Lab Manual for Numerical Analysis

Lab No. 12

NUMERICAL OPTIMIZATION - GOLDEN SECTION SEARCH METHOD

BAHRIA UNIVERSITY KARACHI CAMPUS
Department of Software Engineering

NUMERICAL ANALYSIS

LAB EXPERIMENT # 12

Numerical Optimization - Golden Section Search Method

OBJECTIVE:

The objective of this lab is to delve into Numerical Optimization techniques, specifically emphasizing the **Golden Section Search Method.**

Introduction

Numerical optimization is a field within numerical analysis focused on finding the best solution for a given problem by **minimizing** or **maximizing** a function. It involves employing computational algorithms to efficiently search for the optimal values of parameters within a defined range. For instance, optimizing the design of a structure by minimizing material usage while maintaining structural integrity is a typical numerical optimization problem. These computations often involve functions where analytical solutions are complex or impractical to derive.

Various numerical techniques are utilized in optimization problems. **Gradient-based methods**, like Gradient Descent, use the derivative information of the function to iteratively converge towards the optimal solution. **Evolutionary algorithms**, such as Genetic Algorithms or Particle Swarm Optimization, mimic natural selection principles to search for the best solution in a population-based manner. Additionally, methods like the **Golden Section Search**, which doesn't require derivatives, iteratively narrows down the search space to identify the optimal point.

Numerical optimization has wide-ranging applications across numerous domains. It's extensively used in engineering for designing structures, optimizing mechanical systems, and parameter tuning in control systems. In finance, it's applied for portfolio optimization and risk management. Furthermore, in machine learning, optimizing model parameters is crucial for enhancing predictive accuracy. With its relevance in diverse fields, numerical optimization continually drives innovation and problem-solving.

In this lab, our focus will be on the **Golden Section Search Method**, a unimodal function optimization technique. The Golden Section Search doesn't rely on derivatives and efficiently narrows down intervals by utilizing the golden ratio. Through this exploration, we aim to understand the principles and practical applications of this method in solving optimization problems efficiently.

Implementation of Golden Section Search

The **Golden Section Search** method is a technique used for finding the **minimum** or **maximum** of a **unimodal function** within a given interval. It's an iterative algorithm based on the principle of the golden ratio that efficiently narrows down the search interval to locate the optimal value.

The method starts with an interval known to contain the **optimal value** and iteratively divides this interval into smaller sub-intervals based on the golden ratio. By evaluating function values at specific points within these intervals, it determines which sub-interval is more likely to contain the optimal solution. The search continues until the interval becomes sufficiently small, indicating a convergence to the optimal value. Golden Section Search is applicable to unimodal functions, meaning it possesses only **one minimum** or **maximum** within the given interval. It assumes that the function is continuous over the interval and that the optimum lies within this range.

The iteration in Golden Section Search involves selecting two points within the interval, ***a* and *b*,** such that the ratio of the larger sub-interval to the whole interval is the golden ratio **(ϕ, approximately 1.618)**). The updated points are determined based on comparisons of function evaluations at these points, progressively narrowing down the interval.

Implementation in Python

The following is the Python implementation that finds the minimum as well as maximum of a function **f** within the interval **[a, b]** using the Golden Section Search method.

```
def golden_section_search_min(f, a, b, tol=1e-5):
    phi = (1 + 5 ** 0.5) / 2  # Golden ratio

    while abs(b - a) > tol:
        x1 = b - (b - a) / phi
        x2 = a + (b - a) / phi

        if f(x1) < f(x2):
            b = x2
        else:
            a = x1

    return (a + b) / 2, f((a + b) / 2)


def golden_section_search_max(f, a, b, tol=1e-5):
    phi = (1 + 5 ** 0.5) / 2  # Golden ratio

    while abs(b - a) > tol:
        x1 = b - (b - a) / phi
        x2 = a + (b - a) / phi

        if f(x1) > f(x2):  # Inverted comparison for finding maximum
            b = x2
        else:
            a = x1

    return (a + b) / 2, f((a + b) / 2)
```

```python
# Example usage:
# Define the function to be optimized (modify accordingly)
def func(x):
    return (x - 3) ** 2  # Example quadratic function


# Set the initial interval [a, b] (modify as needed)
a, b = 0, 5


# Find the minimum of the function within the interval
min_value = golden_section_search_min(func, a, b)
print("Minimum value and position:", min_value)


# Find the maximum of the function within the interval
max_value = golden_section_search_max(func, a, b)
print("Maximum value and position:", max_value)
```

Lab Tasks

1. Write a Python program that implements golden section search method to find the max and min value of the following functions:

    a. $x^2 + 2\ ;\ -3 \le x \le 5$

    b. $0.5 - x\, e^{-x^2} + 2\ ;\ 0 \le x \le 2$