UNIVERSITETET I OSLO
Matematisk Institutt

Masteroppgave i matematikk

Optimale randbetingelser for det diskrete Laplace-problemet

Andreas Brandsæter

mai 2011

Forord

Denne oppgaven representerer avslutningen av mastergraden i matematikk under LAP-programmet ved Matematisk institutt, Universitetet i Oslo. Oppgaven er en kort masteroppgave (30 studiepoeng), og er skrevet i løpet våren 2011.

Oppgaven kombinerer teori om partielle differensialligninger og lineær optimering, og leseren forventes å ha grunnleggende forkunnskaper innen disse emnene. I tillegg vil noe kjennskap til lineær algebra være nyttig.

Arbeidet med oppgaven har vært svær interessant og spennende, ikke minst takket være min veileder Geir Dahl (MI/Ifi/CMA), som stadig har hjulpet meg til å se nye resultater og utvidelser.

Blindern, 10. mai 2011,
Andreas Brandsæter

Innhold

Kapittel 1

Introduksjon

Hovedmålet med denne oppgaven er å studere et optimeringsproblem i forbindelse med en partiell differensialligning (PDE) kalt Laplace-ligningen. Laplace-ligningen gir opphav til Laplace-problemet med Dirichlet-randbetingelser som kan defineres som følger

$$\begin{aligned} \Delta u &= 0 \quad \text{i} \quad \Omega \\ u &= g \quad \text{på} \quad \partial\Omega \end{aligned} \tag{1.1}$$

der den ukjente u er en passende deriverbar funksjon $u : \mathbb{R}^n \to \mathbb{R}$. Problemet er definert for dimensjon n, men vi vil begrense oss til å studere det to-dimensjonale tilfellet. I to dimensjoner er Laplace-operatoren Δ gitt ved $\Delta = \frac{\partial^2}{\partial x^2} + \frac{\partial^2}{\partial y^2}$. Videre vil vi begrense oss til å studere det åpne enhetskvadratet $\Omega = \{(x, y) \in \mathbb{R}^2 : 0 < x, y < 1\}$, og vil la $\partial\Omega$ betegne randa til området. Funksjonen $g : \partial\Omega \to \mathbb{R}$ angir det vi vil omtale som Dirichlet-randbetingelser.

I denne oppgaven vil vi hovedsakelig konsentrere oss om en approksimasjon til (1.1) kalt det diskrete Laplace-problemet. Problemet oppstår fra (1.1) ved å innføre et uniformt grid i Ω, og vi vil la Ω_h betegne diskretiseringen av Ω. Hvordan dette gjøres vil grundig bli forklart i kapittel 3.

Hensiktsmessig valg av randverdier

Etter at vi i kapittel 2 og 3 henholdsvis har redegjort for det kontinuerlige og det diskrete Laplace-problemet, og nevnt noen sentrale resultater, vil vi undersøke hvordan vi kan oppnå ulike bestemte løsninger av det diskrete Laplace-problemet ved hensktsmessig valg av randverdier gitt ved funksjonen g. Det er vanligst å studere Laplace-problemet der funksjonen g er gitt på forhånd, men i denne oppgaven vil vi la denne funksjonen være variabel i forbindelse med følgende problem:

Treffpunktsproblemet: *Gitt en endelig mengde punkter $P \subseteq \Omega_h$, med vilkårlige verdier gitt ved $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Bestem randverdier $g(x_i, y_j)$ for $(x_i, y_j) \in \partial\Omega_h$ slik at*

$$v_g(x_i, y_j) - \alpha(x_i, y_j) \quad = \quad 0 \qquad \text{for } (x_i, y_j) \in P \tag{1.2}$$

der v_g er den unike løsningen av det diskrete Laplace-problemet med Dirichlet-randbetingelser g.

Gitt en mengde punkter P med verdier gitt ved funksjonen $\alpha : P \to \mathbb{R}$ ønsker vi altså å finne randverdier g slik at løsningen av det diskrete Laplace-problemet "treffer" de gitte punktene P i verdiene $\alpha(x_i, y_j)$ for alle $(x_i, y_j) \in P$.

Vi vil blant annet vise at dersom delmengden P består av 4 punkter vil det alltid eksistere en løsning av treffpunktsproblemet (1.2), for vilkårlige verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Vi vil også vise at dette alltid gjelder i et spesialtilfelle der alle punktene i P er fordelt langs randa av et rektangel.

Behov for optimering

Selv om det i de ovennevnte tilfeller, samt en mengde andre, er mulig å finne en løsning av treffpunktsproblemet gjelder dette ikke bestandig. Vi vil blant annet vise at dersom P består av 5 eller flere punkter vil det finnes konstellasjoner av punkter slik at det ikke eksisterer en løsning av treffpunktsproblemet. I tilfeller der det ikke er mulig å finne en løsning som "treffer" de gitte verdiene α i punktene i P vil vi ofte være interessert i å finne en løsning som ligger så nær denne løsningen som mulig. Motivert av dette vil vi i kapittel 5 gi en meget kort innføring i lineær optimering, slik at vi i det påfølgende kapittelet kan definere følgende optimeringsproblem

Optimeringsproblemet: *Gitt en delmengde P av Ω_h og en funksjon $\alpha : P \to \mathbb{R}$, bestem en randfunksjon g som minimerer funksjonen*

$$\Phi(g) := \sum_{(x_i, y_j) \in P} |v_g(x_i, y_j) - \alpha(x_i, y_j)| \tag{1.3}$$

der v_g er den unike løsningen av det diskrete Laplace-problemet med Dirichlet-randbetingelser g.

Anvendelser og et eksempel

Problemet vi har beskrevet har flere anvendelser. En anvendelse finner vi innenfor varmebehandling av metalliske materialer. Gjennom oppvarming eller avkjøling av et materiale eller ved opphold ved bestemte temperaturer kan materialets egenskaper endres. I mange tilfeller vil det være hensiktsmessig med ulike egenskaper i ulike deler av et materiale.

Som et eksempel vil vi studere en lang metallisk stav med kvadratisk tverssnitt som varmebehandles. Gitt en ønsket temperaturfordeling i stavens tverrsnitt vil vi undersøke hvordan vi kan varme opp eller kjøle ned staven langs randa slik at temperaturfordelingen i materialet blir en så god tilnærming til denne ønskede fordelingen som mulig. Dersom vi lar diskretiseringsgraden n være 3, vil det diskretiserte området Ω_h av området Ω inneholde 3×3 punkter. Vi kan for eksempel la den ønskede temperaturfordelingen i området, $\bar{\Omega}_h = \Omega_h \cup \partial\Omega_h$, være gitt ved

$$\begin{bmatrix} * & * & * & * & * \\ * & 2 & 1 & 3 & * \\ * & 0 & * & * & * \\ * & * & 5 & 2 & * \\ * & * & * & * & * \end{bmatrix}$$

der $*$ betegner punkter i temperaturfordelingen der ingen ønsket temperatur er gitt. Løsningen av optimeringsproblemet vil gi randverdier g langs $\partial\Omega_h$, som i en viss forstand gir den beste tilnærmingen til den ønskede temperaturfordelingen i Ω_h.

Ved å løse optimeringsproblemet finner vi følgende randverdier

$$\begin{bmatrix} 3.50 & 3.50 & -3.01 & 4.47 & 4.47 \\ 3.50 & & & & 4.47 \\ -6.36 & & & & 1.16 \\ 2.19 & & & & 0.47 \\ 2.19 & 2.19 & 13.63 & 0.47 & 0.47 \end{bmatrix}$$

Laplace-problemet med Dirichlet-randverdier g som skissert i matrisen over har følgende løsning, som altså er den optimale løsningen av optimeringsproblemet (1.3)

$$\begin{bmatrix} 3.50 & 3.50 & -3.01 & 4.47 & 4.47 \\ 3.50 & \mathbf{2.00} & \mathbf{1.00} & \mathbf{3.00} & 4.47 \\ -6.36 & \mathbf{0.00} & 2.01 & 2.04 & 1.16 \\ 2.19 & 2.34 & \mathbf{5.00} & \mathbf{2.00} & 0.47 \\ 2.19 & 2.19 & 13.63 & 0.47 & 0.47 \end{bmatrix}$$

Vi observerer at løsningen gir en eksakt tilnærming til den ønskede temperaturfordelingen. Dermed er løsningen også en løsning av treffpunktsproblemet. En grundig forklaring av dette problemet vil bli gitt i eksempel 6.1.

Effektiv løsning av optimeringsproblemet

Før vi avslutter oppgaven med en oppsummering og konklusjon, vil vi i kapittel 7 presentere løsninger av optimeringsproblemet for en rekke eksempler. Problemet vil løses ved hjelp av effektive metoder og algoritmer i programmene MATLAB og CPLEX. De ulike metodene vil sammenlignes på grunnlag av løsninger og beregningstider for en rekke eksempler returnert fra de ulike metodene.

Beregningstider vil presenteres underveis, men de er også å finne i tillegg A. I tillegg B finnes en beskrivelse av et program med grafisk brukergrensesnitt som vi har utviklet for å løse optimeringsproblemer på formen (1.3). Programmet, som krever at MATLAB og toolboxen optimtool er forhåndsinstallert, kan lastes ned fra internettsiden www.brandseter.com/OBVLP.html. Utvalgte øvrige programkoder er vedlagt i tillegg C.

Hovedresultater

Hovedresultatene fra arbeidet med denne oppgaven er å finne i kapittel 4 og 6, der resultater i forbindelse med treffpunktsproblemet (1.2) og optimeringsproblemet (1.3) blir presentert. Av sentrale resultater kan vi nevne et resultat som sier at det alltid eksisterer en optimal løsning av optimeringsproblemet. Vi vil også presentere et teorem som karakteriserer sammenhengen mellom optimeringsproblemet og treffpunktsproblemet. Videre vil vi presentere uklike versjoner av optimeringsproblemet med ulike egenskaper og kjennetegn.

I vårt arbeid har vi brukt betydelige resurser på å utvikle metoder for å løse optimeringsproblemet effektivt. Vi har valgt å ikke gi en teknisk, detaljert presentasjon av disse metodene her, da en oppgave som denne er lite egnet for en slik presentasjon. Vi vil likevel fremheve metodene som viktige resultater av arbeidet.

Kapittel 2

Laplace-problemet

Teorien om differensialligninger er den viktigste disiplinen i den moderne matematikken.

Sophus Lie (1842-1899)
Leipziger Berichte, 1895

Innledningsvis i dette kapittelet vil vi definere Poisson-problemet med Dirichlet-randbetingelser dets spesialtilfelle kalt Laplace-problemet. I den påfølgende seksjonene vil vi nevne noen anvendelser av problemet. Deretter vil vi studere maksimum- og minimumsprinsippet for harmoniske funksjoner, og bruke dette til å vise at Poisson-problemet har entydig løsning. Dette etterfølges av at vi utleder analytiske løsning av Laplace-problemet med svært enkle Dirichlet-randbetingelser på enhetskvadratet, før vi avslutter med å gi løsningen av problemet med mer generelle randbetingelser. Det hele vil illustreres med noen eksempler.

2.1 Poisson- og Laplace-problemet

For en gitt funksjon f definert på et åpent, sammenhengende og begrenset område Ω, og en gitt funksjon g definert på randa $\partial\Omega$, er Poisson-problemet med Dirichlet-randbetingelser gitt ved

$$\begin{aligned} -\Delta u &= f \quad \text{i} \quad \Omega \\ u &= g \quad \text{på} \quad \partial\Omega \end{aligned} \tag{2.1}$$

der den ukjente $u : \Omega \to \mathbb{R}$ er en passende deriverbar funksjon, og Δ betegner Laplace-operatoren, definert ved $\Delta u = \sum_j \frac{\partial^2 u}{\partial x_j^2}$. Den øverste linjen kalles Poisson-ligningen og den nederste linjen angir Dirichlet-randbetingelser. Poisson-ligningen er en fundamental partiell differensialligning med anvendelser blant annet innenfor mange områder av matematisk fysikk [5, 10, 14]. Vi har definert Poisson-problemet med Dirichlet-randbetingelser for et generelt åpent, sammenhengende og begrenset område Ω i $\mathbb{R}^n$, men vi skal begrense oss til å studere kvadratiske områder Ω i $\mathbb{R}^2$.

Under passende betingelser på funksjonene f og g og området Ω kan det vises at det alltid finnes en entydig løsning av Poisson-problemet. I denne oppgaven vil vi konsentrere oss om en klasse slike problemer, der funksjonen u er harmonisk i Ω, og u i en viss forstand bestemmes ut fra funksjonen g på randa $\partial\Omega$.

Definisjon 2.1. *Vi sier at en funksjon $u \in C^2(\Omega) \cap C(\bar{\Omega})$ er harmonisk hvis*

$$\Delta u(x, y) = 0 \qquad \text{for alle } (x, y) \in \Omega \tag{2.2}$$

der $u \in C^2(\Omega)$ betyr at alle partiellderiverte av orden ≤ 2 er kontinuerlige i Ω, og $\bar{\Omega} = \Omega \cup \partial\Omega$.

Spesialtilfellet av Poisson-problemet med Dirichlet-randbetingelser der u er harmonisk kalles Laplace-problemet og er gitt ved

$$\begin{aligned} \Delta u(x, y) &= 0 && \text{for alle } (x, y) \in \Omega \\ u(x, y) &= g(x, y) && \text{for alle } (x, y) \in \partial\Omega \end{aligned} \tag{2.3}$$

2.2 Anvendelser

Løsningen av Laplace-problemet med Dirichlet-randbetingelser har mange anvendelser innenfor matematisk fysikk, med anvendelser innenfor blant annet varmeledning, fluidmekanikk, elastisitet og elektrostatikk [2, 5, 10, 14]. I dette avsnittet vil vi kort beskrive et par av dem.

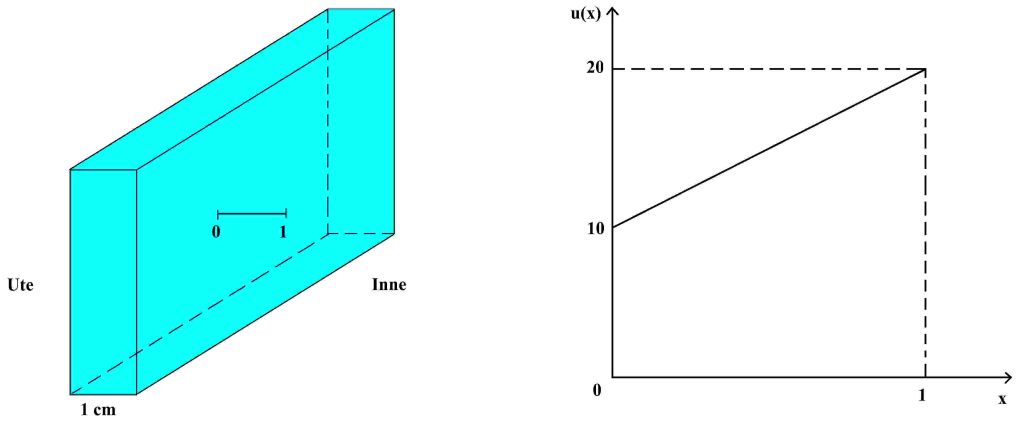
2.2.1 Elektrostatikk

Vi lar Ω være et åpent, sammenhengende og begrenset område som representere elektriske ledere. Vi induserer en konstant spenning langs randa $\partial\Omega$. Denne spenningen angir Dirichlet-randbetingelsene og varierer ikke med tiden. Løsningen av Laplace-problemet med Dirichlet-randbetingelsene gir da det elektriske potensialet ved likevekt i det indre av området.

2.2.2 Varmeledning

Løsningen av Laplace-problemet kan også representere temperaturfordelingen i området Ω ved likevekt. Temperaturen langs randa $\partial\Omega$ vil angi Dirichlet-randbetingelser for problemet. Randa $\partial\Omega$ kan for eksempel være tilkoblet en varmeovn eller en avkjølende væske slik at temperaturen langs randa ikke endrer seg over tid.

Eksempel 2.1 (Varmeledning i vindusrute): Som et enkelt eksempel i én dimensjon kan vi studere varmefordelingen i en vindusrute. Vi antar at temperaturen ute er konstant lik 10 grader og temperaturen inne er konstant lik 20 grader, og ønsker å bestemme hvordan varmen fordeler seg gjennom glasset i et punkt på vindusruta (i x-retning). Vi lar vindusruta være 1 cm tykk. Vi antar også at punktet vi studerer er langt unna



Figur 2.1: Figuren til venstre viser en vindusrute beskrevet i eksempel 2.1. Til høyre vises temperaturen som funksjon av posisjon i glasset.

vinduskarmene slik at temperaturfordelingen er tilnærmet homogen i y- og z-retning. Problemet er illustrert til venstre i figur 2.1.

Laplace-ligningen med Dirichlet-randbetingelsene i én dimensjon er gitt ved

$$\begin{aligned} \Delta u(x) &= \frac{\partial^2 u(x)}{\partial x^2} = 0 \\ u(0) &= 10 \\ u(1) &= 20 \end{aligned} \tag{2.4}$$

Laplace-ligningen (øverste linje av (2.4)) har generell løsning $u(x) = a + bx$. Dette kan lett verifiseres ved å derivere uttrykket. Ved hjelp av Dirichlet-randbetingelsene (de to nederste linjene i (2.4)) kan vi bestemme konstantene a og b, og varmefordelingen ved likevekt er dermed gitt ved

$$u(x) = 10 + 10x$$

Varmefordelingen ved likevekt er skissert til høyre i figur 2.1.

I samsvar med vår intuisjon observerer vi at funksjonen u oppnår sin maksimum- og minimumsverdi på randa av området. Dette stemmer godt med maksimum og minimumsprinsippet for harmoniske funksjoner som vi skal studere nærmere i neste seksjon.

2.3 Maks/min-prinsippet for harmoniske funksjoner

La først $u = u(x)$ være en harmonisk funksjon i én variabel, definert på området (α, β), der $\alpha, \beta \in \mathbb{R}$. I én variabel vil harmoniske funksjoner være lineære, og da er det klart at u vil oppnå sin maksimum- og minimumsverdi i randpunktene, og vi har følgende

$$\min(u(\alpha), u(\beta)) \leq u(x) \leq \max(u(\alpha), u(\beta)) \qquad \text{for alle } (x, y) \in (\alpha, \beta)$$

Maksimum- og minimumsprinsippet for harmoniske funksjoner i to variable sier at en tilsvarende ulikhet holder for slike funksjoner.

Teorem 2.2 (Maks/min-prinsippet for harmoniske funksjoner). *Anta at u er harmonisk i et åpent, sammenhengende og begrenset område $\Omega \subseteq \mathbb{R}^2$. Da tilfredsstiller u følgende ulikhet*

$$M_0 \leq u(x,y) \leq M_1 \qquad \text{for alle } (x,y) \in \Omega$$

der

$$M_0 = \min_{(x,y)\in\partial\Omega} u(x,y) \qquad \textit{og} \qquad M_1 = \max_{(x,y)\in\partial\Omega} u(x,y)$$

For bevis av teorem 2.2 vil vi henvise til Tveito og Winther [14].

Korollar 2.3. *Hvis u er harmonisk i Ω, vil*

$$|u(x,y)| \leq M \qquad \text{for alle } (x,y) \in \Omega$$

der

$$M = \max_{(x,y)\in\partial\Omega} |u(x,y)|.$$

Bevis. Fra teorem 2.2 følger det at

$$|u(x,y)| \leq \max(u(x,y), -u(x,y)) \leq \max(M_1, -M_0) = M.$$

□

Nå har vi alt vi trenger for å vise at Poisson-problemet (2.1) i to dimensjoner maksimalt har én løsning.

Teorem 2.4 (Unik løsning av Poisson-problemet). *Anta at $u^1, u^2 \in C^2(\Omega) \cap C(\bar{\Omega})$ er to løsninger av Poisson-problemet (2.1) med samme funksjon f og samme Dirichlet-randbetingelser g. Da er*

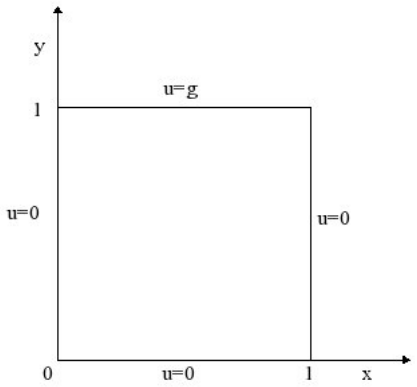
$$u^1 \equiv u^2$$

Bevis. Vi lar $\hat{u} = u^1 - u^2$. Dersom u^1 og u^2 begge er løsninger av Poisson-problemet med samme funksjon f følger det at

$$\Delta\hat{u} = \Delta(u^1 - u^2) = \Delta u^1 - \Delta u^2 = f - f = 0 \tag{2.5}$$

Dermed er $\hat{u}$ harmonisk. Vi har antatt at u^1 og u^2 har samme Dirichlet-betingelser g, så $u^1 = u^2 = g$ på $\partial\Omega$. Dette medfører at $\hat{u} \equiv 0$ på $\partial\Omega$.

Fra korollar 2.3 vet vi at $|\hat{u}|$ er begrenset av den maksimale verdien på randa. Siden denne verdien er 0 følger det at $v \equiv 0$ på Ω, og det impliserer at $u^1 \equiv u^2$. Vi har dermed vist at løsningen av Poisson-problemet er entydig. □



Figur 2.2: Dirichlet-randbetingelser på enhetskvadratet

2.4 Analytisk løsning av Laplace-problemet

I forrige avsnitt viste vi at det alltid eksisterer en unik løsning av Poisson-problemet med Dirichlet-randbetingelser, og dermed også spesialtilfellet Laplace-problemet (2.3). I dette avsnittet vil vi undersøke hvordan vi kan finne denne unike løsningen av Laplace-problemet med Dirichlet-betingelser i to dimensjoner for kvadratiske områder. For å finne løsningen av Laplace-problemet analytisk bruker vi teknikker utviklet av Joseph Fourier (1768-1830). Gjennom sitt arbeid med varmeledning utviklet den franske fysikeren det som frem til i dag er den mest betydningsfulle metoden for å løse partielle differensialligninger [10, 14].

For å løse Laplace-problemet med Dirichlet-randbetingelser vil vi her bruke Fouriers metode, og vise hvordan løsningen kan uttrykkes ved uendelige rekker. Konvergens av fourier-rekker, som er nødvendig for å bevise at vi faktisk har en løsning, vil ikke bli behandlet i denne oppgaven. For en grundig innføring i konvergens av fourier-rekker vil vi henvise til [10, 14].

Innledningsvis vil vi analysere og utlede en løsning av Laplace-problemet der bare én av kvadratets sider har ikke-homogene randbetingelser, før vi vil gi løsningen for det mer generelle problemet der alle sidene kan ha ikke-homogene randbetingelser. For enkelhets skyld vil vi la området vi studerer være enhetskvadratet, $\Omega = \{(x, y) \mid 0 < x, y < 1\}$. For mer kompliserte eksempler og detaljer kan vi anbefale [11, 14], som denne utledningen bygger på.

2.4.1 Konstante ikke-homogene Dirichlet-randbetingelser langs én side

Vi ønsker altså å finne den unike løsningen av Laplace-ligningen $-\Delta u(x, y) = 0$ for $(x, y) \in \Omega$, med følgende Dirichlet-randbetingelser (se figur 2.2)

$$\begin{array}{rcll} u(0,y) & = & 0 & 0 \leq y \leq 1, \\ u(1,y) & = & 0 & 0 \leq y \leq 1, \\ u(x,0) & = & 0 & 0 \leq x \leq 1, \\ u(x,1) & = & g(x) & 0 < x < 1, \end{array} \tag{2.6}$$

Det første vi gjør er å separere variablene. Vi gjør følgende antagelse

$$u(x,y) = X(x)Y(y). \tag{2.7}$$

Setter vi dette inn i Laplace-ligningen $-\Delta u(x,y) = 0$, får vi

$$-X''(x)Y(y) - X(x)Y''(y) = 0.$$

Ved å dele med XY på begge sider og flytte det ene leddet over får vi

$$-\frac{X''(x)}{X(x)} = \frac{Y''(y)}{Y(y)} \tag{2.8}$$

Nå observerer vi at venstresiden bare avhenger av x og høyre siden bare avhenger av y. Dermed er det klart at begge sider må være lik en konstant λ som ikke avhenger av hverken x eller y. Vi kan derfor dele opp problemet (2.8) i to separate egenverdiproblemer; et som avhenger av x og et som avhenger av y.

x-avhengig

For funksjonen $X(x)$ får vi følgende egenverdiproblem

$$\begin{array}{ll} -X''(x) = \lambda X(x) & 0 < x < 1 \\ X(0) = 0 & \\ X(1) = 0 & \end{array} \tag{2.9}$$

der randbetingelsen for $x = 0$ og $x = 1$ er utledet fra (2.6). Vi kan nå velge egenverdi

$$\lambda_k = (k\pi)^2 \qquad k = 1, 2, ...$$

og korresponderende egenvektor

$$X_k(x) = \sin(k\pi x) \qquad k = 1, 2, ...$$

Deriverer vi egenvektoren $X_k(x)$ to ganger kan vi observere at egenverdien λ_k og egenvektoren $X_k(x)$ tilfredsstiller egenverdiproblemet med randbetingelser (2.9).

y-avhengig

Funksjonen $Y(y)$ må tilfredsstille følgende

$$Y''(y) = \lambda Y(y) \qquad 0 < y < 1 \tag{2.10}$$
$$Y(0) = 0 \tag{2.11}$$

der randbetingelsen for $y = 0$ er utledet fra (2.6). De ikke-homogene randbetingelsene for $y = 1$ som også er utledet fra (2.6) vil vi ta hensyn til senere. Den generelle løsningen av (2.10) med $\lambda = (k\pi)$ (husk at det ikke er noe minustegn forran Y'') er en lineærkombinasjon av $e^{k\pi y}$ og $e^{-k\pi y}$. Randbetingelsene tatt i betraktning får vi følgende løsning av Y

$$Y_k(y) = (e^{k\pi y} - e^{-k\pi y})/2 = \sinh(k\pi y) \qquad k = 1, 2, ... \tag{2.12}$$

Løsningen

Kombinerer vi uttrykket for $X(x)$ og $Y(y)$ får vi fra ligning 2.7

$$u_k(x, y) = X_k(x)Y_k(y) = \sin(k\pi x)\sinh(k\pi y) \qquad \text{for} \qquad k = 1, 2, ...$$

Funksjonene over tilfredsstiller Laplace-ligningen samt randbetingelsene (2.6). Dette vil også gjelde for lineærkombinasjoner, og for reelle koeffisienter c_k har vi følgende løsning

$$\boxed{u_k(x, y) = \sum_{k=1}^{\infty} c_k \sin(k\pi x)\sinh(k\pi y) \qquad \text{for} \qquad k = 1, 2, ...} \tag{2.13}$$

Nå gjenstår det bare å bestemme koeffisientene c_k. La oss derfor se på den siste randbetingelsen

$$u(x, 1) = g(x) \qquad 0 < x < 1,$$

Vi skal anta at funksjonen $g(x)$ kan skrives som en fourier-sinus-rekke på formen

$$g(x) = \sum_{k=1}^{\infty} g_k \sin(k\pi x) \tag{2.14}$$

der g_k er gitt ved

$$g_k = 2\int_0^1 g(x)\sin(k\pi x)dx$$

Kombinerer vi uttrykket for $u(x, 1)$ innsatt i (2.13) og uttrykket for $g(x)$ gitt ved (2.14), får vi

$$c_k = g_k / \sinh(k\pi) \qquad \text{for } k = 1, 2, ...,$$

Vi har dermed funnet et uttrykk for løsningen av Laplace-problemet med Dirichlet-randbetingelser (2.6).

Eksempel 2.1 (V=0, H=0, B=0, T=1): Det er på tide med et eksempel. Vi lar Laplace-problemet med Dirichlet-randbetingelser være definert som over (2.6), og lar $g(x) = 1$ for $0 < x < 1$. Området Ω har altså randbetingelsene 0 på bunnen (B), og på høyre (H) og venstre (V) side, og 1 på toppen (T).

Funksjonen $g(x) = 1$ for $0 < x < 1$ kan skrives som en fourier-rekke [10, 14]

$$g(x) \quad = \quad 1 \quad = \quad \sum_{k=1,3,5,...}^{\infty} \frac{4}{k\pi} \sin(k\pi x) \tag{2.15}$$

Fra (2.14) ser vi at $g_k = 4/k\pi$, som igjen gir

$$c_k = \frac{4}{k\pi \sinh(k\pi)}$$

Den unike løsningen av problemet er dermed definert ved

$$u(x, y) = \frac{4}{\pi} \sum_{k=1,3,5,...}^{\infty} \frac{1}{k \sinh(k\pi)} \sin(k\pi x) \sinh(k\pi y) \tag{2.16}$$

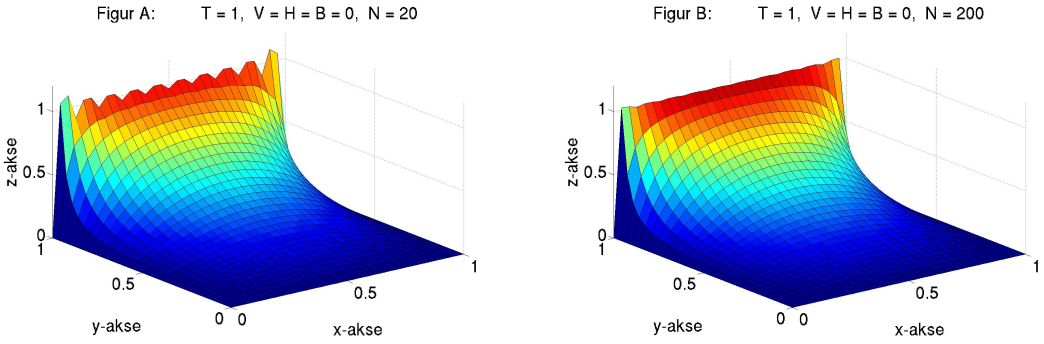
Ved å derivere uttrykket over (2.16) finner vi raskt at løsningen $u(x, y)$ tilfredsstiller Laplace-ligningen (2.3). Også Dirichlet-randbetingelsene (2.6) er tilfredsstilt. Dette kan verifiseres ved innsetting.

I figur 2.3 er løsningen skissert. Figurene er laget ved å innføre et uniformt grid i området, og beregne verdien i hvert punkt fra den analytiske løsningen gitt over. Verdiene til den konstante funksjonen $g(x) = 1$ er uttrykt som en fourier-rekke gitt ved (2.15), der det summeres fra $k = 1, 3, 5, ...N$. I figur A til venstre vises en løsning beregnet med $N = 20$, og i figur B til høyre er N økt til 200. Figuren indikerer randbetingelsen langs toppen vil gå mot den eksakte verdien når N går mot uendelig.

2.4.2 Vilkårlige ikke-homogene Dirichlet-randbetingelser

La oss nå studere et mer generelt problem der randbetingelsene på alle kvadratets sider er gitt ved vilkårlige endimensjonale funksjoner som kan uttrykkes ved fourier-rekker. Vi lar området Ω være enhetskvadratet som tidligere, og betrakter nå Laplace-problemet med Dirichlet-randbetingelser gitt ved

$$\begin{aligned} u(0, y) &= V(y) \qquad 0 < y < 1, \\ u(1, y) &= H(y) \qquad 0 < y < 1, \\ u(x, 0) &= B(x) \qquad 0 < x < 1, \\ u(x, 1) &= T(x) \qquad 0 < x < 1, \end{aligned}$$



Figur 2.3: Løsning av Laplace-problemet med konstante Dirichlet-randbetingelser på enhetskvadratet som beskrevet i eksempel 2.1. x- og y-aksene representerer posisjon, mens z-aksen for eksempel representerer temperatur.

der $V, H, B, T : (0,1) \to \mathbb{R}$ er vilkårlige funksjoner. Vi antar at funksjonene kan uttrykkes ved endimensjonale fourier-rekker

$$\begin{aligned} V(y) &= \sum_{k=1,3,5,\ldots}^{\infty} V_k \sin(k\pi y) \qquad 0 < y < 1 \\ H(y) &= \sum_{k=1,3,5,\ldots}^{\infty} H_k \sin(k\pi y) \qquad 0 < y < 1 \\ B(x) &= \sum_{k=1,3,5,\ldots}^{\infty} B_k \sin(k\pi x) \qquad 0 < x < 1 \\ T(x) &= \sum_{k=1,3,5,\ldots}^{\infty} T_k \sin(k\pi x) \qquad 0 < x < 1 \end{aligned} \tag{2.17}$$

Det kan da vises, se for eksempel [10], at den unike løsningen av problemet er funksjonen $u : \Omega \to \mathbb{R}$ definert ved

$$\boxed{u(x,y) = v(x,y) + h(x,y) + b(x,y) + t(x,y)\,, \qquad \text{for alle } (x,y) \in \Omega} \tag{2.18}$$

der

$$\begin{aligned} v(x,y) &= \sum_{k=1,3,5,\ldots}^{\infty} \frac{V_k}{\sinh(k\pi)} \sinh(k(\pi - x\pi)) \cdot \sin(k\pi y) \\ h(x,y) &= \sum_{k=1,3,5,\ldots}^{\infty} \frac{H_k}{\sinh(k\pi)} \sinh(k\pi x) \cdot \sin(k\pi y) \\ b(x,y) &= \sum_{k=1,3,5,\ldots}^{\infty} \frac{B_k}{\sinh(k\pi)} \sinh(k(\pi - y\pi)) \cdot \sin(k\pi x) \\ t(x,y) &= \sum_{k=1,3,5,\ldots}^{\infty} \frac{T_k}{\sinh(k\pi)} \sinh(k\pi y) \cdot \sin(k\pi x) \end{aligned} \tag{2.19}$$

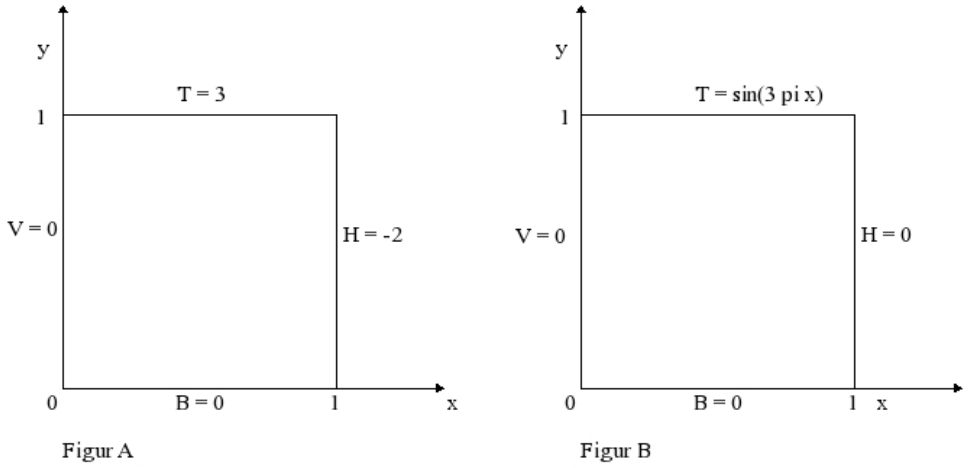
for alle $(x, y) \in \Omega$.

La oss illustrere problemet med et par eksempler. Eksemplene er hentet fra [10].

Eksempel 2.2 (V=0, H=-2, B=0, T=3): Vi lar Dirichlet-randbetingelsene være gitt ved

$$\begin{aligned} V(y) &= 0 && \text{for } 0 < y < 1 \\ H(y) &= -2 && \text{for } 0 < y < 1 \\ B(x) &= 0 && \text{for } 0 < x < 1 \\ T(x) &= 3 && \text{for } 0 < x < 1 \end{aligned} \tag{2.20}$$

som illustrert i figur 2.4 A.



Figur 2.4: Dirichlet-randbetingelser for eksempel 2.2 (til venstre) og 2.3 (til høyre).

Funksjonene $H(y)$ og $T(x)$ kan skrives som fourier-rekker

$$\begin{aligned} H(y) &= \sum_{k=1,3,5,\ldots}^{\infty} H_k \sin(k\pi y) = \sum_{k=1,3,5,\ldots}^{\infty} (-2)\frac{4}{k\pi}\sin(k\pi y) \qquad 0 \le y \le 1 \\ T(x) &= \sum_{k=1,3,5,\ldots}^{\infty} T_k \sin(k\pi x) = \sum_{k=1,3,5,\ldots}^{\infty} 3\frac{4}{k\pi}\sin(k\pi x) \qquad 0 \le x \le 1 \end{aligned}$$

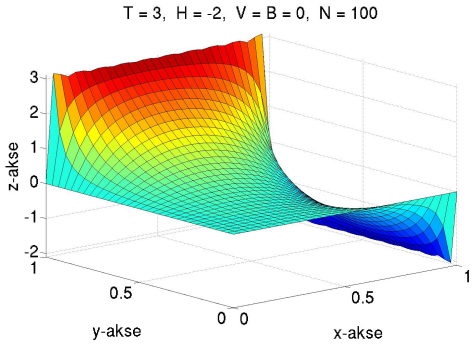
Vi observerer i henhold til (2.14) at $H_k = -\frac{8}{\pi k}$ og $T_k = \frac{12}{\pi k}$. Fra (2.19) ser vi da at

$$\begin{aligned} h(x,y) &= -\frac{8}{\pi}\sum_{k=1}^{\infty}\frac{1}{k\sinh(k\pi)}\sinh(k\pi x)\cdot\sin(k\pi y) \\ t(x,y) &= \frac{12}{\pi}\sum_{k=1}^{\infty}\frac{1}{k\sinh(k\pi)}\sinh(k\pi y)\cdot\sin(k\pi x) \end{aligned}$$

Den unike løsningen av dette Laplace-problemet er dermed gitt ved funksjonen $u : \Omega \to \mathbb{R}$ definert ved

$$u(x,y) = h(x,y) + t(x,y)$$

Løsningen er skissert i figur 2.5.



Figur 2.5: Løsning av Laplace-problemet i eksempel 2.2 med ikke-homogene Dirichlet-randbetingelser

Eksempel 2.3 (V=0, H=0, B=0, T= sin(3πx)): Vi avslutter dette kapittelet med et siste eksempel. Vi lar Dirichlet-randbetingelsene for toppen (T) være gitt ved funksjonen $T(x) = \sin(3\pi x)$ for $0 < x < 1$, og lar $V(y) = H(y) = B(x) = 0$. Se figur 2.4 B.

Fra (2.17) får vi at

$$T(x) \approx \sum_{k=1}^{\infty} T_k \sin(k\pi x) = \sin(3\pi x)$$

der T_k er gitt ved

$$T_k = \begin{cases} 1 & \text{for } k = 3 \\ 0 & \text{ellers} \end{cases}$$

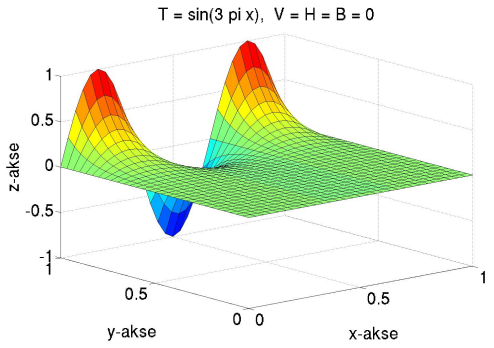
Vi setter dette inn i uttrykket fra (2.19), og får

$$t(x,y) \approx \sum_{k=1}^{\infty} \frac{T_k}{\sinh(k\pi)} \sinh(k\pi y) \cdot \sin(k\pi x) = \frac{\sinh(3\pi y) \cdot \sin(3\pi x)}{\sinh(3\pi)}$$

Fra (2.18) følger det nå at den unike løsningen av dette problemet er gitt ved funksjonen $u : \Omega \to \mathbb{R}$ definert ved

$$u(x,y) = t(x,y)$$

Løsningen er skissert i figur 2.6.



Figur 2.6: Løsning av Laplace-problemet med ikke-homogene Dirichlet-randbetingelser som beskrevet i eksempel 2.3.

Kapittel 3

Det diskrete Laplace-problemet

I forrige kapittel viste vi hvordan vi kan finne en analytisk løsning av det kontinuerlige Poisson-problemet, og spesialtilfellet Laplace-problemet (2.3), definert på enhetskvadratet med noen utvalgte Dirichlet-randbetingelser. For et generelt område Ω er det ikke mulig å finne en løsning av det kontinuerlige problemet [6, 14]. Motivert av dette vil vi i dette kapittelet formulere og studere et diskret problem som er en approksimasjon til det kontinuerlige Poisson-problemet, og vi vil vise hvordan dette problemet kan løses numerisk.

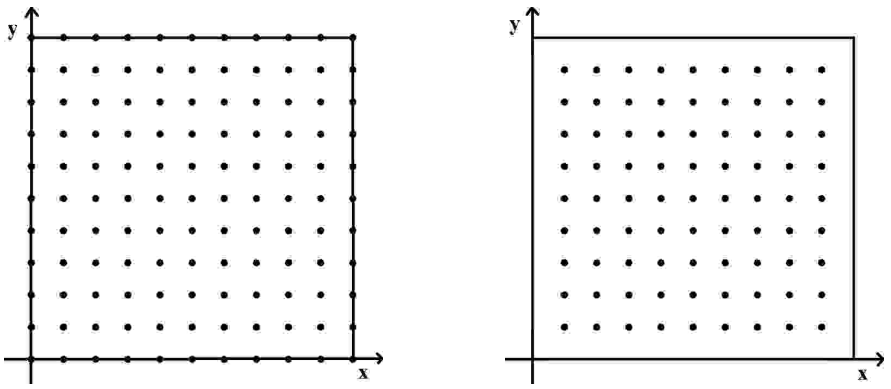
Selv om vår konsentrasjon hovedsakelig er rettet mot Laplace-problemet vil vi innledningsvis studere det mer generelle Poisson-problemet. Resultatene vi viser vil selvsagt være gjeldende også for spesialtilfellet kalt Laplace-problemet.

Vi begynner med å redegjøre for diskretisering av Poisson-problemet ved bruk av fem-punkts skjema. Med bakgrunn i blant annet Tveito og Winthers introduksjonsbok i partielle differensialligninger [14] vil vi vise at det eksisterer et maksimum- og minimumsprinsipp for det diskrete Laplace-problemet som tilsvarer maksimum- og minimumsprinsippet i det kontinuerlige tilfellet. Vi vil bruke dette for å vise at det diskrete Poisson-problemet med Dirichlet-randbetingelser alltid vil ha en unik løsning. Dernest vil vi undersøke hvordan vi kan finne denne løsningen. Avslutningsvis vil vi presentere noen eksempler der vi sammenligner den analytiske løsningen av det kontinuerlige Laplace-problemet med den numeriske løsningen av det tilsvarende diskrete problemet.

En av de store fordelene med å benytte numeriske metoder er at de lett kan anvendes på generelle områder Ω. For enkelhets skyld, og for å tydeliggjøre sammenhengen med drøftingen av det kontinuerlige problemet, vil vi likevel også her begrense oss til å betrakte enhetskvadratet $\Omega = \{(x, y) \mid 0 < x, y < 1\}$.

3.1 Diskretisering

Vi ønsker som sagt å finne en løsning av det diskrete Poisson-problemet og dets spesialtilfelle kalt Laplace-problemet, som vi vil vie mest oppmerksomhet til i denne oppgaven. Vi begynner med å diskretisere området. Diskretiseringen av område vil representere det kontinuerlige området med et endelig antall punkter. Avstanden mellom hvert punkt vil



Figur 3.1: Definisjon av $\bar{\Omega}_h$ og Ω_h

betegnes h, der $h = 1/(n+1)$ og n er et gitt ikke-negativt heltall som angir diskretiseringsgraden. Store verdier av n vil gi et grid[1] med fin inndeling, mens små verdier vil gi et grovt grid. En finere inndeling vil gi en mer nøyaktig tilnærming til det analytiske problemet, men det vil også medføre at problemet blir større og mer tidkrevende å løse. Gridpunktene betegnes (x_i, y_j) der $(x_i, y_j) = (ih, jh)$ for $0 \leq i, j \leq n+1$.

Mengden av gridpunktene i det indre av området og mengden av gridpunktene på randa vil betegnes av henholdsvis Ω_h og $\partial\Omega_h$. Mengden av alle gridpunktene vil betegnes $\bar{\Omega}_h$. (Se figur 3.1).

$$\begin{array}{lcl} \bar{\Omega}_h & = & \{(x_i, y_j) = (ih, jh) \mid 0 \leq i, j \leq n+1\} \\ \Omega_h & = & \{(x_i, y_j) = (ih, jh) \mid 1 \leq i, j \leq n\} \\ \partial\Omega_h & = & \bar{\Omega}_h \setminus \Omega_h \end{array}$$

3.1.1 Fem-punkts skjema

Idéen bak nesten alle numeriske metoder for å løse differensialligninger er å approksimere differensialligningen ved et sett av algebraiske ligninger [1]. Ligningene kan settes opp på ulike måter, og tilstreber en så god approksimasjon som mulig. Her vil vi begrense oss til å beskrive den vanligste metodene som bruker det som kalles Fem-punkts skjema[2].

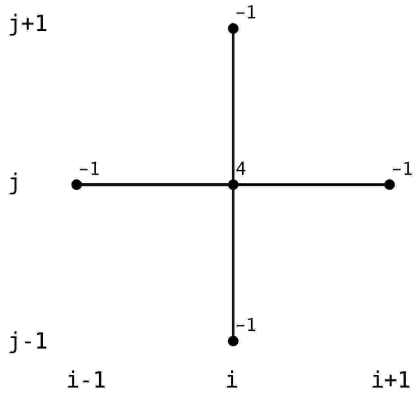
Vi ønsker å formulere en diskret approksimasjon til Poisson-problemet med Dirichlet-randbetingelser i to dimensjoner, og spesialtilfellet kalt Laplace-problemet. For å gjøre dette approksimerer vi Laplace-operatoren Δ ved å bruke den såkalte fem-punkts-operatoren som vi har valgt å betegne Δ_h[3], definert ved

$$\begin{array}{rcl} \Delta_h v_{i,j} & = & \frac{1}{h^2}[(-v_{i-1,j} + 2v_{i,j} - v_{i+1,j}) + (-v_{i,j-1} + 2v_{i,j} - v_{i,j+1}] \\ & = & \frac{1}{h^2}[4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1}] \end{array} \tag{3.1}$$

[1] gitter, rutenett med konstant avstand mellom punktene
[2] *eng.*: Five point scheme
[3] Fem-punkts-operatoren Δ_h betegnes ofte også L_h.

for alle $(x_i, y_j) \in \Omega_h$, der $v_{i,j}$ angir funksjonsverdien $v(x_i, y_j)$ for en grid-funksjon v definert i området $\bar{\Omega}_h$.

Fem-punkts-operatoren har fått sitt navn fordi funksjonsverdien i et gridpunkt bestemmes ut fra verdien i de fire nabopunktene i tillegg til verdien i det samme punktet, som illustrert i figur 3.2.



Figur 3.2: Illustrasjon av fem-punkts-operatoren Δ_h

3.1.2 Det diskrete Poisson-problemet

Ved å substituere Laplace-operatoren Δ med fem-punkts-operatoren Δ_h i uttrykk (2.1), får vi en endelig differanse approksimasjon til Poisson-problemet med Dirichlet-betingelser i to dimensjoner gitt ved

$$\begin{aligned} \Delta_h v(x_i, y_j) &= f(x_i, y_j) \qquad \text{for alle} \quad (x_i, y_j) \in \Omega_h \\ v(x_i, y_j) &= g(x_i, y_j) \qquad \text{for alle} \quad (x_i, y_j) \in \partial\Omega_h \end{aligned} \tag{3.2}$$

3.1.3 Det diskrete Laplace-problemet

Dersom vi lar grid-funksjonen f i uttrykket (3.2) over være konstant lik 0, får vi et uttrykk for en diskrete versjon av Laplace-problemet gitt ved

$$\begin{aligned} \Delta_h v(x_i, y_j) &= 0 \qquad \text{for alle} \quad (x_i, y_j) \in \Omega_h \\ v(x_i, y_j) &= g(x_i, y_j) \qquad \text{for alle} \quad (x_i, y_j) \in \partial\Omega_h \end{aligned} \tag{3.3}$$

3.2 Maks/min-prinsippet og eksistens av unik løsning

I dette avsnittet skal vi se at diskrete harmoniske funksjoner tilfredsstiller et maksimum- og minimumsprinsipp tilsvarende maksimum- og minimumsprinsippet for analytiske harmoniske funksjoner, som vi studerte i seksjon 2.3. Vi skal deretter bruke dette for å vise at det diskrete Poisson-problemet med Dirichlet-betingelser (3.2) som vi formulerte i forrige avsnitt har unik løsning. Vi begynner med å gi en definisjon av diskrete harmoniske funksjoner.

Definisjon 3.1. *En grid-funksjon v kalles diskret harmonisk dersom*

$$\Delta_h v(x_i, y_j) = 0 \qquad \text{for alle } (\mathrm{x_i}, \mathrm{y_j}) \in \Omega_\mathrm{h} \tag{3.4}$$

Teorem 3.2 (Maks/min-prinsippet for diskrete harmoniske funksjoner). *Dersom v er en diskret harmonisk funksjon holder følgende*

$$M_0 \;\leq\; v(x_i, y_j) \;\leq\; M_1 \quad \text{for alle } (x_i, y_j) \in \bar{\Omega}_h$$

der

$$M_0 \;=\; \min_{(x_i,y_j)\in\partial\Omega_h} v(x_i, y_j) \quad \text{og} \quad M_1 \;=\; \max_{(x_i,y_j)\in\partial\Omega_h} v(x_i, y_j)$$

For bevis av teorem 3.2 vil vi henvise til Tveito og Winther [14]. I analogi med det analytiske tilfellet har vi følgende korollar.

Korollar 3.3. *Hvis v er en diskret harmonisk funksjon, vil*

$$|v(x_i, y_j)| \leq M \qquad \text{for alle } (x_i, y_j) \in \Omega_h$$

der

$$M = \max_{(x_i,y_j)\in\partial\Omega_h} |v(x_i, y_j)|.$$

Bevis. Beviset er identisk med beviset for korollar 2.3. Fra teorem 3.2 følger det at

$$|v(x_i, y - j)| \leq \max(v(x_i, y_j), -v(x_i, y_j)) \leq \max(M_1, -M_0) = M.$$

□

Vi vil nå bruke resultatene over til å vise at det diskrete Poisson-problemet med Dirichlet-betingelser (3.2) alltid har en unik løsning.

Teorem 3.4 (Eksistens av unik løsning av Poisson-problemet). *Det finnes en unik grid-funksjon v_g som løser det diskrete Poisson-problemet (3.2) med Dirichlet-randbetingelser g.*

Bevis. Som vi skal se i neste avsnitt, kan Poisson-problemet (3.2) ved passende variabelskifte betraktes som et ligningssystem $\mathbf{Ax}=\mathbf{b}$ med n^2 ligninger og n^2 ukjente $\mathbf{x}=\{v_{i,j}\}_{i,j=1}^{n}$. Fra lineær algebra [7] vet vi at et kvadratisk lineært ligningssystem har unik løsning dersom nullrommet til matrisen er 0. Det er dermed tilstrekkelig å vise at det korresponderende homogene systemet $\mathbf{Ax}=\mathbf{0}$ kun har den trivielle løsningen $\mathbf{x}= v \equiv 0$.

Vi antar derfor at v tilfredsstiller

$$\begin{aligned} \Delta_h v(x_i, y_j) &= 0 \qquad \text{for alle} \quad (x_i, y_j) \in \Omega_h \\ v(x_i, y_j) &= 0 \qquad \text{for alle} \quad (x_i, y_j) \in \partial\Omega_h \end{aligned}$$

Vi må vise at $v \equiv 0$ er eneste løsning, men dette følger direkte fra korollar 3.3. □

3.3 Diskret løsning av Laplace-problemet

I det foregående avsnittet viste vi at det alltid eksisterer en unik løsning av det diskrete Poisson-problemet med Dirichlet-randbetingelser (2.1). Dette gjelder også spesialtilfellet Laplace-problemet (3.3). Vi vil vie mest konsentrasjon til Laplace-problemet, men vi begynner med å studere det mer generelle Poisson-problemet (3.2). Ved å sette inn for fem-punkts-operatoren Δ_h i (2.1) får vi følgende ligningssystem

$$\begin{aligned} 4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1} &= h^2\, f_{i,j} && \text{for } \; i,j = 1,2,...,n \\ v_{0,j} &= g_{0,j} && \text{for } \; i,j = 0,1,...,n,n+1 \\ v_{n+1,j} &= g_{n+1,j} && \text{for } \; i,j = 0,1,...,n,n+1 \\ v_{i,0} &= g_{i,0} && \text{for } \; i,j = 0,1,...,n,n+1 \\ v_{i,n+1} &= g_{i,n+1} && \text{for } \; i,j = 0,1,...,n,n+1 \end{aligned} \tag{3.5}$$

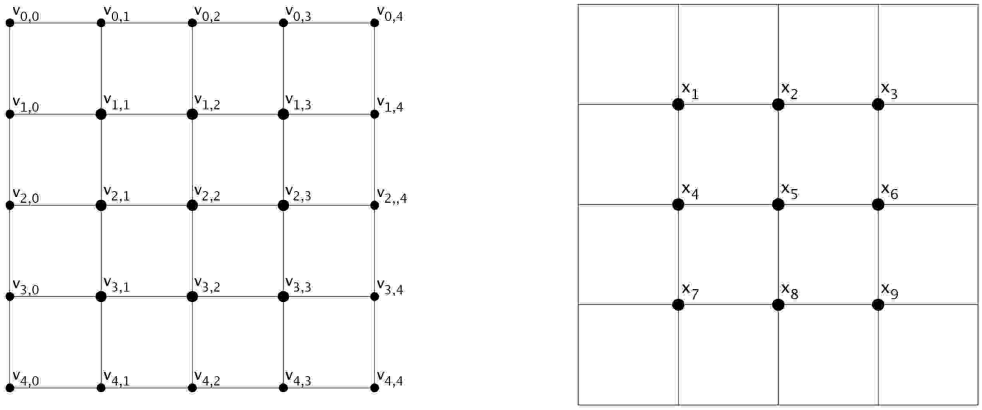
3.3.1 Formulering av problemet på matriseform

Vi ønsker å skrive ligningssystemet (3.5) på formen $\mathbf{Ax}=\mathbf{b}$, der $\mathbf{A}$ er en matrise, $\mathbf{x}$ er en vektor av variablene $v_{i,j}$, og $\mathbf{b}$ er en vektor som representerer funksjonsverdiene $f_{i,j}$. Ved å løse dette ligningssystemet løser vi det diskrete problemet som er en approksimasjon til det analytiske problemet.

Vi observerer at verdiene for variablene $v_{i,j}$ på randa er gitt ved $g_{i,j}$, og vi vil derfor ikke behandle disse som variable. Vi vil kun beregne løsningen for punktene i det indre av området, beskrevet av den øverste linja i (3.5). Denne linja representerer n^2 ligninger, én for hvert punkt i det indre av området, og matrisen $\mathbf{A}$ vil dermed ha dimensjon $n^2 \times n^2$.

Variabelskifte

Vi skriver variablene $v_{i,j}$ og funksjonsverdiene $f_{i,j}$ som vektorene $\mathbf{x}$ og $\mathbf{b}$ ved hjelp av *vec*-operatoren som stabler kolonnene i matrisene oppå hverandre på følgende måte (se



Figur 3.3: Viser variabelskifte fra $v_{i,j}$ til $\mathbf{x}_k$

figur 3.3)

Definisjon 3.5. *For enhver matrise* $\boldsymbol{B} \in \mathbb{R}^{m,n}$ *definerer vi*

$$vec(\boldsymbol{B}) = [b_{1,1}, ..., b_{m,1}, b_{1,2}, ..., b_{m,2}, ..., b_{1,n}, ..., b_{m,n}]^T \in \mathbb{R}^{mn}.$$

Dette gir $\mathbf{x}$=$vec(\mathbf{V})$ og $\mathbf{b}$=$vec(\mathbf{F})$, der $\mathbf{V}$ er matrisen av variablene $v_{i,j}$ og $\mathbf{F}$ er matrisen av funksjonsverdiene $f_{i,j}$. For Laplace-problemet vil $\mathbf{b}$ opprinnelig være gitt ved $\mathbf{0}$-vektoren.

Problemets struktur

Ligningssystemet har en spesiell struktur som vi skal utnytte. Hvis vi begynner i øvre venstre hjørne, med $i = j = 1$, har vi følgende

$$4v_{1,1} - v_{0,1} - v_{2,1} - v_{1,0} - v_{1,2} \quad = \quad h^2 f_{1,1} \tag{3.6}$$

Vi gjør nå variabelskiftet som forklart over og illustrert i figur 3.3, og setter inn verdien for $g_{i,j}$ for punktene på randa $\partial\Omega_h$, og får

$$4x_1 - g_{0,1} - x_{1+n} - g_{1,0} - x_{1+1} \quad = \quad h^2 f_1$$

Vi flytter nå de gitte randverdiene $g_{0,1}$ og $g_{1,0}$ over på høyre side i ligningen og får

$$4x_1 - x_{1+n} - x_{1+1} \quad = \quad h^2 f_1 + g_{0,1} + g_{1,0}.$$

Vi gjør tilsvarende for neste punkt $i = 1, j = 2$. Fra (3.5) har vi

$$4v_{1,2} - v_{0,2} - v_{2,2} - v_{1,1} - v_{1,3} \quad = \quad h^2 f_{1,2} \tag{3.7}$$

Vi gjør variabelskifte og setter inn verdien for $g_{i,j}$ for punktene på randa på samme måte som over, og får

$$4x_2 - g_{0,2} - x_{2+n} - x_{2-1} - x_{2+1} \quad = \quad h^2 f_2$$

Til slutt flytter vi de gitte randverdiene $g_{0,1}$ og $g_{1,0}$ over på høyre side i ligningen og får

$$4x_2 - x_{2+n} - x_{2-1} - x_{2+1} \quad = \quad h^2 f_2 + g_{0,2}.$$

Vi kan fortsette på samme måte for alle n^2 punktene. I hjørnene av området vil to randverdier flyttes over på høyre side i ligningen og legges til verdien i vektoren $\mathbf{b}$, langs randa vil én randverdi flyttes over og legges til verdien i vektoren $\mathbf{b}$, og i punkter som ikke har randa som nabopunkter vil ingen randverdi flyttes over. For Laplace-problemet vil funksjonen $f \equiv 0$ og $\mathbf{b}$ vil dermed i utgangspunktet være $\mathbf{0}$-vektoren før randverdiene legges til.

Prosedyren vi har beskrevet gir en matrise $\mathbf{A}$ med en spesiell struktur, og vi kaller denne matrisen for Poisson-matrisen. Generelt er Poisson-matrisen $\mathbf{A}$ gitt ved følgende

$$\begin{array}{rcrcll} & & a_{i,i} & = & 4, & \text{for } i = 1, ..., n^2 \\ a_{i+1,i} & = & a_{i,i+1} & = & -1, & \text{for } i = 1, ..., n^2 - 1, \quad i \neq n, 2n, ..., (n-1)n \\ a_{i+n,i} & = & a_{i,i+n} & = & -1, & \text{for } i = 1, ..., n^2 - n \\ & & a_{i,j} & = & 0, & \text{ellers} \end{array} \tag{3.8}$$

For $n = 3$ gir det følgende matrise

$$\mathbf{A} = \begin{bmatrix} 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 & 0 \\ -1 & 4 & -1 & 0 & -1 & 0 & 0 & 0 & 0 \\ 0 & -1 & 4 & 0 & 0 & -1 & 0 & 0 & 0 \\ -1 & 0 & 0 & 4 & -1 & 0 & -1 & 0 & 0 \\ 0 & -1 & 0 & -1 & 4 & -1 & 0 & -1 & 0 \\ 0 & 0 & -1 & 0 & -1 & 4 & 0 & 0 & -1 \\ 0 & 0 & 0 & -1 & 0 & 0 & 4 & -1 & 0 \\ 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 & -1 \\ 0 & 0 & 0 & 0 & 0 & -1 & 0 & -1 & 4 \end{bmatrix} \tag{3.9}$$

3.3.2 Effektive metoder for å løse matriseligningen $\mathbf{Ax} = \mathbf{b}$

I det foregående avsnittet skrev vi det diskrete Laplace-problemet på formen $\mathbf{Ax} = \mathbf{b}$, der $\mathbf{A}$ er Poisson-matrisen (3.8). Det som gjenstår for å løse det diskrete Laplace-problemet er å løse denne matriseligningen.

Matrisen $\mathbf{A}$ i matriseligningen $\mathbf{Ax} = \mathbf{b}$ kan bli svært stor, for eksempel hvis approksimasjonen til det tilsvarende analytiske problemet skal være svært god. Det er derfor viktig å ha effektive metoder for å løse ligningsystemet. Ved å ta hensyn til den spesielle

strukturen i matrisen vil vi kunne skrive effektive algoritmer for dette. Både direkte og iterative metoder kan benyttes [8].

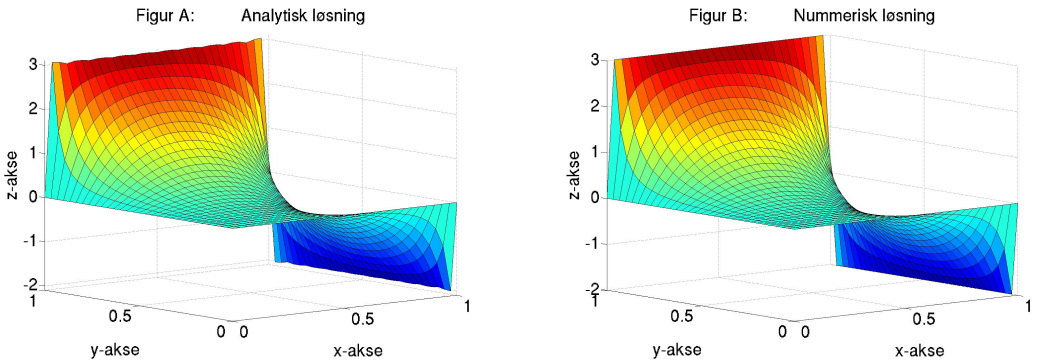
I eksemplene som følger er en algoritme basert på diagonalisering benyttet. Det vil føre for langt i en oppgave som denne å gi en grundig innføring i denne eller andre metoder, og vi vil derfor henvise til [8] for detaljer. Programkode er vedlagt i tillegg C.2.

3.4 Sammenligning av analytisk og numerisk løsning

Til slutt i dette kapittelet vil vi illustrere sammenhengen mellom det kontinuerlige og det diskrete Laplace-problemet med Dirichlet-randbetingelser. Vi vil ta et tilbakeblitt på et par eksempler fra kapittel 2, og sammenligne den analytiske løsningen av det kontinuerlige Laplace-problemet med den numeriske løsningen av det tilsvarende diskrete problemet.

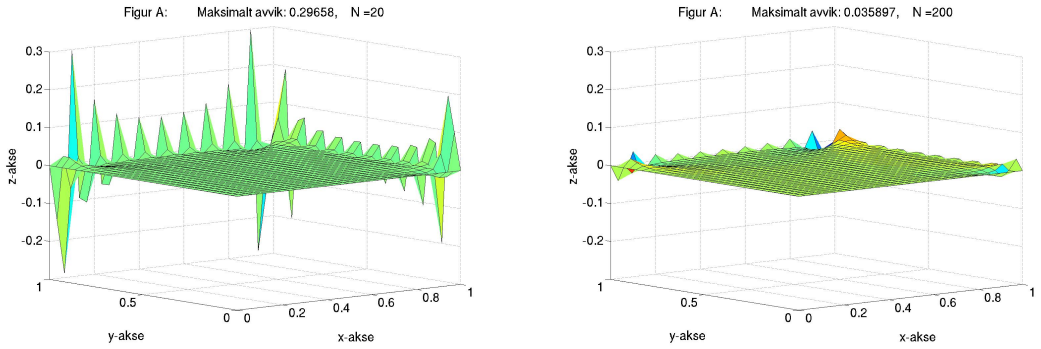
Sammenligningen er gjennomført i MATLAB. Programkode for diskret løsning av Laplace-problemet med Dirichlet-randbetingelser, **DIRICHLET.m**, er vedlagt i tillegg C.1.

Eksempel 3.1 (V=0, H=-2, B=0, T=3): Vi skal løse det diskrete Laplace-problemet med samme Dirichlet-randbetingelser som i det kontinuerlige problemet vi studerte i eksempel 2.2. Vi husker at Dirichlet-randbetingelsene for toppen (T) og høyre side (H) er henholdsvis gitt ved de konstante funksjonen $T(x) = 3$ og $H(y) = -2$. Langs venstre side ($x = 0$) og langs bunnen ($y = 0$) er verdien 0 (se figur 2.4 B).



Figur 3.4: Analytisk og numerisk løsning for problemet beskrevet i eksempel 3.1. Diskretiseringsgrad $n = 30$.

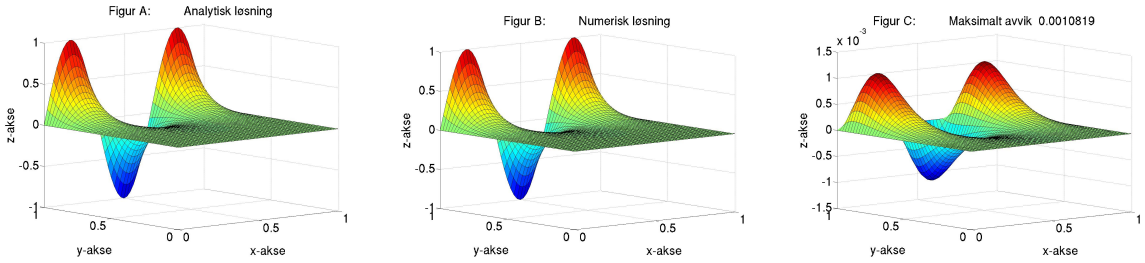
Den analytiske løsningen, for $N = 200$, og den numeriske løsningen er skissert ved siden av hverandre i figur 3.4. I figur 3.5 er differansen mellom den numeriske og den analytiske løsningen for ulik N skissert.



Figur 3.5: Viser differansen mellom de numerisk og analytisk løsning av problemet beskrevet i eksempel 3.1 for ulik N. Til venstre: $N = 20$. Til høyre: $N = 200$.

Eksempel 3.2 (V=0, H=0, B=0, T= sin(3πx)): La oss nå studere løsningen av det diskrete Laplace-problemet med samme Dirichlet-randbetingelser som i det kontinuerlige problemet vi studerte i eksempel 2.3. Dirichlet-randbetingelsene for toppen (T) er gitt ved funksjonen $T(x) = \sin(3\pi x)$ for $0 < x < 1$, og $V(y) = H(y) = B(x) = 0$. Løsningen er skissert i figur 2.4 B.

Vi skisserer den analytiske og numeriske løsningen i tillegg til differansen mellom de to løsningene i figur 3.6.



Figur 3.6: Analytisk og numerisk løsning er skissert i figur A og B. Figur C viser differansen mellom den numeriske og analytiske løsningen. Diskretiseringsgrad $n = 30$.

Kapittel 4

Egenskaper ved løsningen

I dette kapittelet vil vi studere forskjellige egenskaper ved løsningen av det diskrete Laplace-problemet med Dirichlet-randbetingelser i to dimensjoner som vi formulerte og studerte i forrige kapittel.

Innledningsvis vil vi beskrive en egenskap ved løsningen av Laplace-problemet som kort fortalt sier at dersom løsningen krummer oppover i en retning (for eksempel x-retning), vil den krumme nedover i den andre retningen (y-retning) i dette punktet. Deretter vil vi vie vår konsentrasjon til å studere løsningen av Laplace-problemet med Dirichlet-randbetingelser i forbindelse med følgende problem:

> **Treffpunktsproblemet:** *Gitt en endelig mengde punkter $P \subseteq \Omega_h$ med vilkårlige verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Bestem randverdier $g(x_i, y_j)$ for $(x_i, y_j) \in \partial\Omega_h$ slik at*
>
> $$v_g(x_i, y_j) - \alpha(x_i, y_j) = 0 \qquad \text{for } (x_i, y_j) \in P \tag{4.1}$$
>
> *der v_g er den unike løsningen av det diskrete Laplace-problemet med Dirichlet-randbetingelser g.*

Vi vil blant annet vise at dersom P består av 4 eller færre punkter vil det alltid eksistere en løsning av treffpunktsproblemet for alle verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Vi skal også se eksempler på at dette ikke alltid gjelder dersom P består av 5 eller flere punkter dersom disse punktene er plassert på en spesiell måte.

Vi bemerker at vi gjennom hele kapittelet vil anta at det ikke finnes begrensninger på randverdiene. I praktiske tilfeller vil ofte denne antagelsen ikke være gyldig. For eksempel vil vi måtte ta hensyn til det absolutte nullpunkt på -273,15°C, eller 0 K, dersom løsningen v_g representerer en temperaturfordeling.

4.1 Krumminger i løsningen

Vi begynner med å beskrive en egenskap som sier noe om hvordan løsningen krummer i x- og y-retning. I det analytiske tilfellet sier Laplace-ligningen at

$$\Delta u(x,y) = \frac{\partial^2 u}{\partial x^2} + \frac{\partial^2 u}{\partial y^2} = 0 \tag{4.2}$$

som impliserer at

$$\frac{\partial^2 u}{\partial x^2} = -\frac{\partial^2 u}{\partial y^2} \tag{4.3}$$

Dersom den andrederiverte i en retning er positiv vil dette medføre at den andrederiverte er negativ i den andre retningen. Dersom løsningen krummer oppover i en retning, vil den altså krumme nedover i den andre retningen. Det er også klart at i tilfeller der den andrederiverte er 0 i en retning, vil den også være 0 i den andre retningen. I denne seksjonen vil vi vise at dette også gjelder i det diskrete tilfellet. For å gjøre dette vil vi innføre litt ny notasjon, men først definerer vi strengt konvekse og strengt konkave funksjoner i én variabel.

Definisjon 4.1 (Strengt konveks funksjon i én variabel). *En funksjon $f : \mathbb{R} \to \mathbb{R}$ kalles strengt konveks dersom ulikheten*

$$f((1-\lambda)x + \lambda y) < (1-\lambda)f(x) + \lambda f(y) \tag{4.4}$$

holder for alle $x, y \in \mathbb{R}$ og alle $0 < \lambda < 1$.

Definisjon 4.2 (Strengt konkav funksjon i én variabel). *En funksjon $f : \mathbb{R} \to \mathbb{R}$ kalles strengt konkav dersom funksjonen $-f$ er strengt konveks.*

4.1.1 Diskret strengt konveks, strengt konkav og lineær i et punkt

I analogi med definisjonene over vil vi her innføre ny (ikke-standard) notasjon for å kunne omtale den diskrete løsningen $v_g(x_i, y_j)$ som henholdsvis diskret konveks, konkav og lineær i x- eller y-retning i et punkt (x_i, y_j).

Definisjon 4.3 (Konvekse, konkave og lineære funksjoner i et punkt:). *La U være en diskret delmengde av $\mathbb{R}^2$, og la $f : U \to \mathbb{R}$ være en diskret funksjon. Da gjelder følgende*

Konveks: *Funksjonen f er diskret strengt konveks i x-retning i punktet (x_i, y_j) dersom*

$$\frac{f(x_{i-1}, y_j) + f(x_{i+1}, y_j)}{2} > f(x_i, y_j) \tag{4.5}$$

Konkav: *Funksjonen f er diskret strengt konkav i x-retning i punktet (x_i, y_j) dersom*

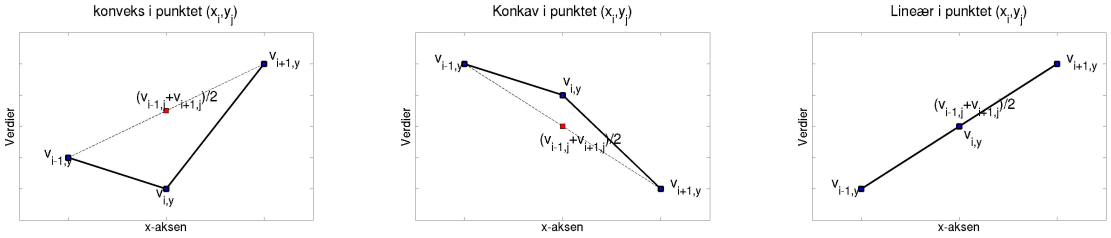
$$\frac{f(x_{i-1}, y_j) + f(x_{i+1}, y_j)}{2} \quad < \quad f(x_i, y_j) \tag{4.6}$$

Lineær: *Funksjonen f er diskret lineær i x-retning i punktet (x_i, y_j) dersom*

$$\frac{f(x_{i-1}, y_j) + f(x_{i+1}, y_j)}{2} \quad = \quad f(x_i, y_j) \tag{4.7}$$

Tilsvarende definisjoner gjelder for funksjonen f i y-retning i punktet (x_i, y_j) og dets nabopunkter (x_i, y_{j-1}) og (x_i, y_{j+1}).

De ulike tilfellene for punkter i x-retning er skissert i figur 4.1.



Figur 4.1: Skisserer verdien av punktene som funksjon av posisjonen i x-retning.

Vi kan nå formulere følgende teorem.

Teorem 4.4 (Ulik krumming i x- og y-retning). *La $v : \Omega_h \to \mathbb{R}$ være løsningen av det diskrete Laplace-problemet med Dirichlet-randbetingelser g. Da gjelder følgende for alle punkter $(x_i, y_j) \in \Omega_h$:*

1. *v er strengt konveks i x-retning i punktet $(x_i, y_j) \iff v$ er strengt konkav i y-retning i punktet (x_i, y_j)*
2. *v er strengt konkav i x-retning i punktet $(x_i, y_j) \iff v$ er strengt konveks i y-retning i punktet (x_i, y_j)*
3. *v er lineær i x-retning i punktet $(x_i, y_j) \iff v$ er lineær i y-retning i punktet (x_i, y_j)*

Bevis. Vi vil bevise de tre implikasjonene ($\Rightarrow$) etter tur. De motsatte implikasjonene kan ($\Leftarrow$) bevises på tilsvarende måte, og vi vil derfor utelate detaljer om dette her.

Siden v er løsningen av Laplace-problemet med Dirichlet-randbetingelser følger det at

$$4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1} = 0 \tag{4.8}$$

Ved å flytte over og dividere hele uttrykket med 2 får vi følgende uttrykk

$$\frac{v_{i,j-1} + v_{i,j+1}}{2} = 2v_{i,j} - \frac{v_{i-1,j} + v_{i+1,j}}{2} \tag{4.9}$$

1. **Konveks i x-retning $\Rightarrow$ konkav i y-retning:** Anta først at $v_{i,j}$ er konveks i x-retning. Per definisjon (4.5) følger det da at

   $$\frac{v_{i-1,j} + v_{i+1,j}}{2} > v_{i,j}$$

   Men det medfører at

   $$\frac{v_{i,j-1} + v_{i,j+1}}{2} = 2v_{i,j} - \frac{v_{i-1,j} + v_{i+1,j}}{2} < 2v_{i,j} - v_{i,j} = v_{i,j}$$

   Vi har dermed vist at

   $$\frac{v_{i,j-1} + v_{i,j+1}}{2} < v_{i,j}$$

   som per (4.6) viser at $v_{i,j}$ er konkav i y-retning.

2. **Konkav i x-retning $\Rightarrow$ konveks i y-retning:** La oss nå anta at $v_{i,j}$ er konkav i x-retning. Fra (4.6) følger det nå at

   $$\frac{v_{i-1,j} + v_{i+1,j}}{2} < v_{i,j}$$

   Men det vil medføre at

   $$\frac{v_{i,j-1} + v_{i,j+1}}{2} = 2v_{i,j} - \frac{v_{i-1,j} + v_{i+1,j}}{2} > 2v_{i,j} - v_{i,j} = v_{i,j}$$

   som per (4.5) viser at $v_{i,j}$ er konveks i y-retning.

3. **Lineær i x-retning $\Rightarrow$ lineær i y-retning:** Dersom $v_{i,j}$ er lineær i x-retning følger det fra (4.7) at

   $$\frac{v_{i-1,j} + v_{i+1,j}}{2} = v_{i,j}$$

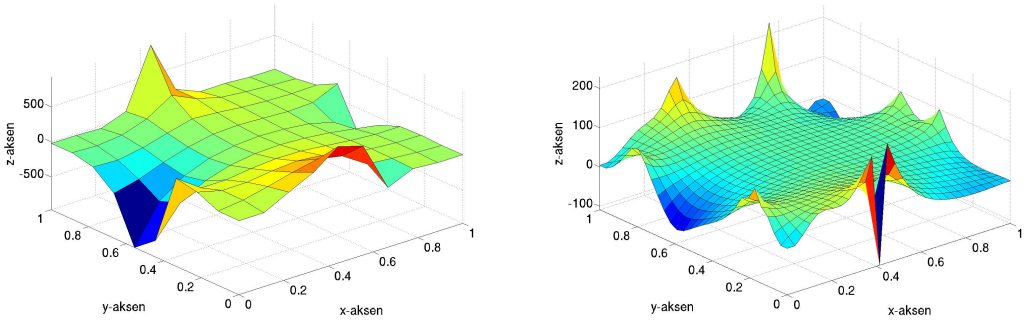
   noe som vil medføre at

$$\frac{v_{i,j-1} + v_{i,j+1}}{2} = 2v_{i,j} - \frac{v_{i-1,j} + v_{i+1,j}}{2} = 2v_{i,j} - v_{i,j} = v_{i,j}$$

og vi har dermed vist at $v_{i,j}$ også er lineær i y-retning.

□

I figur 4.2 er et par løsninger av Laplace-problemet med forskjellige Dirichlet-randbetingelser skissert. Vi observerer i henhold til teorem 4.4 at der løsningen krummer oppover i en retning, krummer den nedover i den andre retningen.



Figur 4.2: Skisse av løsningen v_g av Laplace-problemet med forskjellige randverdier g, og ulik diskretiseringsgrad n.

Definisjon 4.5 (Diskret strengt lokalt maksimum og minimum). *La U være en diskret delmengde av $\mathbb{R}^n$, og la $f : U \to \mathbb{R}$ være en diskret funksjon. Vi sier at $a \in U$ er et diskret strengt lokalt maksimum dersom det eksisterer et nabolag N til a slik at*

$$f(a) > f(u) \qquad \text{for alle} \quad u \in N \setminus \{a\} \tag{4.10}$$

Tilsvarende sier vi at $b \in U$ er et diskret strengt lokalt minimum dersom det eksisterer et nabolag N til b slik at

$$f(b) < f(u) \qquad \text{for alle} \quad u \in N \setminus \{b\} \tag{4.11}$$

Med denne definisjonen får vi følgende resultat som en direkte følge av 4.4.

Korollar 4.6 (Diskret strengt lokalt maksimum- og minimumsprinsipp). *Løsningen v av Laplace-problemet med Dirichlet-randbetingelser har ingen diskrete strengt lokale maksimum- eller minimumspunkter i det indre av området, Ω_h.*

Bevis. Anta for motsigelse at det finnes et lokalt maksimumspunkt $(x_i, y_j) \in \Omega_h$. Det medfører at nabopunktene $v_{i-1,j}$, $v_{i+1,j}$, $v_{i,j-1}$ og $v_{i,j+1}$ alle vil ha lavere verdi enn $v_{i,j}$, og dermed må $v_{i,j}$ være konveks i både x- og y-retning. Fra teorem 4.4 vet vi at det er umulig og antagelsen vår må være feil, og resultatet følger. □

4.2 Plan

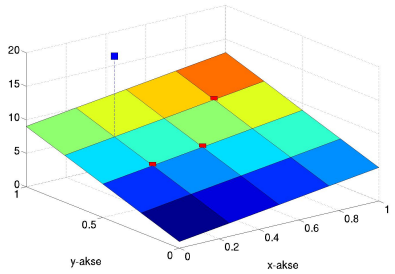
Et spesialtilfelle av løsninger v av Laplace-problemet med Dirichlet-randbetingelser er løsninger som kan beskrives ved et plan.

Teorem 4.7. *Gitt et diskret plan p definert ved $p(x_i, y_j) = ax_i + by_j$ for $(x_i, y_j) \in \Omega_h$ der $a, b \in \mathbb{R}$. Da finnes det Dirichlet-randbetingelser $g(x_i, y_j) = ax_i + by_j$ for $(x_i, y_j) \in \partial\Omega_h$ slik at $p = v_g$, der v_g er løsningen av det diskrete Laplace-problemet.*

Teorem 4.4 er selvsagt oppfylt for denne typen løsninger, da hvert punkt $v_{i,j}$ er lineært i x- og y-retning.

Fra elementær geometri vet vi at for tre (eller færre) vilkårlig valgte punkter vil det alltid finnes et plan som skjærer disse punktene. Vi vet derfor at dersom $|P|$, antall treffpunkter, er 3 vil det alltid eksistere en løsning av treffpunktsproblemet. Det vil si at det alltid vil finnes randverdier g slik at løsningen av Laplace-problemet v_g tilfredsstiller (4.1). Dette gjelder for vilkårlig verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$.

Dersom antall treffpunkter er 4, vil det ikke generelt eksistere et plan som skjærer disse punktene. Et eksempel på dette er vist i figur 4.3.



Figur 4.3: Viser et plan som skjærer tre gitte punkter, markert med røde bokser. Planet skjærer ikke det fjerde punktet, markert med en blå boks.

4.3 Løsninger av treffpunktsproblemet (4.1)

Dersom vi ikke begrenser randverdiene g til å være lineært fordelt er det rimelig å forvente at vi kan finne randverdier som gir løsninger v_g som skjærer flere enn 3 generelle punkter med vilkårlige verdier.

4.3.1 4 treffpunkter

I teorem 4.8 viser vi at dette er tilfelle. Teoremet sier at dersom antall treffpunkter er 4 eller færre, vil det alltid finnes randverdier g slik at det eksisterer en løsning av treffpunktsproblemet. Det vil si slik at løsningen v_g sammenfaller fullstendig med verdiene $\alpha(x_i, y_j)$ i punktene $(x_i, y_j) \in P$, for vilkårlig verdier gitt ved funksjonen $\alpha : P \to \mathbb{R}$.

Teorem 4.8 (Treffpunktsproblemet har alltid en løsning for $|P| \leq 4$). *La $P \subseteq \Omega_h$ bestå av maksimalt 4 forskjellige tilfeldig valgte treffpunkter med verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Da finnes Dirichlet-randbetingelser g slik at løsningen v_g av Laplace-problemet også tilfredsstiller (4.1).*

Bevis. Vi må vise at det eksisterer en løsning av følgende ligningssystem

$$\begin{aligned} \Delta_h v_g(x_i, y_j) &= 0 && \text{for } (x_i, y_j) \in \Omega_h \\ v_g(x_i, y_j) &= \alpha(x_i, y_j) && \text{for } (x_i, y_j) \in P \end{aligned} \tag{4.12}$$

der Δ_h er fem-punkts-operatoren som beskrevet tidligere (3.1) og Ω_h er det indre av det diskretiserte kvadratiske området.

For å vise at det eksisterer en løsning av ligningssystemet (4.12) vil vi skrive det på matriseform

$$\begin{bmatrix} \mathbf{A} & \mathbf{N} \\ \mathbf{C} & \mathbf{0} \end{bmatrix} \begin{bmatrix} \mathbf{x} \\ \mathbf{z} \end{bmatrix} = \begin{bmatrix} \mathbf{0} \\ \boldsymbol{\alpha} \end{bmatrix} \tag{4.13}$$

Vi har foretatt et variabelskifte slik at variablene $\mathbf{x}$ og $\mathbf{z}$ representerer henholdsvis punktene i det indre av området, Ω_h, og punktene langs randa, $\partial\Omega_h$. Høyresiden i uttrykket vil betegnes $\mathbf{b}$ og er en vektor sammensatt av $\mathbf{0}$-vektoren og en vektor $\boldsymbol{\alpha}$ der de n^2 første elementene er 0 og de fire gjenværende elementene $\boldsymbol{\alpha}$ er gitt ved $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$.

Vi vil la $\mathbf{M}$ betegne blokkmatrisen bestående av de fire matrisene $\mathbf{A}$, $\mathbf{N}$, $\mathbf{C}$ og $\mathbf{0}$. Her er $\mathbf{A}$ Poisson-matrisen som definert i (3.8) med dimensjon $n^2 \times n^2$. Matrisen $\mathbf{N}$ har dimensjon $n^2 \times 4n$, og består av rader med opp til to elementer med verdi -1. De resterende elementene er 0. Hver kolonne består av kun ett ikke-null element. Matrisen $\mathbf{C}$ er en $4 \times n^2$ matrise med ett ikke-null element i hver rad. Dette ikke-null elementet forekommer i forskjellige rader som følge av at de gitte punktene i P er forskjellige.

Lineært uavhengige rader: Vi ønsker å vise at radene i matrisen $\mathbf{M}$ er lineært uavhengige. Da vet vi fra lineær algebra [7] at det underdeterminerte ligningssystemet (4.13) har en løsning for enhver $\mathbf{b} \in \mathbb{R}^m$. I så tilfelle vil det eksistere en løsning av ligningssystemet for vilkårlige verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ og det viser teoremet.

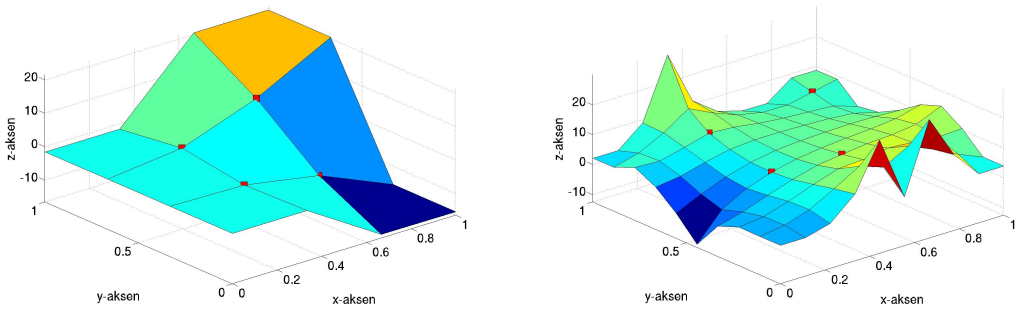
La $r = [r_A \; r_N]$ betegne en rad i matrisen $\mathbf{[A \; N]}$, der r_A betegner elementene i $\mathbf{A}$ og r_N betegner elementene i $\mathbf{N}$. Som følge av at hver kolonne i $\mathbf{N}$ kun inneholder ett ikke-null element vil raden r ikke kunne uttrykkes som lineærkombinasjon av noen av de andre radene i $\mathbf{M}$ dersom r_N består av ikke-null elementer, og disse radene er dermed lineært uavhengige.

La oss nå se nærmere på de resterende radene i **[A N]**. I hver av de resterende radene i **[A N]** består r_A av 5 ikke-null elementer, der to rader maksimalt har to felles ikke-null elementer. Grunnet strukturen i Poisson-matrisen vil ingen rad kunne uttrykkes som en lineærkombinasjon av de resterende radene. Dersom vi forsøker vil det alltid være minst 5 elementer som ikke "passer". Derfor vil raden heller ikke kunne uttrykkes som en lineærkombinasjon av radene i **[A N]** og radene i **[C 0]** så lenge **[C 0]** kun består av 4 eller færre rader.

Dermed er det vist at radene i matrisen $\mathbf{M}$ er lineært uavhengige, som igjen viser at det eksisterer en løsning av ligningssystemet (4.12), og teoremet er bevist.

□

I figur 4.4 er to løsninger v_g av Laplace-problemet med forskjellig Dirichlet-randbetingelser g som i tillegg tilfredsstiller (4.1) skissert.



Figur 4.4: Skisse av løsningen v_g av Laplace-problemet med forskjellige randverdier g, og ulik diskretiseringsgrad n.

4.3.2 5 treffpunkter

Vi har vist at dersom $|P| \leq 4$ vil det alltid eksistere en løsning av treffpunktsproblemet (4.1). Vi vil altså kunne finne randverdier g slik at løsningen v_g av Laplace-problemet vil treffe disse punktene for vilkårlige verdier α. I teorem 4.9 vil vi vise at dette ikke gjelder når P består av 5 eller flere punkter.

Teorem 4.9. *Gitt en vilkårlig mengde P bestående av 5 eller flere treffpunkter. Da er det ikke alltid mulig å finne randverdier g slik at løsningen v_g av Laplace-problemet også tilfredsstiller (4.1) for alle verdier $\alpha(x_i, y_j) \in P$.*

Bevis. Det er nok å vise at det finnes en mengde punkter $P \subset \Omega_h$ med verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ slik at det ikke eksisterer en løsning av treffpunktsproblemet.

La for eksempel P bestå av 5 punkter som er fordelt slik at et av dem har de fire andre som nærmeste naboer. La punktet i midten være gitt verdien $\alpha = 5$, og la nabopunktene være gitt verdier $\alpha = 2$. Problemet er illustrert til venstre i figur 4.5.

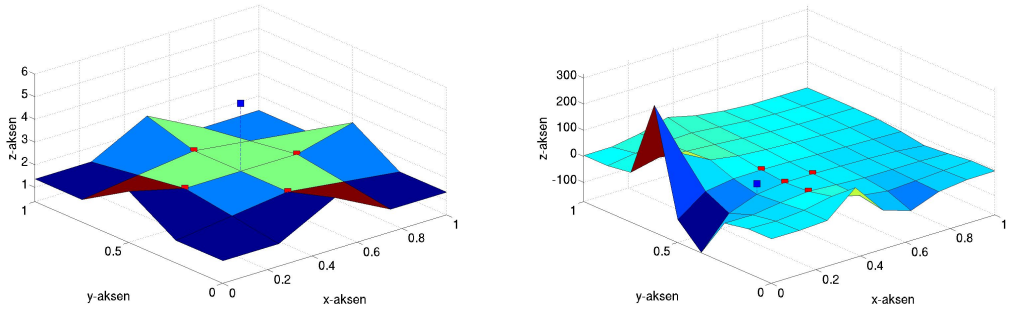
En løsning av treffpunktsproblemet er en løsning av det diskrete Laplace-problemet og følgende må dermed være tilfredsstilt

$$\Delta_h v(x_i, y_j) \quad = \quad 0 \qquad \text{for alle} \quad (x_i, y_j) \in \Omega_h \tag{4.14}$$

Det medfører at verdien i punktet i midten må være lik gjennomsnittet av nabopunktene. Men gjennomsnittet av nabopunktene er $2 \neq 5$, og resultatet følger.

□

I figur 4.5 er løsninger v_g av Laplace-problemet med to forskjellige Dirichlet-randbetingelser g skissert. Løsningene tilfredsstiller ikke treffpunktsproblemet for punktene markert med en blå boks.



Figur 4.5: Skisse av løsningen v_g av Laplace-problemet med forskjellige randverdier g. Figuren til venstre viser løsningen med Dirichlet-randverdier som beskrevet i beviset av teorem 4.9. Punktene markert med en boks angir verdien α for punktene i P. Punktene som ikke "treffes" av løsningen er markert med en blå boks.

4.3.3 Maksimalt antall treffpunkter

Vi har vist at dersom $|P| \geq 5$ finnes det fordelinger av punkter i P som er slik at det ikke eksisterer en løsning av treffpunktsproblemet for vilkårlige verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Det vil likevel finnes mange fordelinger av punkter P slik at det eksisterer en løsning av treffpunktsproblemet, selv når P består av langt flere punkter. I teorem 4.10 gis en øvre grense for hvor mange punkter P kan bestå av samtidig som det eksisterer en løsning av treffpunktsproblemet.

Teorem 4.10. *La en delmengde P av Ω_h være gitt. Anta at det, for vilkårlige verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$, eksisterer Dirichlet-randverdier g slik at løsningen v_g av Laplace-problemet tilfredsstiller (4.1). Da gjelder følgende:*

$$|P| \leq 4n \tag{4.15}$$

der n^2 angir antall punkter i det indre av det kvadratiske området, Ω_h.

Bevis. Igjen vil vi studere ligningsystemet (4.12), og vise at det ikke eksisterer en løsning av systemet for $|P| > 4n$. Vi formulerer følgende utvidede blokkmatrise

$$\left[\mathbf{M} \;\middle|\; \mathbf{b}\right] = \left[\begin{array}{cc|c} \mathbf{A} & \mathbf{N} & \mathbf{0} \\ \mathbf{C} & \mathbf{0} & \boldsymbol{\alpha} \end{array}\right] \tag{4.16}$$

der matrisen i blokkmatrisen er definert som i beviset av teorem 4.8.

Fra lineær algebra [7] er det kjent at et lineært ligningssystem er konsistent, det vil si at det finnes en unik løsning eller uendelig mange løsninger, hvis og bare hvis den høyre kolonnen i den utvidede matrisen **[M | b]** ikke er en pivot-kolonne - det vil si hvis og bare hvis en trappeform[1] av den utvidede matrisen ikke har rader på formen

$$\begin{bmatrix} 0 & \dots & 0 & b \end{bmatrix} \qquad \text{der } b \neq 0 \tag{4.17}$$

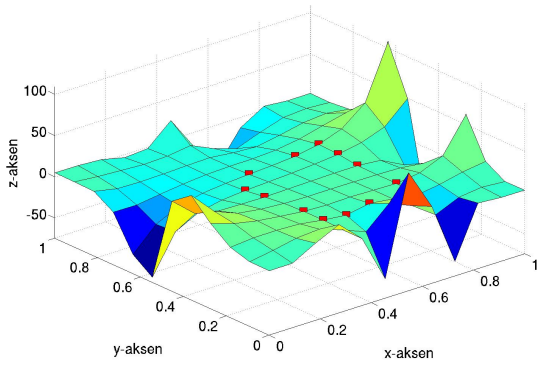
Matrisen $\mathbf{M}$ har dimensjon $n^2 + |P| \;\times\; n^2 + 4n$. Dersom $|P| > 4n$ vil matrisen $\mathbf{M}$ ha flere rader enn kolonner, den reduserte trappeformen av den utvidede matrisen **[M | b]** må bestå av minst en rad på formen

$$\begin{bmatrix} 0 & \dots & 0 & b \end{bmatrix} \tag{4.18}$$

der b er en lineærkombinasjon av elementene i $\boldsymbol{\alpha}$. Da finnes det verdier $\boldsymbol{\alpha}$ slik at $b \neq 0$. Ligningsystemet er følgelig inkonsistent, og det beviser teoremet.

□

Vi observerer at det maksimale antallet punkter i P er det samme som problemets frihetsgrad som er gitt ved antall randpunkter. Dette følger fra teorem 3.4 som sier at Laplace-problemet med Dirichlet-randbetingelser g har unik løsning v_g. Av den grunn bestemmes punktene i det indre av området, Ω_h, entydig av randverdiene og kun disse er "virkelige" variable.



Figur 4.6: Skisse av løsning av treffpunktsproblemet der punktene i P er fordelt slik at det finnes et rektangel slik at alle punktene i P ligger på randa av dette rektangelet.

[1] *eng.*: echelon form

4.3.4 Et spesialtilfelle

Til slutt i dette kapittelet vil vi undersøke noen egenskaper ved et spesialtilfelle av treffpunktsproblemet. Vi vil studere et problem der punktene i mengden P har en fordeling som er slik at det finnes en konveks mengde K slik at alle punktene i P ligger på randa denne mengden. I henhold til [3] definerer vi konvekse mengder på følgende måte

Definisjon 4.11 (Konveks mengde). *En mengde $K \subseteq \mathbb{R}^n$ er konveks dersom $(1-\lambda)x_1 + \lambda x_2 \in KC$ når $x_1, x_2 \in K$ og $0 \leq \lambda \leq 1$.*

Teorem 4.12 (Treffpunkter langs randa av en konveks mengde). *La P bestå av punkter som er fordelt slik at det finnes en konveks mengde $K \subseteq \Omega_h$ slik at alle punktene i P ligger langs randa av den konvekse mengden. Da eksisterer det en løsning av treffpunktsproblemet.*

Bevis. Fra [14] vet vi at det eksisterer en løsning av Laplace-problemet med Dirichlet-randbetingelser for åpne, sammenhengende og begrensede områder. Dette vil også gjelde med randverdier gitt ved $\alpha(x_i, y_j)$ for $(x_i, y_j) \in \partial K$. Vi har dermed vist at $\Delta_h v(x_i, y_j) = 0$ for $(x_i, y_j) \in K$. Det gjenstår å vise at vi kan finne verdier $v(x_i, y_j)$ for $(x_i, y_j) \in \Omega_h \setminus K$, slik at $\Delta_h v(x_i, y_j) = 0$ også for $(x_i, y_j) \in \Omega_h \setminus K$.

Vi lar p^j betegne punktene i P. Punktene er sortert slik at p^j har p^{j-1} og p^{j+1} som nabopunkter langs randa av området, ∂K. I tillegg er p^1 og p^N nabopunkter langs randa, der $N = |P|$. Videre lar vi Q^j betegne nabopunktene til p^j slik at $Q^j \subseteq \Omega_h \setminus \{K \cup \partial K\}$. Videre velger vi $p^1 \in P$ slik at minst ett av nabopunktene $q \in Q^1$, er slik at det eneste nabopunktet til q som ligger i P er dette punktet p^1.

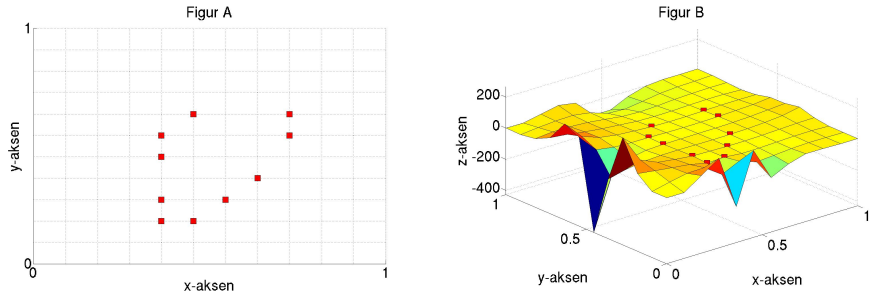
Punktet $p^2 \in P$ er et av punktene som ligger som nabopunkt til p^1 langs ∂K. Nå bestemmer vi verdier $v(x_i, y_j)$ for $(x_i, y_j) \in Q^2$, slik at $\Delta_h v(x_i, y_j) = 0$ for $(x_i, y_j) = p^2$. Siden K er konveks vil nabopunktet langs ∂K, p^3, ha nabopunkter Q^3 som ikke ligger i Q^2, og vi kan derfor bestemme verdier $v(x_i, y_j)$ for punktene $(x_i, y_j) \in Q^3 \setminus Q^2$, slik at $\Delta_h v(x_i, y_j) = 0$ for $(x_i, y_j) = p^3$.

Vi kan fortsette på samme måte for alle punktene p^4 ... p^N. Siden vi har valgt p^1 slik at minst ett av nabopunktene $q \in Q^1$, er slik at det eneste nabopunktet til q som ligger i P er punktet p^1, vil $Q^1 \setminus \{Q^2 \cup Q^N\} \neq \emptyset$. Vi kan derfor bestemme verdier $v(x_i, y_j)$ for punktene $(x_i, y_j) \in Q^1 \setminus \{Q^2 \cup Q^N\}$, slik at $\Delta_h v(x_i, y_j) = 0$ for $(x_i, y_j) = p^1$.

Vi har dermed angitt verdi i alle nabopunktene Q til P. På tilsvarende måte kan vi angi verdi for nabpounktene til Q, og således "arbeide oss utover" inntil vi når randa $\partial\Omega_h$.

□

Et eksempel på en fordelingen som beskrevet i teorem 4.12 er skissert i figur 4.7.



Figur 4.7: Til venstre er en fordeling av punktene P vist. Til høyre er løsningen for gitte verdier α skissert.

4.4 Behov for optimering

I dette kapittelet har vi sett at det i mange tilfeller ikke vil være mulig å finne randverdier g slik at løsningen v_g av Laplace-problemet tilfredsstiller (4.1) for vilkårlige verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Dersom vi ikke klarer å finne en løsning som tilfredsstiller (4.1) vil vi ofte være interessert i å finne de randverdiene g som gir en løsning v_g av Laplace-problemet som er en god tilnærming til de gitte verdiene $\alpha(x_i, y_j)$ for treffpunktene $x_i, y_j) \in P$. For å finne disse randverdiene vil vi ta i bruk teknikker og avanserte algoritmer fra et felt innenfor matematisk optimering kalt lineær optimering. I kapittel 6 vil vi formulere et optimeringsproblem for å løse problemet vi nettopp beskrev. Først vil vi i kapittel 5 gi en kort innføring i lineær optimering.

Kapittel 5

Lineær optimering

For the first time in history, managers were given a powerful and practical method of formulating and comparing extremely large numbers of interdependent alternative courses of action to find one that was optimal.

Arthur F. Veinott Jr., Professor ved Stanford University

I dette kapittelet vil vi gi en svært kort innføring i en viktig felt innenfor matematikk kalt lineær optimering. Ved bruk av enkle eksempler vil vi illustrere viktige konsepter og anvendelser. For detaljer og en grundig innføring til emnet henvises det til [4, 15], som mye av denne innføringen bygger på. Innledningsvis vil vi nevne et par hverdagslige eksempler der lineære optimeringsproblemer oppstår.

Eksempel 5.1 (Ukentlig diett for en familie): En familie på fem ønsker å planlegge en ukentlig diett. De har alle behov for en gitt mengde kalorier hver dag. I tillegg ønsker å spise sunt, og de har derfor satt opp begrensninger på fettinnholdet, og de sørger for at maten blant annet inneholder nok vitaminer og proteiner. De har innhentet priser på ulike matvarer, og ønsker å sette opp en diett som minimere familiens matutgifter samtidig som dietten tilfredsstiller betingelsene som sikrer et sunt kosthold.

Eksempel 5.2 (Produksjonsplanlegging på møbelfabrikk): Faren i familien jobber på en møbelfabrikk som produserer ulike møbler. Produksjonsprosessen er todelt. Først skal produktene snekres i snekkerverkstedet, og deretter sendes videre til ferdigbehandling der de lakkes, males eller vokses. Produksjonslederen er kjent med hvor lang tid de ulike verkstedene bruker på produksjonen av de ulike produktene, hvor mye tid de ulike verkstedene har tilgjengelig samt fortjenesten fabrikken har per produkt. Vi antar at fabrikken får solgt alle produktene den produserer. Produksjonslederen ønsker å planlegge produksjonsprosessen slik at fortjenesten maksimeres innenfor de gitte begrensningene.

Eksemplene over er hentet fra George B. Dantzigs innføringsbok i lineær optimering [4]. Sammen med John von Neumann og Leonid Kantorovich grunnla Dantzig lineær opti-

mering, også kalt lineær programmering, på 1940-tallet. Lineær optimering er teknikker for å maksimere eller minimere lineære objektivfunskjoner, også kalt målfunksjoner, i mange variable, med begrensninger som er lineære ulikheter og likheter [4]. Feltet fikk sitt gjennomslag i 1947 da Dantzig formulerte simplex-metoden som er en metode for å løse lineære optimeringsproblemer (LP-problemer). Feltet har mange ikke-lineære og heltallige utvidelser som går inn under betegnelsen matematisk optimering.

Lineær optimering har utallige praktiske anvendelser og brukes innenfor ulike grener av industri og næringsliv. Frakt og transportselskaper bruker lineær optimering i ruteplanlegging og effektiv bruk av personell. Internett- og telekommunikasjonsselskaper bruker det når de skal "rute" datatrafikk. Fabrikanter kan bruke lineær optimering for å finne beste og kanskje billigste blanding i et kompositttmateriale. I tillegg kan vi nevne at lineær optimering blant annet brukes innenfor områder som økonomi og porteføljeforvaltning, elektronisk industri, reklame og arkitektur, for å nevne noen [4, 12, 15].

Uten å tenke over det forsøker vi ofte å løse lineære optimeringsproblemer i dagliglivet. Et eksempel er når vi skal gå eller kjøre mellom to steder, og ønsker å minimere avstanden. Å finne den korteste veien kan være svært vanskelig og mange har derfor en GPS i bilen, som løser problemet effektivt.

5.1 Formulering av et lineært optimeringsproblem

I dette avsnittet skal vi se hvordan vi kan formulere lineære optimeringsproblemer, ofte kalt LP-problemer, matematisk. Et generelt LP-problem er på formen

$$\begin{array}{lrcll} \max & \sum_{j=1}^{n} c_j x_j & & & \\ \text{f.a.} & \sum_{j=1}^{n} a_{i,j} x_j & \leq & b_i & \text{for } i = 1, 2, ..., m \\ & x_j & \geq & 0 & \text{for } j = 1, 2, ..., n \end{array} \tag{5.1}$$

der den øverste linja $\sum_{j=1}^{n} c_j x_j$ kalles objektivfunksjonen. I andre linje ser vi betingelsene gitt ved lineære ulikheter, og nederst er det en linje som sikrer at variablene er ikkenegative. 'f.a.' er en forkortelse for 'forutsatt at'.

Ofte skriver vi LP-problemet (5.1) på matriseform

$$\begin{array}{lrcl} \max & c^T x & & \\ \hline \text{f.a.} & Ax & \leq & b \\ & x & \geq & 0 \end{array} \tag{5.2}$$

La oss nå se nærmere på eksempel 5.2 over om produksjonsplanlegging på møbelfabrikken, og formulere dette matematisk som et LP-problem på standardform.

Eksempel 5.3 (Produksjonsplanlegging som LP-problem): Vi antar at møbelfabrikken produserer tre forskjellige produkter; stoler, bord og skrivepulter. Vi lar x_j angi antall produserte enheter av produkt j. c_j vil være fortjenesten per solgt produkt av type j, og vi antar at $c_1 = 200$ kr, $c_2 = 500$ kr og $c_3 = 300$ kr. Videre husker vi at produksjonsprosessen var todelt, og vi lar maksimal tidskapasitet på hver av de to verkstedene være

gitt ved b_i, der $b_1 = 32$ timer og $b_2 = 16$ timer. Til slutt lar vi $a_{i,j}$ angi hvor lang tid de ulike verkstedene (i) bruker på produksjonen av de ulike produktene (j). På produkt 1, 2 og 3 bruker snekkerverkstedet ($a_{1,j}$) henholdsvis 2, 4 og 3 timer, mens det tar henholdsvis 1, 4 og 1 timer for ferdigbehandling ($a_{2,j}$). Oppsummert gir dette følgende problem:

$$\begin{array}{lrcrcrcl} \max & 200x_1 & + & 500x_2 & + & 300x_3 & & \\ \hline \text{f.a.} & 2x_1 & + & 4x_2 & + & 3x_3 & \leq & 32 \\ & 1x_1 & + & 4x_2 & + & 1x_3 & \leq & 16 \\ & & & & & x_1, x_2, x_3 & \geq & 0 \end{array}$$

Vi observerer at $x_1, x_2, x_3 \geq 0$ fordi fabrikken ikke kan produsere negative bord. I enkelte problemer, for eksempel i forbindelse med økonomiske problemer, vil denne begrensningen ikke være til stede.

Ved bruk av Simplex-metoden, som vi gir en kort forklaring av til slutt i dette kapittelet, finner vi løsningen av problemet. Problemet har en optimal løsning gitt ved $x_1 = 0$, $x_2 = 2$ og $x_3 = 8$. Dette viser at fabrikken maksimerer sin fortjeneste ved å produserer 0 stoler, 2 bord og 8 skrivepulter, noe som gir en optimal verdi, v^* på $300 \cdot 0 + 500 \cdot 2 + 300 \cdot 8 = 3400$.

5.1.1 Standardform

I henhold til Vanderbeis [15] notasjon vil vi omtale problemer på formen (5.1) og (5.2) som lineære optimeringsproblemer på standardform. LP-problemer kan formuleres på forskjellig måte, men ved passende omskrivninger kan alle LP-problemer formuleres på standardform. For eksempel kan minimeringsproblemer på formen $\min c^T x$ skrives som $\max -c^T x$. Ved å multiplisere med -1 kan betingelser på formen $\sum_j a_{i,j} x_j \geq b_i$ omformuleres til betingelser på formen

$$-\sum_j a_{i,j} x_j \leq -b_i.$$

Ligninger på formen $\sum_j a_{i,j} x_j = b_i$ kan formuleres som ulikheter ved å skrive ligningen som to ulikheter

$$\begin{aligned} \sum_j a_{i,j} x_j &\leq b_i \\ \sum_j a_{i,j} x_j &\geq b_i. \end{aligned}$$

Dersom betingelsen at $x_j \geq 0$ for alle j mangler, kan problemet skrives om ved å substituere x med $(x^+ - x^-)$, der $x^+, x^- \geq 0$.

5.2 Eksistens av optimal løsning av LP-problem

I eksempel 5.3 om produksjonsplanlegging fant vi en optimal løsning. Løsningen var både tillatt og begrenset, men som vi skal se i dette avsnittet gjelder ikke dette alle problemer.

Et av hovedresultatene i lineær optimering, ofte kalt "Fundamentalteoremet for lineær optimering", sier blant annet at dersom det ikke finnes optimal løsning av et problem, er problemet enten ikke tillatt eller ubegrenset. Her vil vi nøye oss med å illustrere teoremet med et par eksempler. For en detaljert beskrivelse og bevis av teoremet vil vi igjen henvise til [4, 15].

Eksempel 5.4 (Ikke tillatt løsning): Vi begynner med å se på et eksempel der ingen løsning er tillatt.

$$\begin{array}{lrcrcl} \max & 4x_1 & + & 2x_2 & & \\ \hline \text{f.a.} & x_1 & + & x_2 & \leq & 1 \\ & x_1 & + & & \geq & 2 \\ & & & x_1, x_2 & \geq & 0 \end{array}$$

Vi ser fra den andre betingelsen at $x_1 \geq 2$, men da kan ikke den øverste ulikheten tilfredsstilles uten at $x_2 < 0$ som vil medføre at den tredje betingelsen brytes. Det finnes altså ingen $x = (x_1, x_2)$ som oppfyller betingelsene, og problemet har derfor ingen løsning.

Eksempel 5.5 (Ubegrenset løsning): La oss nå se på et eksempel der løsningen er ubegrenset.

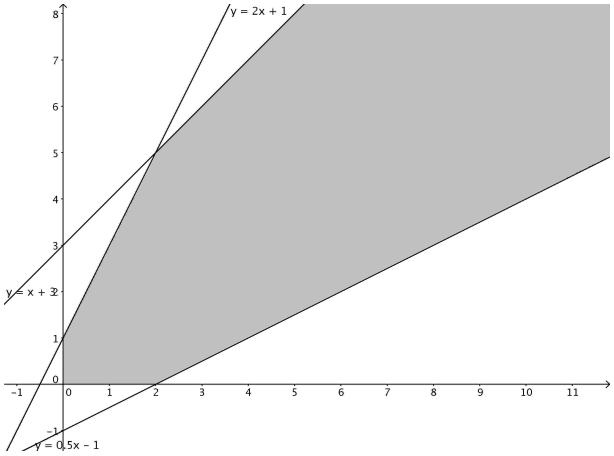
$$\begin{array}{lrcrcl} \max & 2x_1 & + & x_2 & & \\ \hline \text{f.a.} & -x_1 & + & x_2 & \leq & 3 \\ & -4x_1 & + & 2x_2 & \leq & 2 \\ & -\frac{1}{2}x_1 & + & x_2 & \geq & -1 \\ & & & x_1, x_2 & \geq & 0 \end{array}$$

I figur 5.1 ser vi det tillatte området (markert i grått). Betingelsene legger ikke begrensninger på objektivfunksjonen $2x_1 + x_2$, som dermed kan vokse ubegrenset.

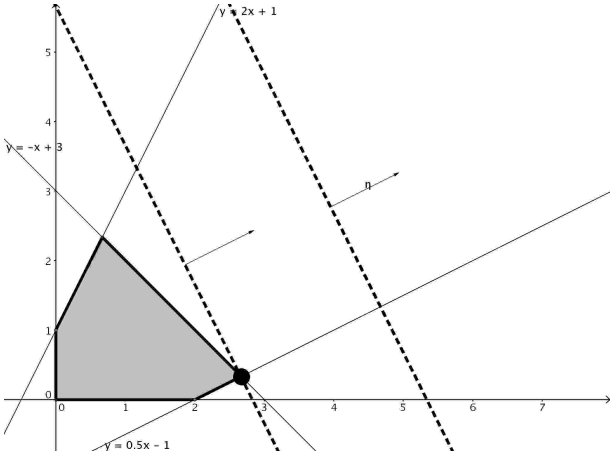
Eksempel 5.6 (Optimal løsning): Dersom vi endrer fortegnet foran den første variabelen i den øverste ulikheten i eksempelet over fra - til +, får vi følgende problem

$$\begin{array}{lrcrcl} \max & 2x_1 & + & x_2 & & \\ \hline \text{f.a.} & -x_1 & + & x_2 & \leq & 3 \\ & -4x_1 & + & 2x_2 & \leq & 2 \\ & -\frac{1}{2}x_1 & + & x_2 & \geq & -1 \\ & & & x_1, x_2 & \geq & 0 \end{array}$$

Dette problemet er både tillatt og begrenset, og har derfor optimal løsning.



Figur 5.1: Grafisk løsning av det ubegrensede LP-problemet beskrevet i eksempel 5.5. Området med tillatt løsning er markert i grått.



Figur 5.2: Grafisk løsning av LP-problemet beskrevet i eksempel 5.6. Området med tillatt løsning er markert i grått. Vi observerer at det eksisterer en optimal løsning av problemet.

5.2.1 Grafisk løsning

Problemer med to variable, som eksempelet over, kan løses grafisk. Vi kjenner objektfunksjonens retning, og kan tegne inn normalen (stiplet linje i figur 5.2). Vi tenker oss at vi flytter denne normalen nærmere det tillatte området, og finner optimal løsning i det punktet som først treffes av normalen. Løsningen ligger i det punktet av den tillatte mengden som befinner seg "lengst ut" i retningen til objektivfunksjonen. I dette eksempelet er løsningen $x_1 = \frac{8}{3}$ og $x_2 = \frac{1}{3}$.

Eksempel 5.7 (Uendelig mange optimale løsninger): En løsning er optimal dersom den gir optimal verdi. Et problem som har optimal løsning har enten en unik optimal løsning eller uendelig mange optimale løsninger [15]. Problemer med uendelig mange optimale løsninger er problemer der betingelsene er slik at den tillatte mengden har en side som står normalt på objektivfunksjonen. Problemet i eksempelet over hadde unik optimal løsning, men dersom vi endrer objektivfunksjonen til $\eta = -1 \cdot x_1 + 0 \cdot x_2$, og beholder betingelsene ellers slik de var, får vi et problem med uendelig mange optimale løsninger. I dette eksempelet vil optimal verdi være 0, og oppnås for $x_1 = 0$ og $x_2 \in [0, 1]$.

5.3 Algoritmer

Som vi har sett eksempler på over kan LP-problemer med to variable løses grafisk, men for å løse problemer med flere variable trenger vi andre metoder. Med unntak av de aller minste problemene med få variable og få betingelser løses LP-problemer ved hjelp av datamaskin. Det finnes mange ulike algoritmer for å løse LP-problemer. Her vil vi svært kort redegjøre for de to viktigste typene; Simplex-algoritmer og indrepunktsmetoder. For en grundig innføring i begge algoritmene henviser vi til [4].

5.3.1 Simplex-algoritmen

Som nevnt tidligere ble Simplex-algoritmen utviklet av George B. Dantzig i 1947. Simplex-algoritmen løser LP-problemer ved å konstruere en tillatt løsning, og deretter bevege seg langs randa av det tillatte området, fra et hjørne/ekstrempunkt til det neste, slik at objektivfunksjonens verdi er ikke-synkende. Etter et antall iterasjoner, kalt pivoteringer, vil objektivfunksjonen nå sin maksimumsverdi, og identifisere en optimal løsning [4].

Hvor effektivt algoritmen finner optimal løsning på et gitt problem avhenger av antall pivoteringer som gjøres før optimal løsning identifiseres. Simplex-algoritmen har vist seg å være svært effektiv for å løse store praktiske problemer, men for enkelte oppkonstruerte problemer (blant annet Klee og Minty-problemet) vokser antall pivoteringer eksponentielt med problemets dimensjon [4, 15].

5.3.2 Indrepunktsmetoder

Disse oppkonstruerte problemene som ikke lot seg løse effektiv ved hjelp av Simplex-algoritmen motiverte til å finne andre metoder. Et teoretisk gjennombrudd kom i 1979

da den russiske matematikeren L.G. Khachian fant en algoritme som i det verst tenkelige tilfelle var betydelig mer effektiv enn Simplex-algoritmen, og i 1984 fant N. Karmarkar en indrepunkstmetode som var betydelig mer effektiv enn det igjen.

I motsetning til den klassiske Simplex-algoritmen som beveger seg langs randa av området, beveger indrepunktsmetodene seg, som navnet tilsier, i det indre av det tillatte området. Metoden starter med en tillatt løsning, og forbedrer denne mens den beveger seg igjennom området.

Kapittel 6

Optimale randbetingelser

Med teoretisk bakgrunn fra de foregående kapitlene skal vi i resten av denne oppgaven vie vår konsentrasjon til å studere optimeringsproblemet som ble formulert allerede i oppgavens introduksjon. Problemet kombinerer løsning av det diskrete Laplace-problemet og lineær optimering. Som motivasjon, før vi studerer problemet i detalj, vil vi nevne en anvendelse innenfor fabrikasjon av metalliske materialer.

I fabrikasjon av metalliske materialer er varmebehandling en essensiell prosess. Gjennom oppvarming eller avkjøling av et materiale eller ved opphold ved bestemte temperaturer kan materialets egenskaper forandres. Blant annet kan økt formendringsevne, økt slitebestandighet og maskinell bearbeidbarhet oppnås. I tillegg kan egenspenninger i materialet fjernes [6, 9].

Ofte er det hensiktsmessig med ulike egenskaper i ulike deler av et materiale. For å tilstrebe de ønskede egenskapene kan materialet varmes opp eller kjøles ned slik at temperaturenfordelingen varierer gjennom materialet. Slik kan egenskapene i materialet forandres hensiktsmessig.

Vi lar en lang metallisk stav med kvadratisk tverrsnitt varmebehandles. Vi antar at en temperaturfordeling i tverrsnittet som vil gi optimale egenskaper er gitt. Kan vi varme opp eller kjøle ned staven langs randa slik at vi får nøyaktig denne temperaturfordelingen? Hvordan kan vi få en så god tilnærming til den gitte fordelingen som mulig?

6.1 Optimale randverdier for Laplace-problemet

En kontinuerlig versjon av optimeringsproblemet vi skal studere kan formuleres som følger: Gitt en delmengde S av Ω og en funksjon $\gamma(x, y) : S \to \mathbb{R}$, finn randverdier $g(x, y) : \partial\Omega \to \mathbb{R}$ som minimerer funksjonen

$$\Phi(g) := \int_S |u_g(x, y) - \gamma(x, y)| \, dx \, dy \tag{6.1}$$

forutsatt at $u_g(x, y)$ er den unike løsningen av det kontinuerlige Laplace-problemet (2.3) med Dirichlet randbetingelser $g(x, y)$.

Vi skal konsentrere oss om en approksimasjon til problemet over, og vil studere problemet definert for det diskretiserte kvadratiske området Ω_h, og den diskretiserte randa $\partial\Omega_h$, som forklart i kapittel 3. Det diskrete problemet kan da formuleres som følger

Optimeringsproblemet: *Gitt et endelig antall punkter $P \subseteq \Omega_h$ med verdier $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Bestem randverdier $g(x_i, y_j)$ slik at v_g er en optimal løsning av optimeringsproblemet*

$$\begin{array}{lrcll} \min & \displaystyle\sum_{(x_i,y_j)\in P} |v_g(x_i,y_j) & - & \alpha(x_i,y_j)| & \\ \hline \text{f.a.} & \Delta_h v_g(x_i,y_j) & = & 0 & (x_i,y_j) \in \Omega_h \\ & v_g(x_i,y_j) & = & g(x_i,y_j) & (x_i,y_j) \in \partial\Omega_h \end{array} \tag{6.2}$$

I optimeringsproblemet over minimeres summen av avvikene. Selv om vi i denne oppgaven utelukkende vil konsentrere oss om dette optimeringskriteriet, er det verdt å nevne at det i mange praktiske tilfeller vil være hensiktsmessig å benytte andre optimeringskriterier. Det vil for eksempel kunne være av større interesse å finne en løsning av Laplace-problemet der summen av det kvadratiske avviket er minimal, heller enn summen av avviket.

Vi kan tenke oss at det kvadratiske området Ω_h representerer tverrsnittet i metallstaven vi beskrev over, og at verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ representerer den ønskede temperaturfordelingen i tverrsnittet. Målet er å finne Dirichlet-randverdier g slik at løsningen av Laplace-problemet v_g har minimal avstand til de ønskede verdiene $\alpha(x_i, y_j)$ i punktene $(x_i, y_j) \in P$. La oss illustrere problemet med et konkret eksempel.

Eksempel 6.1 (Varmebehandling av en stav med kvadratisk tverrsnitt): Vi lar diskretiseringsgraden n være 3, og lar den ønskede temperaturfordelingen i tverrsnittet av en lang metallisk stav være gitt i 6 punkter, det vil si at $|P| = 6$. Verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ kan da illustreres ved følgende matrise

$$\begin{bmatrix} * & * & * & * & * \\ * & 2 & 1 & 3 & * \\ * & 0 & * & * & * \\ * & * & 5 & 2 & * \\ * & * & * & * & * \end{bmatrix}$$

der $*$ betegner punktene $(x_i, y_j) \in \bar{\Omega}_h \setminus P$ som ikke er gitt noen ønsket verdi $\alpha(x_i, y_j)$. Løsningen av optimeringsproblemet vil gi verdier g langs $\partial\Omega_h$, som gir den beste tilnærmingen til den ønskede temperaturfordelingen.

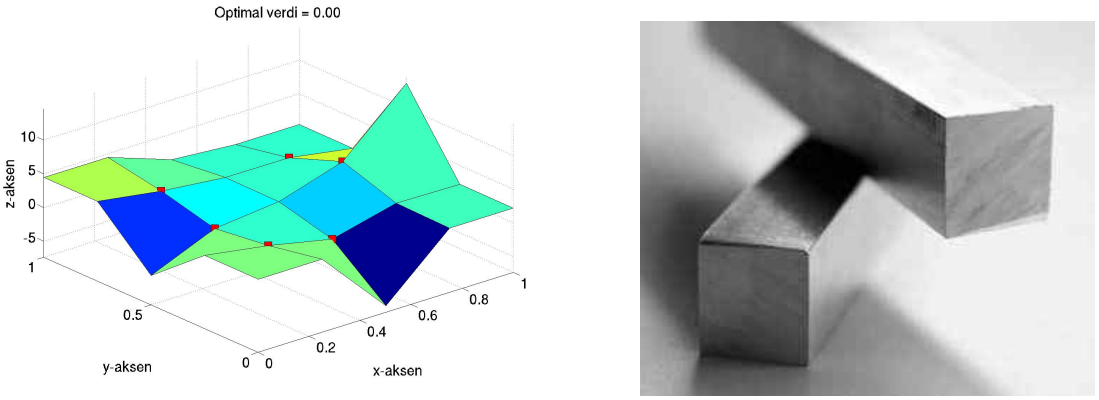
Ved å løse optimeringsproblemet finner vi følgende randverdier

$$\begin{bmatrix} 3.50 & 3.50 & -3.01 & 4.47 & 4.47 \\ 3.50 & & & & 4.47 \\ -6.36 & & & & 1.16 \\ 2.19 & & & & 0.47 \\ 2.19 & 2.19 & 13.63 & 0.47 & 0.47 \end{bmatrix}$$

Laplace-problemet med Dirichlet-randverdier g som skissert i matrisen over har følgende løsning, som altså er den optimale løsningen av optimeringsproblemet (6.2).

$$\begin{bmatrix} 3.50 & 3.50 & -3.01 & 4.47 & 4.47 \\ 3.50 & \mathbf{2.00} & \mathbf{1.00} & \mathbf{3.00} & 4.47 \\ -6.36 & \mathbf{0.00} & 2.01 & 2.04 & 1.16 \\ 2.19 & 2.34 & \mathbf{5.00} & \mathbf{2.00} & 0.47 \\ 2.19 & 2.19 & 13.63 & 0.47 & 0.47 \end{bmatrix}$$

Løsningen er skissert til venstre i figur 6.1. Vi observerer at løsningen v_g sammenfaller fullstendig med de gitte verdiene α i punktene i P· Den optimale verdien er dermed 0 og løsningen er det vi vil omtale som en eksakt løsning.



Figur 6.1: Til venstre er løsning v_g av optimeringsproblemet beskrevet i eksempel 6.1 skissert. Punktene markert med en rød boks viser gitte verdier for α. Til høyre vises et bilde av to aluminiumstaver med kvadratisk tverrsnitt.

6.1.1 LP-problem

Som følge av at absoluttverdi ikke er en lineær funksjon, er optimeringsproblemet (6.2) over ikke et lineært optimeringsproblem, LP-problem. Ved å innføre en hjelpevariabel δ kan vi likevel skrive om problemet til et LP-problem som kan løses ved hjelp av moderne algoritmer for å løse generelle LP-problemer.

Optimeringsproblemet kan skrives om på følgende måte: Vi minimerer $\sum_P \delta(x_i, y_j)$ forutsatt at $-\delta(x_i, y_j) \leq (v_g(x_i, y_j) - \alpha(x_i, y_j)) \leq \delta(x_i, y_j)$. Dette gir følgende

$$\begin{array}{ll} \min & \displaystyle\sum_{(x_i,y_j)\in P} \delta(x_i, y_j) \\ \hline \\ \text{f.a.} & \begin{array}{rcrcll} v_g(x_i, y_j) & - & \alpha(x_i, y_j) & \leq & \delta(x_i, y_j) & (x_i, y_j) \in P \\ -v_g(x_i, y_j) & + & \alpha(x_i, y_j) & \leq & \delta(x_i, y_j) & (x_i, y_j) \in P \\ & & \Delta_h v_g(x_i, y_j) & = & 0 & (x_i, y_j) \in \Omega_h \\ & & v_g(x_i, y_j) & = & g(x_i, y_j) & (x_i, y_j) \in \partial\Omega_h \end{array} \end{array} \tag{6.3}$$

som kun består av lineære betingelser og lineær objektivfunksjon. Problem 6.3 er derfor et LP-problem. Vi flytter α over til høyre side, og samler modelleringsvariablene v_g og δ på venstre side. Det gir følgende problem

$$\begin{array}{ll} \min & \displaystyle\sum_{(x_i,y_j)\in P} \delta(x_i, y_j) \\ \hline \\ \text{f.a.} & \begin{array}{rcrcll} v_g(x_i, y_j) & - & \delta(x_i, y_j) & \leq & \alpha(x_i, y_j) & (x_i, y_j) \in P \\ -v_g(x_i, y_j) & - & \delta(x_i, y_j) & \leq & -\alpha(x_i, y_j) & (x_i, y_j) \in P \\ & & \Delta_h v_g(x_i, y_j) & = & 0 & (x_i, y_j) \in \Omega_h \\ & & v_g(x_i, y_j) & = & g(x_i, y_j) & (x_i, y_j) \in \partial\Omega_h \end{array} \end{array} \tag{6.4}$$

Fra Fundamentalteoremet for lineær optimering (LP), skissert i kapittel 5.2, får vi nå følgende viktige resultat:

Teorem 6.1 (Eksistens av optimal løsning). *Det eksisterer alltid en optimal løsning av optimeringsproblemet (6.2).*

Bevis. Som vi har vist kan problem (6.2) skrives som et lineært optimeringsproblem, LP-problem (6.4). Fra Fundamentalteoremet for lineær optimering følger det at et LP-problem har optimal løsning dersom problemet er begrenset og det eksisterer en tillatt løsning. Det er lett å se at det alltid vil eksistere en tillatt løsning av (6.4). La for eksempel $v_g(x_i, y_j) = 0$ og la $\delta(x_i, y_j) = |\alpha(x_i, y_j)|$ for alle $(x_i, y_j) \in P$. Som følge av at problemet stammer fra et uttrykk med absoluttverdi-funksjonen er problemet i tillegg begrenset, og resultatet følger.

□

Vi vil la v^* betegne verdien av objektivfunksjonen innsatt den optimale løsningen. Denne verdien er et mål på hvor godt løsningen tilnærmer de gitte verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$, og vil omtales som optimeringsproblemets optimale verdi.

Som beskrevet i kapittel 5 kjenner vi effektive metoder for å løse LP-problemer, blant annet simplex-algoritmer og indrepunktsmetoder. Som følge av at vi har skrevet om optimeringsproblemet (6.2) til et LP-problem (6.4), kan vi benytte disse effektive metodene for å løse problemet. Før vi i kapittel 7 beskriver hvordan dette gjøres, vil vi vie vår oppmerksomhet til å studere ulike egenskaper og varianter av optimeringsproblemet (6.2).

6.2 Eksistens av eksakt løsning

Den optimale verdien v^* er et mål på hvor godt løsningen av optimeringsproblemet tilnærmer de ønskede verdiene α definert for den endelige mengden P. I noen tilfeller, blant annet i eksempel 6.1, vil løsningen sammenfalle fullstendig med de ønskede verdiene. I slike tilfeller er den optimale verdien 0, og vi kaller løsningen eksakt.

Definisjon 6.2 (Eksakte løsninger). *Vi vil omtale løsninger av optimeringsproblemet (6.2) som sammenfaller fullstendig med verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ for eksakte løsninger.*

Sammenhengen mellom eksakte optimale løsninger av optimeringsproblemet, optimal verdi v^*, og treffpunktsproblemet karakteriseres i teorem 6.3.

Teorem 6.3. *Følgende tre utsagn er ekvivalente*

1. *En optimal løsning v_g av optimeringsproblemet (6.2) er eksakt*
2. *Optimal verdi v^* av optimeringsproblemet (6.2) er 0*
3. *Det korresponderende treffpunktsproblemet (4.1) er konsistent*

Bevis. (***1***⇒***2***) La v_g være en eksakt løsning av optimeringsproblemet (6.2). Per definisjon 6.2 sammenfaller løsningen v_g fullstendig med verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Det betyr at $v_g(x_i, y_j) = \alpha(x_i, y_j)$, som impliserer at $|v_g(x_i, y_j) - \alpha(x_i, y_j)| = 0$ og den optimale verdien er dermed 0.

(***2***⇒***3***) La v_g være en optimal løsning av optimeringsproblemet (6.2), og la den optimale verdien være 0. Det medfører at $\sum_P |v_g(x_i, y_j) - \alpha(x_i, y_j)| = 0$. Grunnet tallverdifunksjonen medfører dette at $v_g(x_i, y_j) = \alpha(x_i, y_j)$ for hver $(x_i, y_j) \in P$. Da er v_g også en løsning av treffpunktsproblemet, og da er treffpunktsproblemet konsistent.

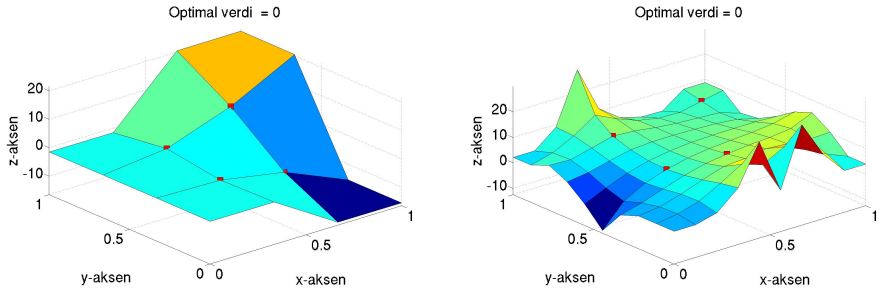
(***3***⇒***1***) La v_g være en løsning av treffpunktsproblemet. Da er v_g en løsning av Laplaceproblemet ($\Delta v_g = 0$) som i tillegg tilfredsstiller $v_g(x_i, y_j) = \alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Følgelig er v_g en eksakt optimal løsning av optimingsproblemet (6.2), per definisjon 6.2.

Vi har dermed vist at ***1***⇒***2***⇒***3***⇒***1***. Dette viser at uttrykkene ***1***, ***2*** og ***3*** er ekvivalente.

□

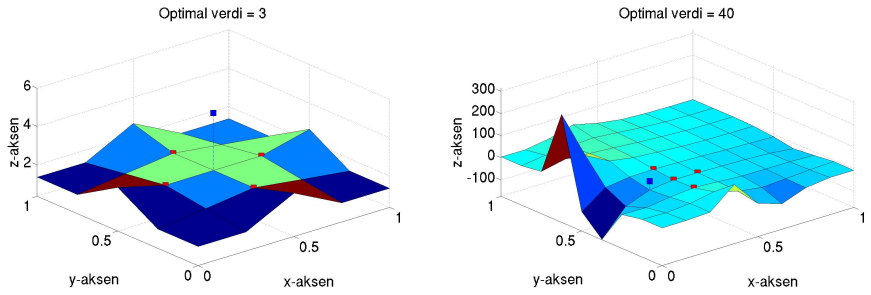
La oss illustrere teoremet over med et par eksempler.

Eksempel 6.2 (Optimal verdi = 0): I figur 4.4 i kapittel 4 viste vi to løsninger av treffpunktsproblemet. Løsningene sammenfaller fullstendig med de ønskede verdiene α for de fire punktene i P, og løsningene er dermed eksakte. Løsninger av det tilsvarende optimeringsproblemet vil dermed ha optimal verdi 0. Løsningene er skissert i figur 6.2.



Figur 6.2: Viser løsningen v_g av optimeringsproblemet, og de ønskede verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ markert med røde bokser. Vi observerer at løsningen sammefaller fullstendig med de ønskede verdiene.

Eksempel 6.3 (Optimal verdi $\neq$ 0): Vi studerte også eksempler der det ikke fantes en løsning. Løsningene er skissert i figur 6.3. Vi observerer at den optimale verdien er forskjellig fra 0.



Figur 6.3: Viser løsningen v_g av optimeringsproblemet, og de ønskede verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ markert med røde bokser. Vi observerer at løsningen ikke sammefaller med de ønskede verdiene i alle punktene.

6.3 Mengden av optimale løsninger

Løsningen av optimeringsproblemet (6.2) er en optimal tilnærming til verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Generelt finnes det flere ulike optimale løsninger. Vi vil utforske mengden av optimale løsninger ved å skrive om optimeringsproblemet (6.4), og ønsker å finne nye løsninger med ulike egenskaper, samtidig som de er løsninger av det "originale" optimeringsproblemet (6.2). Vi begynner med å vise hvordan dette kan gjøres ved å innføre en ny objektivfunksjon.

6.3.1 Ny objektivfunksjon

For å finne nye optimale løsninger ved utskiftning av objektivfunksjon, vil vi begynne med å finne en optimal løsning og tilhørende optimal verdi av optimeringsproblemet (6.2). Deretter vil vi skrive om problemet ved å la den originale objektivfunksjonen

$$\min \sum_{(x_i,y_j)\in P} \delta(x_i, y_j)$$

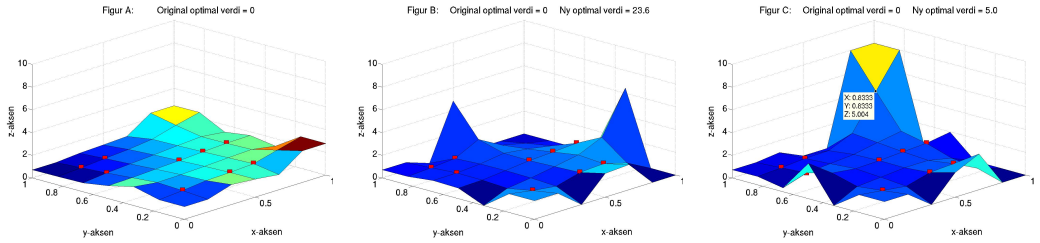
fra (6.4) opptre som en betingelse, og forutsetter at verdien er mindre eller lik den optimale verdien av det originale problemet, pluss en liten ϵ for avrundingsfeil. De øvrige originale betingelsene fra (6.4) vil bli beholdt, og vi vil innføre en ny vilkårlig objektivfunksjon η.

$$\begin{array}{ll} \min & \eta \\ \hline \text{f.a.} & \displaystyle\sum_{(x_i,y_j)\in P} \delta(x_i, y_j) \;\leq\; \text{optimal verdi} + \epsilon \\ & \\ & \text{originale betingelser fra (6.4) beholdes} \end{array} \tag{6.5}$$

Dermed har vi skrevet et nytt LP-problem, med vilkårlig objektivfunksjon η, som vil ha en ny optimal løsning v_g.

Eksempel 6.4 (Forskjellige objektivfunksjoner): La oss se på et problem med diskretiseringsgrad $n = 5$, der P består av 9 punkter. Figur 6.4 A viser løsningen av det originale problemet. Dette problemet har optimal verdi lik 0, som betyr at løsningen sammenfaller fullstendig med de gitte verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$. Ved å skrive om problemet som forklart over, kan vi nå finne nye løsninger som alle har samme optimale verdi $v^* = 0$.

Vi kan for eksempel finne en løsning der summen av verdien langs randa av området er så lav som mulig, som skissert i figur 6.4 B. Eller kanskje vi ønsker å maksimere verdien i et gitt punkt, si $(x_i, y_j) = ((n-1)h, (n-1)h)$, som skissert i figur 6.4 C.



Figur 6.4: Løsning v_g av optimeringsproblemet beskrevet i eksempel 6.4 for ulike objektivfunksjoner. Figur A: Original objektivfunksjon. Figur B: Minimere verdiene langs randa. Figur C: Maksimere verdiene i punktet $((n-1)h, (n-1)h)$.

6.4 Begrenset avstand mellom verdiene på randa

En annen type løsninger som kan være av interesse er løsninger der differansen mellom verdiene i nabopunktene langs randa er begrenset. Si vi ønsker å varmebehandle metallstaven med kvadratisk tverrsnitt som beskrevet tidligere, og ikke ønsker eller har mulighet til å ha store temperaturvariasjoner mellom nabopunktene.

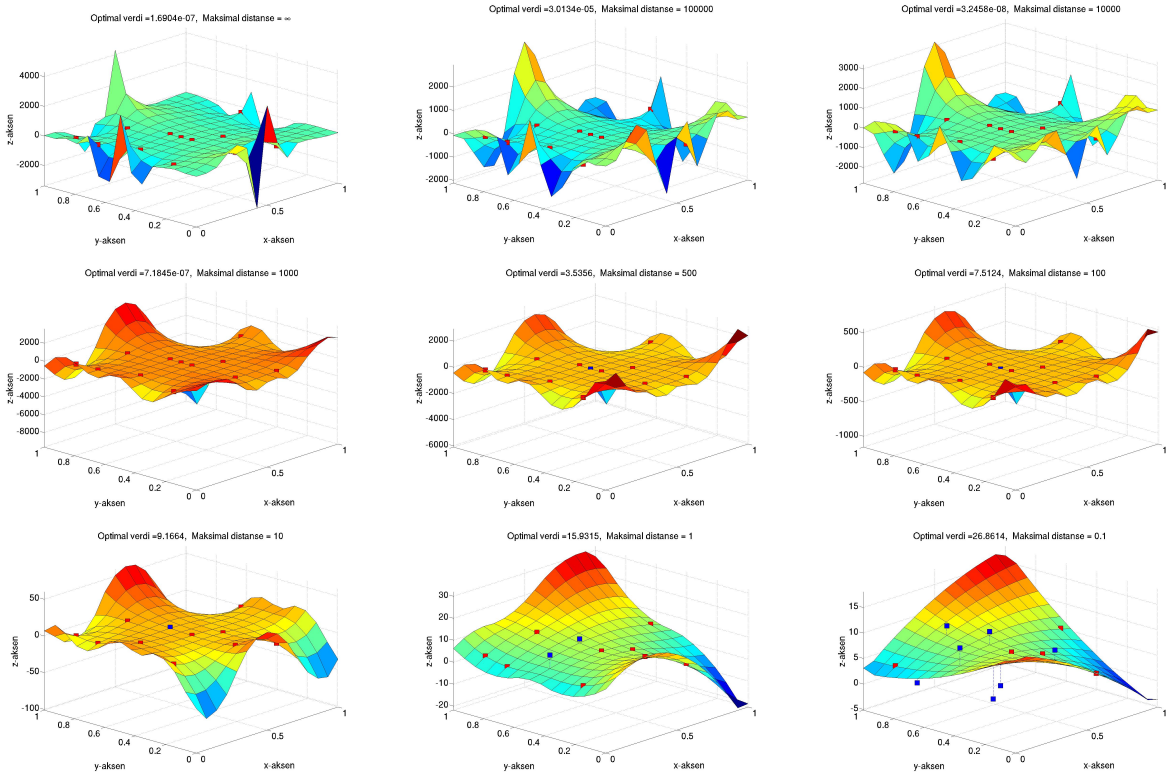
Vi supplerer det originale optimeringsproblemet (6.4) med betingelser som begrenser absoluttverdien av differansen mellom verdien i et punkt og gjennomsnittet av nabopunktene langs randa; $|v_g(x_{i-1}, y_j) - 2v_g(x_i, y_j) + v_g(x_{i+1}, y_j)| \leq M_d$, der M_d er en gitt maksimal differanse. Dette gir følgende optimeringsproblem

$$\begin{aligned} \min \quad & \sum_{(x_i,y_j)\in P} \delta(x_i, y_j) \\ \text{f.a.} \quad & -M_d \leq v_g(x_{i-1}, y_j) - 2v_g(x_i, y_j) + v_g(x_{i+1}, y_j) \leq M_d \\ & \text{originale betingelser fra (6.4) beholdes} \end{aligned} \tag{6.6}$$

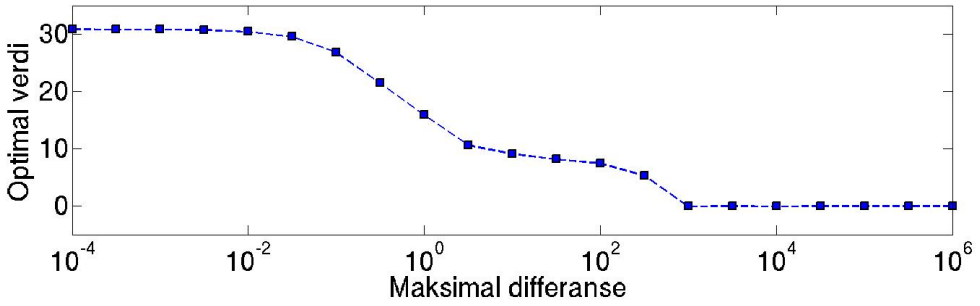
La oss illustrere problemet over med et eksempel.

Eksempel 6.5 (Begrenset variasjon langs randa): Vi lar diskretiseringsgraden n være 13 og antall punkter i mengden P være 12. Løsningen av det originale problemet, uten begrensninger på avstanden mellom verdiene på randa er skissert i figur 6.5 A. I figur 6.5 B-F ser vi løsninger for problemet med avtagende maksimal differanse M_d.

I figur 6.6 har vi plottet optimal verdi for problemet som funksjon av maksimal differanse. Vi observerer at for $M_d \leq 10^{-2}$ vil optimal verdi være tilnærmet lik 30. I området $M_d \in \{10^{-2}, 10^3\}$ vil optimal verdi reduseres i takt med økt maksimal differanse, før den optimale verdien er tilnærmet 0, for maksimale differanse $M_d \geq 10^3$. I dette eksempelet vil dermed begrensningen av avstanden mellom verdiene på randa ikke medføre økt optaiml verdi dersom $M_d \geq 10^3$.



Figur 6.5: Løsning v_g av optimeringsproblemet beskrevet i eksempel 6.5 med begrenset avstand mellom randverdiene. Maksimal differanse avtar. Merk: ulike verdier på z-aksen



Figur 6.6: Viser forholdet mellom maksimal differanse og optimal verdi for eksempel 6.5

6.5 Vekting

Vi har sett flere eksempler på ulike løsninger av optimeringsproblemet, både løsninger som sammenfaller fullstendig med de gitte verdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$, og løsninger som ikke gjør det. I tilfellene der det ikke eksisterer en eksakt løsning, det vil si i tilfellene

der løsningene ikke sammenfaller fullstendig med de gitte verdiene, vil det ofte være hensiktsmessig å fastslå om noen punkter er viktigere å "treffe" enn andre. Generelt finnes det mange løsninger av optimeringsproblemet (6.2) som alle er optimale, og vi ønsker å finne den den løsningen som sammenfaller med de "viktigste" punktene.

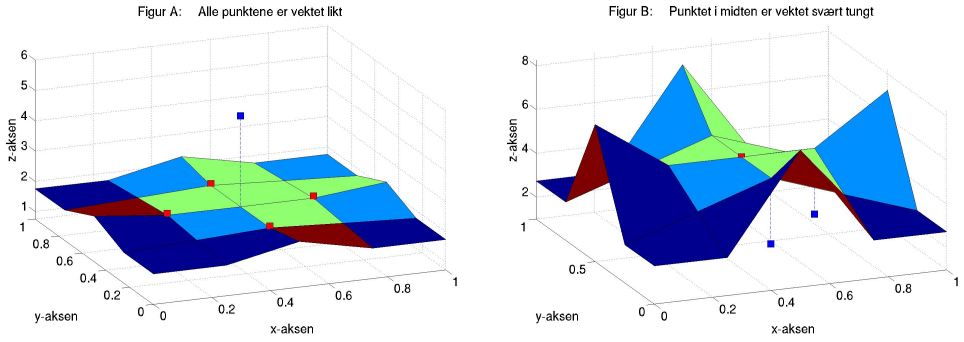
6.5.1 Vekting av et utvalg punkter

Vi kan ta hensyn til hvilke punkter det er viktigst å tilnærme ved å innføre vekter $w(x_i, y_j)$ i hvert punkt $(x_i, y_j) \in P$. I tillegg til verdien $\alpha(x_i, y_j)$ som vi ønsker å tilnærme, er vi altså gitt en vekt $w(x_i, y_j)$ som sier noe om hvor viktig det er å treffe dette punktet.

Objektivfunksjonen i optimeringsproblemet (6.2) forandres slik at vekten $w(x_i, y_j)$ multipliseres med absoluttverdien av differansen mellom $v_g(x_i, y_j)$ og $\alpha(x_i, y_j)$ i hvert punkt. Dermed får vi et nytt otimeringsproblemet. Optimeringsproblemet beholder de opprinnelige betingelsene, men får følgende nye objektivfunksjon

$$\min \sum_{(x_i,y_j)\in P} w(x_i, y_j) \cdot |v_g(x_i, y_j) - \alpha(x_i, y_j)|$$

Eksempel 6.6 (Nye vekter): La oss igjen se på eksempelet med gitte verdier $\alpha(3h, 3h) = 5$, og $\alpha(x_i, y_j) = 2$ for $(x_i, y_j) = \{(2h, 3h), (3h, 2h), (3h, 4h), (4h, 3h)\}$. Vi har illustrert løsningen av dette problemet med lik vekting i alle punkter i figur 6.7 A. I figur 6.7 B ved siden av har vi skissert løsningen for det samme problemet, men nå har vi lagt større vekt på punktet i midten.



Figur 6.7: Viser løsningen v_g med ulike vekter

6.5.2 Vekting av alle punktene i området

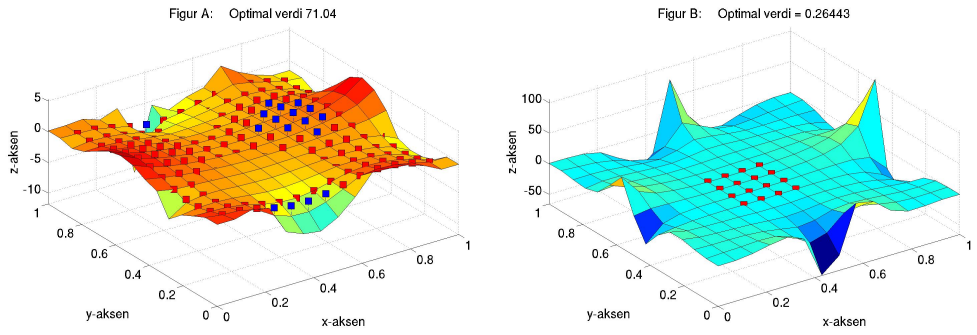
Vi vil nå studere problemer der vi er gitt funksjonsverdier og vekter i hvert enkelt punkt i hele området Ω_h. Ved å formulere problemet på denne måten kan vi blant annet finne approksimasjoner til løsninger av problemer der vi er gitt en analytisk funksjon definert i hele området Ω, og vi ønsker å finne en løsning av Laplace-problemet som tilnærmer den

analytiske funksjonen i gitte deler av området Ω. Ved å la $w(x_i, y_j) = 0$ for $(x_i, y_j) \notin P$ vil vi som tidligere kunne løse problemer der bare noen av punktene i Ω_h er gitt en verdi vi ønsker å tilnærme.

Eksempel 6.7: La $\gamma : \Omega \to \mathbb{R}$ være en analytisk funksjon. Vi ønsker å finne en diskret approksimasjon, α, til funksjonen γ i et gitt delområde S av Ω. Nå lar vi $w(x_i, y_j) = 1$ for punktene $(x_i, y_j) \in P$, der P er en approksimasjon til delmengden S. Vektene i de resterende punktene settes lik 0.

Vi kan for eksempel la $\gamma(x, y) = \sin(\pi^2 x) \cdot \cos(\pi^2 y)$ og $S = \{(x, y) \in \mathbb{R}^2 : 1/3 < x, y < 2/3\}$. Dette gir $\alpha(x_i, y_j) = \sin(\pi^2 x_i h) \cdot \cos(\pi^2 y_j h)$ for $(x_i, y_j) \in P$, der $P = \{(x_i, y_j) \in \mathbb{R}^2 : 1/3 < x_i h, y_j h < 2/3\}$. Med diskretiseringsgrad $n = 15$, gir dette løsningen illustrert i figur 6.8 B.

Vi observerer at løsningen er en god approksimasjon, med optimal verdi 0.26, når vi begrenser oss til å tilnærme verdiene i delmengden P. Forsøker vi derimot å tilnærme funksjonsverdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in \Omega_h$, som skissert i figur 6.7 A, blir tilnærmingen mindre god, og optimeringsproblemet har optimal verdi 71.04.

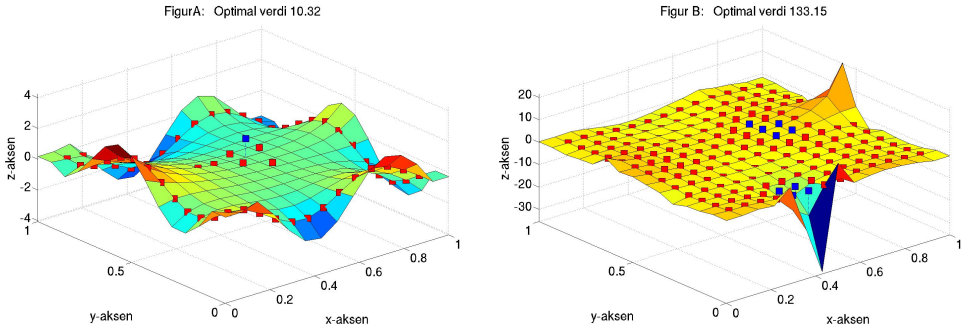


Figur 6.8: Viser løsning v_g av problemene beskrevet i eksempel 6.7

Eksempel 6.8: I eksempel 6.7 over ble alle punktene i mengden P gitt en konstant vekt lik 1, og $w(x_i, y_j) = 0$ for $(x_i, y_j) \notin P$. Generelt gis hvert enkelt punkt i P en egen vekt $w(x_i, y_j)$.

La nå $P = P_1 \cup P_2$, der P_1 definerer området $P_1 = \{(x_i, y_j) \in \mathbb{R}^2 : 1/3 < x_i h, y_j h < 2/3\}$ som vi studerte i forrige eksempel, og la P_2 angi mengden av nabopunkter til randpunktene. Vi gir punktene i P_1 vekt 1 som tidligere, og gir punktene i P_2 vekt 2. Med funksjonen γ som i eksempelet over får vi løsningen som er skissert i figur 6.9 A.

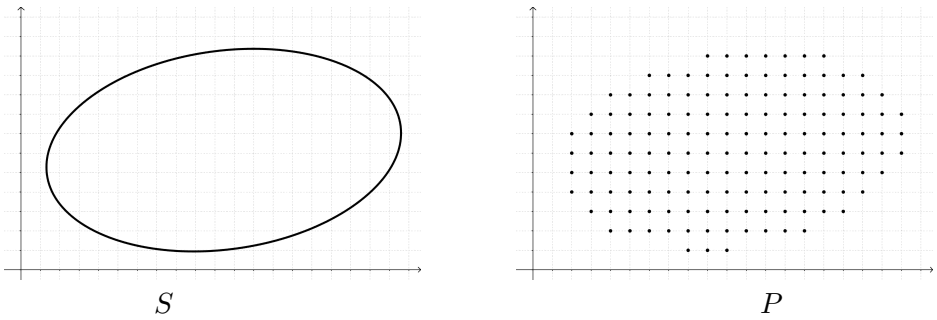
Til slutt lar vi alle punktene i hele området vektes, men punktene i P_1 (midten av området) vektlegges ti ganger mer. Løsningen er skissert i figur 6.9 B.



Figur 6.9: Viser løsning v_g av problemene beskrevet i eksempel 6.8

6.6 Kontinuerlig mengde av punkter

Delmengden S av Ω og funksjonen $\gamma : S \to \mathbb{R}$ i (6.1) beskriver henholdsvis hvilke punkter og hvilke verdier vi ønsker at løsningen av det kontinuerlige Laplace-problemet (2.3) skal ha minimal avstand til. I praktiske anvendelser vil det ofte være naturlig at mengden S av punkter vi ønsker å tilnærme er en kontinuerlig delmengde eller delområde av Ω. For varmebehandling av et materiale vil det for eksempel være naturlig å angi områder med en ønsket temperaturprofil heller enn en ønsket temperatur i et endelig antall punkter. For å løse denne typen problemer vil vi approksimere den kontinuerlige delmengden S med et endelig antall punkter P og løse det diskrete optimeringsproblemet (6.2).



Figur 6.10: Delmengden $P \subseteq \Omega_h$ er en diskret approksimasjon til det kontinuerlig område $S \subseteq \Omega$.

I tilfeller der S ikke er en endelig mengde punkter er det en utfordring at den optimale verdien av en løsning av optimeringsproblemet vil øke med diskretiseringsgraden n. Dette skyldes at den optimale verdien av problemet beregnes ved å summere avstanden mellom løsningen vi finner og de ønskede verdiene $\gamma(x, y)$ for $(x, y) \in S$, og dersom diskretiseringsgraden øker vil mengden S bestå av flere punkter som gir bidrag til denne verdien. For bedre å kunne sammenligne de ulike problemene vil vi innføre en ny størrelse kalt relativ optimal verdi.

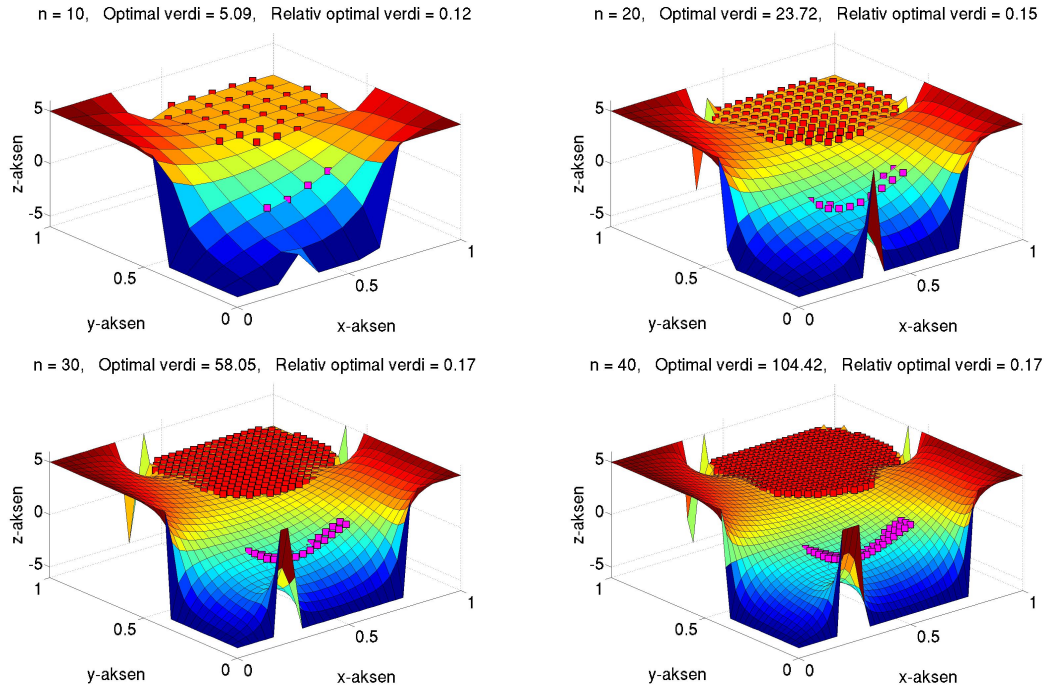
Definisjon 6.4. *Vi vil la v_r^* betegne relativ optimal verdi av optimeringsproblemet (6.2), gitt ved*

$$v_r^* = \frac{v^*}{|P|} \tag{6.7}$$

der v^* betegner optimal verdi av optimeringsproblemet og $|P|$ betegner antall punkter i mengden P, der P er en approksimasjonen til mengden S.

La oss nå studere et eksempl der mengden S kan uttrykkes ved forskjellige kontinuerlige delmengder eller delområder av Ω, og sammenligne de ulike løsningene og deres optimale og relative optimale verdi. I eksempel 6.9 lar vi mengden S uttrykkes ved to forskjellige kontinuerlige mengder eller områder S_1 og S_2 med en ønsket temperaturprofil gitt ved funksjonene $\gamma_1, \gamma_2 : S \to \Omega$.

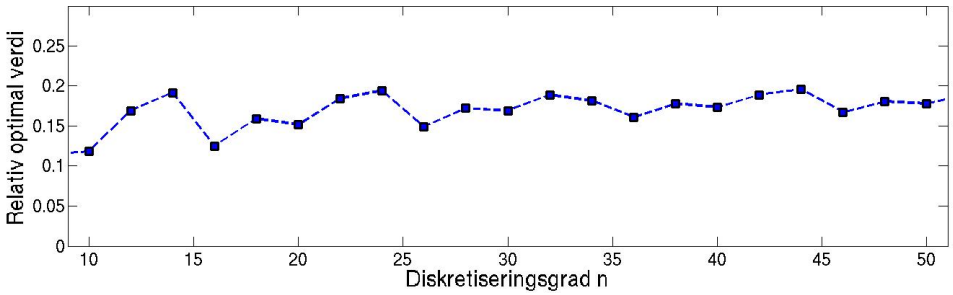
Eksempel 6.9 (To kontinuerlige mengder): Vi lar området $S \in \Omega$ være definert ved $S = S_1 \cup S_2$, der $S_1 = \{(x, y) \in \mathbb{R}^2 : 0.3 \leq x \leq 0.7,\ 0.2 \leq y \leq 0.4\}$, og $S_2 = \{(x, y) \in \mathbb{R}^2 : [(x-0.7)^2 + (y-0.7)^2] \leq 0.1\}$. Temperaturprofilen er gitt ved funksjonene $\gamma_1(x, y) = -1$ for $(x, y) \in S_1$ og $\gamma_2(x, y) = \sin(x\pi^2)$ for $(x, y) \in S_2$.



Figur 6.11: Viser løsningen av problemet beskrevet i eksempel 6.9 for ulike diskretiseringsgrad n. Punktene med opprinnelse i delmengden S_1 er markert med rosa bokser, mens punktene markert med røde bokser har sin opprinnelse i delmengden S_2.

Vi har formulert optimeringsproblemet (6.2) for ulike diskretiseringsgrader n. Løsningen for diskretiseringsgrad $n = 10$, 20, 30, og 40 er skissert i figur 6.11.

I figur 6.12 har vi plottet relativ optimal verdi v_r^* som funksjon av diskretiserigsgraden n for $n = 10, 12, ...50$. For dette problemet kan vi observere at variasjonen i v_r^* som ønsket er relativt liten. Den relative optimale verdien v_r^* vil derfor i dette tilfellet kunne fungere som et bedre mål på hvor godt løsningen "treffer" de gitte punktene, enn den optimale verdien v^*.



Figur 6.12: Viser relativ optimal verdi v_r^* av problemet beskrevet i eksempel 6.9 som funksjon av diskretiserigsgraden n.

Kapittel 7

Effektiv løsning av optimeringsproblemet

I think there is a world market for maybe five computers.

Thomas Watson, formann ved IBM, 1943

I dette kapittelet vil vi se nærmere på hvordan vi kan finne en løsning av optimeringsproblemet (6.2) som vi formulerte i forrige kapittel, på en effektiv måte ved hjelp av moderne optimeringsverktøy. Vi vil betrakte ulike problemer på formen (6.2), og sammenligne resultater og effektivitet ved anvendelse av ulike metoder og algoritmer.

Med fin diskretisering vil optimeringsproblemet inneholde et stort antall begrensninger og variable. Generelt kreves det mye regnekraft og lang beregningstid for å løse slike problemer. Det finnes mange typer optimeringsproblemer som vi ikke har effektive metoder for å løse, men som nevnt i kapittel 5 kan lineære optimeringsproblemer, LP-problemer, løses effektivt ved hjelp av blant annet simplex-algoritmer eller indrepunktsmetoder. Og i forrige kapittel viste vi hvordan vi ved innføring av passende variable, δ, kan skrive om optimeringsproblemet (6.2) til et lineært optimeringsproblem, LP-problem, på formen

$$
\begin{array}{lrcrcll}
\min & \displaystyle\sum_{(x_i,y_j)\in P} \delta(x_i,y_j) & & & & & \\
\hline
\text{f.a.} & v_g(x_i,y_j) & - & \delta(x_i,y_j) & \leq & \alpha(x_i,y_j) & (x_i,y_j)\in P \\
 & -v_g(x_i,y_j) & - & \delta(x_i,y_j) & \leq & -\alpha(x_i,y_j) & (x_i,y_j)\in P \\
 & & - & \Delta_h v_g(x_i,y_j) & = & 0 & (x_i,y_j)\in \Omega_h
\end{array}
\tag{7.1}
$$

Vi vil beskrive flere ulike metoder for å løse optimeringsproblemet (6.2) som alle tar utgangspunkt i en omformulering av problemet til et LP-problem:

Opprinnelig LP-problem i MATLAB I seksjon 7.1 vil vi beskrive en metode som benytter optimerings-verktøykassen optimtool i matematikkprogrammet MATLAB for å løse LP-problemet (7.1).

Opprinnelig LP-problem i CPLEX En tilsvarende metode som benytter modellerings- og optimeringsverktøyet CPLEX for å løse LP-problemet (7.1) vil bli beskrevet i seksjon 7.2.

Redusert LP-problem i MATLAB Deretter, i seksjon 7.3, vil vi beskrive en metode der optimeringsproblemet (6.2) reduseres til et LP-problem med færre variable og begrensninger, før dette løses ved hjelp av optimerings-verktøykassen optimtool i MATLAB.

Gradient i MATLAB Også metoden beskrevet i seksjon 7.4 tar utgangspunkt i det reduserte LP-problemet. Denne metoden er en iterativ gradient-metode. Metoden begynner med en approksimasjon til den optimale løsningen, og med denne som utgangspunkt beregnes retningsvektorer og skrittlengder som gir en sekvens av løsninger som konvergerer mot den optimale løsningen av optimeringsproblemet.

MATLAB og CPLEX tilbyr ulike algoritmer for å løsne LP-problemer. Vi har valgt å konsentrere oss om ulike varianter av indrepunktsmetoden og simplex-algoritmen. Som nevnt vil ulike eksempler med problemer på formen (6.2) betraktes. Problemene formuleres underveis, og løsningene og beregningstid ved anvendelse av de ulike metodene og algoritmene vil bli presentert. Se tillegg A for en oversikt over beregningstid for alle problemene som studeres.

7.1 Opprinnelig LP-problem i MATLAB

I denne seksjonen vil vi studere hvordan optimeringsproblemet (7.1) kan løses ved hjelp av det matriseorienterte matematikkprogrammet MATLAB. MATLAB ble opprettet på slutten av 1970-tallet av Cleve Moler som på den tiden var ansatt ved Universitetet i New Mexico. Det er et teknisk programmeringsspråk og utviklingsmiljø brukt innenfor en rekke områder, blant annet bilde- og signalbehandling, kontrollsystemer, finansiell modellering og bioinformatikk [www.mathworks.com]. Programmet er mye brukt innenfor forsknings- og utdanningsinstitusjoner i tillegg til industrielle foretak [13].

MATLAB har mange spesialiserte rutiner kalt "toolboxer" eller verktøykasser. En av disse er optimerings-verktøykassen optimtool som benyttes for å løse ulike typer matematiske optimeringsproblemer. For å løse (7.1) har vi benyttet funksjonen **linprog** som benyttes for å løse LP-problemer. Funksjonen **linprog**($\mathbf{c}$, $\mathbf{A}_{in}$, $\mathbf{b}_{in}$, $\mathbf{A}_{eq}$, $\mathbf{b}_{eq}$) returnerer optimal løsning og optimal verdi for problemet

$$\begin{array}{lrcl} \min & \mathbf{c}^T\mathbf{x} & & \\ \hline \text{f.a.} & \mathbf{A}_{in}\mathbf{x} & \leq & \mathbf{b}_{in} \\ & \mathbf{A}_{eq}\mathbf{x} & = & \mathbf{b}_{eq} \end{array} \tag{7.2}$$

For å løse optimeringsproblemet (7.1) ved hjelp av **linprog** må vi skrive problemet på formen (7.2) over. Under vil vi kort skissere hvordan dette kan gjøres. Programkode er vedlagt i tillegg C.3.

7.1.1 Optimeringsproblemet på matriseform

Vi lar diskretiseringsgraden n være gitt, og antar at funksjonsverdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ er gitt i en matrise $\mathbf{W}$ med rader på formen

$$[x_i \quad y_j \quad \alpha(x_i, y_j)]$$

Vi lar $\mathbf{x}$ være en vektor som representerer alle variablene g, v_g og δ i (7.1) og konstruerer en matrise $\mathbf{A}_{in}$ og vektor $\mathbf{b}_{in}$ slik at matriseligningen $\mathbf{A}_{in}\mathbf{x} \leq \mathbf{b}_{in}$ er ekvivalent med

$$\begin{array}{rcrcll} v(x_i,y_j) & - & \delta(x_i,y_j) & \leq & \alpha(x_i,y_j) & \text{for} \quad (x_i,y_j) \in P \\ -v(x_i,y_j) & - & \delta(x_i,y_j) & \leq & -\alpha(x_i,y_j) & \text{for} \quad (x_i,y_j) \in P \end{array}$$

der α er gitt ved $\mathbf{W}$.

Videre konstruerer vi matrisen $\mathbf{A}_{eq}$ slik at matriseligningen $\mathbf{A}_{eq}\mathbf{x} = \mathbf{b}_{eq}$, der $\mathbf{b}_{eq} = \mathbf{0}$-vektoren, er ekvivalent med

$$-\Delta_h v_g(x_i, y_j) = \frac{1}{h^2}[4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1}] = 0$$

Objektivfunksjonen skrives på formen $\mathbf{c}^T\mathbf{x}$ der $\mathbf{c}$ er en vektor som består av 0-er på alle plasser unntatt plassene som representerer δ.

Dermed har vi skrevet optimeringsproblemet (7.1) på formen (7.2) og problemet kan løses ved hjelp av funksjonen **linprog**.

7.1.2 Ulike optimeringsalgoritmer

Vi skal betrakte to ulike algoritmer som er tilgjengelige i MATLAB ved bruk av **linprog**. Med standardinnstillinger benyttes en primal-dual indrepunktsmetode. I følge produktbeskrivelsen er metoden basert på en variant av Mehrotras "predictor-corrector" algoritme [9], og er designet for å utnytte blant annet MATLABs sparse-matrise[1] funksjon. En videreutviklet versjon av den originale simplex-algoritmen som beskrevet i kapittel 5 kan også benyttes.

7.1.3 Resultater

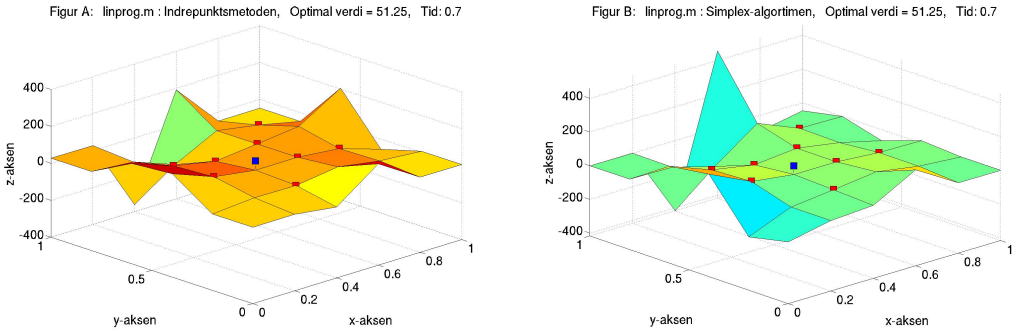
Vi har løst et utvalg optimeringsproblemer på formen (7.1) ved anvendelse av algoritmene beskrevet over. Beregningstiden ved anvendelse av indrepunktsmetoden og simplex-algoritmen er estimert ved hjelp av stoppeklokkefunksjonene *tic* og *toc* i MATLAB. Den returnerte beregningstiden er avhengig av datamaskinen som benyttes, og vi må anta at beregningstiden vil variere. Den returnerte beregningstiden vil likevel indikere forholdet mellom

[1] sparse-matrise (eng.: sparse-matrix): en matrise bestående av et stort antall 0-er

de ulike metodene hva gjelder effektivitet. For å begrense avvik er alle målingene foretatt flere ganger, og aritmetisk middel oppgis.

Eksempel 7.1 (Ni tilfeldig valgte punkter): Vi lar diskretiseringsgraden n være 4, og og lar P bestå av 9 tilfeldig valgte punkter, (x_i, y_j), som gis tilfeldige heltallige verdier $\alpha(x_i, y_j)$ i intervallet $[0, 100]$. Vi ønsker at løsningen v_g av Laplace-problemet skal ligge så nær disse verdiene som mulig.

Løsningen av problemet ved bruk av indrepunktsmetoden og simplex-algoritmen er skissert i figur 7.1. Vi observerer at løsningene er svært forskjellige, samtidig som begge løsningene gir samme optimale verdi på 51.25. Beregningstiden er tilnærmet den samme for begge metodene.



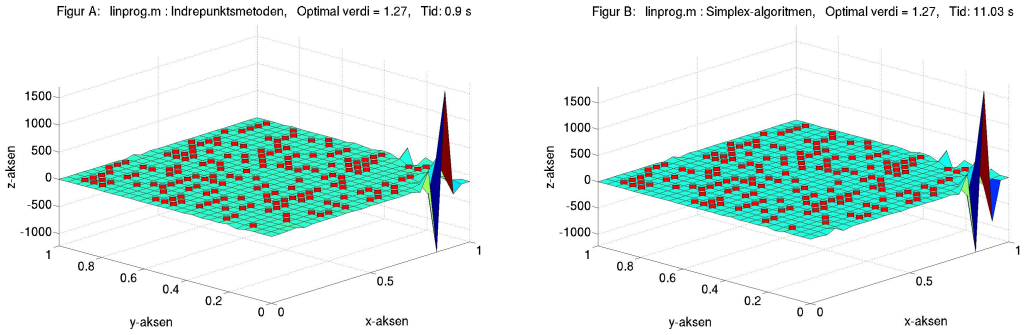
Figur 7.1: Skisse av løsning av problemet beskrevet i eksempel 7.1. Løsning ved anvendelse av indrepunktsmetoden er skissert i figur A. Figur B viser løsning ved anvendelse av simplex-algoritmen

Eksempel 7.2 (Flere punkter): La oss nå studere et problem med diskretiseringsgrad $n = 23$, som gir et grid med 25×25 punkter. Vi velger ut 200 vilkårlige punkter som gis verdier $\alpha(x_i, y_j) = \sin(x_i h) \cdot \cos(y_j h)$.

I figur 7.2 A og B er løsningen av problemet ved anvendelse av indrepunktsmetoden og simplex-algoritmen skissert. Algoritmene gir tilnærmet lik optimal verdi på 1.27.

Vi observerer at indrepunktsmetoden er betydelig mer effektiv enn simplex-algoritmen. Mens indrepunktsmetoden bare bruker i underkant av 1 sekund, er beregningstiden ved anvendelse av simplex-algoritmen omkring 11 sekunder. Det er også verdt å merke seg at løsningene, tross likhet, ikke er identiske. Løsningene skiller seg markant i enkelte punkter langs randa.

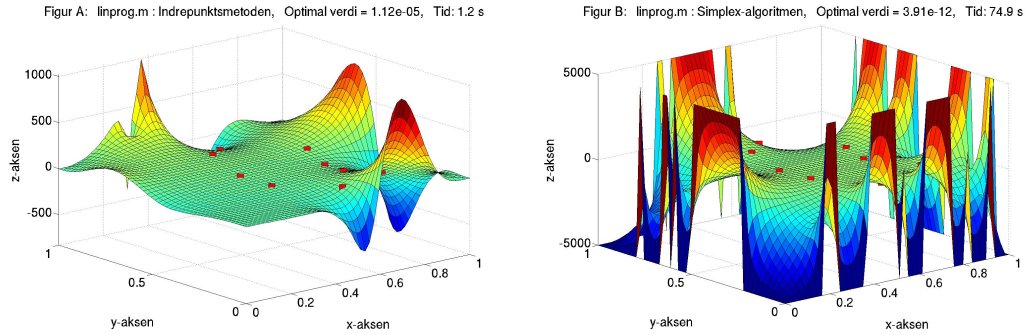
Eksempel 7.3 (Høy diskretiseringsgrad): Til slutt i denne seksjonen betrakter vi et problem med høy diskretiseringsgrad og svært få punkter som skal tilnærmes. Vi lar diskretiseringsgraden n være 48, som gir et grid med 50×50 punkter, og lar P bestå av 10 vilkårlig valgte punkter. På samme måte som i problem 7.1 gis hvert punkt i P



Figur 7.2: Skisse av løsning av problemet beskrevet i eksempel 7.2. I figur A er løsning ved anvendelse av indrepunktsmetoden skissert. Løsning ved anvendelse av simplex-algoritmen er skissert i figur B.

en vilkårlig heltallig verdi mellom 0 og 100. I tillegg legger vi inn begrensninger på alle punktene slik at løsningen må ligge i intervallet $[-5000, 5000]$. Løsningen er skissert i figur 7.3.

Også i dette tilfellet er indrepunktsmetoden betraktelig mer effektiv enn simplex-algoritmen. Simplex-algoritmen returnerer en løsning der randverdiene i flere punkter tar maksimale og minimale verdier. Dette er ikke tilfelle i løsningen returnert av indrepunktsmetoden der punktet med maksimal verdi har verdi omkring 1000. Variasjonen mellom punktene i løsningen er svært stor i løsningen returnert ved anvendelse av simplex-algoritmen noe som i de fleste praktiske tilfeller vil være uheldig. Begge algoritmene returnerer en løsning med optimal verdi v^* tilnærmet 0.



Figur 7.3: Skisse av løsning av problemet beskrevet i eksempel 7.3. Løsning ved anvendelse av indrepunktsmetoden er skissert i figur A. Figur B viser løsning ved anvendelse av simplex-algoritmen.

Vi oppsummerer resultatetene fra problem 7.1-7.3 i tabell 7.1.

	Eksempel 7.1		**Eksempel 7.2**		**Eksempel 7.3**	
	v^*	tid	v^*	tid	v^*	tid
linprog indrepunkts	51.25	0.7	1.27	0.9	1.1e-05	1.9
linprog simplex	51.25	0.7	1.27	11.0	3.9e-09	74.9

Tabell 7.1: Viser verdi og beregningstid for optimeringsproblemene beskrevet i eksempel 7.1 - 7.3.

7.2 Opprinnelig LP-problem i CPLEX

Vi har som nevnt tidligere benyttet ulike optimeringsverktøy for å løse optimeringsproblemet (6.2). I denne seksjonen vil vi beskrive hvordan problemet kan løses ved hjelp av optimeringsverktøyet CPLEX, og sammenligne resultater og beregningstid ved anvendelse av noen av de ulike algoritmene som tilbys.

CPLEX ble grunnlagt av Robert E. Bixby, og ble lansert kommersielt i 1988 av firmaet CPLEX Optimization Inc. I ettertid er firmaet kjøpt opp av ILOG som igjen er kjøpt opp av IBM, og går i dag under navnet IBM ILOG CPLEX Optimization Studio. Da CPLEX ble grunnlagt baserte det seg på simplex-algoritmen og var implementert med programmeringsspråket C. I dag består programmet av en rekke algoritmer for ulike optimeringsproblemer, og flere brukergrensesnitt tilbys [www.ibm.com].

7.2.1 Formulering av optimeringsproblemet i OPL-CPLEX

Vi har benyttet et programmeringsspråk designet for CPLEX kalt OPL - Optimization Programming Language. Språket er designet for å løse optimeringsproblemer, og gir svært enkel og forholdsvis kort kode sammenlignet med andre mer generelle språk. Syntaksen er enklere enn i MATLABs **linprog**, og i OPL-CPLEX er det ikke nødvendig å skrive optimeringsproblemet (7.1) på matriseform.

I OPL formulerer vi det generelle optimeringsproblemet i en modelleringsfil. Modelleringsfila begynner med å hente data og innstillinger til hvert spesifikke problem fra to separate filer. Dataene er lagret i en separat fil som består av en matrise $\mathbf{W}$ med funksjonsverdiene $\alpha(x_i, y_j)$ for $(x_i, y_j) \in P$ i tillegg til informasjon om diskretiseringsgraden. Også innstillingene er lagret i en separat fil for hvert enkelt problem. I denne fila ligger blant annet detaljert informasjon om hvilken algoritme og type av denne som skal benyttes.

Etter at data og innstillinger er hentet formulerer vi Poisson-matrisen A_{eq} på tilsvarende måte som i forrige seksjon, før beslutningsvariablene v og δ deklareres

```
dvar float  v[ver];
dvar float+ delta[P];
```

Til slutt skriver vi optimeringsproblemet på samme form som (7.1):

```
minimize sum(j in P) delta[j];
subject to{
   forall(j in P){
      v[(n*(ftoi(W[j][1])-1)+ftoi(W[j][2]))] -delta[j] <=  W[j][3];
     -v[(n*(ftoi(W[j][1])-1)+ftoi(W[j][2]))] -delta[j] <= -W[j][3];
   }
   forall(i in veq){
      sum(j in ver) Aeq[i][j]*v[j] ==0;
   }
}
```

der **ftoi** er en funksjon som konverterer flytall (float) til heltall (int). Programkode er vedlagt i tillegg C.4.

7.2.2 Ulike optimeringsalgoritmer

CPLEX tilbyr en rekke forskjellige algoritmer for å løse lineære optimeringsproblemer effektivt. Vi har benyttet primal og dual simplex-algoritme i tillegg til barrier-indrepunktsmetoden for å løse optimeringsproblemet (7.1). I følge produktbeskrivelsen er den duale simplex-algoritmen anbefalt førstevalg. Den primale simplex-algoritmen vil imidlertid ofte fungerer bedre for problemer der antall variable overskrider antall begrensninger betydelig, eller for problemer med liten variasjon i kost-variablene. Barrier-metoden er spesielt designet for å løse store, sparse problemer. For utfyllende informasjon om de ulike metodene henvises det til [4, 15].

7.2.3 Resultater

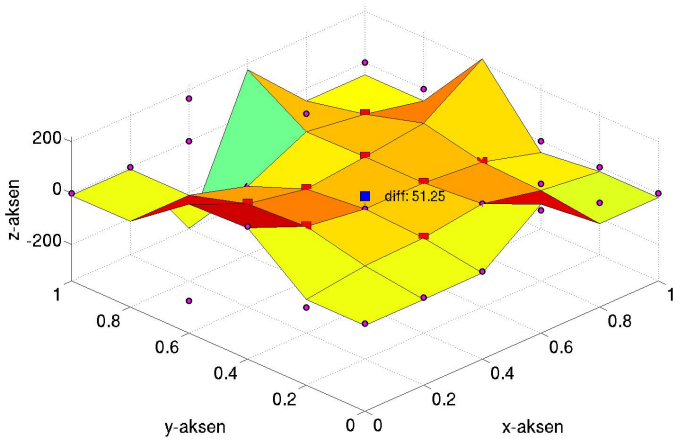
Vi ønsker å sammenligne løsninger og beregningstid ved anvendelse av OPL-CPLEX med resultatene beregnet med MATLABs **linprog**, for noen problemer på formen (7.1). Vi begynner med å studere problemene i eksempel 7.1 - 7.3 som vi formulerte i forrige seksjon, før vi presenterer noen nye problemer.

Eksempel 7.1 - 7.3 (gjensyn): Problemet vi studerte i eksempel 7.1 hadde diskretiseringsgrad $n = 4$ og mengden P bestod av 9 tilfeldig valgte punkter. Løsningene fra MATLAB og CPLEX er illustrert i figur 7.4. Løsningen fra MATLAB er skissert kontinuerlig som tidligere, mens løsningen fra CPLEX er markerert med runde punkter.

I dette tilfellet returnerer CPLEX den samme løsningen ved anvendelse av den primale og duale simplex-algoritmen og barrier indrepunktsmetoden. Vi observerer at løsningene i MATLAB og CPLEX er forskjellige, men sammenfallende i punktene i P.

For problemene i eksempel 7.2 og 7.3 returnerer CPLEX samme optimale verdi som MATLAB, henholdsvis 1.27 og tilnærmet 0. Løsningene er også her de samme ved anvendelse av de ulike algoritmene.

Beregningstiden i CPLEX og MATLAB er presentert i tabell 7.2. Vi observerer at for problemet i eksempel 7.1 og 7.2 er beregningstiden tilsynelatende den samme ved



Figur 7.4: Skisserer løsning av problemet i eksempel 7.1. Løsningen fra MATLAB er skissert kontinuerlig som tidligere, mens løsningen fra CPLEX er markerert med runde punkter.

bruk av de ulike algoritmene i CPLEX, henholdsvis omkring 0.2 og 0.7 sekunder, mens beregningstiden varierer for problemet i eksempel 7.3. For den primale og duale simplex-algoritmen og barrier indrepunktsmetoden er gjennomsnittlig beregningstid henholdsvis 6.2, 7.7 og 6.7 sekunder for dette problemet.

EKSEMPEL:	**7.1**	**7.2**	**7.3**	**7.4**
Antall punkter i $\bar{\Omega}_h$	36	625	2 500	10 000
Antall punkter i P:	9	200	10	50
Optimal verdi (tilnærmet):	51.3	1.3	0.0	0.0
MATLAB				
indrepunktsmetoden	0.7	0.9	1.2	7.8
simplex	0.7	11.0	74.9	1647.9
CPLEX				
simplex primal	0.2	0.7	6.2	173.6
simplex dual	0.2	0.7	7.7	176.0
barrier indrepunktsmetoden	0.2	0.7	6.7	185.4

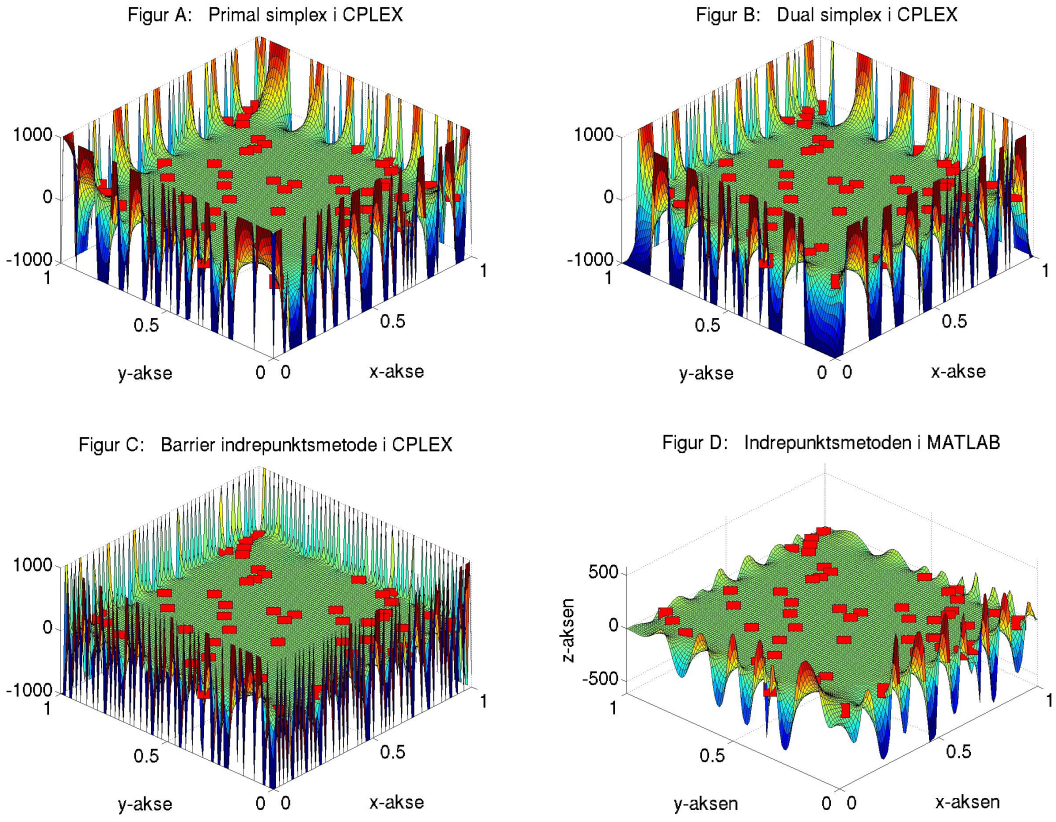
Tabell 7.2: Tabell som viser beregningstiden i sekunder for problemene i eksempel 7.1 - 7.3 og 7.4 ved bruk av ulike algoritmer i MATLAB og CPLEX.

Beregningstidene vi presenterer er som nevnt gjennomsnittlig beregningstid for et gitt problem. Det viser seg at beregningstidene varierer mye, noe som indikerer at feilmarginen er betydelig, og vi vil anslå at feilmarginen i eksempel 7.3 er større enn differansen mellom beregningstiden med de ulike algoritmene. Vi kan likevel med rimelig

sikkerthet fastslå at CPLEX løser problemet i eksempel 7.1 noe raskere enn både simplex-algoritmen og indrepunktsmetoden i MATLAB. CPLEX løser problemet i eksempel 7.2 omlag like effektivt som indrepunktsmetoden i MATLAB. For problemet i eksempel 7.3 er indrepunktsmetoden i MATLAB betydelig mer effektiv. Simplex-algoritmen i MATLAB er i begge de to sistnevnte eksemplene betydelig mindre effektiv enn de ulike algoritmene i CPLEX og indrepunktsmetoden i MATLAB.

La oss nå studere et par nye eksempler. Vi begynner med et eksempel der vi skal se at løsningen ved anvendelse av de ulike algoritmene i CPLEX ikke er sammenfallende.

Eksempel 7.4 (Forskjellige løsninger med ulike metoder): Vi lar diskretiseringsgraden n være 98, noe som gir et grid med 100×100 punkter. Mengden P lar vi bestå av 50 tilfeldig valgte punkter som gis verdier $\alpha(x_i, y_j) = \sin(x_i h) + \cos(x_j h)$ for $(x_i, y_j) \in P$. I tillegg lar vi øvre og nedre skranke være satt til henholdsvis 1000 og -1000.



Figur 7.5: Figur A-C viser løsning av optimeringsproblemet beskrevet i eksempel 7.4 ved bruk av henholdsvis primal og dual simplex-algoritme og indrepunktsmetoden i CPLEX. I figur D er løsning returnert ved bruk av indrepunktsmetoden i MATLAB skissert.

Løsningen returnert av den primale og duale simplex algoritmen og indrepunktsme-

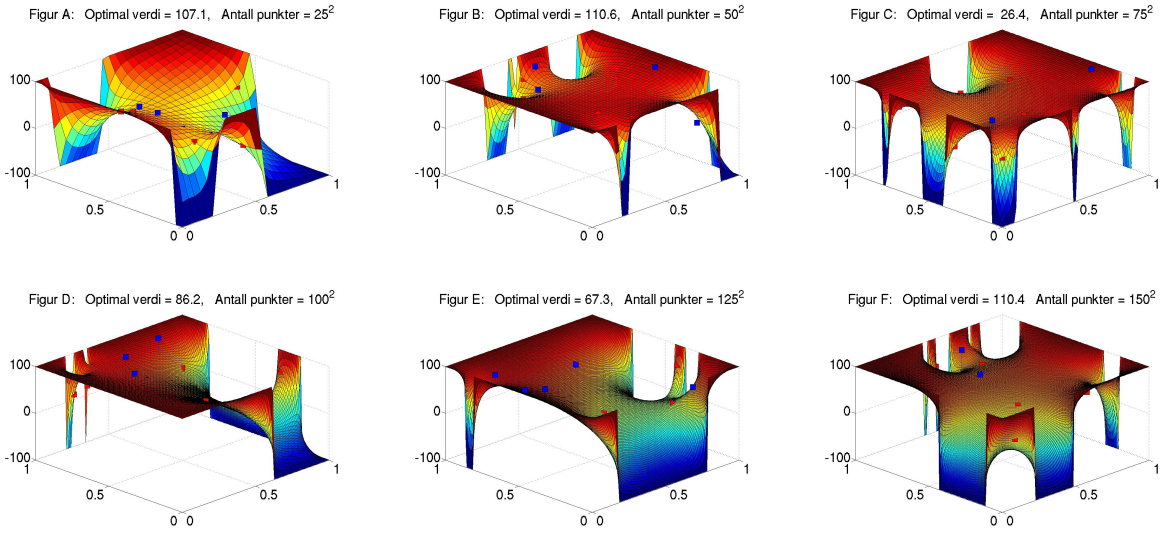
toden i CPLEX er henholdsvis illustrert i figur 7.5 A-C. Figur 7.5 D viser løsning ved bruk av indrepunktsmetoden i MATLAB.

De ulike algoritmene og metodene i CPLEX og MATLAB returnerer alle sammen optimal verdi svært nær 0, men løsningene er likevel svært forskjellige. Vi observerer blant annet at løsningene fra CPLEX har svært store variasjoner langs randa sammenlignet med løsningen returnert ved bruk av indrepunktsmetoden i MATLAB. Spesielt stor variasjon finner vi ved anvendelse av barrier indrepunktsmetoden i CPLEX. Årsaken til dette kan ligge i ulik programkode og variasjoner i algoritmene som benyttes. Forskjellen kan også ha sin forklaring i forskjellig toleranse, tillatt avvik fra nøyaktig mål.

Vi observerer også at løsningene i CPLEX inneholder flere punkter langs randa som tar verdier øvre og nedre skranke. Til sammenligning har løsningen i MATLAB sin maksimale og minimale verdi ved omkring halvparten av denne skranken.

Beregningstiden for problemet ved anvendelse av de ulike algoritmene er oppgitt i tabell 7.2. Her indikeres det at indrepunktsmetoden i MATLAB er betydelig mer effektiv enn de ulike algoritmene i CPLEX for disse problemene. Vi må ta forbehold om at disse forskjellene i beregningsid helt eller delevis har sin årsak i utenforliggende omstendigheter. De de to programmene er kjørt gjennom ulike operativsystemer, og vi må anta at dette kan påvirke resultatene.

Desverre er maksimal grense for antall variable i MATLAB begrenset slik at vi ikke kan sammenligne metodene for større problemer enn problemet vi studerte i det foregående eksempelet. Som vi skal se i de to neste eksempelene gjelder ikke dette i CPLEX.



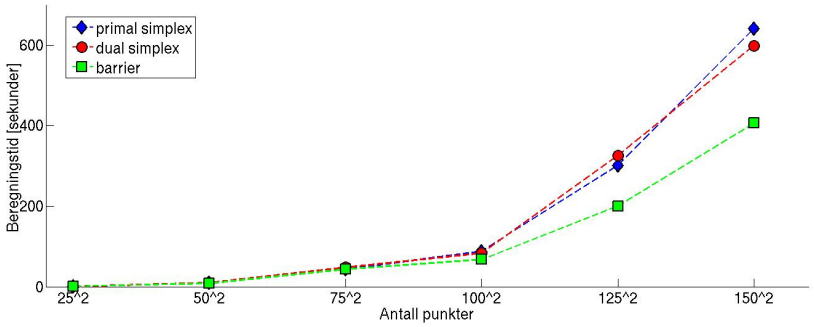
Figur 7.6: Skisse av løsning av de ulike problemene beskrevet i eksempel 7.5 ved anvendelse av optimeringsverktøyet CPLEX.

7.2.4 Sammenligning av ulike algoritmer i CPLEX

Eksempel 7.5: I dette eksempelet vil vi sammenligne beregningstiden ved anvendelse av den primal og dual simplex-algoritmen og barrier indrepunktsmetoden i CPLEX for en rekke problemer med ulik diskretiseringsgrad n. I hvert problem vil mengden P av punkter vi ønsker å tilnærme bestå av 10 tilfeldig valgte punkter som alle er gitt tilfeldige heltallige verdier mellom 0 og 100. Øvre og nedre skranke er satt til henholdsvis 100 og -100.

De ulike algoritmene i CPLEX returnerer identiske løsninger for hvert enkelt problem. Løsningen av de ulike problemene med ulik diskretiseringsgrad er skissert i figur 7.6.

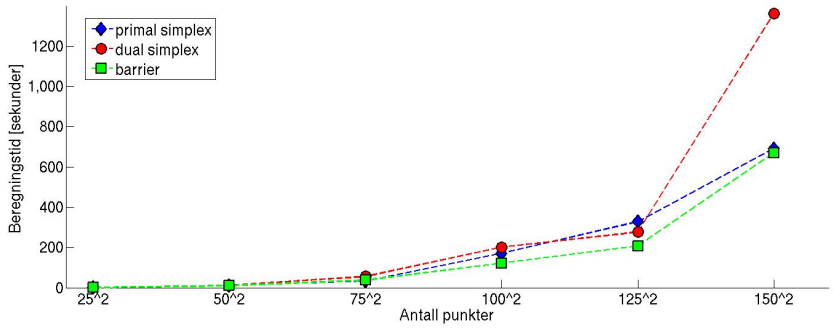
I figur 7.7 sammenligner vi beregningstiden for de ulike algoritmene anvendt på problemene beskrevet over. Tross noe usikkerhet, kan vi med bakgrunn i beregningstidene som skissert i figur 7.7 med rimelig sikkerthet hevde at barrier indrepunktsmetoden løser denne typen problemer mer effektivt enn den primale og den duale simpleksalgoritmen. Vi observerer at forskjellen i beregningstid ved anvendelse av de ulike algoritmene først bli tydelig når diskretiseringsgraden $n > 100$.



Figur 7.7: Viser sammenhengen mellom beregningstid og antall punkter i Ω_h for problemet beskrevet i eksempel 7.5 ved anvendelse av ulike algoritmer i CPLEX.

Eksempel 7.6: Vi avslutter denne seksjonen med et eksempel der betingelsene er identisk med betingelsene i eksempel 7.5, med unntak av antall punkter vi ønsker å tilnærme. I eksempel 7.5 bestod P av 10 tilfeldig valgte punkter. Nå lar vi P bestå av 1000 punkter. Vi konstruerer problemet ved å velge 1000 tilfeldige punkter og gir hvert punkt en tilfeldig heltallig verdi mellom 0 og 100. På den måten kan vi velge ut 1000 punkter vi ønsker å tilnærme også for problemet der Ω_h bare består av $25^2 = 625$ punkter. Beregningstiden for de ulike algoritmene i CPLEX er presentert i figur 7.8.

Vi hevdet i avsnitt 7.2.2 at store sparse problemer løses effektivt ved hjelp av barrier metoden. For problemer av typen beskrevet i eksempel 7.5 observerer vi av figur 7.7 at dette stemmer svært godt. Figur 7.8 indikerer at den primale simplex-algoritmen er omtrent like effektiv som barrier indrepunktsmetoden for problemer av typen beskrevet i eksempel 7.6. Til sammenligning antydes det at den duale simplex-algoritmen er betydelig



Figur 7.8: Viser sammenhengen mellom beregningstid og antall punkter i Ω_h for problemet beskrevet i eksempel 7.6 ved anvendelse av ulike algoritmer i CPLEX.

mindre effektiv for problemer av denne typen dersom diskretiseringsgraden er fin.

Igjen ser vi behov for å understreke at feilmarginen i målingen av beregningstidene for de ulike problemene er betydelig. I tillegg er utvalget av problemer svært lite, og våre konklusjoner er derfor kun ment å foreslå noen tendenser hva gjelder effektivitet av de ulike algoritmene.

7.3 Redusert LP-problem i MATLAB

Som nevnt i innledningen av kapittelet vil optimeringsproblemet (6.2) kunne inneholde et stort antall begrensninger og variable. Generelt kreves det mye regnekraft og lang beregningstid for å løse slike problemer, og vi er interessert i finne så effektive metoder som mulig for å løse problemet.

I denne seksjonen vil vi beskrive en metode der den inverse Poisson-matrisen benyttes for å redusere optimeringsproblemet (6.2) til et LP-problem med betydelig færre variable og begrensninger enn LP-problemet (7.1). Det reduserte problemet løses deretter ved hjelp av **linprog** som beskrevet i seksjon 7.1. Det reduserte LP-problemet er mindre, og vil kunne løses mer effektivt en det opprinnelige problemet, noe som vil bidra til redusert beregningstid. Samtidig vil det være svært tidkrevende å beregne den inverse Poisson-matrisen, spesielt for problemer med fin diskretisering.

Vi begynner med å redegjøre for hvordan optimeringsproblemet kan reduseres, før vi sammenligner resultater og beregningstider ved benyttelse av denne metoden for noen utvalgte eksempler.

7.3.1 Formulering av det reduserte optimeringsproblemet

I avsnitt 6.5 skisserte vi hvordan vi ved å innføre vekter $w(x_i, y_j)$ for $(x_i, y_j) \in \Omega_h$, der $w(x_i, y_j) = 0$ for $(x_i, y_j) \notin P$, kan skrive optimeringsproblemet (6.2) på formen

$$\begin{aligned} &\min \sum_{(x_i,y_j)\in\bar{\Omega}_h} w(x_i,y_j)\cdot|v_g(x_i,y_j)-\alpha(x_i,y_j)| \\ &\text{f.a.} \quad -\Delta_h v_g(x_i,y_j)=0 \qquad\qquad (x_i,y_j)\in\Omega_h \end{aligned} \tag{7.3}$$

der den eneste forskjellen er at vi nå summerer over alle punktene i $\bar{\Omega}_h$ og ikke bare punktene i P, i tillegg til innføringen av vektene w.

Ved å vektorisere $v_g(x_i,y_j)$, $g(x_i,y_j)$, $\alpha(x_i,y_j)$ og $w(x_i,y_j)$ til henholdsvis $\mathbf{x}$, $\mathbf{z}$, $\boldsymbol{\alpha}$ og $\mathbf{w}$ kan vi skrive (7.3) på formen

$$\begin{aligned} &\min \quad \mathbf{w}\cdot|\mathbf{x}-\boldsymbol{\alpha}| \\ &\text{f.a.} \quad \mathbf{Ax}=\mathbf{Qz} \end{aligned} \tag{7.4}$$

der $\mathbf{A}$ er Poisson-matrisen som definert i (3.8), $\mathbf{Q}$ er en permutasjonsmatrise, og absoluttverdien av en vektor er definert ved $|\mathbf{x}|=\begin{bmatrix}|x_1| & |x_2| & ... & |x_n|\end{bmatrix}$. Ved å multiplisere begrensningen med den inverse Poisson-matrisen på begge sider får vi følgende begrensning

$$\mathbf{x}=\mathbf{A}^{-1}\mathbf{Qz}$$

Vi kan nå substituere dette i objektivfunksjonen og får følgende optimeringsproblem uten begrensninger

$$\min \quad \mathbf{w}\cdot|\mathbf{A}^{-1}\mathbf{Qz}-\boldsymbol{\alpha}| \tag{7.5}$$

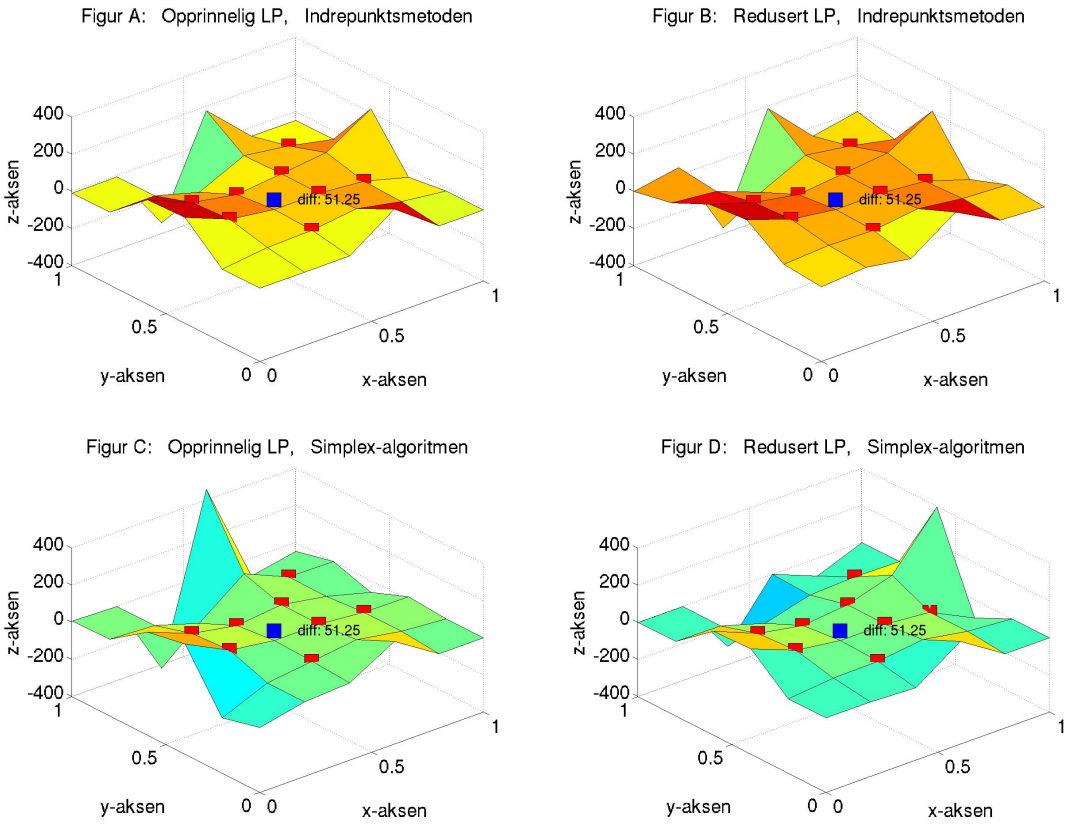
Ved å innføre hjelpevariable som forklart i avsnitt 6.1.1 kan (7.5) skrives som et LP-problem. Dette LP-problemet vil bestå av et redusert antall begrensninger og variable sammenlignet med metoden beskrevet i seksjon 7.1, og vil kunne løses effektivt ved hjelp av for eksempel **linprog**.

Som følge av den fine strukturen i Poisson-matrisen kan vi finne den inverse matrisen $\mathbf{A}^{-1}$ relativt effektivt [8]. Med fin diskretiseringsgrad n vil det tross dette være tidkrevende å beregne den inverse Poisson-matrisen som er en $n^2\times n^2$ matrise. Dersom flere beregninger med samme diskretiseringsgrad skal gjennomføres, vil det derfor kunne være hensiktsmessig å lagre matrisen slik at den kan anvendes på nytt. Dette vil bidra til å redusere beregningstiden betydelig.

7.3.2 Resultater

Vi vil her presentere løsninger og beregningstid for noen eksempler med optimeringsproblemer på formen (6.2), og sammenligne de ulike metodene. Vi begynner med å studere problemene i eksempel 7.1 - 7.3, før vi formulerer noen nye problemer.

Eksempel 7.1 - 7.3 (gjensyn): Løsningen av problemet i eksempel 7.1 ved anvendelse av indrepunktsmetoden og simplex-algoritmen for det opprinnelige og det reduserte LP-problemet er skissert i figur 7.9.



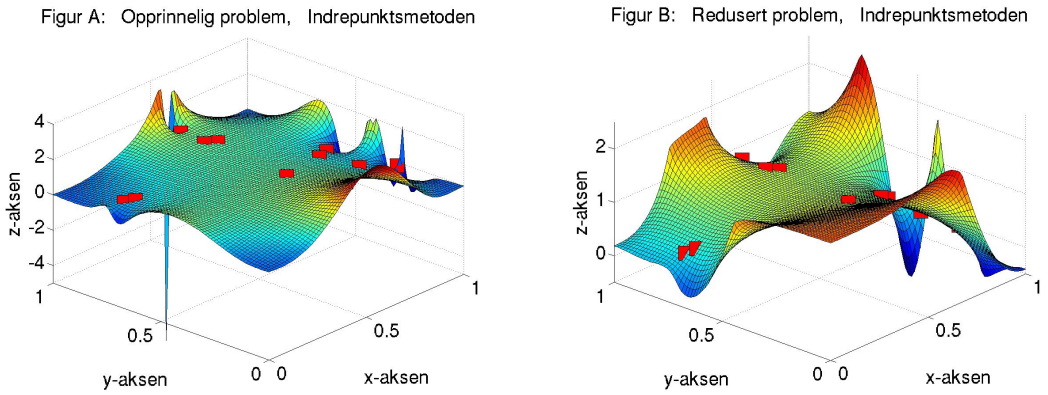
Figur 7.9: Viser løsning av optimeringsproblemet beskrevet i eksempel 7.1 ved bruk av indrepunktsmetoden (figur A og B) og simplex-algoritmen (figur C og D) i MATLAB for å løse det opprinnelige og det reduserte LP-problemet.

Vi observerer at løsningene er signifikant forskjellige, samtidig som den optimale verdien er den samme, og alle fire løsningene "bommer" på det samme punktet. Også de ulike løsningene av problemene i eksempel 7.2 og 7.3 er noe forskjellige. Beregningstider for de ulike metodene og problemene er presentert i tabell 7.3.

Eksempel 7.7 (75×75 punkter): Vi vil nå studere et par eksempler på problemer med finere diskretiseringsgrad. Vi lar $\bar{\Omega}_h$ bestå av 75×75 punkter og lar P bestå av 10 tilfeldige punkter i det indre av området. Punktene i P gis verdier $\alpha(x_i, y_j) = \sin(x_i h\pi)$. Vi har skissert løsningen av det opprinnelige og det reduserte LP-problemet ved anvendelse av indrepunktsmetoden og simplex-algoritmen i figur 7.10.

Også for dette problemet observerer vi at løsningene er svært forskjellige. Begge metodene returnerer optimal verdi tilnærmet 0. Beregningstiden for det reduserte LP-problemet ved anvendelse av indrepunktsmetoden og simplex-algoritmen er henholdsvis omkring 5.3 og 5.4 sekunder dersom den inverse Poisson-matrisen er beregnet og lagret på forhånd. Beregningstiden er likevel noe høyere enn beregningstiden for det opprinnelige

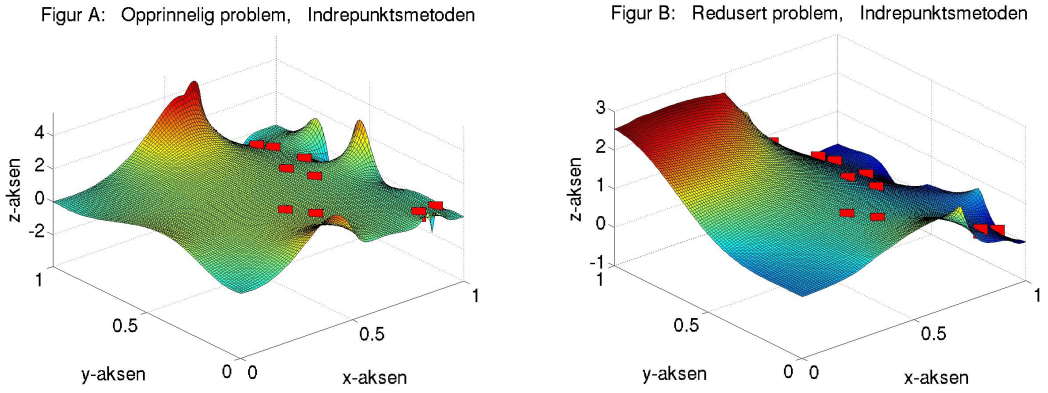
LP-problemet ved anvendelse av indrepunktsmetoden som er omkring 3.9 sekunder.



Figur 7.10: Viser løsningen av det opprinnelige og det reduserte LP-problemet i eksempel 7.7.

Eksempel 7.8 (100×100 punkter): Til slutt i denne seksjonen skal vi studere et problem der $\bar{\Omega}_h$ består av 100×100 punkter, og betingelsene ellers er som i eksempel 7.7. Løsningen av det opprinnelige og det reduserte LP-problemet er skissert i figur 7.11.

Med datamaskinen og programvaren vi har benyttet tar det omkring 10 minutter å beregne den inverse Poisson-matrisen vi trenger for å formulere det reduserte LP-problemet. Dersom denne er beregnet og lagret på forhånd, vil beregningstiden av det reduserte LP-problemet ved anvendelse av indrepunktsmetoden og simplex-algoritmen være omkring 20 sekunder. Til sammenligning er beregningstiden av det opprinnelige LP-problemet ved anvendelse av indrepunktsmetoden omkring 5 minutter. Forutsatt at den inverse Poisson-matrisen er beregnet og lagret på forhånd, kan beregningstiden altså reduseres drastisk for dette problemet. Dessverre krever lagring av den inverse Poisson-matrisen stor plass. Matrisen vi behøver for å beregne dette problemet opptar i underkant



Figur 7.11: Viser løsningen av det opprinnelige og det reduserte LP-problemet i eksempel 7.8

av 700 MB. Dersom $\bar{\Omega}_h$ i stedet hadde bestått av 125×125 punkter ville $\mathbf{A}^{-1}$ opptatt omkring 1.5 GB.

EKSEMPEL:	**7.1**	**7.2**	**7.3**	**7.7**	**7.8**
Antall punkter i $\bar{\Omega}_h$	6^2	25^2	50^2	75^2	100^2
Antall punkter i P:	9	200	10	10	10
Optimal verdi (tilnærmet):	51.3	1.3	0.0	0.0	0.0
Opprinnelig problem					
indrepunktsmetode	0.7	0.9	1.9	3.9	297.3
simplex	0.7	11.0	74.9	-	-
Redusert problem					
indrepunktsmetode					
uten beregning av $\mathbf{A}^{-1}$	0.8	4.1	2.6	5.3	18.8
med beregning av $\mathbf{A}^{-1}$	0.9	4.2	3.5	44.8	594.5
simplex					
uten beregning av $\mathbf{A}^{-1}$	0.7	3.0	2.5	5.4	20.6
med beregning av $\mathbf{A}^{-1}$	0.9	3.3	3.3	44.9	594.7

Tabell 7.3: Tabellen viser beregningstid i sekunder for løsning av problemene i eksempel 7.1 - 7.3 og 7.7 - 7.8 på opprinnelig og redusert form, ved anvendelse av ulike algoritmer i MATLAB.

7.4 Gradient-metoden

Metoden vi vil beskrive i denne seksjonen er en iterativ gradient-metode. Som i forrige seksjon vil vi ta utgangspunkt i det reduserte LP problemet

$$\min \quad f(\mathbf{z}) \quad = \quad \min \quad \mathbf{w} \cdot | \, \mathbf{A}^{-1}\mathbf{Q}\mathbf{z} - \boldsymbol{\alpha} \, | \tag{7.6}$$

men problemet løses på en ny måte. I uttrykket (7.6) angir $\mathbf{w}$ vekter, $\mathbf{A}^{-1}$ er den inverse Poisson-matrisen, $\mathbf{Q}$ er en permutasjonsmatrise, $\boldsymbol{\alpha}$ angir verdien av de punktene vi ønsker å tilnærme løsningen, og $\mathbf{z}$ er en vektorisering av variablene $g_{i,j}$ langs randa av området Ω_h.

I det følgende vil vi beskrive metoden og illustrere den med et enkelt eksempel, før vi anvender metoden på noen av problemene vi har studert i tidligere eksempler. Resultatene vil sammenlignes med resultatene returnert av metodene vi har betraktet tidligere.

7.4.1 En iterativ metode

Som nevnt er metoden vi vil beskrive her en iterativ metode. En iterativ metode begynner med en approksimasjon $\mathbf{z}^{(0)}$ til den eksakte løsningen $\mathbf{z}$. Startløsningen $\mathbf{z}^{(0)}$ kan

for eksempel være **0**-vektoren. Deretter beregnes en sekvens av løsninger $\{\mathbf{z}^{(k)}\}$ slik at $\mathbf{z}^{(k)} \rightarrow \mathbf{z}$. Iterative metoder anvendes vanligvis på store sparse systemer, og fordelen med disse metodene er hovedsakelig redusert lagringskrav. I tillegg er de ofte lette å implementere [8].

I gradient-metoden vi vil beskrive her er hvert nye ledd $\mathbf{z}^{(k+1)}$ i sekvensen gitt ved

$$\mathbf{z}^{(k+1)} \;=\; \mathbf{z}^{(k)} \;-\; \mu^{(k)} \nabla f(\mathbf{z}^{(k)}) \tag{7.7}$$

der $\mu^{(k)}$ betegner skrittlengde og $\nabla f(\mathbf{z}^{(k)})$ betegner retningsvektoren. Den optimale løsningen $\mathbf{z}$ angir optimale randbetingelser for det diskrete Laplace-problemet (3.3) som vi formulerte og studerte i kapittel 3. I kapittel 3 viste også hvordan Laplace-problemet kan løses, og slik finner vi den optimale løsningen $v_g(x_i, y_j)$ og vektoriseringen $\mathbf{x}$ som representerer verdiene i det indre av området, Ω_h.

Hensiktsmessig valg av skrittlengden $\mu^{(k)}$ er avgjørende for metodens effektivitet. Vi vil undersøke denne nærmere når vi studerer noen eksempler senere, men først vil vi konsentrere oss om hvordan retningsvektoren $\nabla f(\mathbf{z}^{(k)})$ bestemmes.

7.4.2 Hvordan bestemme retningsvektoren

Vi ønsker å benytte $\nabla f(\mathbf{z}^{(k)})$ som retningsvektorer for å finne nye løsninger $\mathbf{z}^{(k+1)}$ i sekvensen. Problemet er at $f(\mathbf{z})$ bare er stykkevis deriverbar på grunn av absoluttverdien. I det følgende vil vi vise hvordan vi kan omgå dette problemet slik at vi likevel kan beregne $\nabla f(\mathbf{z}^{(k)})$.

For å bidra til mer oversiktlig notasjon lar vi $\mathbf{R} = \mathbf{A}^{-1}\mathbf{Q}$, og lar $\mathbf{r}_i$ betegne radene i $\mathbf{R}$ slik at (7.6) kan skrives

$$\min \sum_{i=1}^{n^2} \mathbf{w}_i |\, \mathbf{r}_i\, \mathbf{z} - \boldsymbol{\alpha_i}| \tag{7.8}$$

For hver iterasjon k kan problemet (7.8) omformuleres ved å gruppere indeksene i i mengdene $I_+^{(k)}$, $I_-^{(k)}$ og $I_0^{(k)}$

$$\begin{aligned} I_+^{(k)} &= \{\, i : \mathbf{r}_i\, \mathbf{z}^{(k)} > \boldsymbol{\alpha}_i \,\} \\ I_-^{(k)} &= \{\, i : \mathbf{r}_i\, \mathbf{z}^{(k)} < \boldsymbol{\alpha}_i \,\} \\ I_0^{(k)} &= \{\, i : \mathbf{r}_i\, \mathbf{z}^{(k)} = \boldsymbol{\alpha}_i \,\} \end{aligned} \tag{7.9}$$

slik at vi for hver iterasjon k kan skrive (7.8) på formen

$$\min \Big[\sum_{i \in I_+^{(k)}} \mathbf{w}_i (\, \mathbf{r}_i\, \mathbf{z}^{(k)} - \boldsymbol{\alpha_i}) \;-\; \sum_{i \in I_-^{(k)}} \mathbf{w}_i (\, \mathbf{r}_i\, \mathbf{z}^{(k)} - \boldsymbol{\alpha_i}) \;+\; 0 \Big] \tag{7.10}$$

der vi har antatt at $I_0^{(k)}$ er tom. Med denne antagelsen har vi uttrykt optimeringsproblemet uten absoluttverdi, og objektivfunksjonen som minimeres i (7.10) er deriverbar.

Ved elementære derivasjonsregler er dermed retningsvektoren for hver iterasjon k gitt ved

$$\nabla f(\mathbf{z}^{(k)}) \;=\; \sum_{i \in I_+^{(k)}} \mathbf{w}_i \mathbf{r}_i \;-\; \sum_{i \in I_-^{(k)}} \mathbf{w}_i \mathbf{r}_i$$

7.4.3 Resultater

Til å begynne med vil vi illustrere metoden skissert over med en detaljert betraktning av et lite eksempel. Deretter vil ta et tilbakeblikk på noen av problemene vi har studert i eksempelene i foregående seksjoner og sammenligne de ulike metodene.

Eksempel 7.9 (Et lite eksempel for å illustrere metoden): Vi lar diskretiseringsgraden $n = 2$ som gir et område bestående av 4×4 punkter. Vi illustrerer punktene i P og verdiene vi ønsker å tilnærme følgende matrise

$$\begin{bmatrix} * & * & * & * \\ * & 2 & * & * \\ * & -3 & 9 & * \\ * & * & * & * \end{bmatrix}$$

der punktene som ikke er merket med stjerne har vekt $\mathbf{w}_i$=1, og punktene som er merket med stjerne har vekt $\mathbf{w}_i$=0.

Vi vil la skrittlengden $\mu^{(k+1)} = \mu^{(k)} \cdot 0.5$, slik at skrittlengden halveres for hver iterasjon. Til å begynne med lar vi skrittlengde $\mu^{(0)}$ være 30.

Iterasjon k=1 I begynnelsen av hver iterasjon beregnes produktet av matrisen $\mathbf{R}$ og løsningen $\mathbf{z}^{(k)}$. I dette tilfellet vil vektoren $\mathbf{z}$ representere 8 variable, og matrisen $\mathbf{R}$ vil være en 4×8 - matrise. Vi begynner med en startløsningen $\mathbf{z}^{(0)} = [0\ 0\ ...\ 0]^T$. Dette gir vektoren $\mathbf{Rz}$= $[0\ 0\ 0\ 0]^T$. Deretter grupperes indeksene i som beskrevet i (7.9)

$$I_+^{(0)} = \{3\} \qquad I_-^{(0)} = \{1, 4\} \qquad I_0^{(0)} = \emptyset$$

Fordi $\mathbf{w}_2$ =0 kan vi se bort i fra indeksen $i = 2$, som ellers ville ligget i mengden $I_0^{(0)}$.

Retningsvektoren $\nabla f(\mathbf{z}^{(0)})$ beregnes

$$\nabla f(\mathbf{z}^{(0)}) \;=\; \sum_{i \in I_+^{(0)}} \mathbf{w}_i\, \mathbf{r}_i \;-\; \sum_{i \in I_-^{(0)}} \mathbf{w}_i \mathbf{r}_i \;=\; 1 \cdot \mathbf{r}_3 \;-\; \big(1 \cdot \mathbf{r}_1 \;+\; 1 \cdot \mathbf{r}_4\big)$$

Vi lar $\mu^{(0)} = 30$. Dette gir en ny løsning

$$\mathbf{z}^{(1)} \;=\; \mathbf{z}^{(0)} \;-\; \mu^{(0)}\nabla f(\mathbf{z}^{(0)}) = \begin{bmatrix}0\\0\\0\\0\\0\\0\\0\\0\end{bmatrix} \;-\; 30 \begin{bmatrix}-0.2500\\-0.1250\\-0.2500\\-0.1250\\0.1250\\-0.2500\\0.1250\\-0.2500\end{bmatrix} \;=\; \begin{bmatrix}7.5000\\3.7500\\7.5000\\3.7500\\-3.7500\\7.5000\\-3.7500\\7.5000\end{bmatrix}$$

Løsningen $\mathbf{x}$ av Laplace-problemet med randbetingelser $\mathbf{z}^{(1)}$ er skissert til venstre i figur 7.12. Vi observere at verdien av denne løsningen er omtrent 10.6.

Iterasjon k=2 Vi begynner med å beregne produktet av matrisen $\mathbf{R}$ og $\mathbf{z}^{(1)}$, for så å gruppere indeksene i i mengden $I_+^{(1)}$, $I_-^{(1)}$ og $I_0^{(1)}$. Ved å følge prosedyren beskrevet over, og igjen se bort i fra indeksen $i = 2$, får vi

$$I_+^{(0)} = \{1, 3\} \qquad I_-^{(0)} = \{4\} \qquad I_0^{(0)} = \emptyset$$

som gir følgende retningsvektor

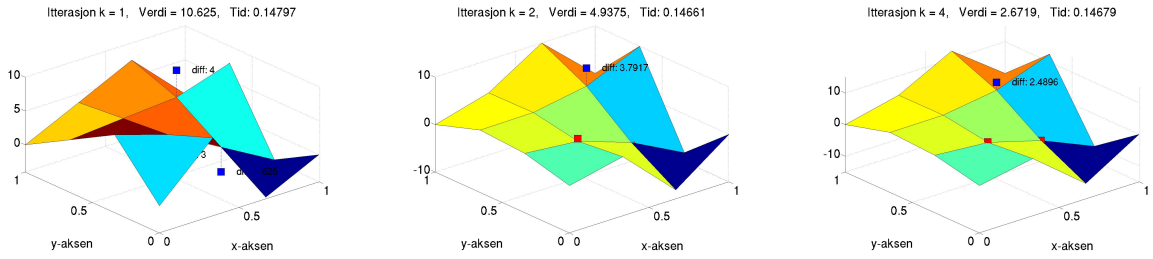
$$\nabla f(\mathbf{z}^{(1)}) \;=\; \sum_{i \in I_+^{(1)}} \mathbf{w}_i\, \mathbf{r}_i \;-\; \sum_{i \in I_-^{(1)}} \mathbf{w}_i \mathbf{r}_i \;=\; \big(1 \cdot \mathbf{r}_1 \;+\; 1 \cdot \mathbf{r}_3\big) \;-\; 1 \cdot \mathbf{r}_4$$

Skrittlengden $\mu^{(1)}$ er halvert siden forrige iterasjon. Det gir følgende løsning

$$\mathbf{z}^{(2)} \;=\; \mathbf{z}^{(1)} \;-\; \mu^{(1)}\nabla f(\mathbf{z}^{(1)}) = \begin{bmatrix}7.5000\\3.7500\\7.5000\\3.7500\\-3.7500\\7.5000\\-3.7500\\7.5000\end{bmatrix} \;-\; 15 \begin{bmatrix}0.3333\\0.0417\\0.3333\\0.0417\\0.2917\\-0.1667\\0.2917\\-0.1667\end{bmatrix} \;=\; \begin{bmatrix}2.5000\\3.1250\\2.5000\\3.1250\\-8.1250\\10.0000\\-8.1250\\10.0000\end{bmatrix}$$

Løsningen $\mathbf{x}$ av Laplace-problemet med randbetingelser $\mathbf{z}^{(2)}$ er skissert i midten av figur 7.12. Vi observere at verdien av denne løsningen er betydelig lavere enn forrige iterasjon med en verdi på omtrent 4.9.

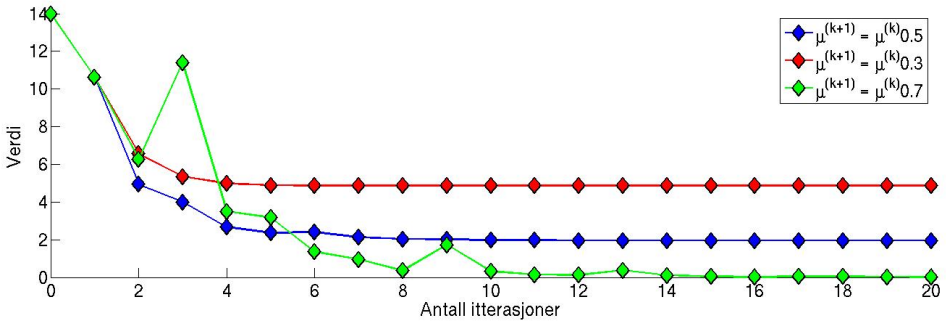
Iterasjon k=3,4,... Etter tre iterasjoner returneres verdien 4.0 og etter fire iterasjoner er verdien ytterligere redusert til 2.7. Løsningen etter fire iterasjoner er skissert til høyre i figur 7.12, og vi kan observere at 2 av punktene er svært nærme punktene vi ønsker å tilnærme.



Figur 7.12: Viser løsningen $\mathbf{x}$ etter 1, 2 og 4 iterasjoner. Boksene indikerer punktene vi ønsker å tilnærme. Dersom boksen er nær løsningen vi ønsker er den farget rød.

Verdien reduseres betydelig frem til omkring den tiende iterasjonen. Da er verdien 1.9806 og etter 20 iterasjon er verdien 1.9610. Etter 100 iterasjoner har verdien bare sunket til 1.9609 og den er den samme etter 1000 iterasjoner.

Ved å anvende en av metodene beskrevet i en av de foregående seksjonene finner vi raskt ut at det eksisterer en løsning av dette problemet med verdi tilnærmet 0. Lite hensiktsmessig valg av skrittlengden μ er trolig årsaken til at vi ikke lykkes med å finne en løsning som gir verdi tilnærmet 0. Allerede etter 20 iterasjoner er skrittlengden redusert fra 30 til 0.0001, og etter 50 iterasjoner er skrittlengden $1.0e-14$. Dermed vil de nye løsningene i sekvensen endre seg minimalt.



Figur 7.13: Viser verdien av løsningen som funksjon av antall iterasjoner med ulik skrittlengde μ.

Forskjellige skrittlengder

Ved å anvende andre skrittlengder μ vil vi kunne finne en løsning som ligger nærmere den optimale. Vi har benyttet et par andre skrittlengder, og plottet de ulike verdiene som funksjon av de første 20 iterasjonene i figur 7.13. I blått, med punktene markert med diamant, er løsningen med skrittlengden $\mu^{(k+1)} = \mu^{(k)}0.5$ som benyttet over skissert. Vi har også benyttet skrittlengder $\mu^{(k+1)} = \mu^{(k)}0.3$ og $\mu^{(k+1)} = \mu^{(k)}0.7$. Disse er skissert i henholdsvis rødt og grønt med sirkelformede og firkantede punkter.

Vi observerer at ved bruk av skrittlengden $\mu^{(k+1)} = \mu^{(k)}0.7$ finner vi verdier bety-

delig lavere optimal verdi enn ved bruk av de andre skrittlengdene. Løsningen etter de første fem iterasjonene har riktignok høyere verdi enn ved anvendelse av skrittlengden $\mu^{(k+1)} = \mu^{(k)}0.5$, men etter dette er gir skrittlengden $\mu^{(k+1)} = \mu^{(k)}0.7$ lavere verdi. Etter 20 iterasjoner er verdien 0.01.

Stopp-kriterie

Som vi har sett er det en stor utfordring å finne skrittlengder som gjør at metoden finner løsninger med verdi nær den optimale verdien på en effektiv måte. En annen utfordring er å bestemme når algoritmen skal terminere. Som regel kjenner vi ikke problemets optimale verdi, og vi kan derfor ikke benytte denne som stopp-kriterium. En mulighet er å stoppe algoritmen dersom den produserer en rekke nye løsninger som gir høyere verdi enn tidligere løsninger. Dersom vi ikke kjenner problemets optimale verdi har vi ingen mulighet til å kontrollere om løsningen som returneres gir optimal verdi, eller noe i nærheten av denne.

I denne oppgaven vil vi ikke gå dypere inn i disse nevnte utfordringene, og vi skal avslutte denne seksjonene med å presentere noen resultater fra beregninger gjort med gradient-metoden. Vi vil se nærmere på et par problemer vi har studert tidligere ved hjelp av sofistikerte optimerings-algoritmermetoder beskrevet i de foregående seksjonene, og vi vil bruke disse resultatene til å kontrollere de nye løsningene.

Eksempel 7.1 (gjensyn): Igjen skal vi betrakte problemet vi beskrev i eksempel 7.1. Mengden P består av ni tilfeldig valgte punkter, som er gitt heltallige verdier mellom 0 og 100. Diskretiseringsgraden n er 4 som gir et område $\bar{\Omega}_h$ bestående av 6×6 punkter.

Vi vil benytte en skrittlengde μ gitt ved $\mu^{(k)} = \max(\alpha) \cdot [(\kappa - k)/\kappa]^2$, der $\max(\alpha)$ er maksimal verdi av punktene i P vi ønsker å tilnærme, og κ er totalt antall iterasjoner.

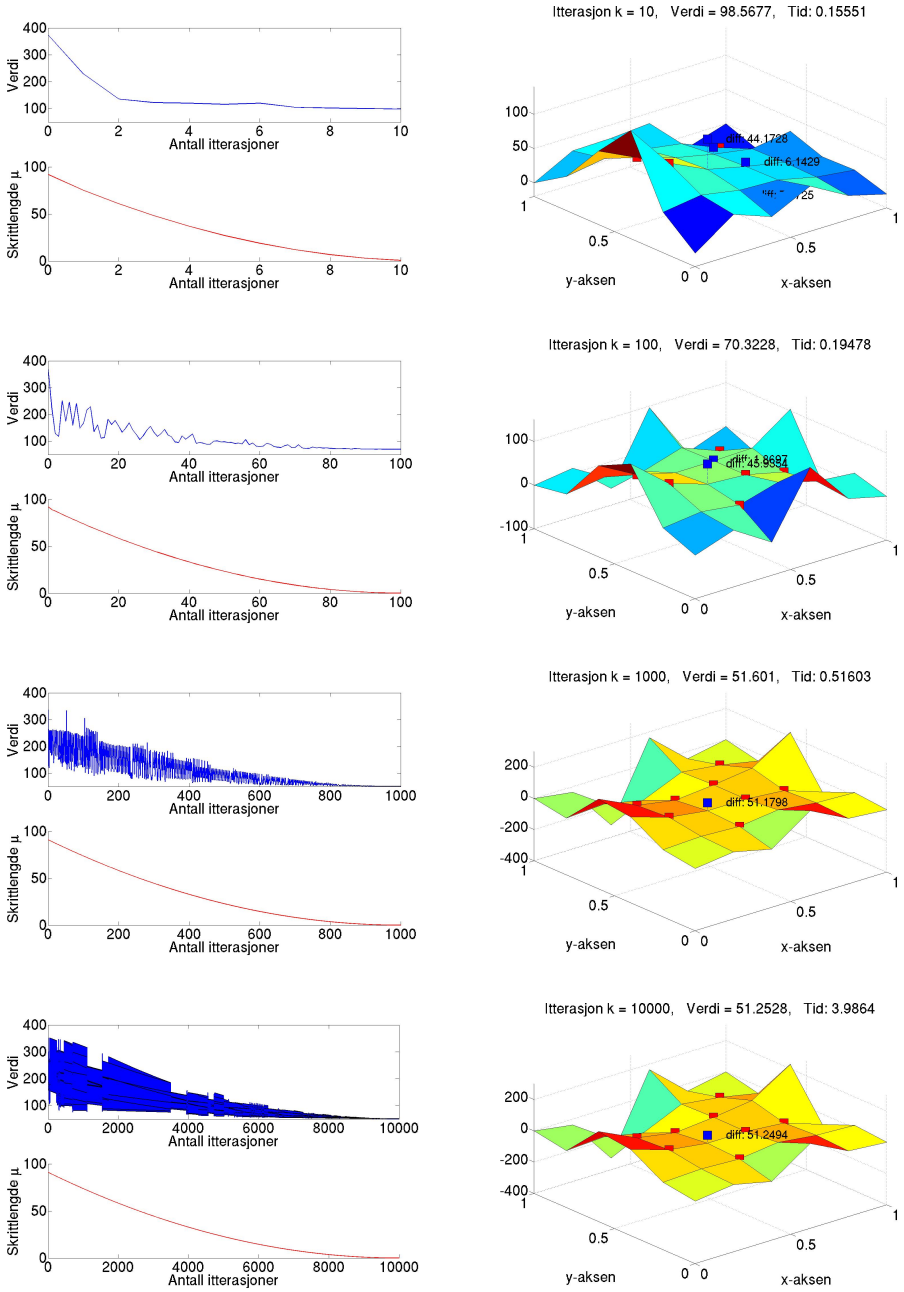
Løsningen $\mathbf{x}$ av problemet formulert i eksempel 7.1 med 10, 100, 1000 og 10000 iterasjoner er skissert i figur 7.14. I tilfellet der totalt antall iterasjoner, κ, er 1000 observerer vi at løsningen har verdi 51.60. Dette er svært nær verdien til løsningen der κ= 10000, og verdien er 51.25. Denne verdien sammenfaller med optimal verdi returnert ved anvendelse av metodene vi har beskrevet tidligere.

For κ= 1000 og 10000 observerer vi at løsningen tilsynelatende er identisk med løsningen returnert ved anvendelse av indrepunktsmetoden på det opprinnelige LP-problemet som beskrevet i seksjon 7.1. Vi skisserte løsningen ved anvendelse av indrepunktsmetoden i figur 7.1.

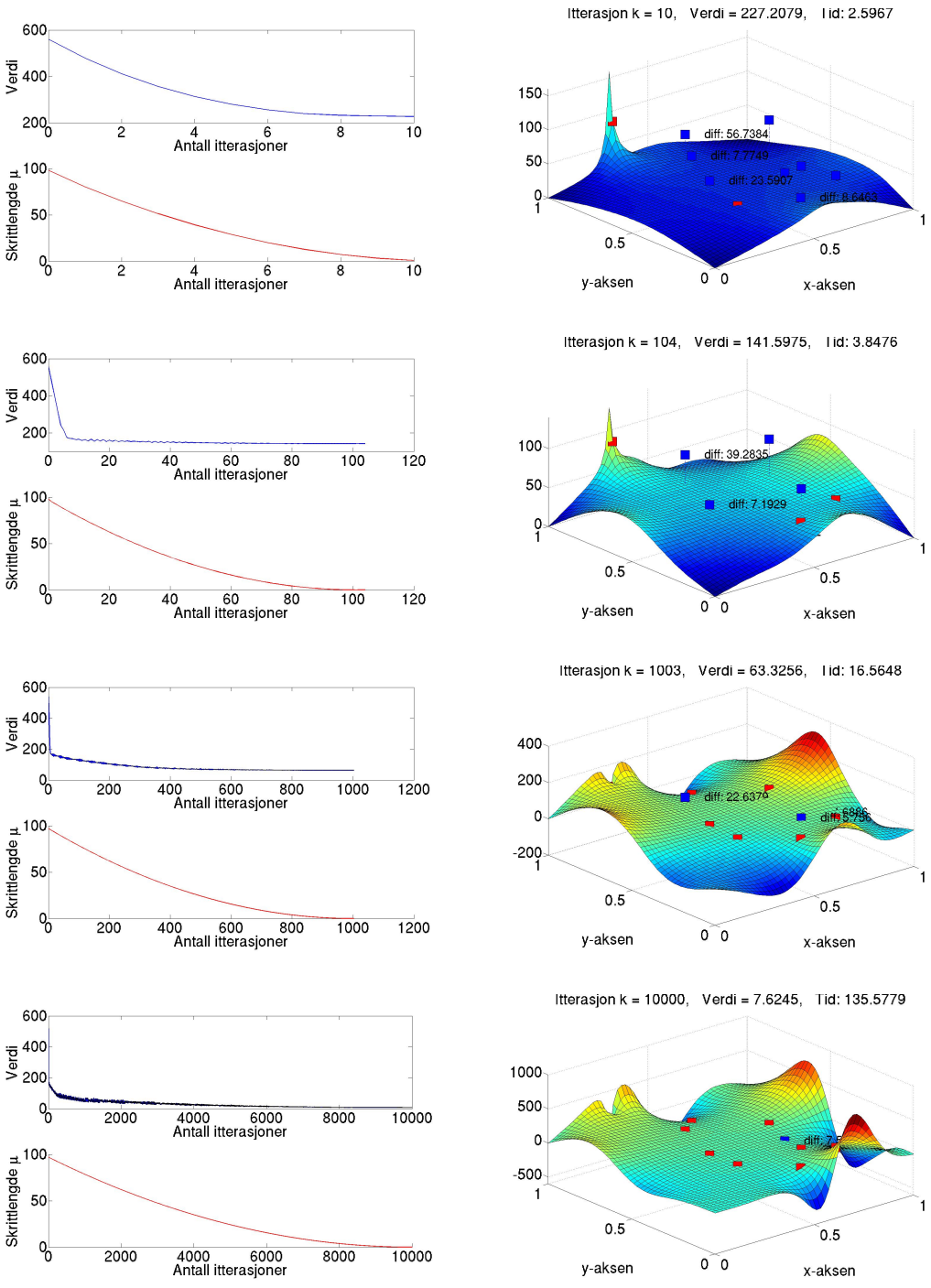
Eksempel 7.3 (gjensyn): Til slutt betrakter vi problemet formulert i eksempel 7.3. Mengden $\bar{\Omega}_h$ består av 50×50 punkter, og mengden P består av 10 punkter som er gitt heltallige verdier mellom 0 og 100.

Vi har valgt å benytte samme skrittlengde μ som i forrige eksempel. Løsningen av dette problemet etter 10, 104, 1003 og 10000 iterasjoner er skissert i figur 7.15. At vi presenterer 104 og 1003 i stedet for det mer opplagte valget 100 og 1000, har sin forklaring i algoritmen vi har benyttet. Algoritmen er skrevet slik at den kan fortsette noen ekstra iterasjoner dersom verdien er lavere enn den foregående.

Vi husker fra seksjon 7.1 at optimal verdi av problemet er tilnærmet 0. Fra figur 7.15 observerer vi at selv etter 10000 iterasjoner, er verdien fortsatt i overkant av 7.6. Vi observerer også at løsningen nesten er lik løsningen returnert ved anvendelse av metoden beskrevet i seksjon 7.1.



Figur 7.14: Til høyre er løsningen $\mathbf{x}$ av problemet formulert i eksempel 7.1 i avsnitt 7.1 med 10, 100, 1000 og 10000 iterasjoner skissert. Øverst til venstre er verdien som funksjon av antall iterasjoner plottet, og nederst til venstre er skrittlengden μ som funksjon av antall iterasjoner plottet.



Figur 7.15: Til høyre er løsningen $\mathbf{x}$ av problemet formulert i eksempel 7.3 i avsnitt 7.1 med 10, 104, 1003 og 10000 iterasjoner skissert. Øverst til venstre er verdien som funksjon av antall iterasjoner plottet, og nederst til venstre er skrittlengden μ som funksjon av antall iterasjoner plottet.

Kapittel 8

Oppsummering og konklusjon

Hovedmålet med denne oppgaven har vært å studere optimeringsproblemet (6.2), som gir optimale Dirichlet-randbetingelser til det diskrete Laplace-problemet. Vi har formulert optimeringsproblemet matematisk og forklart hvordan dette optimeringsproblemet kan skrives som et lineært optimeringsproblem med passende variable. Videre har vi presentert en rekke metoder som tar i bruk forskjellige optimeringsalgoritmer, og drøftet og sammenlignet effektiviteten til de ulike metodene. Vi har også viet mye konsentrasjon til å studere ulike egenskaper ved løsningen av det diskrete Laplace-problemet.

Det er viktig å bemerke at optimeringsproblemet (6.2) kan defineres i flere dimensjoner, selv om vi i denne oppgaven hele tiden har konsentrert oss om det todimensjonale tilfellet. For å forenkle problemet videre har vi kun betraktet kvadratiske områder, nærmere bestemt enhetskvadratet.

8.1 Løsningens egenskaper

Etter at vi i kapittel 2 og 3 definerte og studerte det kontinuerlige og det diskrete Laplace-problemet med Dirichlet-randbetingelser i to dimensjoner, ønsket vi i kapittel 4 å se nærmere på de ulike egenskapene ved løsningen av det diskrete Laplace-problemet. Vi innledet kapittelet ved å vise at dersom løsningen i et punkt krummer oppover i en retning (for eksempel x-retning), vil den krumme nedover i den andre retningen (y-retning) i dette punktet.

Deretter formulerte og studerte vi problemet vi kalte treffpunktsproblemet (4.1). Vi viste blant annet at dersom vi er gitt 4 punkter i det kvadratiske området med vilkårlige verdier, kan vi alltid finne randverdier g slik at løsningen v_g av det diskrete Laplace-problemet skjærer disse punktene. Videre viste vi at dette ikke alltid er mulig dersom vi i stedet er gitt 5 eller flere punkter. Motivert av dette definerte vi i kapittel 6 optimeringsproblemet (6.2) som lar oss finne randverdier som gir en løsning av det diskrete Laplace-problemet som er en så god tilnærming til verdiene i de gitte punktene som mulig.

8.2 Ulike varianter av optimeringsproblemet

Vi har formulert og redegjort for ulike varianter av optimeringsproblemet, og presentert en rekke eksempler. Vi begynte med å forklare hvordan optimeringsproblemet kunne formuleres som et lineært optimerings problem, LP-problem, og argumenterte for at optimeringsproblemet alltid har en optimal løsning. Denne løsningen er den løsningen av det diskrete Laplace-problemet som best tilnærmer de ønskede verdiene for de gitte punktene. Vi har studert en rekke eksempler der denne optimale verdien har vært 0, og vist at i slike tilfeller vil det korresponderende treffpunktsproblemet være konsistent.

Vi har også studert eksempler der den optimale verdien er betydelig høyere enn 0, og vi har sett at den optimale løsningen av optimeringsproblemet i slike tilfeller ikke treffer mange av de ønskede verdiene.

8.2.1 Løsninger med andre egenskaper

Vi har viet en del konsentrasjon til å studere ulike løsninger som alle er optimale, men som i tillegg har andre egenskaper. Blant annet har vi studert et problem der summen av verdiene langs randa av området var så liten som mulig. Dette kan for eksempel være hensiktsmessig dersom vi ønsker å oppnå en gitt temperaturfordeling i et område uten å benytte mer energi til oppvarming og nedkjøling enn nødvendig.

Gjennom vårt arbeid har vi sett mange eksempler der variasjonene mellom nabopunkter langs randa har vært svært stor, og i mange praktiske tilfeller vil det være ønskelig å begrense denne avstanden. Motivert av dette formulerte vi en versjon av optimeringsproblemet der denne avstanden var begrenset. Vi illustrerte det nye problemet med et eksempel der avstanden mellom nabopunktene i utgangspunktet var stor. Ved å innføre de nye begrensningene observerte vi at vi kunne oppnå betydelig reduksjon i variasjonen mellom nabopunktene uten at den optimale verdien økte mye.

Vi har sett flere eksempler på løsninger som har omtrent samme verdi samtidig som andre viktige egenskaper kan være svært forskjellig. I praktiske tilfeller vil vi derfor anta at det ofte vil være av stor betydning å undersøke de ulike optimale løsningene av optimeringsproblemet. I mange tilfeller vil det for eksempel være hensiktsmessig å kombinere problemet der variasjonen mellom nabopunktene begrenses, med problemet der summen av verdiene langs randa av området er minimal.

8.2.2 Vekting av treffpunkter

Vi har også formulert en generalisering av optimeringsproblemet som tar hensyn til vekting av de ønskede verdiene i de gitte punktene. På den måten kan vi for eksempel formulere en ønsket temperaturfordeling der noen av punktene i området er viktigere å "treffe" enn andre, og løsningen som returneres vil dermed ligge nærmest de "viktigste" punktene.

8.3 Ulike metoder for å løse optimeringsproblemet

I kapittel 7 presenterte vi en rekke metoder for å løse optimeringsproblemet (6.2). Metodene tar utgangspunkt i formuleringen av optimeringsproblemet som et lineært optimeringsproblem. I seksjon 7.1 og 7.2 beskrev vi en metode for å løse det lineære optimeringsproblemet (6.4) i MATLAB og CPLEX. I seksjonen etter beskrev vi hvordan vi ved å beregne den inverse Poisson-matrisen kan redusere LP-problemet slik at det består av færre variable og begrensninger, før dette reduserte problemet løses i MATLAB. Også ved anvendelse av den siste metoden vi beskrev beregnes den inverse Poisson-matrisen. Denne metoden er en iterativ gradient-metode.

Ved bruk av MATLAB og CPLEX kan ulike algoritmer anvendes for å løse lineære optimeringsproblemer. Vi har valgt å begrense oss til å studere indrepunktsmetoden og simplex-algoritmen, og løst en rekke ulike eksempler ved anvendelse av de ulike metodene og algoritmene. Vi har observert store ulikheter hva gjelder beregningstid. Som nevnt tidligere vil mye av forskjellene potensielt ha sin forklaring i datautstyret som er benyttet, men vi tør hevde at våre beregninger likevel gir grunnlag for å sammenligne de ulike metodenenes effektivitet. Beregningene er foretatt flere ganger, og aritmetisk middel av beregningstidene oppgis.

8.3.1 Effektivitet

Tabellen i tillegg A inneholder beregningstider for et utvalg eksempler ved anvendelse av de forskjellige algoritmene og metodene vi har benyttet. Beregningstidene antyder at indrepunktsmetoden i MATLAB anvendt på det originale LP-problemet ser ut til å være mest effektiv for eksemplene vi har studert. Resultatet stemmer godt med vår antagelse om at MATLAB løser store sparse problemer effektivt.

Metoden beskrevet i seksjon (7.3) som løser det reduserte LP-problemet, er relativt effektiv så lenge vi ser bort ifra beregning av den inverse Poisson-matrisen. Denne matrisen kan lagres, men den vil kreve svært mye harddiskplass. Allerede for et grid bestående av $100\times$ 100 punkter krever den inverse Poisson-matrisen opp mot 700 MB lagringsplass, og for et grid bestående av $125\times$ 125 punkter kreves omkring 1.5 GB. Matrisen må i tillegg uansett lastes inn i datamaskinens minne noe som også er tidkrevende. Styrken med metoden er at det reduserte optimeringsproblemet består av få begrensninger og variable selv for fin diskretiseringsgrad, og kan derfor løses svært effektivt.

Det har kommet tydelig frem at simplex-algoritmen i MATLAB er lite effektiv sammenlignet med indrepunktsmetoden. I CPLEX er de ulike algoritmene tilsynelatende omkring like effektive, men som vi viste i figur 7.7 og 7.8 er indrepunktsmetoden også her noe mer effektiv enn den primale og den duale simplex-algoritmen.

I arbeidet med denne oppgaven har vi måttet begrense oss til å studere et lite utvalg eksempler. Videre forsøk er derfor nødvendige for å bekrefte tendensene vi har observert.

Tillegg A

Tabell med beregningstider

Følgende tabeller er vedlagt:

Tabellene viser beregningstid i sekunder for en et utvalg av problemene vi har formulert og studert. Tidene er beregnet ved hjelp av stoppeklokkefunksjoner i MATLAB og CPLEX, og beregningstdien vil derfor kunne variere betydelig mellom ulike datamaskiner.

A.1 Beregningstider ved anvendelse av ulike metoder og algoritmer

EKSMPEL:	**7.1**	**7.2**	**7.3**	**7.4**	**7.7**	**7.8**
Antall punkter i $\hat{\Omega}_h$:	6^2	25^2	50^2	100^2	75^2	100^2
Antall punkter i P:	9	200	10	50	10	10
Optimal verdi (tilnærmet):	51.3	1.3	0.0	0.0	0.0	0.0
Opprinnelig LP i MATLAB						
indrepunktsmetode	0.7	0.9	1.2	7.8	3.9	10.9
simplex	0.7	11.0	74.9	1647.9	-	-
Opprinnelig LP i CPLEX						
simplex primal	0.2	0.7	6.2	173.6	38.8	131.5
simplex dual	0.2	0.7	7.7	176.0	38.1	127.8
barrier indrepunktsmetode	0.2	0.7	6.7	185.4	41.3	128.0
Redusert LP i MATLAB						
indrepunktsmetode						
uten beregning av $\mathbf{A}^{-1}$	0.8	4.1	2.6	47.2	5.3	18.8
med beregning av $\mathbf{A}^{-1}$	0.9	4.2	3.5	622.9	44.8	594.5
simplex						
uten beregning av $\mathbf{A}^{-1}$	0.7	3.0	2.5	52.9	5.4	20.6
med beregning av $\mathbf{A}^{-1}$	0.9	3.3	3.3	628.6	44.9	594.7

Tabell A.1: Tabellen viser beregningstider i sekunder for problem 7.1 - 7.4 og 7.7 - 7.8 ved anvendelse av ulike algoritmer. Matrisen $\mathbf{A}$ betegner Poisson-matrisen beskrevet i kapittel 3.3.1

A.2 Beregningstider for eksempel 7.5 ved anvendelse av CPLEX

EKSMPEL:	**7.5 A**	**7.5 B**	**7.5 C**	**7.5 D**	**7.5 E**	**7.5 F**
Antall punkter i $\hat{\Omega}_h$:	25^2	50^2	75^2	100^2	125^2	150^2
Antall punkter i P:	10	10	10	10	10	10
Optimal verdi (tilnærmet):	107.1	110.6	26.4	86.2	67.3	110.4
Opprinnelig LP i CPLEX						
simplex primal	0.7	10.1	42.6	88.0	300.5	641.2
simplex dual	0.8	9.8	47.6	83.2	326.4	598.5
barrier indrepunktsmetode	0.8	7.8	43.3	68.4	184.6	405.7

Tabell A.2: Tabell som viser beregningstider i sekunder for problemet beskrevet i eksempel 7.5 for ulik diskretiseringsgrad n.

A.3 Beregningstider for eksempel 7.6 ved anvendelse av CPLEX

EKSMPEL:	**7.6 A**	**7.6 B**	**7.6 C**	**7.6 D**	**7.6 E**	**7.6 F**
Antall punkter i $\hat{\Omega}_h$:	25^2	50^2	75^2	100^2	125^2	150^2
Antall punkter i P:	1000	1000	1000	1000	1000	1000
Optimal verdi (tilnærmet):	20680	22887	21756	22208	23235	22562
Opprinnelig LP i CPLEX						
simplex primal	1.4	11.4	33.6	170.2	329.1	691.8
simplex dual	1.2	11.3	56.2	201.7	276.3	1362.8
barrier indrepunktsmetode	1.1	9.9	37.9	121.2	208.3	670.5

Tabell A.3: Tabell som viser beregningstider i sekunder for problemet beskrevet i eksempel 7.6 for ulik diskretiseringsgrad n.

Tillegg B

Beskrivelse av programet OBVLP

Vi har skrevet et program **Optimal Boundary Values for the Poisson Problem - OBVLP** med grafisk brukergrensesnitt som kan benyttes for å løse de ulike problemene på formen (6.2) som vi har studert i denne oppgaven. I tillegg til generelle problemer kan de fleste problemene vi har presentert i eksempelene løses og skisseres med en tredimensjonal modell.

Nedlasting

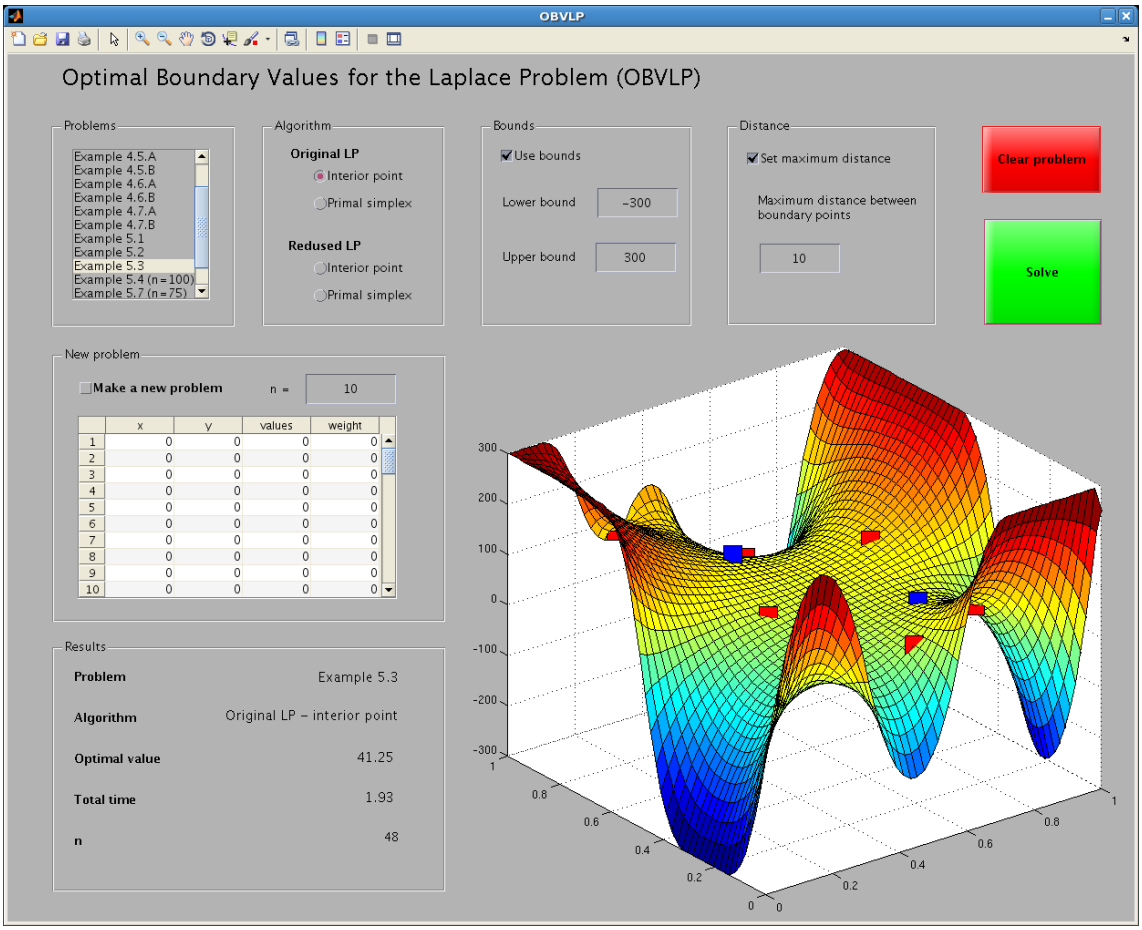
Programmet **OBVLP** kan lastes ned fra internettsiden http://www.brandseter.com/OBVLP.html. For at programmet skal fungere må MATLAB og toolboxen **optimtool** være installert på forhånd. **OBVLP.zip** lastes ned og pakkes ut i en valgfri mappe. Sett "Current Folder" i MATLAB til denne mappa, og gi kommandoen "OBVLP" i kommando-vinduet i MATLAB. Et vindu som vist i figur B.1 vil vises.

Forklaring

Øverst til venstre kan man velge hvilket eksempel man vil studere. Dersom man ønsker å lage et nytt problem eller endre på et av eksempelne huker man av i boksen "Make a new problem", velger diskretiseringsgrad n og angir punkter (x, y) med tilhørende verdier og vekt.

Ulike algoritmer for å løse det originale og det reduserte optimeringsproblemet kan benyttes. Vi gjør oppmerksom på at beregningstiden kan være lang for enkelte problemer, spesielt ved anvendelse av simplex-algoritmen ved løsning av det originale problemet.

I boksen "Bounds" kan øvre og nedre skranke settes. Maksimal avstand mellom randpunktene kan begrenses ved å sette "maximum distance" M_d. Da suppleres det originale optimeringsproblemet (6.4) med betingelser som begrenser absoluttverdien av differansen mellom verdien i et punkt og gjennomsnittet av nabopunktene langs randa; $|v_g(x_{i-1}, y_j) - 2v_g(x_i, y_j) + v_g(x_{i+1}, y_j)| \leq M_d$. Se avsnitt 6.4.



Figur B.1: Skjermdump av OBVLP.m

Resultater

Programmet skisserer løsningen i området nederst til høyre. Ved å benytte verktøyene i verktøymenyen øverst kan man blant annet rotere modellen av løsningen med verktøyet "Rotate 3D". Optimal verdi og beregningstid returneres når man trykker på "Solve". Dersom algoritmen ikke finner en optimal løsning blir det opplyst om dette i boksen "Results".

Tillegg C

Programkoder

Følgende programkoder er vedlagt:

C.1 5-punkts approksimasjon

Programkode for å løse Laplaceproblemet med Dirichlet-randbetingelser (3.3) ved 5-punkts approksiamsjon i MATLAB.

DIRICHLET.m

DIRICHLET.m

```
function [V] = DIRICHLET(gt,gb,gv,gh)

% ***************  LAPLACE-PROBLEMET MED DIRICHLET-BETINGELSER  **********

% 5-punkt approksimasjon til løsning av det diskrete Laplace-problemet med
% Dirichlet-randbetingelser, dvs:

%   -Laplacian u(x,y) = 0        for (x,y) i Omega
%              u(x,y) = g        for (x,y) i randa til Omega

% der g er gitt ved gt,gb,gv,gh, og Omega er et kvadratisk omraade.

n=length(gt)-1;
h = 1/(1+n);

% ***  Lager matrisen F med funksjonsverdier fra randa  ***

F=zeros(n-1,n-1);
F(1,1:n-1)      = F(1,1:n-1)    + 1/(h^2)*gb(2:n) ;      % bunn
F(1:n-1,1)      = F(1:n-1,1)    + 1/(h^2)*gv(2:n)';      % venstre
F(1:n-1,n-1)    = F(1:n-1,n-1)  + 1/(h^2)*gt(2:n)';      % topp
F(n-1,1:n-1)    = F(n-1,1:n-1)  + 1/(h^2)*gh(2:n) ;      % høyre


% ***  Løser TV + VT = h^2 F  ***

V=fastpoisson(F);
V(1,1:n+1)=gv;
V(n+1,1:n+1)=gh;
V(1:n+1,1)=gb;
V(1:n+1,n+1)=gt;


% ***  Skriver ut løsningen  ***

x1=0:1/(n):1;
y1=0:1/(n):1;
surf(x1,y1,V)
title(['Nummerisk løsning av Laplace-problemet gitt randverdier. Metode:
    fastpoisson.m, n=',num2str(n)])
xlabel('y-direction')
ylabel('x-directon')
end
```

DIRICHLET.m

C.2 Fast Poisson Solver basert på diagonalisering

fastpoisson.m

```
function [V]=fastpoisson(F)
m=length(F); h=1/(m+1); hv = pi*h*(1:m)';
sigma=sin(hv/2).^2;
S=sin(hv*(1:m));
G=S*F*S;
X = h^4*G./(sigma*ones(1,m)+ ones(m,1)*sigma');
V= zeros(m+2,m+2);
V(2:m+1,2:m+1)=S*X*S;
```

fastpoisson.m

C.3 Optimeringsproblemet på matriseform

Programkode for å skrive optimeringsproblemet (6.4) på formen (7.2) slik at det kan løses av MATLAB-funksjonen **linprog.m**.

makeMatrix.m

```
function [c,Ain,bin,Aeq,beq]= makeMatrix(W,nn,doWeight)

% Definerer antall variable og storrelsen paa matrisen Ain og bin

    antall_delta=size(W,1);
    antall_delta_eq=antall_delta*2;
    antall_v=nn^2;
    antall_v_eq=(nn-2)^2;          %2*(nn-2)^2; er pga ikke lenger ulikhet men
        likhet
    antall_var=antall_v+antall_delta;

    bin=zeros(antall_delta_eq,1);
    Ain=zeros(antall_delta_eq,antall_var);

    beq=zeros(antall_v_eq,1);
    Aeq=zeros(antall_v_eq,antall_var);


    % Setter inn for v-ene i Aeq
    k=1;
    for ii=2:nn-1
       for jj=2:nn-1
        Aeq(k,(ii-1)*nn+jj)= 4;
        Aeq(k,(ii-1)*nn+jj-1)=-1;
        Aeq(k,(ii-1)*nn+jj+1)=-1;
        Aeq(k,(ii-2)*nn+jj)=-1;
        Aeq(k,(ii-0)*nn+jj)=-1;
        k=k+1;
       end
    end


    % Setter inn for delta-ene i Ain og bin
    k=0;
    for ii=1:(size(W,1))
        bin(k+ii)=  W(ii,3);
        bin(k+ii+1)=-W(ii,3);

        Ain(k+ii,    (W(ii,1)-1)*nn+W(ii,2))=1;
        Ain(k+ii+1,  (W(ii,1)-1)*nn+W(ii,2))=-1;

        Ain(k+ii,    ii+nn^2)=-1;
        Ain(k+ii+1,  ii+nn^2)=-1;

        k=k+1; % siden oppdaterer hver runde med (k+ii) og (k+ii+1)
    end

    % Definere c
    c=zeros(antall_var,1);
    for k=(antall_v+1):(antall_var)
            c(k)=1;
    end
end
```

makeMatrix.m

C.4 Programkode for løsning i OPL-CPLEX

Programkoden **Kontroll.mod** under viser løsning av lineære optimeringsproblemer på formen (7.1) i OPL-CPLEX.

Programmet henter inn data og innstillinger for et spesielt problem fra en **.dat**-fil og en **.ops**-fil. **.dat**-fila inneholder en matrise **W** med informasjon om hvilke punkter vi ønsker å tilnærme samt diskretiseringsgrad, mens **.ops**-fila inneholder informasjon om hvilke algoritmer og typer av disse som skal anvendes av CPLEX.

Kontroll.mod

```
/*********************************************
 * OPL 12.2 Model
 * Author: andrbran
 * Creation Date: 2. feb.. 2011 at 11.05.11
 *********************************************/


int n =...;
int antall_v= n*n;
range ver=1..antall_v;

int antall_delta=...;
range col =1..3;
range P=1..antall_delta;

float W[P][col]=...;

// Lager Poisson-matrisen

int antall_veq=(n-2)*(n-2);
range veq=1..antall_veq;
int Aeq[veq][ver];

range loop=2..n-1;
int k=1;
execute {
  for(var i in loop){
    for(var j in loop){
      Aeq[k][(n*(i-1)+j)]=4;
      Aeq[k][(n*(i-2)+j)]=-1;
```

```
      Aeq[k][(n*(i-0)+j)]=-1;
      Aeq[k][(n*(i-1)+j-1)]=-1;
      Aeq[k][(n*(i-1)+j+1)]=-1;
      k=k+1;
    }
  }
}


// Definerer variable

dvar float v[ver];
dvar float+ delta[P];


// Utfører optimeringen. merk: Funksjon ftoi konverterer flytall til heltall.

minimize sum(j in P) 1*delta[j];
subject to{
  forall(i in P){
     v[(n*(ftoi(W[i][1])-1)+ftoi(W[i][2]))] -delta[i] <=  W[i][3];
    -v[(n*(ftoi(W[i][1])-1)+ftoi(W[i][2]))] -delta[i] <= -W[i][3];
  }
  forall(i in veq){
  sum(j in ver) Aeq[i][j]*v[j] ==0;
  }

}
```

Notasjon

Diskretisering

Ω	kontinuerlig kvadratisk område
$\partial\Omega$	randa til området Ω
$\bar{\Omega}$	$\Omega \cup \partial\Omega$
Ω_h	diskretisert kvadratisk område
$\partial\Omega_h$	randa til området Ω_h
$\bar{\Omega}_h$	$\Omega_h \cup \partial\Omega_h$
n	diskretiseringsgrad
h	$h = 1/(n+1)$
x og y	kontinuerlige variable
$x_i = ih$ og $y_j = jh$	diskret variabele
$u(x, y)$	kontinuerlig funksjon definert for $(x, y) \in \Omega$
$v(x_i, y_j) = v_{i,j}$	diskret funksjon definert for $(x_i, y_j) \in \Omega_h$
$\mathbf{x} = \mathbf{x}_k$	vektorisering av $v_{i,j}$ definert i Ω_h

Poisson- og Laplace-problemet

f	$f : \Omega \to \mathbb{R}$, høyre side i Poisson-problemet
g	$g : \partial\Omega \to \mathbb{R}$, Dirichlet randbetingelser
$\mathbf{z} = \mathbf{z}_k$	vektorisering av $g_{i,j}$ definert i $\partial\Omega_h$
$\Delta u(x, y)$	kontinuerlig Laplace-operator: $\frac{\partial^2 u(x,y)}{\partial x^2} + \frac{\partial^2 u(x,y)}{\partial y^2}$
$\Delta_h v(x_i, y_j)$	diskret Laplace-operator: $\frac{1}{h^2}[4v_{i,j} - v_{i-1,j} - v_{i+1,j} - v_{i,j-1} - v_{i,j+1}]$
$T(x), H(y), B(x), V(y)$	randfunksjon for henholdsvis topp, høyre, bunn, venstre

Optimering

$\mathbf{c}$	objektivfunksjon
$\mathbf{A}_{eq}$	matrise med lineære begrensninger, likheter
$\mathbf{b}_{eq}$	vektor med lineære begrensninger, ulikheter
$\mathbf{A}_{in}$	matrise med lineære begrensninger, likheter
$\mathbf{b}_{in}$	vektor med lineære begrensninger, ulikheter
lb	nedre skranke
ub	øvre skranke

Optimale randbetingelser

$\Phi(g)$	objektivfunksjon i optimeringsproblemet 6.2
S	delmengde av Ω
P	endelig delmengde av Ω_h. Approksimasjon til S
$\gamma(x, y)$	analytisk funksjon $\gamma(x, y) : S \to \mathbb{R}$ med ønskede verdier
$\alpha(x_i, y_j)$	diskret funksjon $\alpha : P \to \mathbb{R}$ med ønskede verdier
$w(x_i, y_j)$	diskret funksjon $w : P \to \mathbb{R}$ med vekting
δ	modelleringsvariabel
v^*	optimal verdi
v_r^*	relativ optimal verdi ($v_r^* = v^*/\lvert P\rvert$)
$\mathbf{W}$	matrise med verdiene fra $\alpha(x_i, y_j)$ og vekting
$\mathbf{A}$	Poissonmatrisen $\mathbf{A}$=$\mathbf{T}_1 \otimes \mathbf{I} + \mathbf{I} \otimes \mathbf{T}_1$
$\mathbf{Q}$	permutasjonsmatrise
$\mathbf{R}$	$\boldsymbol{R = A^{-1}Q}$
$\boldsymbol{r}_i$	rad i av matrisen $\boldsymbol{R}$

Gradientmetoden

$\boldsymbol{\nabla f}(\mathbf{z}^{(k)})$	retningsvektor
$\boldsymbol{\mu}^{(k)}$	skrittlengde
$I_+^{(k)}$	mengde av indekser, $I_+^{(k)} = \{\, i : \mathbf{r}_i\, \mathbf{z}^{(k)} > \boldsymbol{\alpha}_i \,\}$
$I_-^{(k)}$	mengde av indekser, $I_-^{(k)} = \{\, i : \mathbf{r}_i\, \mathbf{z}^{(k)} < \boldsymbol{\alpha}_i \,\}$
$I_0^{(k)}$	mengde av indekser, $I_-^{(k)} = \{\, i : \mathbf{r}_i\, \mathbf{z}^{(k)} = \boldsymbol{\alpha}_i \,\}$

Bibliografi

[1] W. F. Ames. *Numerical methods for partial differential equations.* New York : Academic Press, 2 edition, 1977.

[2] H. S. Carslaw and J. C. Jaeger. *Conduction of heat in solids.* Oxford : Clarendon Press, 2 edition, 1959.

[3] G. Dahl. *An introduction to convexity.* Oslo: UiO (se http://folk.uio.no/geird), 2004.

[4] G. B. Dantzig and M. N. Thapa. *Linear programming 1: Introduction.* New York : Springer, 1997.

[5] E. DiBenedetto. *Partial Differential Equations.* Boston : Birkhäuser Boston, 2 edition, 2010.

[6] C. H. Gur and J. Pan. *Handbook of thermal process modeling of steels.* Boca Raton : Taylor and Francis, 2009.

[7] D. C. Lay. *Linear algebra and its applications.* Boston : Pearson education, 3 edition, 2006.

[8] T. Lyche. *Lecture Notes for Inf-Mat 4350 (Numerical linear algebra).* Oslo : Uio, 2010.

[9] S. Mehrotra. On the implementation of a primal-dual interior point method. *SIAM Journal on Optimization*, 2:575–601, 1992.

[10] M. Pivato. *Linear Partial Differential Equations and Fourier Theory.* UK New York : Cambridge University Press, 2010.

[11] A. D. Polyanin. *Handbook of linear partial differential equations for engineers and scientists.* Boca Raton, Fla. : Chapman and Hall/CRC, 2002.

[12] E. Rothberg and B. Hendrickson. Sparse matrix ordering methods for interior point linear programming. Technical report, Linear Programming, INFORMS Journal on Computing, 1996.

[13] G. Sharma and J. Martin. Matlab: A language for parallel computing. *Int. J. Parallel Program.*, 37:3–36, February 2009.

[14] A. Tveito and R. Winther. *Introduction to Partial Differential Equations. A computational Approach.* Berlin : Springer, 2 edition, 2005.

[15] R. Vanderbei. *Linear programming: Foundations and Extensions.* Boston : Springer, 3 edition, 2008.

Figurer

Tabeller