Report - Assignment # 2: Page 2.

· **Bottom node:**

want to compute: $\dfrac{\bar{J}_{g,1} - \bar{J}_g(0)}{\Delta z}$

$$\bar{J}_{g,1} = -\frac{D_{g,1} \times D_{g,2}}{\Delta z/2} \cdot \frac{\hat{\phi}_{g,2} - \hat{\phi}_{g,1}}{D_{g,1} + D_{g,2}}$$ from equation (A14).

· But $\bar{J}_g(0)$ requires special attention (since there's no node below this):

(Assume Marshak boundary conditions):

$$\begin{cases} \bar{J}_g(0) = -D_{g,1} \cdot \dfrac{\hat{\phi}_{g,1} - \hat{\phi}_g(0)}{\Delta z/2} \\ \bar{J}_g(0) = -\dfrac{\hat{\phi}_g(0)}{2} \end{cases} \Rightarrow \begin{cases} \hat{\phi}_g(0) = \dfrac{\hat{\phi}_{g,1}}{1 + \dfrac{\Delta z}{4D_{g,1}}} \\ \bar{J}_g(0) = -\dfrac{1}{2} \cdot \dfrac{\hat{\phi}_{g,1}}{1 + \dfrac{\Delta z}{4D_{g,1}}} \end{cases}$$

So that:

$$\frac{\bar{J}_{g,1} - \bar{J}_g(0)}{\Delta z} = -\frac{2}{\Delta z^2} \cdot \frac{D_{g,1} D_{g,2}}{D_{g,1} + D_{g,2}} \left(\hat{\phi}_{g,2} - \hat{\phi}_{g,1}\right) + \frac{1}{2\Delta z} \frac{1}{1 + \dfrac{\Delta z}{4D_{g,1}}} \cdot \hat{\phi}_{g,1}$$

$$= \underbrace{\left(+\frac{2}{\Delta z^2} \cdot \frac{D_{g,1} \cdot D_{g,2}}{D_{g,1} + D_{g,2}} + \frac{1}{2\Delta z} \cdot \frac{1}{1 + \dfrac{\Delta z}{4D_{g,1}}}\right)}_{a_{g,1}} \hat{\phi}_{g,1} + \underbrace{-\frac{2}{\Delta z^2} \cdot \frac{D_{g,1} \cdot D_{g,2}}{D_{g,1} + D_{g,2}}}_{b_{g,1}} \hat{\phi}_{g,2}$$

$$\Rightarrow \begin{cases} a_{g,1} = \dfrac{2}{\Delta z^2} \cdot \dfrac{D_{g,1} \cdot D_{g,2}}{D_{g,1} + D_{g,2}} + \dfrac{1}{\Delta z} \cdot \dfrac{1/2}{1 + \dfrac{\Delta z}{4D_{g,1}}} \\ b_{g,1} = -\dfrac{2}{\Delta z^2} \cdot \dfrac{D_{g,1} \cdot D_{g,2}}{D_{g,1} + D_{g,2}} \\ c_{g,1} = 0 \end{cases}$$ → Same that we got in the lecture.

· Now, for the top node:

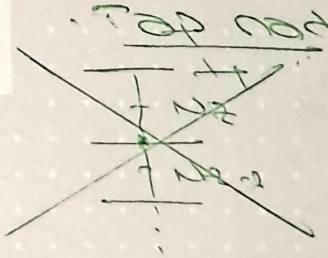
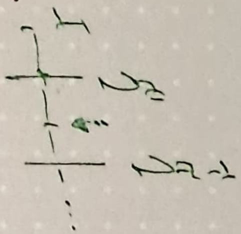
want to compute: $\dfrac{\bar{J}_g(H) - \bar{J}_{g,N_z}}{\Delta z}$.

$$\bar{J}_{g,N_z} = -\frac{D_{g,N_z} \cdot D_{g,N_z-1}}{D_{g,N_z} + D_{g,N_z-1}} \cdot \frac{\hat{\phi}_{g,N_z} - \hat{\phi}_{g,N_z-1}}{\Delta z/2}$$

Check note in the next page.

Boundary conditions:

$$\begin{cases} \bar{J}_g(H) = -D_{g,N_z} \cdot \dfrac{\hat{\phi}_g(H) - \hat{\phi}_{g,N_z}}{\Delta z/2} \\ \bar{J}_g(H) = +\dfrac{\hat{\phi}_g(H)}{2} \end{cases}$$

Top node:  → Preliminaries: 

[illegible]:

$J_{g,Nz}^{-} = -D_{g,Nz}\,\frac{\Phi_{g,Nz} - \Phi_{g,Nz}^{-}}{\Delta z/2}$

$J_{g,Nz}^{-} = -D_{g,Nz-1}\,\frac{\Phi_{g,Nz}^{-} - \Phi_{g,Nz-1}}{\Delta z/2}$

$\Phi_{g,Nz}^{-} = \Phi_{g,Nz-1}^{+}$

$-D_{g,Nz}\cdot\frac{\Phi_{g,Nz} - \Phi_{g,Nz}^{-}}{\Delta z/2} = -D_{g,Nz-1}\cdot\frac{\Phi_{g,Nz}^{-} - \Phi_{g,Nz-1}}{\Delta z/2}$

$\Rightarrow +D_{g,Nz}\cdot\Phi_{g,Nz}^{-} + D_{g,Nz-1}\cdot\Phi_{g,Nz}^{-} = D_{g,Nz-1}\cdot\Phi_{g,Nz-1} + D_{g,Nz}\cdot\Phi_{g,Nz} \Rightarrow$

$\Rightarrow \Phi_{g,Nz}^{-} = \frac{D_{g,Nz-1}\cdot\Phi_{g,Nz-1} + D_{g,Nz}\cdot\Phi_{g,Nz}}{D_{g,Nz} + D_{g,Nz-1}}$

$\Rightarrow J_{g,Nz}^{-} = -\frac{D_{g,Nz}}{\Delta z/2}\left(\Phi_{g,Nz} - \ldots\right) =$

$= -\frac{D_{g,Nz}}{\Delta z/2}\cdot\frac{1}{D_{g,Nz}+D_{g,Nz-1}}\left((D_{g,Nz}+D_{g,Nz-1})\Phi_{g,Nz} - D_{g,Nz-1}\Phi_{g,Nz-1} - D_{g,Nz}\Phi_{g,Nz}\right) =$

$= -\frac{D_{g,Nz}}{\Delta z/2}\cdot\frac{D_{g,Nz-1}}{D_{g,Nz}+D_{g,Nz-1}}\left(\Phi_{g,Nz} - \Phi_{g,Nz-1}\right).$

Want to compute: $\frac{J_g(+) - J_{g,Nz}^{-}}{\Delta z}$.

$J_{g,Nz}^{-} = -\frac{D_{g,Nz}\cdot D_{g,Nz-1}}{D_{g,Nz}+D_{g,Nz-1}}\cdot\frac{\Phi_{g,Nz} - \Phi_{g,Nz-1}}{\Delta z/2}.$

Boundary conditions:

$$\begin{cases} J_g(+) = -D_{g,Nz}\cdot\frac{\Phi_g(+) - \Phi_{g,Nz}}{\Delta z/2} \\ J_g(+) = +\frac{\Phi_g(+)}{2} \end{cases}$$

$\Rightarrow -D_{g,Nz}\cdot\frac{\Phi_g(+) - \Phi_{g,Nz}}{\Delta z/2} = \frac{\Phi_g(+)}{2} \Rightarrow$

$\Rightarrow -\frac{2}{\Delta z}D_{g,Nz}\cdot\Phi_g(+) - \frac{1}{2}\Phi_g(+) = -\frac{2}{\Delta z}D_{g,Nz}\Phi_{g,Nz} \Rightarrow$

$\Rightarrow \left(-\frac{2}{\Delta z}D_{g,Nz} - \frac{1}{2}\right)\Phi_g(+) = -\frac{2}{\Delta z}D_{g,Nz}\cdot\Phi_{g,Nz} \Rightarrow$

Page 3:

$$\Rightarrow \boxed{\bar{\Phi}_g(1) = + \frac{\bar{\Phi}_{g,Nz}}{1 + \frac{\Delta z}{4D_{g,Nz}}}}$$

$$\Rightarrow \boxed{\bar{\Phi}_g(1) = \frac{\bar{\Phi}_g(1)}{2} + \frac{1/2}{1 + \frac{\Delta z}{4D_{g,Nz}}} \cdot \bar{\Phi}_{g,Nz}}$$

Final:

$$\bar{J}_g(1) = \frac{\bar{\Phi}_g(1) - \bar{J}_{g,Nz}}{\Delta z} \qquad (\bar{\Phi}_{g,Nz} - \bar{\Phi}_{g,Nz-1})$$

$$= + \frac{1}{2\Delta z} \cdot \frac{1}{1 + \frac{\Delta z}{4D_{g,Nz}}} \cdot \bar{\Phi}_{g,Nz} + \frac{D_{g,Nz} \cdot D_{g,Nz-1}}{D_{g,Nz} + D_{g,Nz-1}} \cdot \frac{2}{\Delta z^2}$$

$$= \underbrace{\left(+ \frac{1}{2\Delta z} \cdot \frac{1}{1 + \frac{\Delta z}{4D_{g,Nz}}} + \frac{2}{\Delta z^2} \cdot \frac{D_{g,Nz} \cdot D_{g,Nz-1}}{D_{g,Nz} + D_{g,Nz-1}} \right)}_{a_{g,Nz}} \bar{\Phi}_{g,Nz}$$

$$+ \underbrace{\left(- \frac{2}{\Delta z^2} \cdot \frac{D_{g,Nz} \cdot D_{g,Nz-1}}{D_{g,Nz} + D_{g,Nz-1}} \right)}_{c_{g,Nz}} \bar{\Phi}_{g,Nz-1} + \underbrace{(0)}_{b_{g,Nz}} \bar{\Phi}_{g,Nz+1}$$

$$\Rightarrow \begin{cases} c_{g,Nz} = - \frac{2}{\Delta z^2} \cdot \frac{D_{g,Nz} \cdot D_{g,Nz-1}}{D_{g,Nz} + D_{g,Nz-1}} \\ b_{g,Nz} = 0 \\ a_{g,Nz} = \frac{1}{2\Delta z} \cdot \frac{1}{1 + \frac{\Delta z}{4D_{g,Nz}}} + \frac{2}{\Delta z^2} \cdot \frac{D_{g,Nz} \cdot D_{g,Nz-1}}{D_{g,Nz} + D_{g,Nz-1}} \end{cases}$$

Apply.

Wielandt's shift technique: modification to power iteration method.

$A_\omega \cdot \Phi = \lambda \cdot \Phi \Rightarrow$

$\Rightarrow \left(A^{-1} - \frac{1}{k_{est}}\right)\Phi = \left(\frac{1}{k} - \frac{1}{k_{est}}\right)\Phi$ (definitions).

In our case: $\left(M\Phi = \frac{1}{k}F\Phi\right)$.

$A = M^{-1}F$ $\qquad \Rightarrow k\Phi = M^{-1}F\Phi$

$\Rightarrow A^{-1} = F^{-1}M$ $\qquad \Rightarrow k\Phi = A\Phi$

- But, F^{-1} doesn't exist (F has blank of zeros $\Rightarrow$ not invertible!)

So,

$A_\omega \cdot \Phi = \lambda\Phi = \left(\frac{1}{k} - \frac{1}{k_{est}}\right)\Phi \Rightarrow$

$\Rightarrow \left(A^{-1} - \frac{1}{k_{est}}\right)\Phi = \left(\frac{1}{k} - \frac{1}{k_{est}}\right)\Phi \Rightarrow$

$\Rightarrow \left(F^{-1}M - \frac{1}{k_{est}}\right)\Phi = \left(\frac{1}{k} - \frac{1}{k_{est}}\right)\Phi \Rightarrow$ (multiply $\times F$)

$\Rightarrow \left(M - \frac{F}{k_{est}}\right)\Phi = \left(\frac{1}{k} - \frac{1}{k_{est}}\right)F\Phi$

So, we have arrived at a "modified" Wielandt's shift method:

- $\Phi^{(p)} = \frac{1}{\lambda^{(p-1)}}\left(M - \frac{F}{k_{est}}\right)^{-1} F \cdot \Phi^{(p-1)}$; $\qquad A\Phi = S$ ($S = F\cdot\Phi^{(p-1)}$)
- $\lambda^{(p)} = \lambda^{(p-1)} \frac{\Phi^{(p-1)} \cdot \Phi^{(p)}}{\Phi^{(p-1)} \cdot \Phi^{(p-1)}}$; $\rightarrow$ high λ.
- $\frac{1}{\lambda^{(p)}} = \frac{1}{k^{(p)}} - \frac{1}{k_{est}}$;

since $k \neq$ high (λ).

- Wielandt's shift comes in the definition of k_{est}:
  - Fundamental mode: $k_{est} = k^{(p-1)} - k_{shift}$ $\rightarrow$ Simple (Wielandt's) shift.
  - Higher order modes: $k_{est} = k_{est\_mode} - k_{shift}$ $\rightarrow$ Estimation of the "desired" higher order mode.

Step 11: pdf report.

TIF330
Computational Continuum physics

Report: Assignment 2

Date: 2023-05-22

Students:
Afonso Sequeira Azenha

1 Step 11: Compare the different solution strategies

1.1 Plot the spatial distribution of the neutron flux, $\phi_{g,n}$ for all energy groups for the different methods

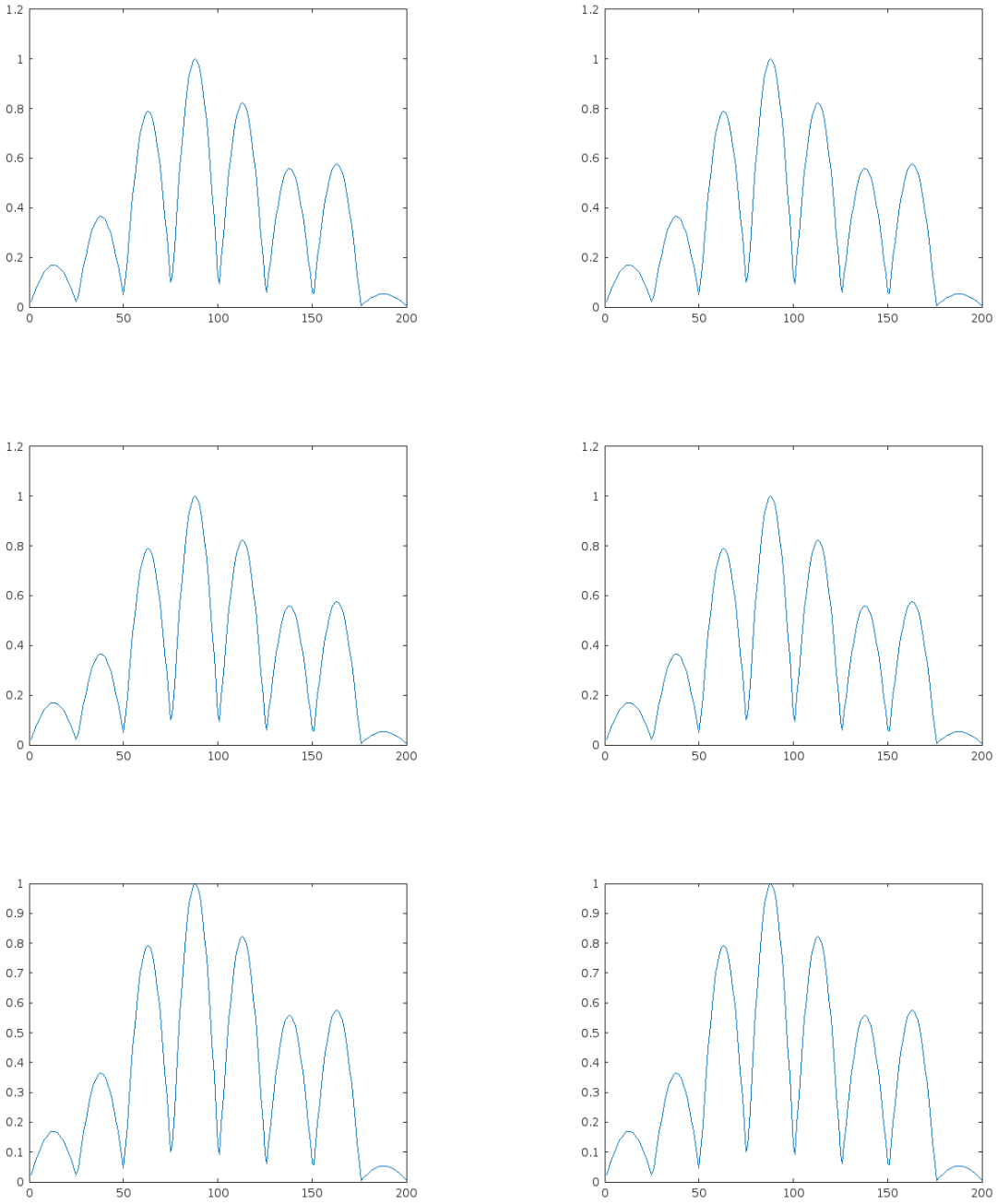


Figure 1: Spatial distribution of the neutron flux, for the 6 different implemented methods. In the order left to right, up to down, we have methods 1, 2, 3, 4, 5 and 6, respectively, exactly like they are defined in MATLAB Grader.

Notes - Input parameters: `solver_nk(0, 100*eps, 1E-5, flag1, flag2, ones(NZ, 1), ones(NZ,1))`. Also, note how `k_est_mode = 0`, which means our code searches for the dominant mode. On the other hand, `k_est_mode = c`, $\forall c \neq 0$ looks for eigenvalues close to c (The plots presented are for `k_est_mode = 0`). As can be seen, there isn't any

noticeable difference: the results appear to be consistent across all the 6 implemented methods.

1.2 Compare the effective multiplication factor, k_{eff}, computed by the different methods

```
keff =

struct with fields:

  met1: 1.127918965406380
  met2: 1.127918965406380
  met3: 1.127918965406379
  met4: 1.127918965406380
  met5: 1.127918965406379
  met6: 1.127918965406367
```

Once again, the results appear to be consistent. Only the last 2 digits sometimes vary ever so slightly between methods.

1.3 Compare the residuals between the different methods

Below, the residuals are presented, in the order: `met1`, `met2`, `met3`, `met4`, `met5`, `met6`.

```
met1: 1.457238798051035e-14
met2: 1.457106025822903e-14
met3: 1.461304304020238e-14
met4: 2.168183648046661e-14
met5: 9.215369235893368e-17
met6: 4.263464898658743e-13
```

This is the first metric that makes it blatantly apparent that method 5 is much more accurate than the rest. This method corresponds to: Power iteration method with acceleration (*Wielandt*'s shift), with LUPQ decomposition.

1.4 Compare the number of iterations between the different methods

```
n_out =

struct with fields:

  met1: 58
  met2: 58
  met3: 58
  met4: 599
  met5: 5
  met6: 40
```

The first three methods show the same number of iterations. This makes perfect sense,

since they all correspond to the same power iteration method (without acceleration), but using different techniques to perform the matrix inversion of `M`. Furthermore, method 4 (Power iteration method without acceleration, using stationary method) appears to be the clear loser, with 599 iterations. This is likely due to an unfortunate decomposition of the M matrix, which results in a slow convergence. Method 5, on the other hand, is noticeably head and shoulders above the rest, whereas method 6 appears to be a little better than methods 1 to 3, but still not as good as 5.

1.5 Compare the CPU times between the different methods

```
t =

struct with fields:

  met1: 0.029258000000000
  met2: 0.077991000000000
  met3: 0.033551000000000
  met4: 0.053623000000000
  met5: 0.006438000000000
  met6: 6.635999000000000
```

The thing that stands out the most here is how method 5 is considerably faster than all the other methods. This is consistent with the fact that method 5 required less (5) iterations to converge. Additionally, `met6` also stands out as significantly slower than all the other techniques. This is due to the nature of the method: In this case, we treat the problem as if it were a non-linear problem and use MATLAB's `fsolve` function. This is clearly less efficient than simply solving the eigenvalue problem using any of the other five methods (as can be seen by the CPU times).

1.6 Method 5: `k_est_mode = 0.8`. Analysis of the results

The obtained results is presented below.

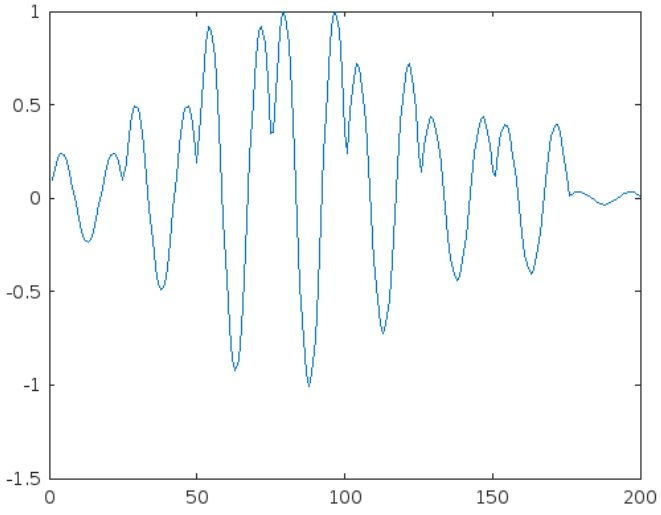


Figure 2: Spatial distribution of the neutron flux, for method 5, using `k_est_mode = 0.8`.

And, k_{eff} value:

```
ans =
    0.7043
```

As can be clearly seen, this mode is distinctly different from the one in Figure 1. As explained previously, using `k_est_mode = c`, $\forall c \neq 0$ makes it so our code looks for eigenvalues close to c. As such, `c = 0.8` results in convergence to $k_{eff} = 0.7043$, which corresponds to the eigenvalue the closest to 0.8. Quick tests show that the other closest eigenvalues would be: 0.9259 (`k_est_mode = 0.85`) and 0.5227 (`k_est_mode = 0.55`).