**고급소프트웨어실습1 - 6주차 과제**

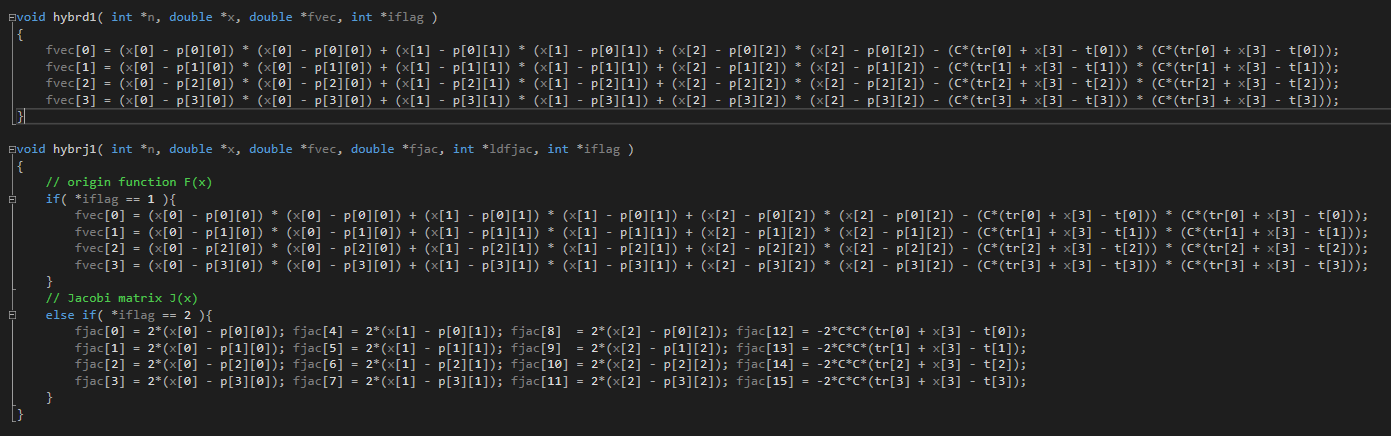
수치 컴퓨팅 실험 3: 공개 소프트웨어를 사용한 문제 풀이

|  |  |  |
| --- | --- | --- |
| **분반** | **:** | 3 |
| **학번** | **:** | 20121635 |
| **이름** | **:** | 장종석 |

***1.  
(ⅰ) minpack에서 제공하는 FORTAN 함수인 HYBRJ1을 사용하여 다음과 같이 정의되는 비선형 방정식 시스템을 풀어 GPS 수신기 위치를 찾아주는 프로그램(프로그램3-1)을 작성하라.***

***(ⅱ) minpack에서 제공하는 FORTRAN 함수인 HYBRD1을 사용하여 위 문제를 반복하라, 결과는 이름이 GPS\_position\_3-2\_i.txt인 파일에 저장하라.***

(i),(ii)를 해결하기 위하여 hybrd1, hybrj1 함수를 다음과 같이 구현하였다.



***(ⅲ) 위 두방법을 사용하여 문제를 풀면서 알게된 내용을 보고서 형식의 파일에 기술하여 제출하라.***

6주차 실습의 경우 FORTAN에서 사용할 수 있는 minpack이라는 패키지를 이용하여 C언어에서 비선형 방정식의 근을 구하였다.  
 이는 FORTRAN 컴파일러로 minpack 에서 사용하고 자하는 함수들을 컴파일하여 오브젝트 코드로 변환하여, C컴파일러에 링크 시킴으로써 가능했다. 이렇게 사용하는 함수에서 2가지 주의할 점이 존재하였으며, 아래와 같다.

1. 매개 변수 전달 방법

C언어는 기본값으로 Call by value로 변수를 함수에 전달하지만, FORTRAN은 Call by reference로 변수를 함수에 전달한다. 그러므로 C에서 FORTRAN 함수를 호출할 때에는 반드시 모든 매개 변수를 Call by reference로 생각하여 넘겨주어야 한다. 즉, 일반적인 변수는 반드시 그 주소를 넘겨주어야 하며, 배열은 그 자체가 포인터이니 배열 이름만 넘겨주면 된다.

1. 행렬 저장 방법

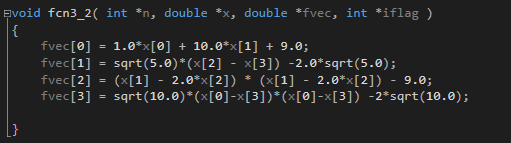
C언어는 2차원 배열을 참조는 순서가 행 우선으로 가장 상단부터 오른쪽으로 흘러가는 방향으로 읽지만, FORTRAN은 열 우선이기 때문에 가장 좌측부터 아래쪽으로 흘러가는 방향으로 읽는다. 그러므로 C에서 FORTRAN 함수로 2차원 배열을 넘겨줄 때에는 FORTRAN 함수가 정상적으로 배열을 읽을 수 있도록 전치 행렬의 형태로 넘겨주어야 한다.

이 두가지를 주의하면, C에서 FORTAN함수를 자유롭게 사용할 수 있었다.

위의 문제(GPS 수신기 위치 찾기)를 풀기 위하여 사용한 함수(HYBRJ1, HYBRD1)의 경우 비선형 방정식의 근을 구하기 위한 함수로, HYBRJ1은 비선형 방정식들의 미분에 대한 정보를 저장하는 야코비 행렬(Jacobian Matrix)을 구해서 넘겨주어야 하는 반면, HYBRD1은 야코비 행렬을 구해서 넘겨주지 않아도 야코비 행렬을 근사적으로 추정하여 근을 구한다.

***2.  
(ⅰ) minpack에서 제공하는 적절한 FORTRAN 함수를 사용하여 다음과 같이 정의되는 비선형 방정식 시스템의 근을 찾아주는 프로그램(프로그램 3-2)를 작성하라.***

문제를 해결하기 위하여 HYBRD1함수를 이용하였다. HYBRJ1의 경우 직접 야코비 행렬을 구해야 하기 때문에 부담이 되어 HYBRD1함수를 사용하였으며, 구현의 경우 아래와 같이 구현하였다.  
HYBRD1에서 x[0] = x, x[1] = y, x[2] = z, x[3] = w 의 값을 담고 있다.



***(ⅱ) 자신이 구한 결과는 이름이 roots\_found\_3-2.txt인 파일에 이해하기 쉬운 포맷으로 저장하라. 이 결과 파일에는 자신이 구한 각 근이 얼마나 정확한 것인지를 보이는 내용을 포함해야 한다.***

테스트를 한 초기값의 경우 문제에서 주어진 것처럼 {0.9, -0.9, 1.25, -1.25}으로 설정을 하였다.  
HYBRD1함수를 돌리고 나면 fvec 행렬에 결과값이 담기게 되는데, 해당값을 확인하여 근의 값을 대입하였을때 0이 되는지 확인을 할 수 있어, 해당값을 확인하면, 우리가 구한 근이 잘 구해졌는지 알 수 있다.

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  | X | Y | Z | W |
| Initial value | 0.9 | -0.9 | 1.25 | -1.25 |
| Solution | 0.442108 | -0.944211 | 1.027895 | -0.972105 |
| F(x,y,z,w) | f1: 0.000000 | f2: 0.000000 | f3: -0.000000 | f4: 0.000000 |

추가로, f3의경우 -0.0000으로 앞에 –부호가 붙어있는 것을 볼 수 있는데, 우리가 구한 근의 경우 근사근이기 때문에 완벽히 0으로 만드는 값은 찾지 못하여 소수점 아래자리에 0이아닌 값이 존재하여 발생하는 경우로, 판단할 수 있다.

***4. (ⅲ) 조교가 제공하는 선형방정식들에 대하여 실험을 해본 후, 오차에 대한 척도를 통하여 자신이 얼마나 정확한 근을 구했는지에 대하여 보고서 형식의 파일에 분석하여 제출하라.***

제공한 선형방정식들에 대한 테스트 결과를 정리하면 다음과 같다.  
<테스트1>

|  |  |
| --- | --- |
| Input | output |
| General\_2 | 2  1.200002  1.200001  0.000000 |

<테스트2>

|  |  |
| --- | --- |
| Input | output |
| General\_3 | 3  1.199997  1.200000  1.200001  0.000000 |

<테스트3>

|  |  |
| --- | --- |
| Input | output |
| General\_4 | 4  1.200000  1.200000  1.200000  1.199998  0.000000 |

<테스트4>

|  |  |
| --- | --- |
| Input | output |
| General\_12 | 12  1.200001  1.200001  1.199999  1.199999  1.200001  1.199998  1.199999  1.199999  1.199999  1.200001  1.200001  1.199998  0.000000 |

<테스트5>

|  |  |
| --- | --- |
| Input | output |
| Hilbert\_9 | 9  1.195502  1.312529  0.501077  2.918451  -0.745466  2.674514  -0.626610  3.060824  0.509489  0.000000 |

<테스트6>

|  |  |
| --- | --- |
| Input | output |
| Hilbert\_16 | 16  1.201568  1.173602  1.260496  1.244426  1.403662  0.444684  0.752322  1.805505  2.525773  1.637263  0.707327  -0.007822  0.139721  1.551384  2.224458  1.141567  0.000000 |

<테스트7>

조교가 제공한 모든케이스에 대하여 근을 구해본 것은 아니지만, 임의로 추출하여 테스트를 진행하였다 output의 가장 마지막에 있는 부분이 오차식을 사용하여 구한 오차값인데, 모두 0.000000(유효숫자 6자리) 인 것으로 보았을 경우, 근이 잘 구해졌다고 말할 수 있다.