Homework #1

Implement Four root finding methods & Disscusion

GIST AI

20231126 황산하

\*코드는 Python으로 작성했고 Report 이외에 스크립트도 Report와 함께 Zip으로 압축하여 제출하였습니다.

INDEX

1. Comparison of Four root finding methods

2. Comparison of secant and regular falsi with a given equation.

3. Appendix: My codes

1. Comparison of Four root finding methods

**1)인수분해를 통해 간단하게 구할 수 있는 함수지만 해가 여러 개인 함수(F1)**와 **2)해를 구하는게 좀 더 복잡한 임의의 함수지만 해가 한 개인 함수(F2)**, **3)지수함수와 같은 다른 복합적인 함수로 표현된 함수(F3)** 로 각각의 성능과 특징을 분석했다. 사용한 함수는 다음과 같다.

또한 엄밀한 비교를 위해 HyperParameter는 최대한 일치시켰다. 시작점을 같게 하고 (Bisection method 같은 경우는 f(a)부터 탐색하기 때문에 이를 시작점으로 생각했다.) Tolerance 변수는 1e-6으로 주어 비슷한 지점까지 iteration이 반복하도록 했다.

[각각의 방법들이 수렴하는 그림 하나 첨부]

1) Bisection Method

Bisection Method는 범위가 반씩 줄어들기 때문에, 3개의 함수 모두 아무리 범위를 크게 줘도 실제 iteration에는 차이가 크게 나지 않았다. 1000이상 범위가 차이나도 iteration은 2~4번 정도 차이가 났고, 이 정도는 Unimodal Root finding에서는 차이가 크게 체감되지 않는다. 따라서 범위를 좁게 주었을 때 그 안에 해가 없다면 오히려 해를 못찾을 수 있기 때문에 범위를 최대한 넓게 주는게 유리하다고 생각된다. 또한 F2, F3 함수의 경우, 범위를 어떻게 잡든 해를 중간에 건너 뛰는 바람에 아주 정확한 해를 찾지는 못했는데, 그럼에도 해에 수렴했기 때문에 최대한 해를 탐색하며 찾아보려는 수치적 기법으로서는 탁월한 방법이라고 생각된다.

2) Newton Method

Newton Method도 해로 잘 수렴하는 모습을 보여주었다. 하지만 시작점이 해로부터 멀어질수록 Bisection Method에 비해서 iteration이 크게 증가했다. Bisecton Method는 (-1000, 1000) 범위에서 32번 iteration을 하는 반면, Newton Method는 49번을 탐색하고서야 알고리즘이 종료되었다. 또한 Newton Method를 구현하는 과정에서 f(x)를 df(x) 값으로 나누는 과정이 있는데, 미분값이 0으로 수렴하게 되면 값을 못구하는 경우가 생겼다. 따라서 이 경우를 방지하기 위해 h라는 작은 변수를 더해서 나누어주었다. Newton Method의 장점은 Bisection Method와는 달리 초기 값을 어떻게 주더라도 해의 근방으로 수렴할 수 있다는 점이었다.

3) Secant Method

4) Regular Falsi Method