**HW#3:**

**Multigroup 1-D Eigenvalue Problem with**

**Chebyshev Acceleration**

과 목 명 : 원자로 수치해석과 설계

제출일자 : 2010.04.19

학 번 : 2006-12015

이 름 : 임 창 현

**Part 1. Input Processing and Composition k∞ Determination**

**1. Modify the input processing routine you wrote for HW#1 such that the group wise cross section data are read from a separate file. Now you will have two input files: the main input file and the cross section (XSEC) file. The XSEC file should have the following structure:**

숙제 1에서 작성한 입력 처리 루틴을 주어진 입력 자료를 읽을 수 있도록 수정하였다. 입력자료의 형식은 대부분 동일하지만, 마지막에 Chebyshev 가속과 관련된 자료가 존재한다. 추가로 주어진 반응단면적 자료를 분리된 파일로 저장하고, 그 파일명을 주 입력파일의 마지막 줄에 써넣어 원할 시 반응단면적 자료를 바꿔가며 구동할 수 있도록 하였다. 명령어 뒤에 주 입력파일명을 써넣지 않으면 기본 입력파일을 읽게 된다. 다음은 반응 단면적 자료를 읽기 위해 새로 추가한 부분이다.

표 1. 반응 단면적 자료 입력 부분

|  |
| --- |
| **…**  **read**(io5, \*) ng, incomp    **allocate**(typedata(ncomp))  **do** i=1, ncomp  **allocate**(typedata(i)%nusigf(ng), typedata(i)%d\_coeff(ng), typedata(i)%sigr(ng), typedata(i)%chi(ng))  **allocate**(typedata(i)%sigs(ng, ng))  **end do**    **do** i=1, incomp  **read**(io5, \*)  **do** j=1, ng  **read**(io5, \*) ig, typedata(i)%d\_coeff(j), typedata(i)%sigr(j), typedata(i)%nusigf(j), typedata(i)%chi(j)  **end do**  **do** j=1, ng  **read**(io5,'(I3, 100(e13.6))') ig,(typedata(i)% sigs(jg, j), jg=1, ng)  **end do**  **end do**  **…**  **do** i=1, ncomp  **allocate**(typedata(i)%gb(ng), typedata(i)%ge(ng))  **do** j=1, ng  igb=500; ige=0  **do** k=1, ng  **if** (typedata(i)%sigs(k, j).ne.0) **then**  **if** (k.lt.igb) igb=k  **if** (k.gt.ige) ige=k  **end if**  **end do**  typedata(i)%gb(j)=igb  typedata(i)%ge(j)=ige  **if** (igb.eq.500) typedata(i)%gb(j)=1  **if** (ige.eq.0) typedata(i)%ge(j)=1  **end do**  **end do**  **…** |

표. 1에 나타난 부분은 각 물질타입 별로 반응 단면적 자료를 저장하고, 산란 단면적을 확인하여 산란 경계를 변수로 저장한다. 이는 후에 산란 단면적을 이용할 때 불필요한 계산을 피할 수 있게 해준다.

**2. Write a routine that determines the k-inf of each fuel composition. Use the cross section set attached at the end and compare your k-inf with those given.**

평판원자로가 하나의 메쉬로 구성되어 있고, 양 끝이 반사면으로 구성되어 있다고 할 때, 외벽을 통한 중성자의 이동이 존재하지 않기 때문에(혹은 들어오고 나가는 양이 동일하기 때문에) 중성자류는 0이된다. 따라서 중성자 균형 방정식은 다음과 같이 표현할 수 있다.



우변의 선원()을 1로 고정시킨 채(고정 선원) Power iteration을 한차례 수행해주면 중성자속을 구할 수 있고, 이를 바탕으로 선원을 갱신할 수 있다. 이를 바탕으로 증배계수를 구할 수 있다.





다음 표는 문제에 주어진 증배계수와 작성한 서브루틴을 이용하여 계산한 증배계수를 비교한 것이다. 계산 결과는 모든 물질 타입에서 주어진 값과 동일했다. 이는 1.에서 작성한 서브루틴이 주어진 입력 자료를 제대로 읽었다는 것을 의미한다.

표 . 증배계수

|  |  |
| --- | --- |
| k\_inf (Input) | k\_inf (Calc) |
| 1.202909 | 1.202909 |
| 0.921402 | 0.921402 |
| 1.128506 | 1.128506 |
| 1.151832 | 1.151832 |
| 1.067965 | 1.067965 |
| 0 | 0 |

**Part 2. Fission Source Iteration**

**1. Implement the fission source iteration scheme which consists of the following steps.**

이전 과제에서 사용하였던 중성자속 대신 핵분열 선원(Fission source)을 이용하여 Power iteration을 수행하는 것을 Fission source iteration이라 한다. 핵분열 선원은 모든 에너지군에 무관한 값으로, 위치에만 의존성을 갖는다. 따라서 중성자속 대신 핵분열 선원을 이용하는 경우 계산량이 에너지군 수만큼 줄어든다는 장점이 있다. 이번 과제를 수행하기 위해 Group Major Ordering 및 Box scheme을 이용하였다. 차분화 후 계수 행렬은 다음과 같이 나타난다.



과제 2에서 다루었던 행렬과 비교해보면, 단일 에너지군 문제에서 존재하던 행렬이 다 에너지군 문제에서는 반복해서 나타나고 있음을 확인할 수 있다. 각 에너지군을 대표하는 행렬의 세부구조는 단일 군 문제와 동일하다.

다 에너지군 문제에는 단일 군 문제에서는 고려하지 않았던 산란 선원 및 산란 누출이 존재한다. 산란 누출은 좌변에 존재하는 제거 단면적(Remover Xsec, 흡수 단면적 + 산란 단면적)에 반영이 되고, 따라서 단일 군에서 흡수 단면적으로 존재하던 항()이 제거 단면적으로 바뀐다.

행렬이 삼대각(Tri-diagonal) 행렬로 나타나기 때문에, 실제 프로그램 내에서 구현할 경우 나머지 항은 고려하지 않아도 된다. 이는 행렬 구성 시 사용되는 메모리를 절약할 수 있게 해주며, 불필요한 계산을 줄여 계산 속도 향상을 기대할 수 있게 한다. 문제를 풀기 위해서는 LU 행렬도 구해야 하는데, 삼대각 행렬을 다루기 때문에 각각 대각 성분과 하대각 성분(하대각 행렬), 상대각 성분(상대각 행렬)만을 갖게 된다. 즉 행렬 은 3열, 상대각 행렬(U)과 하대각 행렬(L)은 2열만을 필요로 한다. 이번 과제에서는 편의를 위해 모든 행렬을 3열로 잡고, 1열은 하대각 성분, 2열은 대각 성분, 3열은 상대각 성분의 정보를 갖게 하였다.



산란 선원은 우변에서 고려하게 된다. 이 문제에서는 upscattering이 없다 가정하였으므로, 이전 차에 구한 에너지군의 중성자속을 이용하여 우변에 더해주면 된다. 산란선원을 고려한 우변의 선원은 다음과 같이 나타난다.



각 행렬은 노드 수 \* 에너지군 수만큼 존재하지만, 각 에너지군 마다 독립적으로 선원 과 이항행렬이 존재하기 때문에, 한번에 푸는 대신 에너지군 마다 단계적으로 문제를 풀 수 있다. 이렇게 풀게 되면 푸는 과정에서 이전 에너지군의 중성자속을 가장 최신값으로 사용할 수 있기 때문에, 좀 더 빠른 수렴속도를 기대할 수 있게 된다. 문제에서 풀게되는 행렬식은 다음과 같다.



다음은 실제 Fission source iteration을 수행하는 부분이다.

|  |
| --- |
| …  !Loop over groups  **do** ig=1, ng  !Determine source at each node for the current group  **do** i=1, tn  gb => typedata(ntype(i))%gb; ge => typedata(ntype(i))%ge  sigs => typedata(ntype(i))%sigs  chi => typedata(ntype(i))%chi  nume=0  **do** jg=gb(ig), ge(ig)  nume=nume+sigs(jg,ig)\*phi(i,jg)  **end do**  s(i)=1/k\_ex\*chi(ig)\*psi(i)+nume  **if** (s(i).le.1E-15) s(i)=0    **nullify**(gb, ge, sigs, chi)    **end do**    !Solve LU  **call** solve\_lu(l,u,phi,s,ig)    **end do**    !Determine fission source  **do** i=1, tn  nusigf=>typedata(ntype(i))%nusigf  nume=0  **do** ig=1, ng  nume=nume+nusigf(ig)\*phi(i,ig)  **end do**  psi(i)=nume    **nullify**(nusigf)  **end do**  … |

LU 행렬은 반복문 전에 생성하고, 초기 분열 선원도 반복문 이전에 계산한다. 그 후 각 그룹에 대해 LU 행렬을 이용하여 Inverse power method 를 수행하게 된다. 모든 계산을 마치고, 분열 선원을 갱신한다.

**2. Solve the problem given below.**

|  |  |  |
| --- | --- | --- |
| k-eff | Iteration Number | Dominance Ratio |
| 1.039104 | 335 | 0.980787 |

코드를 작성한 뒤 주어진 문제를 풀었다. 그 결과는 위와 같이 나타났다. 중성자속 및 분열 선원은 뒤에 첨부하였다.

**Part 3. Chebyshev Acceleration**

**1. Implement the Chebyshev Acceleration scheme with the following logic:**

다 에너지군 문제는 단일 에너지군 문제에 비해 계산량이 많기 때문에 가속을 하지않으면 계샨 효율성이 매우 떨어진다. 따라서 적절한 가속 방법이 고려되어야 한다. 앞의 과제에서 사용하였던 Wielandt shift 방법은 다 에너지군의 경우 행렬을 복잡하게 만들기 때문에 큰 효과를 기대할 수 없다. 때문에 이용하는 것이 Chebyshev 가속 방법이다. 이는 동차 다항식 중 주어진 구간에서 가장 작은 최대값을 갖는 Chebyshev 다항식의 특성을 이용하는 것으로 오차가 가장 작은 최대값을 갖기 때문에 빠른 수렴속도를 기대할 수 있다.



Chebyshev 가속은 우선 Power method를 사용하여 값을 갱신한 후, Chebyshev Polynomial을 이용하여 가중치를 계산 후 외삽을 하여 새로운 값을 최종적으로 얻게 된다. 한 차의 반복계산을 마치면, Dominance ratio 를 재 추정하게 되는데 이는 기존에 가중치를 계산하는데 사용한 Dominance ratio가 정확한 값이 아니라 추정한 값이기 때문에 부정확할 가능성이 존재하기 때문이다. 다음은 실제 코드에서 Chebyshev 가속을 수행하는 부분이다.

|  |
| --- |
| **…**  ! \*\*\*\*\*\*\*\*\* Chebyshev Acceleration \*\*\*\*\*\*\*\*\*\*  **if** (c\_iter.eq.1) **then**  err\_psi\_base=psi\_pow-psi\_ex  alpha=2/(2-domi); beta=0  psi=alpha\*psi\_pow+(1-alpha)\*psi\_ex  **cycle**  **else**  err\_psi=psi\_pow-psi\_ex    inv\_domi=1/domi  gamma=2\*inv\_domi-1  nume=dcosh((c\_iter-1)\*dacosh(gamma))  deno=dcosh((c\_iter)\*dacosh(gamma))    !Determine Extrapolation parameter  alpha=4\*inv\_domi\*nume/deno  beta=(1-domi/2)\*alpha-1    !Perform extrapolation  psi=alpha\*psi\_pow+(1-alpha+beta)\*psi\_ex-beta\*psi\_ex\_ex    !Relative Error Reduction Factor  nume=0;deno=0;  **do** i=1, tn; nume=nume+err\_psi(i)\*err\_psi(i); **end do**  **do** i=1, tn; deno=deno+err\_psi\_base(i)\*err\_psi\_base(i); end do  erf=dsqrt(nume/deno)  xi=erf\*dcosh((c\_iter-1)\*dacosh(gamma))    !New dominance ratio  **if** (xi.gt.1) **then**  gamma\_s=dcosh(dacosh(xi)/(c\_iter-1))  **else**; gamma\_s=dcos(dacos(xi)/(c\_iter-1))  **end if**  domi=domi\*(1+gamma\_s)/2    !reset condition  **if** ((xi.lt.1).or.(c\_iter.ge.max\_cheb)) c\_iter=0    **end if**  **…** |

**2. Compare the performance of the Chebyshev acceleration with the power method solution for the given problem. Examine various orders of Chebyshev polynomial.**

|  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- |
| **parameter** | **Fis\_iteration** | **Wielandt shift**  **(0.05)** | **Wielandt shift**  **(0.1)** | **Wielandt shift**  **(0. 5)** | **Chebyshev acc** |
| **k-eff** | 1.039104 | 1.039104 | 1.039104 | 1.039104 | 1.039104 |
| **Iteration Number** | 335 | 26 | 45 | 131 | 49 |
| **Dominance**  **Ratio** | 0.980787 | 0.700914 | 0.817564 | 0.943129 | 0.980787 |

지난 번에 풀었던 1차원 단일그룹 문제를 Wielandt shift와 Chebyshev 가속 방법을 적용하여 풀어보았다. 그 결과 shift value를 적절히 설정한 경우 Chebyshev 보다 빠른 수렴속도를 보여주었다. Dominance ratio를 살펴보면, Chebyshev 가속의 경우 변화가 없는 반면 Wielandt shift를 사용하였을 때는 약 0.7 정도로 크게 줄어든 것을 볼 수 있다. 이는 두 가속 방법의 개념이 약간 다르기 때문이다. Wielandt shift는 행렬을 바꾸기 때문에 실제로 풀게 되는 행렬식이 바뀐다. 반면 Chebyshev 가속은 동일한 행렬식을 풀되, 외삽을 수행하여 중간 과정을 생략하는 방식이다. 따라서 Dominance ratio는 Chebyshev 가속여부와 상관없이 동일하다.

|  |  |  |
| --- | --- | --- |
| **parameter** | **Fis\_iteration** | **Chebyshev acc** |
| **k-eff** | 1.134779 | 1.134779 |
| **Iteration Number** | 352 | 60 |
| **Dominance**  **Ratio** | 0.970025 | 0. 970027 |

주어진 문제를 Chebyshev 가속을 이용하여 풀었다. 그 결과 고유치와 dominance ratio는 동일하게 나타났으며 반복횟수는 Chebyshev 가속 사용시 약 6배 줄어든 것을 확인할 수 있었다.

C:\Users\cHyun\AppData\Local\Temp\_TS972F.tmp\_TSBF.tmp\flux.emf

위의 그림은 결과로 나온 분열 선원과 중성자속을 정규화 한 후 노드에 따라 그린 것이다. 왼쪽의 그림을 살펴보면 Fission source iteration의 결과와 Chebyshev 가속 사용 결과가 동일하게 나타나는 것을 확인할 수 있다. 실제로 평균 오차도 약 0.02%로 동일한 값으로 여길 수 있다. 이는 Chebyshev 가속이 고유치나 Dominance ratio 뿐 아니라 고유 벡터도 제대로 구한다는 것을 의미한다. (분열 선원은 주어진 핵분열 단면적과 고유 벡터의 곱으로 표현 된다.) 정상을 기준으로 좌우에 불연속인 부분이 존재하는데 이는 고유 벡터는 연속적으로 존재하지만 매질이 바뀌면서 핵반응 단면적이 불연속적으로 변하였기 때문이다. 실제로 불연속점이 나타난 부분에서는 핵분열 양수 값을 지니고 있다가 0으로 변한다. 급격히 변하는 점을 제외한 전체적인 개형은 우측의 중성자속과 유사함을 확인할 수 있다. 중성자 속(우측 그림)을 살펴보면 역시 비슷한 부분에서 가장 높은 점이 나타나고 있고, 마찬가지로 급격히 증가하는 부분을 확인할 수 있다. 이는 분열 선원과 마찬가지로 매질의 변화 때문에 발생한다. 분열 선원의 불연속점이 핵분열 단면적 때문이었다면 이 경우에는 핵분열 단면적뿐 아니라 흡수 단면적도 영향을 미친다.

|  |  |  |  |  |  |  |  |  |
| --- | --- | --- | --- | --- | --- | --- | --- | --- |
| **parameter** | **Max order of Chebyshev Polynomial** | | | | | | | |
| 5 | 10 | 15 | 20 | 25 | 30 | 35 | 40 |
| **k-eff** | 1.134779 | 1.134779 | 1.134779 | 1.134779 | 1.134779 | 1.134779 | 1.134779 | 1.134779 |
| **Iteration Number** | 73 | 54 | 55 | 60 | 51 | 51 | 51 | 51 |
| **Dominance**  **Ratio** | 0.970026 | 0.969961 | 0.970019 | 0.970027 | 0.970129 | 0.970129 | 0.970129 | 0.970129 |
| **Reset**  **Number** | 20 | 7 | 6 | 9 | 4 | 4 | 4 | 4 |

Chebyshev 다항식의 최고 차수 제한을 바꾸어가며 계산을 수행하였다. 그 결과 허용된 최고 차항이 높을수록 수렴 속도가 빨랐다. 하지만 차수가 너무 큰 경우 최고 차에 도달하기 전에 초기화 조건을 만족하여 초기화가 수행되었기 때문에 별다른 효과를 보여주지 못하였다. Chebyshev 다항식의 차수가 큰 경우 오차가 튀는 현상은 발견되지 않았다.