Numerieke Modellering en Benadering

Practicum 1, academiejaar 2017-2018

1 Een toepassing van de bivariate discrete sinustransformatie

We bestuderen in deze opgave het gebruik van de discrete sinustransformatie (DST) voor het numeriek oplossen van partiële differentiaalvergelijkingen.

1.1 Een korte beschrijving van de methode

De Poissonvergelijking modelleert de evenwichtstemperatuursverdeling u van een vlak voorwerp,

$$\frac{\partial^2}{\partial x^2}\,u(x,y) + \frac{\partial^2}{\partial y^2}\,u(x,y) = f(x,y) \;\text{ met }\; (x,y) \in D = [0,1]\times[0,1]\,. \tag{1}$$

We beperken ons hier voor wat het domein D betreft, tot het eenheidsvierkant. De rechterlidfunctie f is gegeven en beschrijft de warmtetoevoer. We beschouwen het geval van Dirichletrandvoorwaarden, d.w.z. we veronderstellen dat de temperatuur op de rand van D vastligt. De opgegeven functies u_w, u_o, u_z en u_n specificeren de temperatuur aan de west-, oost-, zuid- en noordkant van D. Dit betekent:

$$u(0,y) = u_w(y)\;,\quad u(1,y) = u_o(y)\;,\quad u(x,0) = u_z(x) \;\text{ en }\; u(x,1) = u_n(x)\,. \tag{2}$$

De oplossing van een partiële differentiaalvergelijking is meestal niet exact te berekenen. Men gebruikt dan numerieke methoden die de exacte oplossing benaderen, bijvoorbeeld in een aantal discrete punten. Deze punten worden doorgaans op een regelmatig rooster van verticale en horizontale lijnen gekozen, met tussenafstand $h = 1/(N+1)$ en N een natuurlijk getal: $(x_i, y_j),\ i,j = 0,\ldots,N+1$ met $x_i = i\,h$ en $y_j = j\,h$.

De benaderingen van de oplossing u in de roosterpunten zullen we voorstellen als $U_{i,j} \approx u(x_i,y_j)$, de waarden van f als $f_{i,j} = f(x_i,y_j)$. Alleen de $U_{i,j}$-waarden voor $i,j = 1,\ldots,N$ zijn onbekend. De waarden met een index i of j gelijk aan 0 of gelijk aan $N+1$ kunnen immers geëvalueerd worden uit de randvoorwaarden. De onbekenden $U_{i,j}$ stoppen we in een matrix $U \in \mathbb{R}^{N\times N}$.

Vervolgens wordt de partiële differentiaalvergelijking omgezet tot een stelsel algebraïsche vergelijkingen, met behulp van een klassieke benadering voor de tweede-orde afgeleide,

$$\frac{U_{i+1,j} - 2U_{i,j} + U_{i-1,j}}{h^2} + \frac{U_{i,j+1} - 2U_{i,j} + U_{i,j-1}}{h^2} = f_{i,j}\,,$$

voor $i,j = 1,\ldots,N$. Dit kan herschreven worden als

$$U_{i+1,j} + U_{i-1,j} + U_{i,j+1} + U_{i,j-1} - 4U_{i,j} = F_{i,j} \;\text{ met }\; F_{i,j} = f_{i,j}h^2\,. \tag{3}$$

De $F_{i,j}$-waarden zijn de elementen van een matrix $F \in \mathbb{R}^{N\times N}$. Merk op dat deze discretisatie in elk inwendig roosterpunt tot een vergelijking leidt. Al die vergelijkingen samen vormen een zeer groot stelsel met N^2 vergelijkingen en evenveel onbekenden. In (3) komen ook de gekende waarden op de randen voor. Die dienen verplaatst te worden naar het rechterlid, als volgt:

$$F_{1,j} := F_{1,j} - u_w(y_j),\ F_{N,j} := F_{N,j} - u_o(y_j),\ F_{i,1} := F_{i,1} - u_z(x_i),\ F_{i,N} := F_{i,N} - u_n(x_i)\,. \tag{4}$$

Voor het oplossen van de vergelijkingen gebruiken we een tweedimensionale ontbinding in termen van sinusfuncties, en dit zowel voor de $U-$ als voor de $F-$matrix:

$$U_{i,j} = \sum_{k=1}^{N}\sum_{\ell=1}^{N} \widetilde{U}_{k,\ell} \sin(\pi k i h) \sin(\pi \ell j h) \quad \text{en} \quad F_{i,j} = \sum_{k=1}^{N}\sum_{\ell=1}^{N} \widetilde{F}_{k,\ell} \sin(\pi k i h) \sin(\pi \ell j h)\,.$$

De coëfficiënten $\widetilde{U}_{k,\ell}$ en $\widetilde{F}_{k,\ell}$ kunnen berekend worden met een tweedimensionale DST. Dit geeft de matrices $\widetilde{U}, \widetilde{F} \in \mathbb{R}^{N\times N}$. Wanneer we die ontbinding invullen in vergelijking (3), en gebruik maken van de goniometrische formule $\sin\alpha + \sin\beta = 2\sin\frac{\alpha+\beta}{2}\cos\frac{\alpha-\beta}{2}$, verkrijgen we

$$\sum_{k=1}^{N}\sum_{\ell=1}^{N} \widetilde{U}_{k,\ell} \left(2\cos(\pi k h) + 2\cos(\pi \ell h) - 4\right) \sin(\pi k i h) \sin(\pi \ell j h) = \sum_{k=1}^{N}\sum_{\ell=1}^{N} \widetilde{F}_{k,\ell} \sin(\pi k i h) \sin(\pi \ell j h)\,.$$

Controleer dit! Hieruit volgt onmiddellijk dat de coëfficiënten $\widetilde{U}_{k,\ell}$ kunnen berekend worden als

$$\widetilde{U}_{k,\ell} = \frac{\widetilde{F}_{k,\ell}}{2(\cos(\pi k h) + \cos(\pi \ell h) - 2)} = \frac{-\widetilde{F}_{k,\ell}}{4(\sin^2(\frac{\pi k h}{2}) + \sin^2(\frac{\pi \ell h}{2}))}\,. \tag{5}$$

De opeenvolgende stappen van het oplossingsalgoritme kunnen nu samengevat worden als volgt:

1. Evalueer matrix F met behulp van formule (3).
2. Update matrix F voor de randvoorwaarden via formule (4).
3. Bereken matrix $\widetilde{F}$ d.m.v. een 2D DST toegepast op matrix F.
4. Bereken matrix $\widetilde{U}$ met behulp van (5).
5. Bereken matrix U d.m.v. een 2D inverse DST toegepast op matrix $\widetilde{U}$.

Na deze stappen bevat U een benaderende oplossing van vergelijking (1) met randvoorwaarden (2). Merk op dat U enkel de waarden bevat horende bij de *inwendige* punten van het rooster. Wensen we de oplossing te tekenen als een 3D oppervlak of m.b.v. een aantal niveaulijnen, dan moet natuurlijk ook rekening gehouden worden met de gekende waarden op de rand.

1.2 Opgaven

Opgave 1. Implementeer de methode in MATLAB. Controleer de correctheid van je code door het oplossen van (1) met als exacte oplossing $u(x, y) = 1$. Voor dit probleem moet de numerieke oplossing precies gelijk zijn aan de exacte oplossing, eventueel op kleine afrondingsfouten na. Hetzelfde geldt voor een vergelijking met oplossing $u(x, y) = 1 + x + y$ of $u(x, y) = x^2 + y^2$.

Opgave 2. Los de Poissonvergelijking op met als oplossing $u(x, y) = e^{x+y}$. Maak een tabel waarbij je de nauwkeurigheid van de numerieke oplossing uitzet tegenover N. Als maat voor de nauwkeurigheid neem je de maximale (absolute) waarde van de fout in de roosterpunten. Als waarden van N neem je $N = 8, 16, 32, \ldots, 1024$. Hoeveel seconden heeft je computer nodig om het laatste probleem op te lossen (een stelsel met 1048576 vergelijkingen en onbekenden)? Gebruik de MATLAB-commando's `tic` en `toc` om deze tijdsduur te meten.

Opgave 3. Schrijf een MATLAB-script `PDE.m` dat de oplossing berekent van (1) met randvoorwaarden $u_w(y) = 3, u_o(y) = 2, u_z(x) = 2 + \sin(\pi x/2), u_n(x) = 1$ en als rechterlidfunctie $f(x, y) = -100$ voor $(x, y) \in [0.4, 0.6] \times [0.4, 0.6]$ en $f(x, y) = 0$ daarbuiten. Plot het resultaat op twee verschillende manieren. Met het commando `surf` geef je de numerieke benadering weer in 3D. Met `contour` teken je een aantal contourlijnen van je benadering.

Neem de code van `PDE.m`, eventuele niet ingebouwde MATLAB-functies die door `PDE.m` worden opgeroepen en beide figuren op in het verslag. We geven nog volgende hints. Je kan de 2D DST/IDST berekenen met het MATLAB-commando `dst/idst`, gecombineerd met het principe van herhaalde ééndimensionale benadering. Bekijk zeker ook eens de documentatie van het commando `meshgrid`. Met dit commando kun je heel efficiënt functies evalueren op een rooster.

2 Kleinste-kwadratenbenadering met splinefuncties

Beschouw een dataset van meetgegevens of functiewaarden. We wensen een kleinste-kwadraten benadering op te stellen met behulp van splinefuncties. Eén van de voordelen van splines ten opzichte van veeltermen is de vrijheid in de keuze van de knooppunten. Zo is het bijvoorbeeld mogelijk om meer knooppunten te gebruiken in een interval waar de functie sterker variëert.

2.1 De kleinste-kwadratenbenadering met B-splines

De functieruimte van splines van graad k voor een knooppuntenrij $\{t\}_{i-k}^{n+k}$ heeft dimensie $n + k$. We kunnen elke spline dus schrijven als een sommatie van de vorm

$$s(x) = \sum_{i=-k}^{n-1} c_i N_{i,k+1}(x). \tag{6}$$

De genormaliseerde B-splines $N_{i,k+1}(x)$ worden volledig bepaald door de knooppunten.

Een eenvoudige manier om de kleinste-kwadratenbenadering te vinden van een functie f die gegeven is in een aantal punten x_j, $j = 1, 2, \ldots, r$, is door de benaderende kleinste-kwadratenoplossing te zoeken van het overgedetermineerde stelsel

$$Mc = f. \tag{7}$$

Hierin is M een $(r \times (n + k))$-matrix waarvan rij j de functiewaarden $N_{i,k+1}(x_j)$ bevat. Het rechterlid is een kolomvector met de waarden $f(x_j)$, en de onbekende vector c bestaat uit de coëfficiënten van de kleinste-kwadratenbenadering. De knooppunten zijn in dit probleem vrij te kiezen, als hun aantal maar voldoende groot is.

2.2 Efficiënte berekening van de matrix

De genormaliseerde B-splines voldoen aan een interessante recursiebetrekking. Die betrekking vormt de basis voor tal van efficiënte en stabiele algoritmes, zoals het algoritme van de Boor dat gebruikt wordt voor de evaluatie van de som (6). Ook de berekening van de elementen van de

matrix M in (7) kan heel efficiënt gebeuren, steunend op deze recursiebetrekking. Men kan dan één enkele driehoekige tabel opstellen waarin men de waarden van *alle* genormaliseerde B-splines die verschillen van nul in het punt x, terugvindt. Die tabel ziet er als volgt uit:

$$\begin{array}{|cccccc|}
\hline
 & & & & & N_{j-k,k+1}(x) \\
 & & & & \cdot & N_{j-k+1,k+1}(x) \\
 & & & N_{j-2,3}(x) & \dots & N_{j-k+2,k+1}(x) \\
 & & N_{j-1,2}(x) & N_{j-1,3}(x) & \dots & \vdots \\
N_{j,1}(x)=1 & N_{j,2}(x) & N_{j,3}(x) & \dots & & N_{j,k+1}(x) \\
\hline
\end{array}$$

We vinden in de i-de kolom de functiewaarden van alle B-splines van graad $i-1$, die verschillend zijn van nul in het punt x. De berekeningsmethode is zeer stabiel. Elk getal in de tabel wordt immers gevonden als een convexe combinatie van positieve getallen.

2.3 Opgaven

Opgave 4. Schrijf een *efficiënte* MATLAB-functie voor het opstellen en evalueren van een kleinste-kwadratenbenadering met kubische splinefuncties. Gebruik voor de evaluatie van alle B-splines in een punt het algoritme dat in §2.2 van deze opgave gesuggereerd wordt, op basis van de recursiebetrekking. Voor de evaluatie van de benadering zelf maak je een implementatie van het algoritme van de Boor. Pas dit algoritme toe op de som (6). Voor het oplossen van het stelsel (7) mag je gebruik maken van de ingebouwde "\" operator van MATLAB – die berekent automatisch een kleinste-kwadratenoplossing van een overgedetermineerd stelsel. De functie die we willen bekomen is

```
function z = kkb_spline(t,x,f,y,k)
```

Deze functie krijgt als invoer een vector `t` die de knooppuntenrij voorstelt. Parameter `x` is een vector van lengte r die de abscissen bevat, met r bijhorende functiewaarden in `f`. Parameter `y` is een vector van lengte N die de punten bevat waarin we de splinefunctie willen evalueren. De parameter `k` geeft de graad van de spline. Als uitvoer geeft de functie de N-vector `z` terug met de functiewaarden van de splinebenadering in de punten `y`.

Opgave 5. Teken een aantal B-splines ter illustratie van de correctheid van het algoritme van de Boor. Zet hiervoor alle coëfficiënten op 0 uitgezonderd één enkele coëfficiënt.

Opgave 6. Illustreer het belang van de ligging van de knooppunten bij het construeren van een kleinste-kwadraten splinebenadering.

Opgave 7. Wanneer de data onderhevig is aan ruis (bijvoorbeeld door meetfouten) betekent een kleiner residu $\|Mc - f\|_2$ niet noodzakelijk een betere benadering van het onderliggende model. We benaderen de functie

$$f(x) = \frac{\sin(20x)}{100x^2 + 5}$$

op het interval $[-1, 1]$ op basis van een bemonstering van $f(x)$ in 200 equidistante abscissen die door additieve ruis geperturbeerd wordt:

```
x = linspace(-1,1,200)';
f = sin(20*x)./(100*x.^2+5);
f_ruis = f + 0.04*randn(size(x));
```

Gebruik kubische B-splines. Bereken voor een toenemend aantal knooppunten `t` de splinebenadering `z = kkb_spline(t,x,f_ruis,x,3)` en plot het verloop van de fouten `norm(f-z)` en `norm(f_ruis-z)` op eenzelfde grafiek. Welk aantal knooppunten geeft het beste resultaat voor beide normen? Plot ook voor beide foutmetingen het beste resultaat dat je bekomt samen met de achterliggende functie f. Bespreek de resultaten.

Praktische richtlijnen

- Dit practicum wordt gemaakt in groepjes van twee.
- Stuur het verslag (in PDF) **tesamen met alle Matlab code** door per mail ten laatste op **15 april 2018** om 12:00 uur op onderstaand e-mailadres.
- In het verslag wordt minstens de code `kkb_spline.m` en `PDE.m` verwacht, evenals je implementatie van het algoritme van de Boor en van het algoritme voor het opstellen van de matrix M. Neem ook alle gevraagde figuren op in het verslag.
- Bespreek bondig je resultaten en je aanpak in een duidelijke vergezellende tekst. Hou de lengte evenwel onder de 10 pagina's.

Succes!

Simon Telen (200A 01.152)
`simon.telen@cs.kuleuven.be`