in Figure 8. it suggest that $f3$ has 4 roots and three stationary points out of which two are minima and one is a local maxima.

We will use this information and reduce the interval to $[-3, 3]$ to find the roots and stationary points. As there cannot be any more roots than 4 and more than 3 stationary points, for a quartic polynomial, we can conclude that we have found all the roots and stationary points. To find the root we can use the bisection method and to find the stationary points we can use the NewtonBisection method again on the approximated first derivative.

Bisection Method

Let $f : \mathbb{R} \mapsto \mathbb{R}$ be a continuous function. If $f(a)f(b) < 0$ then from the intermediate value theorem $\exists\ x \in (a, b)$ such that $f(x) = 0$. The bisection method is based on this same principle. If $f(a)f(b) < 0$ then we check $f\left(\frac{a+b}{2}\right)$. If $f\left(\frac{a+b}{2}\right) = 0$ then we have found the root. If $f\left(\frac{a+b}{2}\right) f(a) < 0$ then the root lies in $(a, \frac{a+b}{2})$ else it lies in $(\frac{a+b}{2}, b)$. We can repeat this process to find the root. The method converges exponentially. To see the convergence, let x_n be the n-th approximation of the root. Then

$$x_{n+1} = \frac{a_n + b_n}{2} \implies |x_{n+1} - x^*| \leq \frac{1}{2}|b_n - a_n| = \frac{1}{2^n}|b - a|$$

For bisection method to converge close to the root, the interval must contain the root (else the algorithm terminates due to increasing the number of iterations). From the figure 8, we can see that the roots are in the intervals $[-3, -1.5]$, $[-1.5, 0]$, $[0, 1.5]$ and $[1.5, 3]$. Upon using this method, we find the roots to be:

$$\begin{aligned}
&\text{Root } 1 = -2.0026369585830253 \\
&\text{Root } 2 = -0.40414530897396617 \\
&\text{Root } 3 = 0.690238843759289 \\
&\text{Root } 4 = 2.416543423983967
\end{aligned}$$

The stationary points lie in the interval $[-2, -1]$, $[-1, 1]$ and $[1, 2]$. We will use bisection method on the first derivative of $f3$ on these intervals to find the stationary points and compare the double derivatives at these points with 0 to classify them as local minima, local maxima or saddle point. The observed results are:

$$\begin{aligned}
&\text{Extrema } 1 = -1.4206155920983292 \quad \text{label: Local minima} \\
&\text{Extrema } 2 = 0.1467414572252892 \quad \text{label: Local maxima} \\
&\text{Extrema } 3 = 1.798872635292355 \quad \text{label: Local minima}
\end{aligned}$$

2: Gradient Descent (15 points)

Problem 1

Given a function $f4$ of the form $\frac{1}{2}\mathbf{x^T A x} + \mathbf{b^T x}$, where $\mathbf{A} \in \mathbb{R}^{5\times5}$ and $\mathbf{b}, \mathbf{x} \in \mathbb{R}^5$. Consider the initialization as $\mathbf{x_0} = [\mathbf{0.0}, \mathbf{0.0}, \mathbf{0.0}, \mathbf{0.0}, \mathbf{0.0}]$ The function call for $f4$ is $fx, gradfx = f4(\mathsf{Sr.No}, x)$, where x is a NumPy array like $\mathbf{x_0}$. Write code for the following methods.

(a) Gradient Descent with fixed step size: Implement an optimization algorithm, start from x0 and consider $\alpha = 10^{-5}$, direction at every iteration $\mathbf{p_k} = -\nabla\mathbf{f}(\mathbf{x_k})$, update rule is $\mathbf{x_{k+1}} = \mathbf{x_k} + \alpha\mathbf{x_k}$. Name your function as ConstantGradientDescent(alpha, initialx).

(b) Gradient Descent with Diminishing step-size: Consider the same setup as above, but the step size $\alpha_k k = \frac{\alpha_0}{k+1}$ and $\alpha_0 = 10^{-3}$. Run this algorithm for minimum of 10000 iterations, and report the values of x_T and $f(x_T)$ where $T = 10000$. Are these matching with the previous answer, if not what might be the reason. Name your function as DiminishingGradientDescent(InitialAlpha, initialx).

(c) Inexact Line Search Using Wolfe Conditions: In the previous two questions we are kind of making α as deterministic. Now set the alpha in every iteration using the following two conditions, with $c_1 + c_2 = 1$ and $\mathbf{p_k} = -\nabla\mathbf{f}(\mathbf{x_k})$:

$$f(\mathbf{x_k} + \alpha_\mathbf{k}\mathbf{p_k}) \leq \mathbf{f}(\mathbf{x_k}) + \mathbf{c_1}\alpha_\mathbf{k}\mathbf{p_k}^\mathbf{T}\nabla\mathbf{f}(\mathbf{x_k}) \tag{1}$$

and,

$$-\mathbf{p_k}^\mathbf{T}\nabla\mathbf{f}(\mathbf{x_k} + \alpha_\mathbf{k}\mathbf{p_k}) \leq -\mathbf{c_2}\mathbf{p_k}^\mathbf{T}\nabla\mathbf{f}(\mathbf{x_k}) \tag{2}$$

We choose the α in every iteration as:

$$\alpha_k = 1$$

while any of (i) or (ii) not satisfied:

$$\alpha = \gamma \cdot \alpha$$

End while

Report the values of x^* and $f(x^*)$. Name your function as InExactLineSearch(c1, c2, gamma).

(d) Exact Line Search: Unlike doing a search for α in every iteration, we solve a optimization problem over α like follows, $\alpha_k k = \arg\min f(\mathbf{x_k} - \alpha\nabla\mathbf{f}(\mathbf{x_k}))$.The solution α of the optimization problem for the quadratic form is

$$\alpha = -\frac{\nabla f(\mathbf{x_k})^\mathbf{T}(\mathbf{p_k})}{\mathbf{p_k}^\mathbf{T}\mathbf{A}\mathbf{p_k}}$$

where $\mathbf{p_k} = -\nabla\mathbf{f}(\mathbf{x_k})$. Name your function as ExactLineSearch().

For all the above methods report $\mathbf{x}^*, \mathbf{f}(\mathbf{x}^*)$, number of iterations ($T$) and plot the following at each iteration:

- $|| f(\mathbf{x_k}) ||_\mathbf{2}$,
- $f(\mathbf{x_k}) - \mathbf{f}(\mathbf{x_T})$,
- The ratio $\frac{f(\mathbf{x_k})-\mathbf{f}(\mathbf{x_T})}{f(\mathbf{x_{k-1}})-\mathbf{f}(\mathbf{x_T})}$,
- $|| \mathbf{x_k} - \mathbf{x_T} ||_\mathbf{2}^\mathbf{2}$, and,
- The ratio $\frac{||\mathbf{x_k}-\mathbf{x_T}||_\mathbf{2}^\mathbf{2}}{||\mathbf{x_{k-1}}-\mathbf{x_T}||_\mathbf{2}^\mathbf{2}}$.

Solution

(a)

Number of iterations: $T = 10000$

$$\mathbf{x}^* = [\mathbf{-4.5e-05}, \mathbf{-5.e-04}, \mathbf{-1.e-03}, \mathbf{-2.e-03}, \mathbf{-9.99954827e-03}]$$

$$f(\mathbf{x}^*) = \mathbf{-0.006772501227364909}$$

(b)

Number of iterations: $T = 10000$

$$\mathbf{x}^* = [\mathbf{-0.00014865}, \mathbf{-0.00091097}, \mathbf{-0.001}, \mathbf{-0.00104778}, \mathbf{-0.00108767}]$$

$$f(\mathbf{x}^*) = \mathbf{-0.00228607141276548 82}$$

They are not matching with the previous answer. The reason for this discrepancy is that the step size is diminishing at every iteration. Hence, the step size is too small to reach the global minima in 10000 iterations.

(c)

Number of iterations: $T = 10000$

$$\mathbf{x}^* = [\mathbf{-4.50e-05}, \mathbf{-5.e-04}, \mathbf{-1.e-03}, \mathbf{-2.e-03}, \mathbf{-9.998e-03}]$$

$$f(\mathbf{x}^*) = \mathbf{-0.006772501036203483}$$

(d)

Calculating A:

$$f(x) = \frac{1}{2}x^T Ax + b^T x$$

$$\nabla f(x) = Ax + b$$

$$\nabla^2 f(x) = A$$

At $x = (0, 0, 0, 0, 0)$, $\nabla f(0) = b$. From the code, we get $b = (1, 1, 1, 1, 1)^T$. Notice that $A \in \mathbb{R}^{5\times 5}$.

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & a_{14} & a_{15} \\ a_{21} & a_{22} & a_{23} & a_{24} & a_{25} \\ a_{31} & a_{32} & a_{33} & a_{34} & a_{35} \\ a_{41} & a_{42} & a_{43} & a_{44} & a_{45} \\ a_{51} & a_{52} & a_{53} & a_{54} & a_{55} \end{pmatrix}$$

$$A \times \begin{pmatrix} 1 \\ 0 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} a_{11} \\ a_{21} \\ a_{31} \\ a_{41} \\ a_{51} \end{pmatrix} \quad A \times \begin{pmatrix} 0 \\ 1 \\ 0 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} a_{12} \\ a_{22} \\ a_{32} \\ a_{42} \\ a_{52} \end{pmatrix} \quad A \times \begin{pmatrix} 0 \\ 0 \\ 1 \\ 0 \\ 0 \end{pmatrix} = \begin{pmatrix} a_{13} \\ a_{23} \\ a_{33} \\ a_{43} \\ a_{53} \end{pmatrix} \quad A \times \begin{pmatrix} 0 \\ 0 \\ 0 \\ 1 \\ 0 \end{pmatrix} = \begin{pmatrix} a_{14} \\ a_{24} \\ a_{34} \\ a_{44} \\ a_{54} \end{pmatrix} \quad A \times \begin{pmatrix} 0 \\ 0 \\ 0 \\ 0 \\ 1 \end{pmatrix} = \begin{pmatrix} a_{15} \\ a_{25} \\ a_{35} \\ a_{45} \\ a_{55} \end{pmatrix}$$

From the calculations, we get:

$$A \times (1, 0, 0, 0, 0)^T + (1, 1, 1, 1, 1) = (22222, 1, 1, 1, 1)^T$$
$$\implies A \times (1, 0, 0, 0, 0)^T = (22221, 0, 0, 0, 0)^T$$

$$A \times (0, 1, 0, 0, 0)^T + (1, 1, 1, 1, 1) = (1, 2001, 1, 1, 1)^T$$
$$\implies A \times (0, 1, 0, 0, 0)^T = (0, 2000, 0, 0, 0)^T$$

$$A \times (0, 0, 1, 0, 0)^T + (1, 1, 1, 1, 1) = (1, 1, 1001, 1, 1)^T$$
$$\implies A \times (0, 0, 1, 0, 0)^T = (0, 0, 1000, 0, 0)^T$$

$$A \times (0, 0, 0, 1, 0)^T + (1, 1, 1, 1, 1) = (1, 1, 1, 501, 1)^T$$
$$\implies A \times (0, 0, 0, 1, 0)^T = (0, 0, 0, 500, 0)^T$$

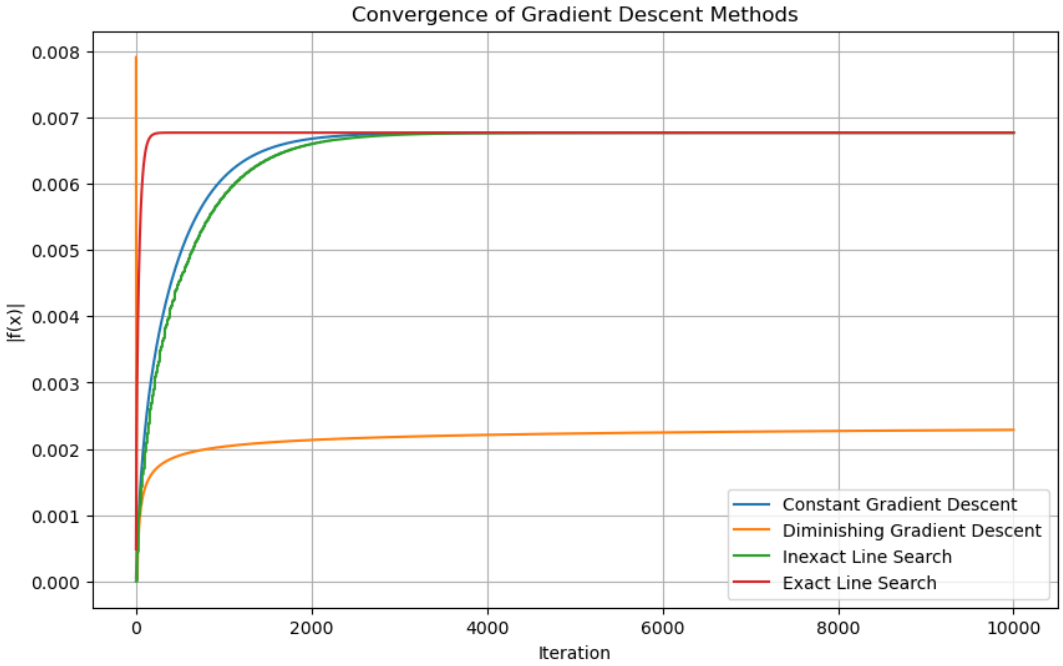

Figure 9: Norm of the function

$$A \times (0,0,0,0,1)^T + (1,1,1,1,1) = (1,1,1,1,101)^T$$
$$\implies A \times (0,0,0,0,1)^T = (0,0,0,0,100)^T$$

Hence, we have:

$$A = \begin{pmatrix} 22221 & 0 & 0 & 0 & 0 \\ 0 & 2000 & 0 & 0 & 0 \\ 0 & 0 & 1000 & 0 & 0 \\ 0 & 0 & 0 & 500 & 0 \\ 0 & 0 & 0 & 0 & 100 \end{pmatrix}$$

Number of iterations: $T = 10000$

$\mathbf{x}^* = [\mathbf{-4.5e-05}, \mathbf{-5.e-04}, \mathbf{-1.e-03}, \mathbf{-2.e-03}, \mathbf{-1.e-02}]$

$f(\mathbf{x}^*) = \mathbf{-0.006772501237568064}$

Plots

- Norm of the gradient is shown in the figure 9.
- $f(\mathbf{x_k}) - \mathbf{f(x_T)}$ is shown in the figure 10.
- The ratio $\frac{f(\mathbf{x_k}) - \mathbf{f(x_T)}}{f(\mathbf{x_{k-1}}) - \mathbf{f(x_T)}}$ is shown in the figure 11.
- $|| \mathbf{x_k} - \mathbf{x_T} ||_\mathbf{2}^\mathbf{2}$ is shown in the figure 12.
- The ratio $\frac{||\mathbf{x_k} - \mathbf{x_T}||_\mathbf{2}^\mathbf{2}}{||\mathbf{x_{k-1}} - \mathbf{x_T}||_\mathbf{2}^\mathbf{2}}$ is shown in the figure 13.

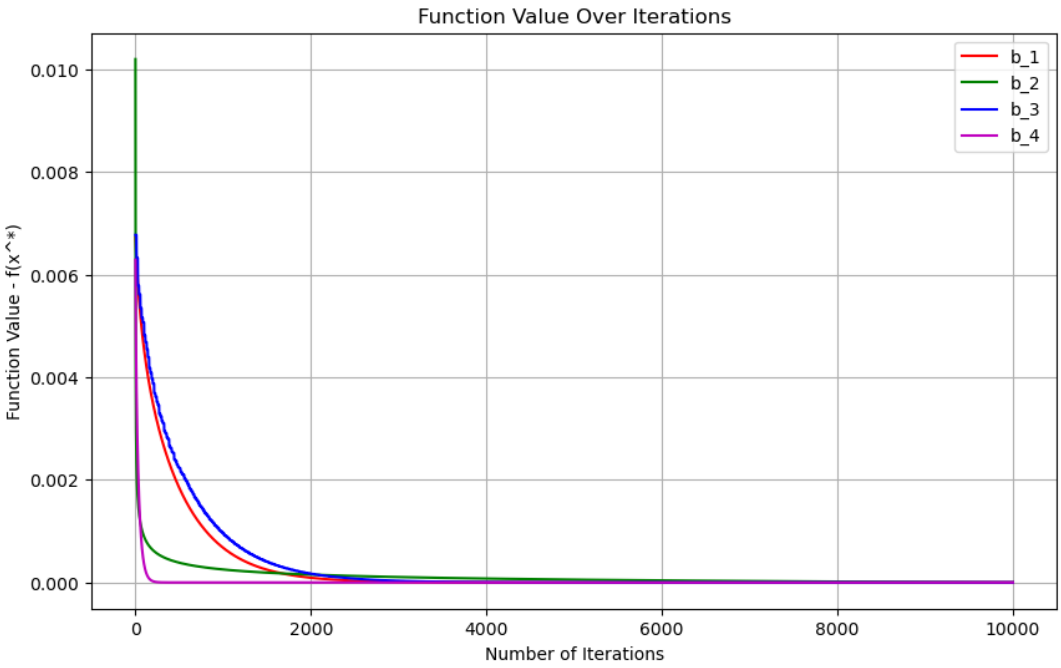

Figure 10: $f(\mathbf{x_k}) - \mathbf{f}(\mathbf{x_T})$

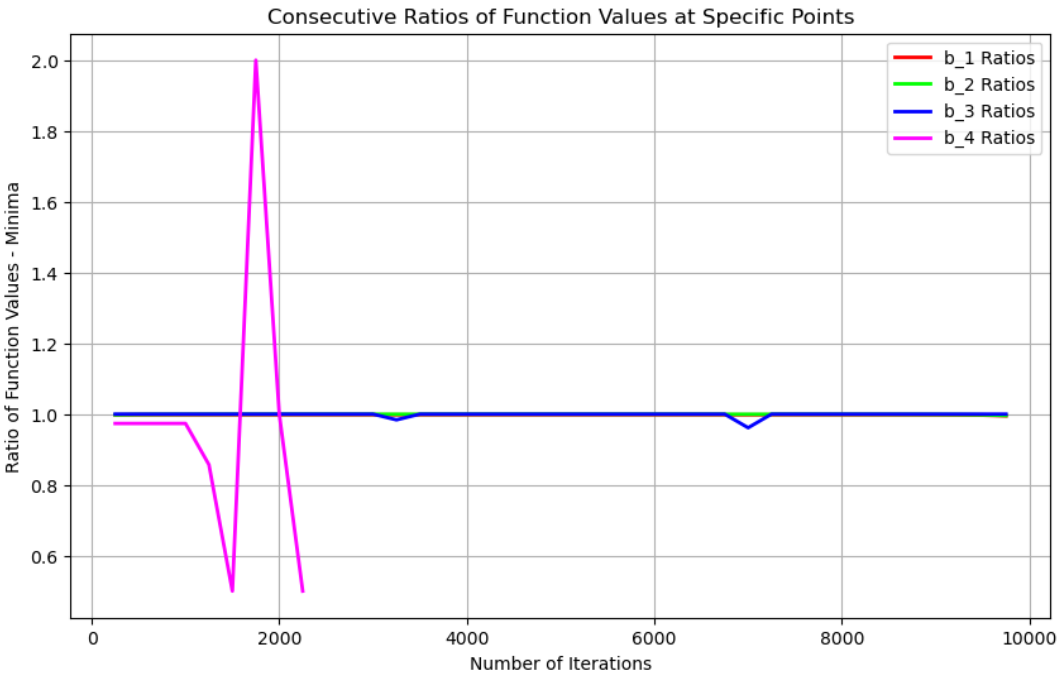

Figure 11: The ratio $\frac{f(\mathbf{x_k}) - \mathbf{f}(\mathbf{x_T})}{f(\mathbf{x_{k-1}}) - \mathbf{f}(\mathbf{x_T})}$

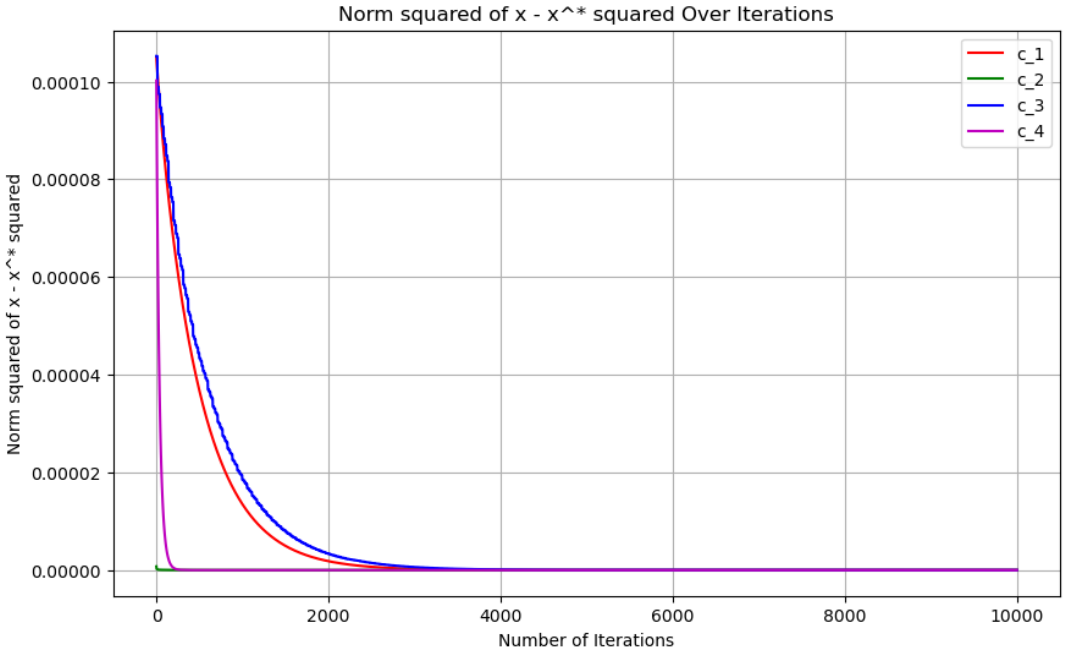

Figure 12: $|| \mathbf{x_k} - \mathbf{x_T} ||_\mathbf{2}^\mathbf{2}$

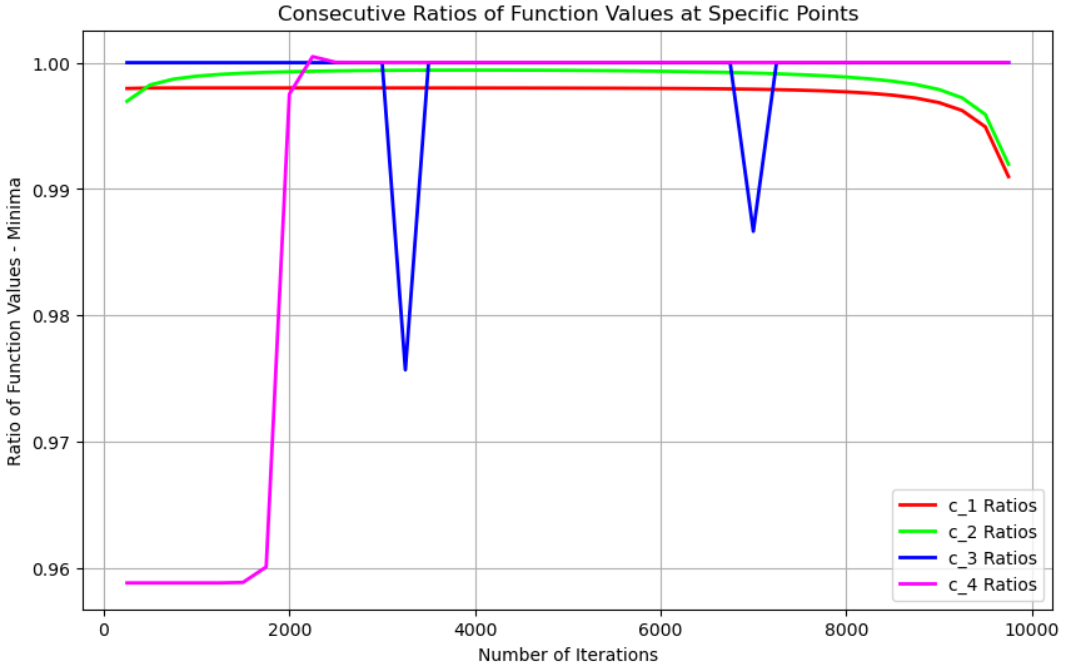

Figure 13: The ratio $\frac{||\mathbf{x_k} - \mathbf{x_T}||_\mathbf{2}^\mathbf{2}}{||\mathbf{x_{k-1}} - \mathbf{x_T}||_\mathbf{2}^\mathbf{2}}$

3: Perturbed Gradient Descent (10 + 5 points)

In this exercise, we will try to check if the intentional addition of noise to gradient descent can help us avoid saddle points. But before doing so, let us revise the necessary and sufficient conditions for identifying (local) minima of continuously differentiable functions. We will start with the first order necessary conditions which state that if a point $\mathbf{x}^*$ is a local minimum in an open neighbourhood of $\mathbf{x}^*$, then it is a stationary point, i.e., $\nabla f(\mathbf{x}^*) = 0$. Note, however, that the converse may not always be true.

To find if a point is indeed a minimum, we would need additional information on the Hessian of the function at that point. Supposing that $\nabla^2 f$ exists and is continuous in an open neighbourhood of $\mathbf{x}^*$:

- If $\mathbf{x}^*$ is a local minimum, then $\nabla f(\mathbf{x}^*) = 0$ and $\nabla^2 f(\mathbf{x}^*)$ is positive semidefinite.
- Conversely, if $\nabla f(\mathbf{x}^*) = 0$ and $\nabla^2 f(\mathbf{x}^*)$ is positive semidefinite at a point $\mathbf{x}^*$, then $\mathbf{x}^*$ is a strict local minimiser of f.
- Alternatively, a stationary point is a local maximiser if the Hessian is negative semidefinite, or a saddle point if the Hessian has a mix of non-positive and non-negative eigenvalues at that point.

Please refer to Chapter 2 of Nocedal and Wright for a deeper and more thorough exposition.

To escape saddle points, we will make use of perturbed gradient descent, whose update-equation is given by:

$$\theta^{t+1} = \theta^t - \alpha_t \left(\nabla f(\theta^{\mathbf{t}}) + \zeta^{(\mathbf{t})} \right)$$

where, $\alpha_t \geq 0$, and $\zeta^{(\mathbf{t})}$ is an artificially-added noise term. Here, we may choose to add uncorrelated Gaussian noise to each coordinate such that: $\zeta^{(\mathbf{t})} \sim \mathcal{N}(\mathbf{0}, \sigma^{\mathbf{2}}\mathbf{I})$.[1] The step-size α_t and the noise-variance may either be constant, or may diminish at each iteration as:

$$\beta_t = \frac{\beta_0}{(t+1)}$$

Now, consider the following function:

$$f(x, y) = e^{xy}$$

with $x, y \in \mathbb{R}$.

Problem 1

Find all the stationary points of $f(x, y)$ and categorise them as minima, maxima, or saddle points.

Solution

$$\begin{aligned} f(x,y) &= e^{xy} \\ \nabla f(x,y) &= \begin{pmatrix} ye^{xy} \\ xe^{xy} \end{pmatrix} \\ \nabla^2 f(x,y) &= \begin{pmatrix} y^2 e^{xy} & e^{xy} + xye^{xy} \\ e^{xy} + xye^{xy} & x^2 e^{xy} \end{pmatrix} \end{aligned}$$

Since f is differentiable and continuous everywhere, every stationary point of f must satisfy $\nabla f = 0 \implies ye^{xy} = 0$ and $xe^{xy} = 0$. This implies that the stationary points are at $(0, 0)$. Furthermore, the Hessian at $(0, 0)$ is:

$$\nabla^2 f(0,0) = \begin{pmatrix} 0 & 1 \\ 1 & 0 \end{pmatrix}$$

The eigenvalues of the Hessian are 1 and -1. Hence, $(0, 0)$ is a saddle point.

[1] One may use the random.multivariatenormal() function from numpy to generate samples from a normal distribution.

Problem 2

Analytically show that on starting gradient descent at an initial point $x = y$, one stays on the line $x = y$.

Solution

Consider the function $f(x, y) = e^{xy}$. The gradient descent update rule is given by:

$$\begin{pmatrix} x^{t+1} \\ y^{t+1} \end{pmatrix} = \begin{pmatrix} x^t \\ y^t \end{pmatrix} - \alpha_t \begin{pmatrix} ye^{xy} \\ xe^{xy} \end{pmatrix}$$

Given that $x = y = k$ (say), the update rule becomes:

$$\begin{pmatrix} x^{t+1} \\ y^{t+1} \end{pmatrix} = \begin{pmatrix} k^t \\ k^t \end{pmatrix} - \alpha_t \begin{pmatrix} ke^{k^2} \\ ke^{k^2} \end{pmatrix}$$

This simplifies to:

$$\begin{pmatrix} x^{t+1} \\ y^{t+1} \end{pmatrix} = \begin{pmatrix} k^t - \alpha_t ke^{k^2} \\ k^t - \alpha_t ke^{k^2} \end{pmatrix}$$

Hence,

$$\begin{aligned} x^{t+1} &= k^t - \alpha_t ke^{k^2} \\ y^{t+1} &= k^t - \alpha_t ke^{k^2} \\ \implies x^{t+1} &= y^{t+1} \end{aligned}$$

Hence, on starting gradient descent at an initial point $x = y$, one stays on the line $x = y$.

Problem 3

Create a contour plot of $f(x, y)$, with $-1 \leq x \leq 1$ and $-1 \leq y \leq 1$.

Solution

The contour plot of $f(x, y)$ is shown in the figure 14.

Problem 4

Run gradient descent with a fixed step-size on $f(x, y)$ starting with a point such that $x = y$. Plot the function value at each iteration, and the trajectory of the points on the contour plot.

Solution

The plot of the function value at each iteration is shown in the figure 15 and the function value at each iteration is shown in the figure 16.

Problem 5

Run gradient descent with a decreasing step-sizes on $f(x, y)$ starting with a point such that $x = y$. Plot the function value at each iteration, and the trajectory of the points on the contour plot.

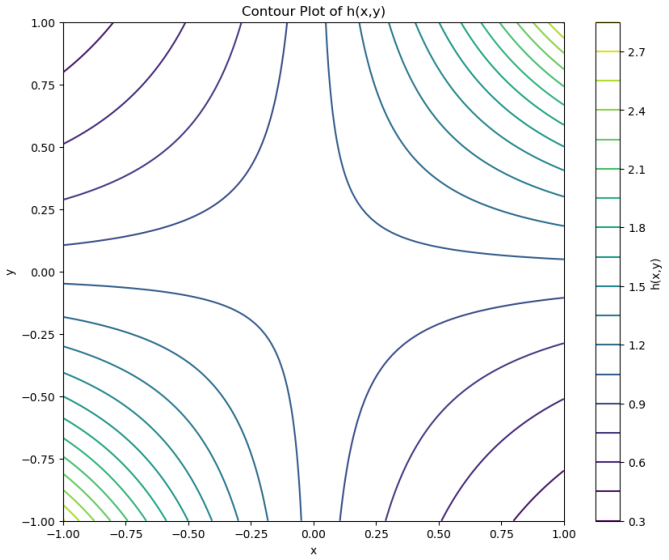

Figure 14: Contour plot of $f(x, y)$

Solution

The plot of the function value at each iteration is shown in the figure 18 and the function value at each iteration is shown in the figure 19.

Problem 6

Run perturbed gradient descent with a fixed step-size and noise-variance on $f(x, y)$ starting with a point such that $x = y$. Plot the "expected" function value at each iteration, and the trajectory of the points on the contour plot.

Solution

The contour plot is shown in the figure 21. The expected function value at each iteration is shown in the figure 22.

Problem 7

Run perturbed gradient descent with a fixed step-size and decreasing noise-variance on $f(x, y)$ starting with a point such that $x = y$. Plot the "expected" function value at each iteration, and the trajectory of the points on the contour plot.

Solution

The contour plot is shown in the figure 24. The expected function value at each iteration is shown in the figure 25.

Problem 8

Run perturbed gradient descent with decreasing step-sizes and a fixed noise-variance on $f(x, y)$ starting with a point such that $x = y$. Plot the "expected" function value at each iteration, and the trajectory of the points on the contour plot.

Solution

The contour plot is shown in the figure 27. The expected function value at each iteration is shown in the figure 28.

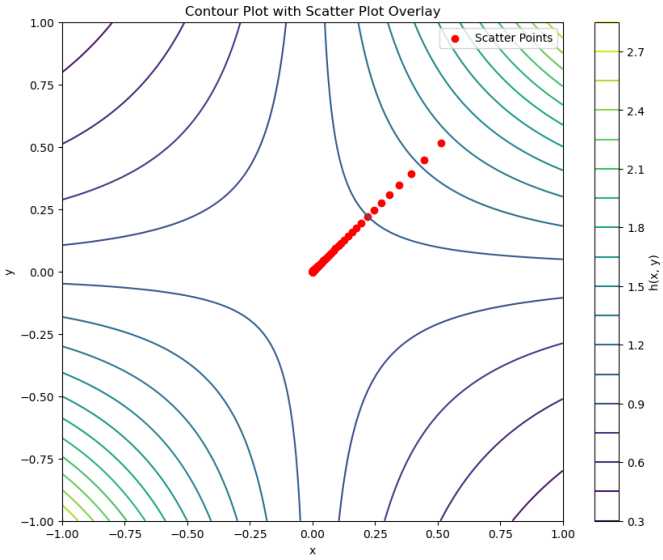

Figure 15: Contour plot alongwith (x_t, y_t) at each iteration with constant learning rate

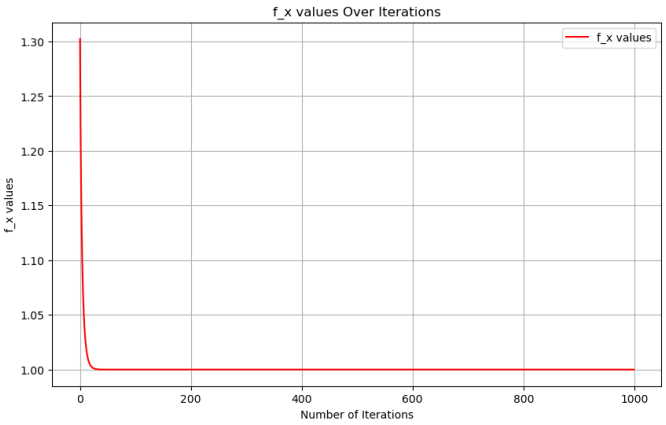

Figure 16: Function value at each iteration with constant learning rate

Figure 17: Combined figures showing contour plot and function value over iterations.

Problem 9

Run perturbed gradient descent with decreasing step-sizes and noise-variances on $f(x, y)$ starting with a point such that $x = y$. Plot the "expected" function value at each iteration, and the trajectory of the points on the contour plot.

Solution

The contour plot is shown in the figure 30. The expected function value at each iteration is shown in the figure 31.

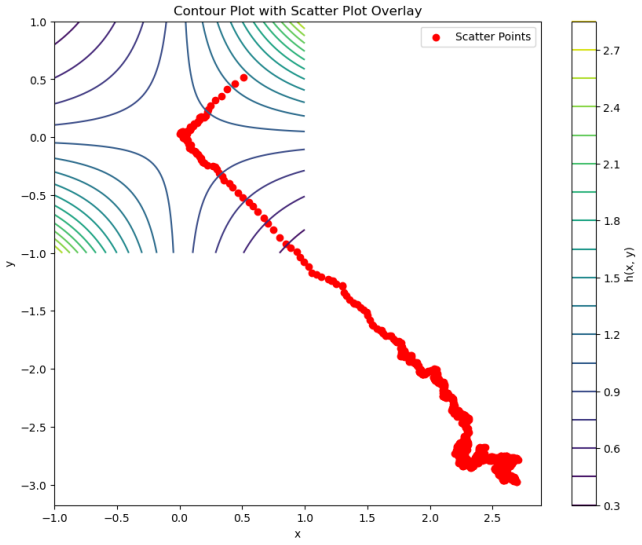

Figure 21: Contour plot alongwith (x_t, y_t) at each iteration with fixed step-size and noise-variance

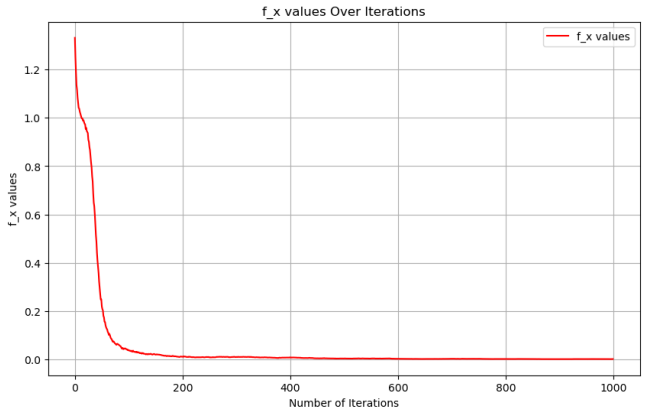

Figure 22: Function value at each iteration with fixed step-size and noise-variance

Figure 23: Combined figures showing contour plot and function value with fixed step-size and noise-variance.

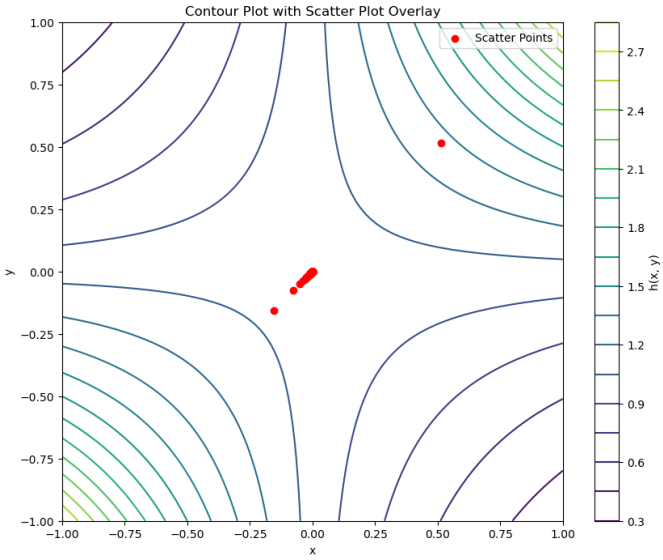

Figure 18: Contour plot alongwith (x_t, y_t) at each iteration with diminishing learning rates

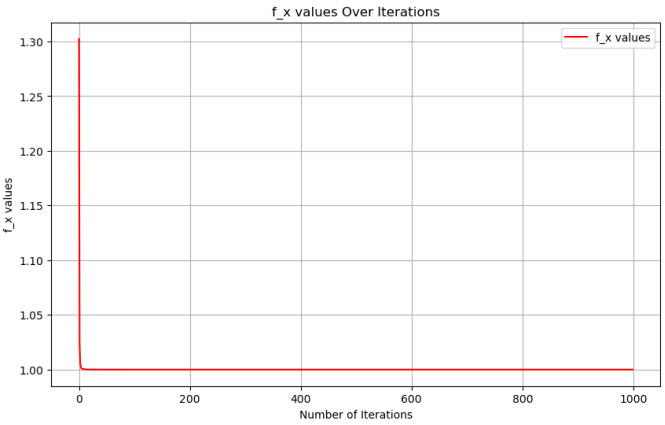

Figure 19: Function value at each iteration with diminishing learning rate

Figure 20: Combined figures showing contour plot and function value with diminishing learning rates.

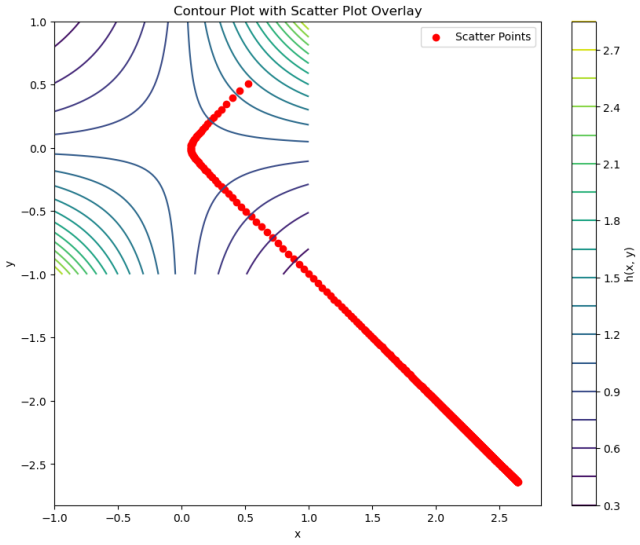

Figure 24: Contour plot alongwith (x_t, y_t) at each iteration with fixed step-size and diminishing noise-variance

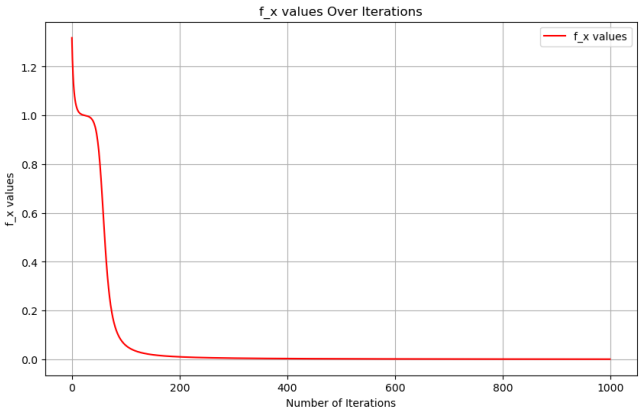

Figure 25: Function value at each iteration with fixed step-size and diminishing noise-variance

Figure 26: Combined figures showing contour plot and function value with fixed step-size and diminishing noise-variance.

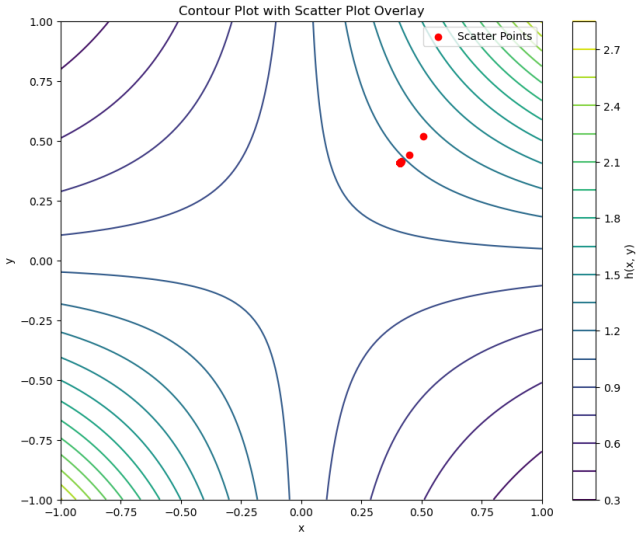

Figure 27: Contour plot alongwith (x_t, y_t) at each iteration with diminishing step-sizes and fixed noise-variance

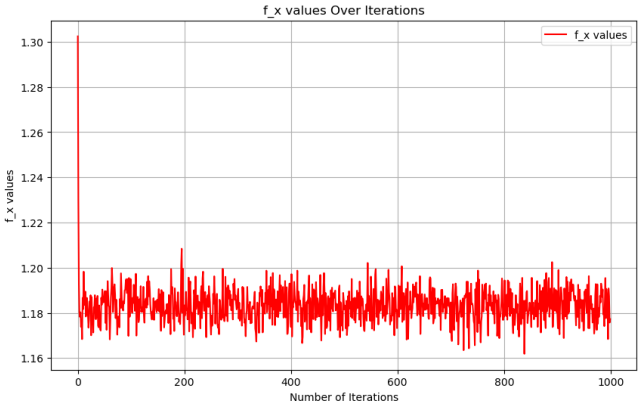

Figure 28: Function value at each iteration with diminishing step-sizes and fixed noise-variance

Figure 29: Combined figures showing contour plot and function value with diminishing step-sizes and fixed noise-variance.

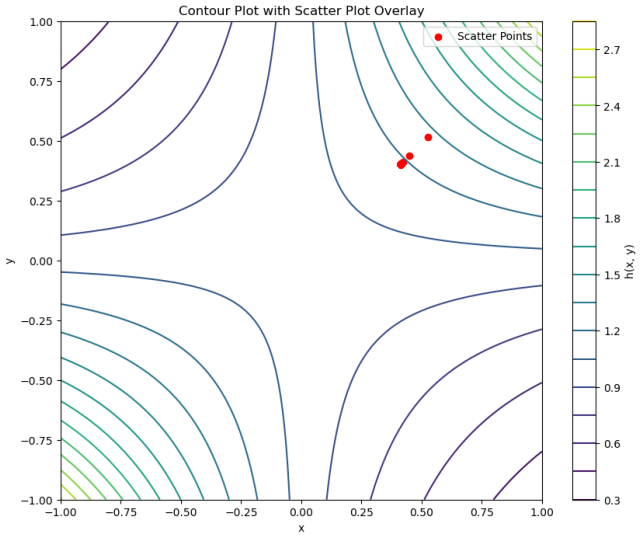

Figure 30: Contour plot alongwith (x_t, y_t) at each iteration with diminishing step-sizes and noise-variances

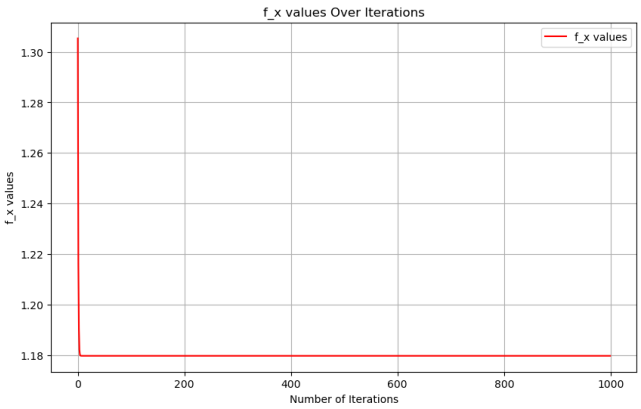

Figure 31: Function value at each iteration with diminishing step-sizes and noise-variances

Figure 32: Combined figures showing contour plot and function value with diminishing step-sizes and noise-variances.

Problem 10

Find the relation between $\mathbb{E}[f^{(t+1)}]$ and $\mathbb{E}[f^{(t)}]$.

Solution

$$
\begin{aligned}
f^{(t+1)} &= e^{x^{(t+1)}y^{(t+1)}} \\
x^{(t+1)} &= x^{(t)} - \alpha_t \left(y^{(t)} e^{x^{(t)}y^{(t)}} + \zeta_x^{(t)} \right) \\
y^{(t+1)} &= y^{(t)} - \alpha_t \left(x^{(t)} e^{x^{(t)}y^{(t)}} + \zeta_y^{(t)} \right)
\end{aligned}
$$

Hence,

$$
\begin{aligned}
x^{(t+1)}y^{(t+1)} &= \left(x^{(t)} - \alpha_t \left(y^{(t)} e^{x^{(t)}y^{(t)}} + \zeta_x^{(t)} \right) \right) \left(y^{(t)} - \alpha_t \left(x^{(t)} e^{x^{(t)}y^{(t)}} + \zeta_y^{(t)} \right) \right) \\
&= x^{(t)}y^{(t)} - \alpha_t \left((x^{(t)})^2 e^{x^{(t)}y^{(t)}} + x^{(t)}\zeta_y^{(t)} + (y^{(t)})^2 e^{x^{(t)}y^{(t)}} + y^{(t)}\zeta_x^{(t)} \right) \\
&+ \alpha_t^2 \left(y^{(t)}x^{(t)} e^{x^{(t)}y^{(t)}} + y^{(t)}\zeta_y^{(t)} e^{x^{(t)}y^{(t)}} + x^{(t)}\zeta_x^{(t)} e^{x^{(t)}y^{(t)}} + \zeta_x^{(t)}\zeta_y^{(t)} \right)
\end{aligned}
$$

For simplicity, let $\alpha_t = \alpha$. Further, $\mathbb{E}[\zeta_x^{(t)}] = \mathbb{E}[\zeta_y^{(t)}] = 0$ and $\mathbb{E}[\zeta_x^{(t)}\zeta_y^{(t)}] = 0$. Hence,

$$
\begin{aligned}
\mathbb{E}[x^{(t+1)}y^{(t+1)}] &= \mathbb{E}[x^{(t)}y^{(t)}] - \alpha\mathbb{E}[\left((x^{(t)})^2 + (y^{(t)})^2 \right) e^{x^{(t)}y^{(t)}}] + \alpha^2\mathbb{E}[y^{(t)}x^{(t)}e^{x^{(t)}y^{(t)}}] \\
&\leq \mathbb{E}[x^{(t)}y^{(t)}] + (\alpha^2 - 2\alpha)\mathbb{E}[x^{(t)}y^{(t)}e^{x^{(t)}y^{(t)}}] \\
&\leq \mathbb{E}[x^{(t)}y^{(t)}] + (\alpha^2 - 2\alpha)\mathbb{E}[x^{(t)}y^{(t)}]\mathbb{E}[e^{x^{(t)}y^{(t)}}] \\
&\leq \mathbb{E}[x^{(t)}y^{(t)}] + (\alpha^2 - 2\alpha)\mathbb{E}[x^{(t)}y^{(t)}]e^{\mathbb{E}[x^{(t)}y^{(t)}]}
\end{aligned}
$$

Hence,

$$
\begin{aligned}
\mathbb{E}[f^{(t+1)}] &= e^{\mathbb{E}[x^{(t+1)}y^{(t+1)}]} \\
&\leq e^{\mathbb{E}[x^{(t)}y^{(t)}]} \times e^{(\alpha^2-2\alpha)\mathbb{E}[x^{(t)}y^{(t)}]e^{\mathbb{E}[x^{(t)}y^{(t)}]}} \\
&\leq \mathbb{E}[f^{(t)}] \times e^{(\alpha^2-2\alpha)\mathbb{E}[x^{(t)}y^{(t)}]\mathbb{E}[f^{(t)}]} \\
&\leq \mathbb{E}[f^{(t)}] \times e^{(\alpha^2-2\alpha)\mathbb{E}[\ln f]\mathbb{E}[f^{(t)}]}
\end{aligned}
$$

Hence,

$$
\mathbb{E}[f^{(t+1)}] \leq \mathbb{E}[f^{(t)}] \times e^{(\alpha^2-2\alpha)\mathbb{E}[\ln f]\mathbb{E}[f^{(t)}]}
$$

4: Zeroth-order Optimisation (10 points)

Until now, we have been relying on gradients of functions to find their minimisers. Such methods are referred to as first-order methods because they use the first derivatives. Similarly, methods that also utilise the Hessian are called second-order methods. But, what if we do not have access to the gradients? Such situations may arise due to a variety of factors:

- if the gradients simply don't exist, i.e., the function is not differentiable (in these cases, however, one may resort to using the subgradients. But this is out of the scope of this exercise),
- or, the gradients may exist, but computing them is expensive (temporally or monetarily),
- or, as it happens very frequently, we are attempting to optimise a black-box.

All is, however, not lost; because there exist methods that aid in optimisation without the use of gradients. These are referred to as zeroth-order methods. In this exercise, we will learn about two such methods that work on functions $f : \mathbb{R} \mapsto \mathbb{R}$. These methods find the minima (or maxima) of the function f in the interval $[a, b]$, with a major caveat being that the function is unimodal (i.e., only one minimum (or maximum) exists) in the given range. If multiple extrema are present, then these methods converge to any of them, and not necessarily the best one.

4.1 Golden Section Search

The method is so named because the search-interval reduces by a factor of $\rho = \left(1 - \frac{1}{\phi}\right)$ at each iteration, where $\phi = \frac{1+\sqrt{5}}{2}$ is the golden ratio. This method uses only a single evaluation of the function at each iteration, and proceeds as follows:

1. Start with the initial search-interval $[a, b]$.
2. Define $x_1 \coloneqq a + \rho(b - a)$ and $x_2 \coloneqq b - \rho(b - a)$.
3. Consider the tuple $[a, x_1, x_2, b]$.
4. If $f(x_1) \leq f(x_2)$, change the tuple to $[a, x_3, x_1, x_2]$ where $x_3 \coloneqq a + \rho(b - a)$. Else, if $f(x_1) \geq f(x_2)$, change the tuple to $[x_1, x_2, x_3, b]$ where $x_3 \coloneqq b - \rho(b - a)$.
5. If convergence is reached, return the final interval; or go back to step 3 with the new tuple.

4.2 Fibonacci Search

This method is similar to golden section search, with the slight modification that instead of using a constant ρ, its value is updated at each iteration t to get:

$$\rho_t = 1 - \frac{F_{T-t-1}}{F_{T-t+2}}$$

where T is the total number of iterations, and $\{F\}$ is the Fibonacci sequence with $F_{-1} = 0$ and $F_0 = 1$. Also, the final search interval after T iterations is given by $\frac{(b-a)}{F_{T+1}}$.

Consider the function:

$$f(x) = x(x - 1)(x - 3)(x + 2); \quad x \in \mathbb{R}$$

Problem 1

Find all the extrema of $f(x)$.

Solution

$$\begin{aligned} f(x) &= x(x-1)(x-3)(x+2) \\ &= x^4 - 2x^3 - 5x^2 + 6x \\ \nabla f(x) &= 4x^3 - 6x^2 - 10x + 6 \\ \nabla^2 f(x) &= 12x^2 - 12x - 10 \end{aligned}$$

Observe that $\nabla f(x) = 2(2x-1)(x^2 - x - 3)$. Hence, the extrema of $f(x)$ are at

$$x \in \left\{ \frac{1}{2}, \frac{1+\sqrt{13}}{2}, \frac{1-\sqrt{13}}{2} \right\}$$

To classify these points as local maxima or minima, we can use the second derivative test. The second derivative test states that if $f''(x) > 0$ at a point x, then x is a local minima. If $f''(x) < 0$ at a point x, then x is a local maxima. If $f''(x) = 0$ at a point x, then the test is inconclusive. From the calculations, we get:

$$\nabla^2 f\left(\frac{1}{2}\right) < 0$$

$$\nabla^2 f\left(\frac{1 \pm \sqrt{13}}{2}\right) > 0$$

Hence, $x = \frac{1}{2}$ is a point of local maxima and $x = \frac{1 \pm \sqrt{13}}{2}$ are points of local minima.

Problem 2

Now, restrict yourself to the interval $[1, 3]$, with the objective of finding the minimum within this interval with a range of 10^{-4} using:

(a) Golden section search. First, report the number of iterations required to achieve the aforesaid precision. Then, implement the method and tabulate the intervals at each iteration. Also, plot $f(a_t)$, $f(b_t)$, $(b_t - a_t)$ and $\frac{(b_t - a_t)}{(b_{t-1} - a_{t-1})}$ at each iteration.

(b) Repeat all of the above for Fibonacci search.

Solution

(a)

The algorithm converges at $x = 2.3027522726894833$ with number of iterations = 21. The intervals at each iteration is shown in table 1. The plots are shown in the figure 33.

(b)

The algorithm converges at $x = 2.3055555555555554$ with number of iterations = 10. The intervals at each iteration is shown in table 2. The plots are shown in the figure 34.

Iteration	a_t	b_t
1	1.0000000000000000	3.0000000000000000
2	1.7639320225002104	3.0000000000000000
3	1.7639320225002104	2.5278640450004204
4	2.0557280900008412	2.5278640450004204
5	2.2360679774997900	2.5278640450004204
6	2.2360679774997900	2.4164078649987380
7	2.2360679774997900	2.3475241575014720
8	2.2786404500042060	2.3475241575014720
9	2.2786404500042060	2.3212129225086224
10	2.2949016875157726	2.3212129225086224
11	2.2949016875157726	2.3111629250273396
12	2.2949016875157726	2.3049516849970560
13	2.2987404449667720	2.3049516849970560
14	2.3011129275460567	2.3049516849970560
15	2.3011129275460567	2.3034854101253410
16	2.3020191352536260	2.3034854101253410
17	2.3025792024177710	2.3034854101253410
18	2.3025792024177710	2.3031392695819166
19	2.3025792024177710	2.3029253429611960
20	2.3027114163404750	2.3029253429611960
21	2.3027114163404750	2.3028436302631790

Table 1: Golden Ratio Search Iterations

Iteration	a_t	b_t
1	1	3
2	1.7638888888888888	3
3	1.7638888888888888	2.5277777777777777
4	2.0555555555555554	2.5277777777777777
5	2.236111111111111	2.5277777777777777
6	2.236111111111111	2.4166666666666665
7	2.236111111111111	2.3472222222222223
8	2.2777777777777777	2.3472222222222223
9	2.2777777777777777	2.3194444444444446
10	2.2916666666666665	2.3194444444444446

Table 2: Fibonacci Search Iterations

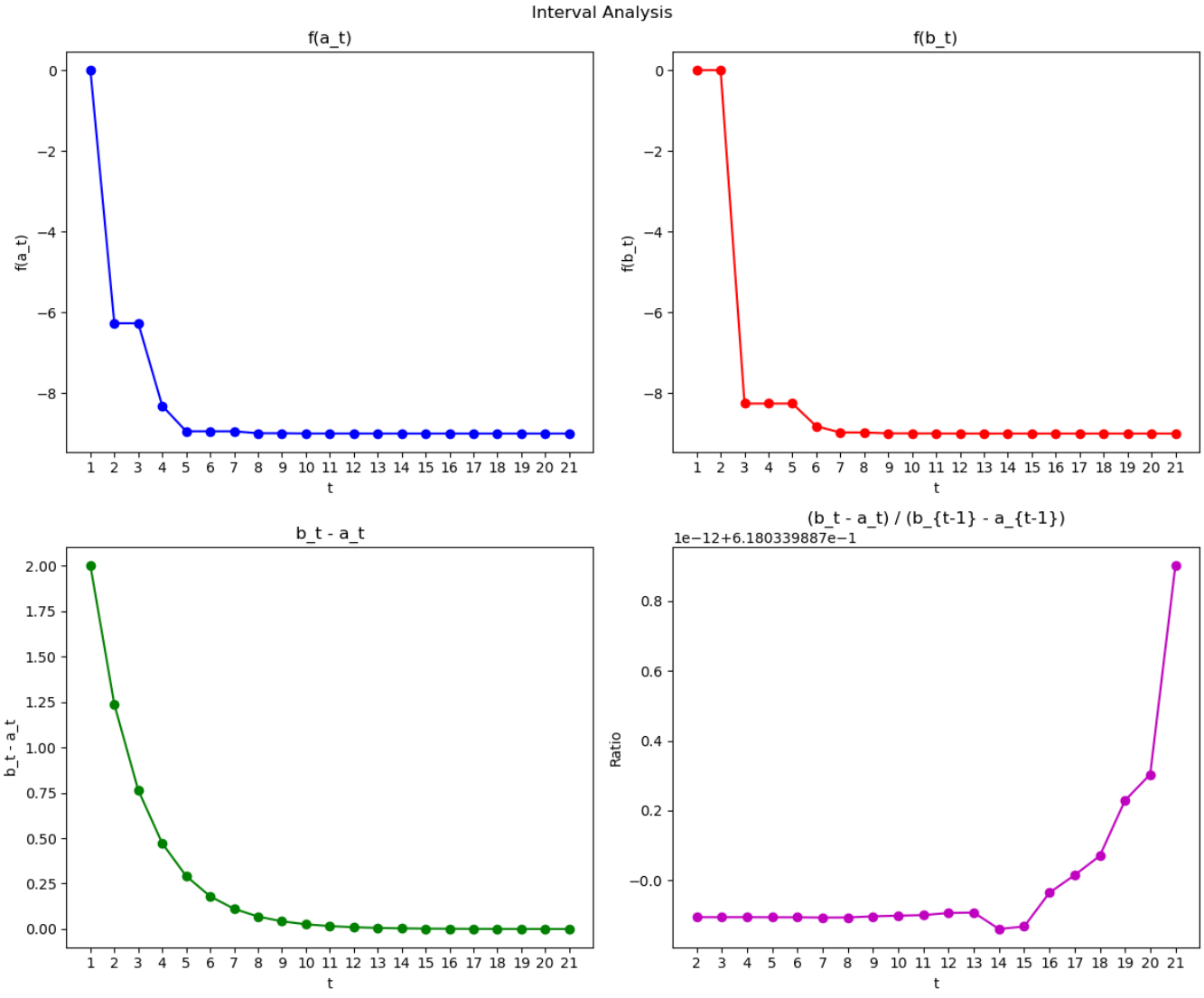

Figure 33: Golden Ratio Search Plots

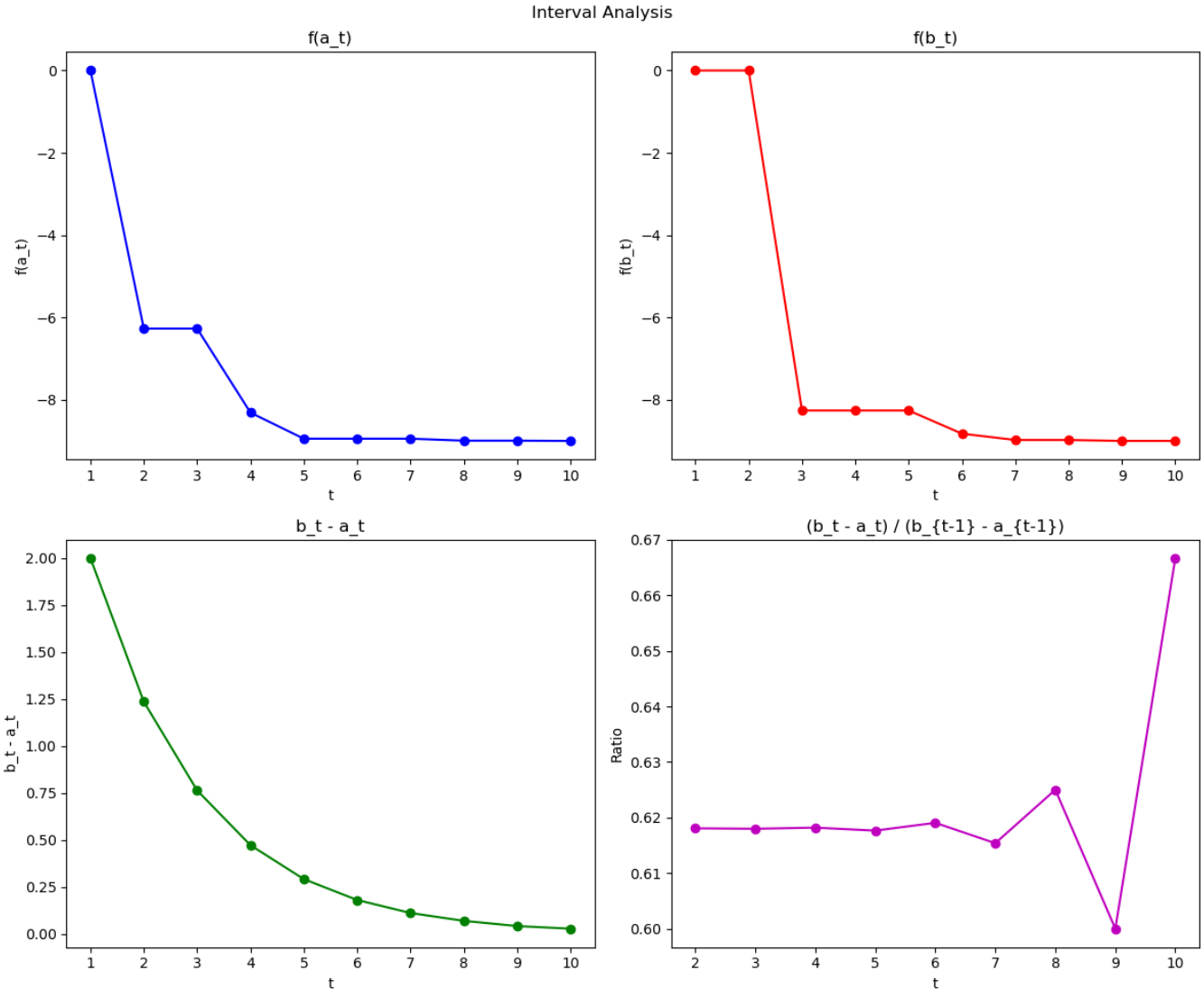

Figure 34: Fibonacci Search Plots