Utrecht Universiteit

Programmeeropdracht

De verglijking $-\Delta u + u = f$ oplossen met *finite element methods*

Student:
Koen de Boer
s0828467, RU

4 juni 2014

Inhoudsopgave

1 Inleiding

Dit verslag gaat over de implementatie en daadwerkelijke toepassing van de *finite element method* die bepaalde stationaire partiële differentiaalvergelijkingen kan benaderen. In deze paper zal eigenlijk alleen worden ingegaan op één specifieke vergelijking, waarvan de *finite element*-benaderingen worden getoetst aan de hand van de exacte oplossingen.

2 Methodiek

2.1 De vergelijking

De volgende stationaire partiële differentiaalvergelijking is bestudeerd:

$$\begin{cases} -\Delta u + u &= f & \text{op } [0,1]\times[0,1] \\ u &= 0 & \text{op } \partial([0,1]^2) \end{cases}$$

Hier is $f : [0,1]^2 \to \mathbb{R}$ een gegeven reëelwaardige functie, en u is de gezochte functie (op hetzelfde domein en codomein). In het vervolg zullen we $[0,1]^2$ ook wel aanduiden met Ω.

Hoewel de finite-elements method voor veel verschillende f voor bovenstaande vergelijking de oplossing kan benaderen, is het natuurlijk handig om een $f : \Omega \to \mathbb{R}$ te kiezen waarvan de oplossing u al bekend is. Namelijk, om de fout te onderzoeken heb je een exacte oplossing nodig om mee te vergelijken. Met deze gedachte is de volgende lijst opgesteld:

f	Matlabroutine f	u
$\sin(\pi x)\sin(\pi y)$	`f_sinsin.m`	$\frac{\sin(\pi x)\sin(\pi y)}{1+2\pi^2}$
$-2\big(y(y-1)+x(x-1)\big)$ $+xy(x-1)(y-1)$	`f_pol.m`	$xy(x-1)(y-1)$
$e^x y(1-y) - 2e^x + 2(e-1)x + 2$ $+(-e^x + (e-1)x + 1)y(1-y)$	`f_exp.m`	$(-e^x + (e-1)x + 1)y(1-y)$
f_n	`f_not.m`	u_n
f_j	`f_jump.m`	u_j

De beschrijvingen van f_n, u_n,f_j en u_j zijn te lang voor deze tabel, en volgen dan ook hieronder:

$$u_n(x,y) = \begin{cases} \left((x-1/2)^3 - \left(\frac{x-1/2}{2}\right)^{3/2}\right) y(1-y) & \text{als } x < 1/2 \\ 0 & \text{als } x = 1/2 \\ -\left((1/2-x)^3 - \left(\frac{1/2-x}{2}\right)^{3/2}\right) y(1-y) & \text{als } x > 1/2 \end{cases}$$

$$\frac{\partial^2}{\partial x^2}u_n(x,y) = \begin{cases} (6(x-1/2) - \frac{3}{4\cdot 2^{3/2}\sqrt{x-1/2}})y(1-y) & \text{als } x \leq 1/2 \\ 0 & \text{als } x = 1/2 \\ -(6(1/2-x) - \frac{3}{4\cdot 2^{3/2}\sqrt{1/2-x}})y(1-y) & \text{als } x > 1/2 \end{cases}$$

$$\frac{\partial^2}{\partial x^2}u_n(x,y) = \begin{cases} -2\left((x-1/2)^3 - \left(\frac{x-1/2}{2}\right)^{3/2}\right) & \text{als } x \leq 1/2 \\ 0 & \text{als } x = 1/2 \\ 2\left((1/2-x)^3 - \left(\frac{1/2-x}{2}\right)^{3/2}\right) & \text{als } x > 1/2 \end{cases}$$

$$f_n = u_n - \frac{\partial^2}{\partial x^2}u_n - \frac{\partial^2}{\partial y^2}u_n$$

$$u_j(x,y) = \begin{cases} x^2y(1-y) & \text{als } x \leq 1/2 \\ -(1-x)^2y(1-y) & \text{als } x > 1/2 \end{cases}$$

$$\frac{\partial^2}{\partial x^2}u_j(x,y) = \begin{cases} 2y(1-y) & \text{als } x \leq 1/2 \\ -2y(1-y) & \text{als } x > 1/2 \end{cases}$$

$$\frac{\partial^2}{\partial x^2}u_j(x,y) = \begin{cases} -2x^2 & \text{als } x \leq 1/2 \\ 2(1-x)^2 & \text{als } x > 1/2 \end{cases}$$

$$f_j = u_j - \frac{\partial^2}{\partial x^2}u_j - \frac{\partial^2}{\partial y^2}u_j$$

Merk op dat de eerste drie beschreven functies wel allemaal in $H^2(\Omega)$ liggen, terwijl de laatste twee dat niet doen. We zullen later zien wat voor invloed dit zal hebben op hun finite element-benaderingen. Overigens – de matlab-routines van de oplossingen u hebben dezelfde naam als die van de functies f, alleen dan eindigen ze met `_sol` (van *solution*). Dus de oplossing u van `f_sinsin.m` is `f_sinsin_sol.m`.

2.1.1 u_j is geen $H^2(\Omega)$-functie

Het bewijs dat u_j niet in $H^2(\Omega)$ ligt, vergt wat uitleg. Deze functie is discontinu. We mogen Morrey's Theorem[1] niet direct toepassen; deze stelling zegt dat $H_2(\Omega)$-functies in $C^0(\Omega)$ liggen, mits Ω een C^1-rand heeft. Aangezien onze $\Omega = [0,1]^2$ dat niet heeft, breiden we Ω uit tot een cirkel $\tilde{\Omega}$ om $[0,1]^2$ heen. De functie u_j kunnen we buiten $[0,1]^2$ als 0 definiëren.

Dit heeft als gevolg dat – als u_j een $H^2(\Omega)$-functie is – $\tilde{u_j}$ ook een $H^2(\tilde{\Omega})$-functie is. Namelijk: op Ω blijven de zwakke eerste- en tweede afgeleides

[1] Zie pagina 280, Partial Differential Equations van Lawrence C. Evans

hetzelfde, en buiten Ω zijn die identiek nul. Als we dan nu Morrey's Inequality toepassen, levert dat een tegenspraak op, aangezien $\tilde{u_j}$ niet continu is.

Merk dus op dat de discontinuïteiten van functies in $H^2(\Omega)$ dus alleen mogen voorkomen op de plekken waarop de rand een 'knik' heeft – en in ons geval komen er ook op inwendige punten discontinuïteiten voor. Conclusie: u_j is niet $H^2(\Omega)$.

2.2 De variationele formulering

De variationele formulering van een partiële differentiaalvergelijking vertaalt deze naar een vergelijking met integralen, waarvan de linkerzijde op te vatten valt als een bilineaire continue vorm, en de rechterzijde als een functionaal:

$$\int_\Omega \nabla u \cdot \nabla v + \int_\Omega uv = \int_\Omega fv \qquad \forall v \in H_0^1(\Omega)$$

Deze variationele formulering correspondeert[2] met de differentiaaloperator $-\Delta + \mathrm{Id}$ en de linkerzijde is[3] een coërcieve begrensde symmetrische bilineaire vorm.

Met Lax-Milgram[4] heeft deze vergelijking een unieke oplossing in de *finite element* functieruimte.

2.3 Triangularizatie van het domein

De triangularizatie van het domein geschiedt als volgt: Allereerst wordt er een verdelingsgrootte m gegeven, deze zal in dit verslag altijd een macht zijn van twee. Vervolgens wordt het domein in m^2 vierkanten verdeeld van gelijke grootte. Tenslotte wordt in elk vierkant de diagonaal van linksonder naar rechtsboven getrokken. Het resultaat is een prachtige vorm-regulaire triangularizatie van het domein $\Omega = [0,1]^2$. Een voorbeeld met $m = 4$ vindt u in figuur 1.

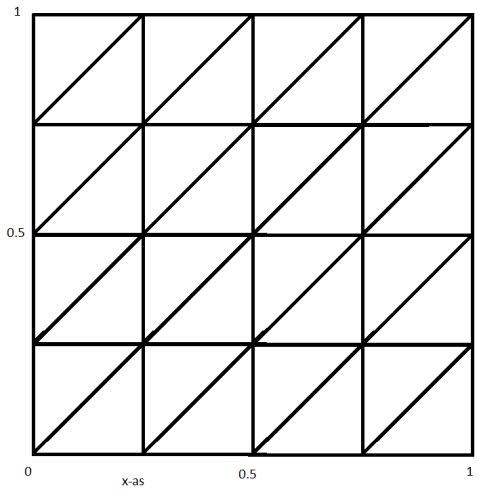
Op elke driehoek zullen we het Lagrange-element toepassen, wat als gevolg heeft dat onze *finite element space* bestaat uit continue functies op $[0,1]^2$ die lineair zijn op elke driehoek van de triangularizatie τ.

De nodale punten zijn natuurlijk de $(m+1)^2$ 'hoekpunten' van de m^2 genoemde vierkanten. De duale basis van de nodale basis (de functionalen 'valuatie in de hoekpunten') bestaat precies uit de (op de driehoeken) stuksgewijs lineaire functies die slechts op één punt de waarde 1 aannemen, en op

[2]Suzanne C. Brenner & L. Ridgway Scott, *The Mathematical Theory of Finite Element Methods*, blz 65

[3]Theorem 2.9.2, Brenner & Scott

[4]Theorem 2.7.7, Brenner & Scott



Figuur 1: De genoemde triangularizatie van het eenheidsinterval met verdelingsgrootte $m = 4$

alle andere hoekpunten 0 zijn. Dit zijn functies die alleen op een klein zeshoekig gedeelte van het domein niet nul zijn, en daar een soort tent-achtige vorm aannemen. In de implementatie wordt deze functie `afflinhill.m` genoemd, een afkorting voor *affine-translated linear hill*. Dit omdat de 'tentfunctie' wordt gedefinieerd in `linearhill.m` en met `afflinhill.m` kan hij 'verschoven' worden.

2.4 Matrixformulering van de variationele formulering

Vanaf nu beschouwen we dus alleen nog functies u_f in de *finite element space* V_f. Doordat we een basis aangewezen hebben, kunnen we u_f dus uitschrijven in basiselementen:

$$u_f = \sum_i \lambda_i v_i$$

De variationele formulering komt dan overeen met (voor alle $v \in V_f$):

$$\sum_i \lambda_i \left(\int_\Omega v v_i + \int_\Omega \nabla v \cdot \nabla v_i \right) = \int_\Omega f v$$

En omdat we elke $v \in V_f$ kunnen uitdrukken in basiselementen, is het dus genoeg als bovenstaande vergelijking klopt voor alle basiselementen van V_f:

$$\sum_i \lambda_i \left(\int_\Omega v_j v_i + \int_\Omega \nabla v_j \cdot \nabla v_i \right) = \int_\Omega f v_j \qquad \forall j$$

Definieer je nu de matrix A door $(A)_{ij} = \int_\Omega v_j v_i + \int_\Omega \nabla v_j \cdot \nabla v_i$, de vector x door: $x_i = \lambda_i$ en de vector y door: $y_j = \int_\Omega f v_j$, dan laat dit zich precies schrijven als een lineaire vergelijking:

$$Ax = y \tag{1}$$

In het *finite element*-jargon wordt A vaak de *stiffness-matrix* genoemd. Moge het duidelijk zijn dat we $x$de 'gevraagde onbekende' is, en dat A en y in principe te berekenen zijn. Over dit berekenen gaan de volgende paragrafen.

2.5 *The stiffness-matrix*

De *stiffnes-matrix* is in principe gewoon te berekenen door middel van exacte integralen: $(A)_{ij} = \int_\Omega v_j v_i + \int_\Omega \nabla v_j \cdot \nabla v_i$. Merk op dat de basisfuncties stuksgewijs lineair zijn op de driehoeken en slechts niet-nul zijn op een kleine zeshoekige deelverzameling van Ω. Met deze gedachte kunnen we onmiddelijk vaststellen dat A hoogstens $O(\mathrm{rk}A)$ elementen heeft, en daarmee sparse is. Verder is hij duidelijk symmetrisch en ook positief definiet, aangezien $x^T Ax = \int_\Omega x_i^2 v_i v_i > 0$ (want v_i heeft een niet-verwaarloosbare verzameling als drager).

Om $(A)_{ij}$ te berekenen, gaan we eerst over op de standaard-driehoek (met lineaire functies). In tabel 1 staat symbolisch weergeven welke twee functies met elkaar worden vermenigvuldigd en worden geïntegreerd: een wit hoekpunt komt overeen met een waarde gelijk aan 1 in dat hoekpunt, en een zwart hoekpunt met een waarde gelijk aan 0. Het symbool $\int_\Delta$ is een afkorting voor: $\int_{x=0}^{x=1} \int_{y=0}^{y=x}$.

Om nu over te gaan op 'kleinere' driehoeken van breedte $h = 1/m$, passen we de substitutieregel toe; we gebruiken als lineaire transformatie:

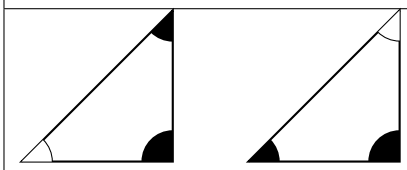
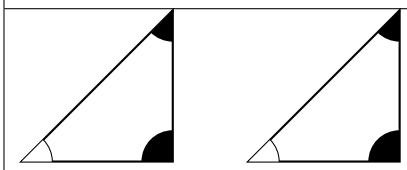
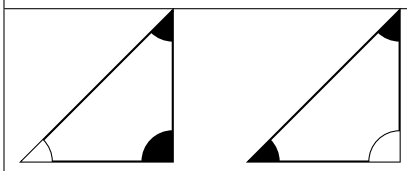
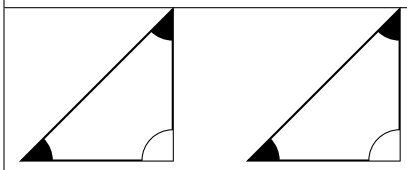
$$\phi : T_h \to T_{std}, (x, y) \mapsto (x/h, y/h)$$

Hierbij wordt met T_{std} de 'standaarddriehoek' bedoeld waar we al eerder mee hebben gewerkt, en met T_h de 'verkleinde standaarddriehoek' met een factor h. Dat is, met hoekpunten: $(0, 0), (h, 0), (0, h)$. Dan laten de integralen van tabel 1 zich naar deze kleinere driehoeken vertalen, en wel als volgt:

$$\int_{T_h} (v \circ \phi)(w \circ \phi) + \int_{T_h} \nabla(v \circ \phi) \cdot \nabla(w \circ \phi)$$

$$= \int_{T_{std}} vw |\det \mathbb{J}(\phi^{-1})| + h^{-2} \int_{T_{std}} \nabla v \cdot \nabla w |\det \mathbb{J}(\phi^{-1})|$$

Hierbij is in beide integralen de substitutieregel toegepast, en op de rechterintegraal ook de kettingregel voor afgeleides. Hiervandaan komt de h^{-2}-term.

Driehoekscombinatie	$\int_\Delta vw$	$\int_\Delta \nabla v \nabla w$
	$\int_\Delta (1-x)y = 1/24$	$\int_\Delta [-1,0]\cdot[0,1] = 0$
	$\int_\Delta (1-x)^2 = 1/12$	$\int_\Delta [-1,0]\cdot[-1,0] = 1/2$
	$\int_\Delta (1-x)(x-y) = 1/24$	$\int_\Delta [-1,0]\cdot[1,-1] = -1/2$
	$\int_\Delta (x-y)^2 = 1/12$	$\int_\Delta [1,-1]\cdot[1,-1] = 1$

Tabel 1: De integralen van lineaire basiselementen op de standaarddriehoeken

Ten slotte merken we op dat de jacobiaan van ϕ^{-1} gelijk is aan: $\begin{pmatrix} h & 0 \\ 0 & h \end{pmatrix}$, en de determinant dus overeenkomt met h^2. Dit resulteert in het volgende:

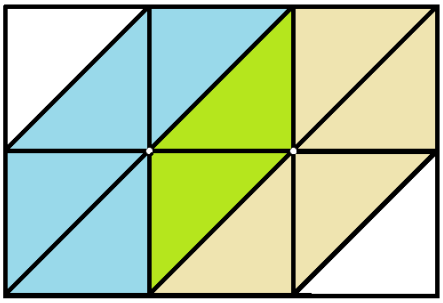
$$\int_{T_h} (v \circ \phi)(w \circ \phi) + \int_{T_h} \nabla(v \circ \phi) \cdot \nabla(w \circ \phi) = h^2 \int_\Delta vw + \int_\Delta \nabla v \cdot \nabla w \quad (2)$$

Omdat er niet al teveel ingegaan moet worden op de berekeningen die te pas komen aan het construeren van de stiffnessmatrix, nemen we één voorbeeld, en hopen dan dat de lezer overtuigd is dat op deze manier de hele stiffnessmatrix gemaakt kan worden. Het voorbeeld is geïllustreerd in figuur 2. De interpretatie van dit figuur is het product van twee basisfuncties, die respectievelijk 1 zijn op de linker en rechter witte stip. Op alle andere punten zijn zij nul. Op het groen-geärceerde gebied is het product van deze functies niet nul, en daarop passen we dan tabel 1 en vergelijking (2) toe.

Nu valt op dat we twee keer in de situatie van rij 3 in tabel 1 vallen; hieruit volgt dat de integraal op het groene gebied gelijk moet zijn aan:

$$h^2/24 - 1/2 + h^2/24 - 1/2 = h^2/12 - 1$$

Dus is te concluderen dat de entries $(A)_{i,i+1}$ gelijk moeten zijn aan $h^2/12 - 1$



Figuur 2: Voorbeeld van een entry van de stiffnessmatrix

(behalve voor de 'randgevallen'). Als u precies wilt weten hoe de stiffnessmatrix eruit ziet voor deze PDE, dan kunt u kijken in de subroutine `stiffness.m`. Deze maakt bij een gegeven verdelingsgrootte m een $(m+1) \times (m+1)$-stiffnessmatrix.

2.6 Integratie van fv

Om fv_i te integreren (en ook om de L_2-norm te berekenen) gebruiken we een tweedimensionale derde-orde kwadratuurmethode. Dit omdat de fout van de integratie dan assymptotisch verwaarloosbaar is en daarmee een verwaarloosbare invloed heeft op de oplossing van het systeem $Ax = y$. De kwadratuurmethode verdeelt het integratiegebied weer – op dezelfde manier als voorheen – in driehoekjes en past dan een Gaussische methode toe op elk driehoekje. Voor een functie f op de eenheidsdriehoek (met punten $(0,0),(1,0),(0,1)$) komt de kwadratuurmethode overeen met de volgende formule[5]:

$$I = -\frac{27}{48} f\left(\frac{1}{3}, \frac{1}{3}\right) + \frac{25}{48} f\left(\frac{1}{5}, \frac{3}{5}\right) + \frac{25}{48} f\left(\frac{1}{5}, \frac{1}{5}\right) + \frac{25}{48} f\left(\frac{3}{5}, \frac{1}{5}\right)$$

Dit wordt uitgevoerd door de Matlabsubroutine `trianquadrature.m`. Maar om op elke willekeurige driehoek deze kwadratuur toe te passen, is er een affiene transformatie nodig, in combinatie met de substitutieregel voor integralen. Dit wordt gedaan door de subroutine `gentrianquadrature.m`.

De hele vector y wordt uiteindelijk geconstrueerd door het programma'tje `make_integrand_vector.m`; die gegeven een verdelingsgrootte m en een functie f op Ω de vector y benadert; dat zijn alle $(m+1)^2$ waarden $\int_\Omega f v_i$.

[5]Deze formule is ontleend uit een dictaat van Shaodeng: `http://math2.uncc.edu/~shaodeng/TEACHING/math5172/Lectures/Lect_15.PDF`, pagina 9

Aangezien de te integreren functie alleen niet nul is op het kleine zeshoekige gebiedje waar het basiselement v_i niet nul is, wordt daar tijdens het integreren rekening mee gehouden.

2.7 Toevoegen van randvoorwaarden

Toevoegen van randvoorwaarden in dit geval komt er op neer dat de basisfuncties v_i die de duale zijn van de valuatie in de randpunten, niet voorkomen in de basisvectorontbinding van de oplossing. Dit wil zeggen dat de vector x uit vergelijking (1) op de plekken die overeenkomen met de 'randbasiselementen', gelijk moet zijn aan nul.

Dit hebben we in de implementatie als volgt toegevoegd: we passen de matrix A en de integratievector y zodanig aan, dat de oplossing x van de vergelijking $Ax = y$ op de gewenste plaatsen – die we vanaf nu de randindices zullen noemen – een 0 heeft staan.

Dit wordt bewerkstelligd door voor alle i die rand-indices zijn, de i-de rij van de matrix A te vervangen door e_i, de standaardbasisvector die op plek i een 1 heeft staan en elders allemaal nullen. Tevens voor dezelfde i, vervangen we y_i door 0. Dit heeft als gevolg dat $x_i = 0$ voor alle randindices i, wil x voldoen aan de vergelijking $Ax = y$.

In de implementatie worden de randindices verkregen door de routine `getIndices.m`. De routines `add_boundary.m` en `add_boundaryvect.m` zorgen voor de juiste aanpassing van respectievelijk A en y.

Het klinkt misschien omslachtig dat wél de hele stiffnessmatrix en de hele y berekend wordt, om er vervolgens op $4m$ plaatsen nullen neer te gaan zetten. Maar op deze manier kunnen er eenvoudig andere randvoorwaarden geïmplementeerd worden. Bovendien valt assymptotisch het werk dat 'voor niets wordt gedaan' in het niet bij al de overige arbeid die de computer moet doen tijdens het berekenen van de stiffnessmatrix en de integratievector.

2.8 De L_2-norm en H_1-seminorm

De L_2-norm wordt eigenlijk precies hetzelfde berekend als de integratie van fv in paragraaf 2.6. Weer wordt het domein van de functie onderverdeeld in evengrote vierkanten en elk vierkant wordt door de rechtsbovendiagonaal in tweeën gedeeld. Merk op dat de L_2-norm wordt gebruikt voor de errorfunctie $E := u_f - u$, die veelal erge 'hobbels' vertoont. Daarom wordt in onze implementatie een grotere verdelingsgrootte genomen, om de fouteneffecten door 'hobbeligheid' van de errorfunctie te verkleinen. Dit heeft wel invloed op de snelheid – het berekenen van de L_2-norm en H_1-seminorm duurt dan ook meestal lang.

De H_1-seminorm wordt eigenlijk precies hetzelfde berekend, alleen wordt niet de functie zelf geïntegreerd, maar een benadering van zijn x-afgeleide en y-afgeleide. Deze benaderingen worden als volgt berekend:

$$\frac{\partial f}{\partial x}(x_0, y_0) \approx \begin{cases} \frac{f(x_0+h,y_0)-f(x_0-h,y_0)}{2h} & \text{als } x_0+h, x_0-h \in \Omega \\ \frac{f(x_0+h,y_0)-f(x_0,y_0)}{h} & \text{als } x_0+h \in \Omega \text{ en } x_0-h \notin \Omega \\ \frac{f(x_0,y_0)-f(x_0-h,y_0)}{h} & \text{als } x_0-h \in \Omega \text{ en } x_0+h \notin \Omega \\ 0 & \text{anders} \end{cases}$$

En een soortgelijke benadering voor de y-partiële afgeleide. Deze benaderingen worden berekend op de punten waar de kwadratuur (`gentrianquadrature.m`) ze nodig heeft. Ook hier kiezen we een grotere verdelingsgrootte.

2.9 De *a posteriori error estimator*

De errorestimator op een driehoekje: $\eta(v, T)$, is als volgt gedefinieerd:

$$\eta(v,T)^2 := h_T^2 \|f + \Delta v - v\|^2_{L_2(T)} + h_T \|[\nabla v]\|_{L_2(\partial T \setminus \partial\Omega)}$$

Met $[\nabla v] := \nabla v|_{T^+} \cdot n_+ + \nabla v|_{T^-} \cdot n_-$, de som van de normaalafgeleiden aan de ene en aan de andere kant van een zijde van de driehoek T. Aangezien $\Delta v = 0$, omdat v stuksgewijs continu is, en de normaalafgeleiden continu zijn op de randen van T, wordt de definitie van $\eta(v,T)^2$ sterk vereenvoudigd:

$$\eta(v,T)^2 := h_T^2 \|f - v\|^2_{L_2(T)} + h_T \sum_{e \in \partial T \setminus \partial\Omega} |e| (\nabla v|_{T^+} \cdot n_+ + \nabla v|_{T^-} \cdot n_-)^2$$

We merken op dat h_T constant blijft over de hele triangularizatie, aangezien elk driehoekje evengroot is. Verder merken we ook op:

$$\sum_{T \in \tau} \|f - v\|^2_{L_2(T)} = \|f - v\|^2_{L_2(\Omega)}$$

Dus:

$$\eta(v,\Omega)^2 := \sum_{T\in\tau} \eta(v,T)^2 = h_T^2 \|f-v\|^2_{L_2(\Omega)} + h_T \sum_{T\in\tau} \sum_{e\in\partial T\setminus\partial\Omega} |e| (\nabla v|_{T^+} \cdot n_+ + \nabla v|_{T^-} \cdot n_-)^2$$

Het linkerdeel van deze som, de L_2-norm, is eenvoudig te berekenen aangezien we al een subroutine hebben voor de L_2-norm. Het rechtergedeelte is echter een beetje detailwerk. De afgeleides van v worden weer berekend aan de hand van de benadering $\frac{v(x+h,y)-v(x,y)}{h}$ etcetera (wat in dit geval geen benadering is, aangezien v lineair is). Verder merken we op dat er maar 3 verschillende normaalvectoren (en hun negatieven) voorkomen in ons geval: $(1,0), (0,1)$ en $(1,1)$. Zo blijven er maar zeer weinig verschillende mogelijkheden over.

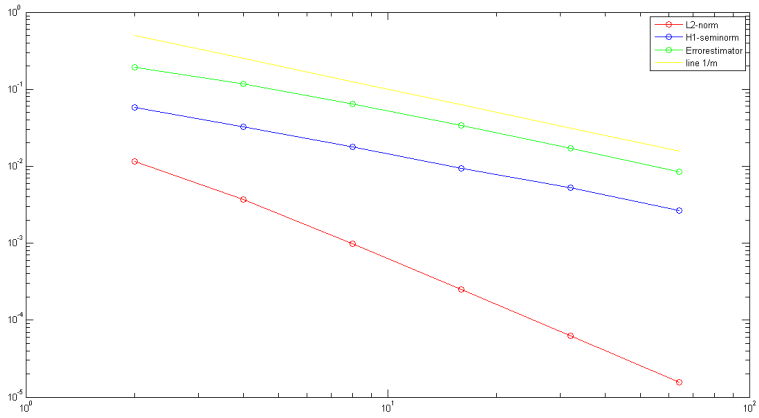
Dit gecompliceerde ding wordt berekend in `error_estimator.m`.

3 Resultaten

De resultaten zijn weergeven in een log-log-plot. Hierin worden de L_2-norm van de error $|u_f - u|$, de H_1-seminorm van de error $|u_f - u|$ en de errorestimator van u_f weergeven. Ook is er ter oriëntatie een gele $1/m$-lijn aangebracht op de plot; voor de $H^2(\Omega)$-functies u zou de error $|u_f - u|$ hoogstens $1/m$ in de H_1-norm moeten zijn.

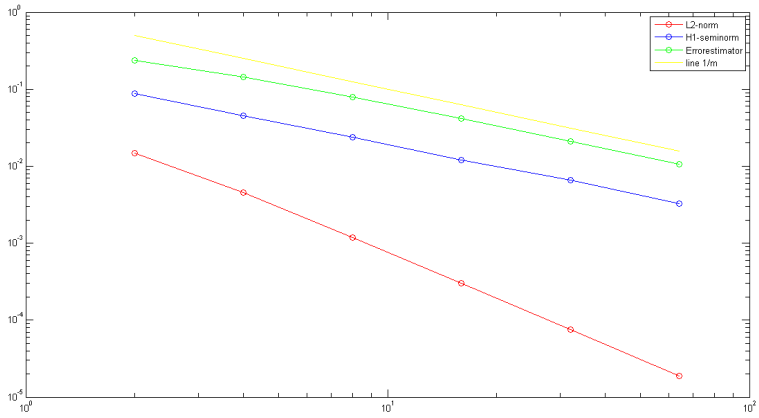
De plots hebben op de $x-as$ de log van de verdelingsgrootte, die waarden $2, 4, 8, 16, 32$ en 64 aanneemt. De bovengrens 64 is uit praktische overwegingen – de hele plot berekenen duurt dan ongeveer 10 minuten. Ter informatie – als de verdelingsgrootte twee keer zo groot wordt, duurt het berekenen vier keer langer.

3.1 `f_sinsin.m`



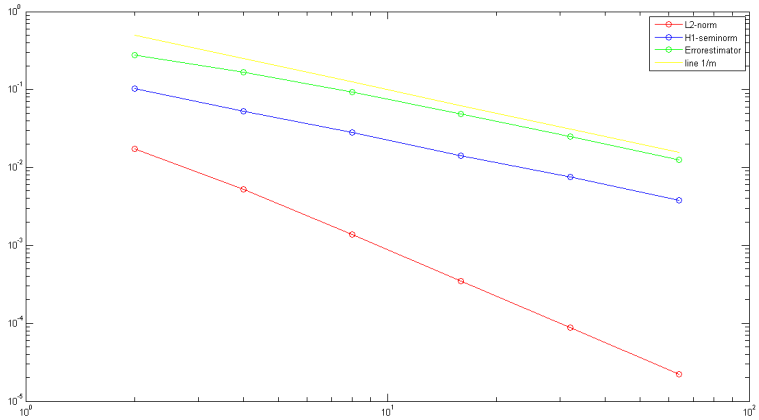
Figuur 3: Plot van verschillende (semi)-normen van de fout van de *finite element*-benadering van `f_sinsin.m`

3.2 `f_exp.m`



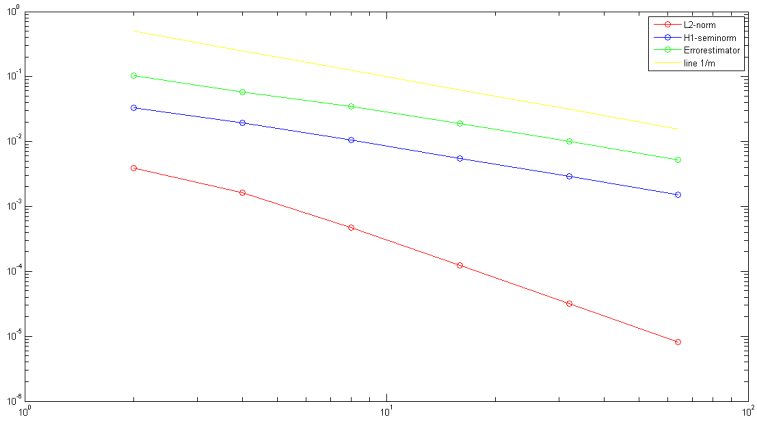
Figuur 4: Plot van verschillende (semi)-normen van de fout van de *finite element*-benadering van `f_exp.m`

3.3 `f_pol.m`



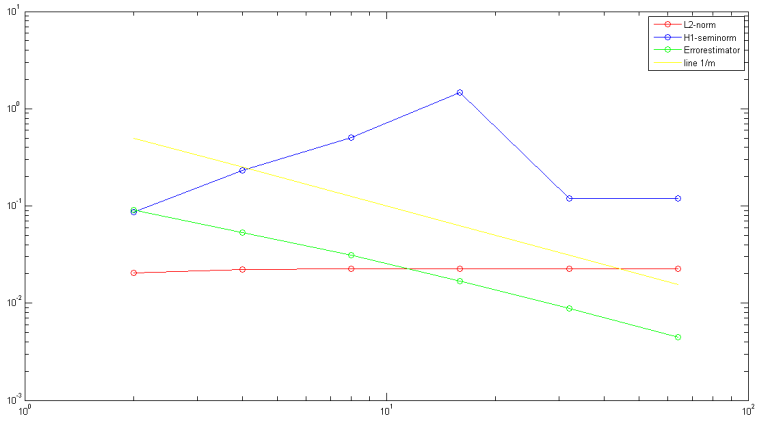
Figuur 5: Plot van verschillende (semi)-normen van de fout van de *finite element*-benadering van `f_pol.m`

3.4 `f_not.m`



Figuur 6: Plot van verschillende (semi)-normen van de fout van de *finite element*-benadering van `f_not.m`

3.5 `f_jump.m`

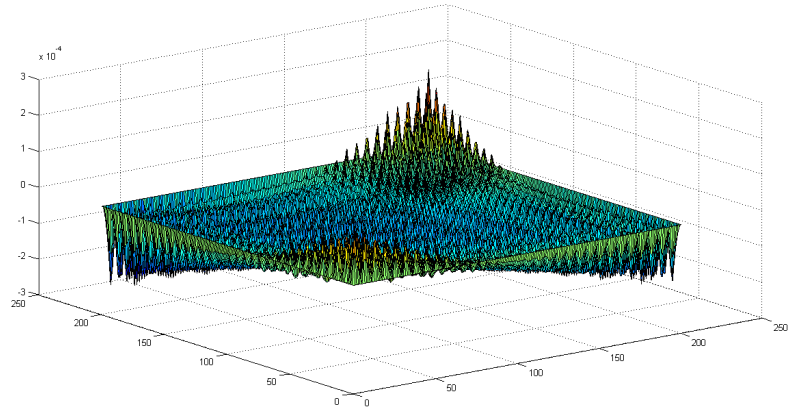


Figuur 7: Plot van verschillende (semi)-normen van de fout van de *finite element*-benadering van `f_jump.m`

3.6 Plot van de error van `f_exp.m`

Om u een idee te geven hoe de plot van de errorfunctie van een 'normale' $H^2(\Omega)$-functie eruitziet, geven we u hier een voorbeeld. Het valt op dat op

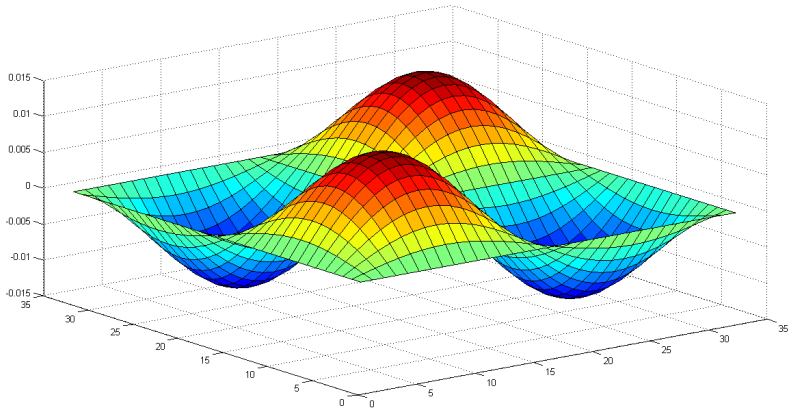
het rooster van de valuatiepunten de error het minst groot is, en daartussen 'bobbels' zijn waarin de error groter wordt. Dit gedrag is bij alle errorplots waargenomen.



Figuur 8: Plot van de error met verdelingsgrootte 32 van `f_exp.m`

3.7 Plot van de benadering van `f_sin2sin2.m`

Ik wilde eigenlijk ook graag gewoon nog een mooi plotje laten zien van de finite-element benadering van de vergelijking voor $\sin(2\pi x)\sin(2\pi y)$.



Figuur 9: Plot van de finite element benadering met verdelingsgrootte 32 van `f_sin2sin2.m`

4 Discussie

4.1 Daling van de L_2-norm en H_1-seminorm

4.1.1 Daling van de H_1-seminorm

In alle bovenstaande gevallen is de H_1-seminorm significant hoger dan de L_2-norm; en daarmee wordt de H_1-norm voornamelijk bepaald door de seminorm. We merken op dat in figuur 3, 4 en 5 het verwachtte patroon van afname van de H_1-norm inderdaad waar te nemen is. Met 'verwacht patroon' doelen we op Theorem 3.4 van Rob Stevensons *Some notes with numerical methods for stationary PDEs*. Deze stelt dat – onder bepaalde voorwaarden, waaraan de besproken partiële vergelijking voldoet – het volgende waar is, voor $k = 1$:

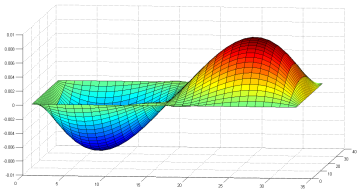
$$\|u - u_f\|_{H_1(\Omega)} \lesssim h^k |u|_{H_{k+1}(\Omega)}$$

Er geldt hier dat $k = 1$, omdat k de graad is van de polynomen die door de interpolatiefunctie 'benaderen met de finite-element method' behouden blijven. Deze graad is in ons geval gelijk aan 1, omdat we met lineaire Lagrange-elementen werken.

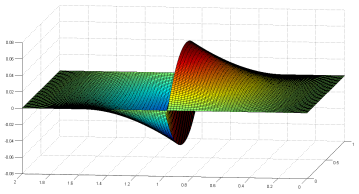
Aangezien $h \sim 1/m$, en in de plotjes van figuur 3, 4 en 5 de blauwe lijn zich min of meer evenwijdig lijkt te bewegen aan de lijn $1/m$, zouden we kunnen stellen dat dit inderdaad overeenkomt met de theoretische voorspellingen.

Het vreemde is echter dat wat plaatsvindt in figuur 6; de oplossing u ligt namelijk niet in $H_2(\Omega)$, en dus is bovengenoemde stelling niet van toepassing. Het feit dat u niet in $H_2(\Omega)$ ligt, is eenvoudig te zien aan het feit dat $\frac{1}{\sqrt{x}}$ voorkomt in de functie $\frac{\partial^2}{\partial x^2}u$ – dat is niet kwadraat-integreerbaar.

Ik heb geen verklaring kunnen vinden voor het fenomeen wat hier plaatsvindt – waarschijnlijk werkt deze *finite element*-method soms ook voor functies die niet in $H^2(\Omega)$ liggen.
Natuurlijk bespreken we ook even wat er gebeurt in figuur 7. De functie u_j ligt niet in $H^2(\Omega)$. Zowel de L_2-norm als de H_2-norm daalt niet bij deze functie. Dit ligt waarschijnlijk aan het feit dat deze functie geen zwakke (dubbele) afgeleide heeft op Ω; de formulering van de PDE met de f_j heeft niet u_j als oplossing, omdat deze functie geen tweede afgeleide heeft. Dit heeft als gevolg dat de finite-element benadering een oplossing probeert te geven die wel een tweede zwakke afgeleide heeft. Dit kun je zien in figuur 10.



(a) De finite element benadering van $-\Delta u + u = f_j$



(b) De 'echte oplossing' u_j

Figuur 10: Het verschil tussen de benadering (links) en de gegeven u_j (rechts)

4.1.2 Daling van de L_2-norm

Voordat we ingaan op de resultaten, zullen we eerst een theoretisch resultaat tonen, de zogenaamde 'Aubin-Nitsche dualiteits-truc'. Deze wordt onder andere beschreven in Rob Stevensons *Some notes with numerical methods for stationary PDEs*, theorem 3.6 en in *FEM methods* van D.Y. Kwak [6], bladzijde 13. In ons geval (zie example 3.7 in Stevensons notes), omdat Ω convex is, wordt er iniedergeval voldaan aan de regulariteitseis van het adjuncte probleem (in geval 1 t/m 3 van onze functies): $\|u\|_{H^2(\Omega)} \lesssim \|f\|_{L^2(\Omega)}$. Bovendien weten we vanwege theorem 3.4 (Stevenson) dat, voor alle $w \in H^2(\Omega) \cap H_0^1(\Omega)$:

$$\inf_{v_h \in V_{\tau_h}} \|w - v_h\|_{H^1(\Omega)} \lesssim h\|w\|_{H^2(\Omega)}$$

Want hier kunnen we voor v_h dan gewoon de finite element benadering van $f = w - \Delta w$ nemen. Nu mogen we met Aubin-Nitsche concluderen dat:

$$\|u - u_h\|_{L_2(\Omega)} \lesssim h\|u - u_h\|_{H^1(\Omega)}$$

Deze bovenstaande uitspraak uit zich in onze grafieken in een scherpere helling in de L_2-lijnen: als u goed kijkt in figuren 3, 4 en 5, ziet u dat de L_2-norm (rode lijn) zich meer als $1/m^2 \sim h^2$ beweegt. Voor de één na laatste plot in figuur 6, ziet u dat er iets soortgelijks gebeurt, hoewel dat niet door de theoretische stellingen wordt voorspeld. Een bespreking van de laatste plot van figuur 7, laten we achterwege aangezien de finite-element method hier niet daadwerkelijk van toepassing is.

[6] mathsci.kaist.ac.kr/ dykwak/Courses/Num765-08/FEM-theory.pdf

4.2 Vergelijking van H_1-seminorm met de error estimator

In de colleges van *Numerical Methods for Stationary PDEs* werd de volgende bovengrens bewezen, voor homogene Dirichlet randvoorwaarden, dimensie twee en de gebruikelijk Poissonvergelijking. Stilletjes aan zullen we aannemen dat deze ook geldt in ons specifieke geval:

$$\|u - u_\tau\|^2_{H^1(\Omega)} \lesssim \eta(u_\tau, \tau)^2$$

Dit komt duidelijk overeen met de plotjes uit figuur 3, 4, 5 en 6 (de groene lijn). Hoewel ik de specifieke voorwaarden voor deze bovengrens niet precies heb opgeschreven in mijn schrijfblok, ben ik geneigd te denken dat u_n van figuur 6 niet aan deze voorwaarden hoeft te voldoen.

Het blijkt eigenlijk dat deze errorestimator 'best wel goed' is: hij (de groene lijn) volgt vrij nauw de lijn (blauw) van de H_1-norm.

5 Matlabcommando's

Figuur	Matlabcommando	Tijdsduur
3	`graph_test('f_sinsin','f_sinsin_sol',64)`	10 minuten
4	`graph_test('f_exp','f_exp_sol',64)`	10 minuten
5	`graph_test('f_pol','f_pol_sol',64)`	10 minuten
6	`graph_test('f_not','f_not_sol',64)`	10 minuten
7	`graph_test('f_jump','f_jump_sol',64)`	10 minuten
8	`test('f_exp','f_exp_sol',32)`	1 minuut
9	`test('f_sin2sin2','f_sin2sin2_sol',32)`	1 minuut
10a	`test('f_jump','f_jump_sol',32)`	1 minuut
10b	`plotfun('f_jump_sol')`	10 seconden

Het programma is ontwikkeld en getest met Matlab 7.8.0 (R2009a).