

1. Week 8

Deze week is besteed aan het doornemen van de reader en doorlezen van overige literatuur

2. Week 9

2.1. Meeting met Kees

2.2. Poisson code in Julia

Om te beginnen met het probleem om een multigrid code te schrijven, moet je eerst een ‘normale’ oplosser voor een PDE schrijven. Dit gaan we doen in de vorm van een directe solver voor de Poisson vergelijking.

Deze staat bekend als de volgende vorm:

$$\frac{\partial^2 u}{\partial^2 x} + \frac{\partial^2 u}{\partial^2 y} = f$$

Door gebruik te maken van de FMD methode, kunnen we deze vergelijking oplossen. De code die we hiervoor gaan schrijven is in Julia. Dit werkt op de volgende manier in 1D:

$$\frac{u_{j+1} - 2u_j + u_{j-1}}{h^2}$$

Of in 2D:

$$\frac{u_{i+1,j} - 2u_{i,j} + u_{i-1,j}}{h^2} + \frac{u_{i,j+1} - 2u_{i,j} + u_{i,j-1}}{h^2}$$

Dit geeft het volgende stencil: $\begin{pmatrix} 0 & 1 & 0 \\ 1 & -4 & 1 \\ 0 & 1 & 0 \end{pmatrix}$

Als de matrix A is opgesteld, kunnen we dit oplossen met behulp van een directe (ingebouwde) solver.

2.2.1. Directe limitaties

Er is geprobeerd om dit te implementeren voor een 2D grid van 201x201 punten. De snelheid was wat traag (ongeveer 2 minuten), maar het grote probleem was de opslag van de punten. Doordat elk punt gemapt is aan elk ander punt, krijg je een matrix A van 201*201 bij 201*201. Dit is een matrix van 40401 bij 40401, wat een groot geheugenverbruik met zich meebrengt. (ongeveer 15Gb...) wat klopt met de opslag van een 64-bit float (komt uit op ongeveer 75 bits). Dit moet beter!