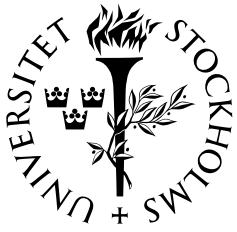


SJÄLVSTÄNDIGA ARBETEN I MATEMATIK

MATEMATISKA INSTITUTIONEN, STOCKHOLMS UNIVERSITET

Mönstersökning - en gradientrelaterad metod

av

Maya Brandi

2009 - No 5

Mönstersökning - en gradientrelaterad metod

Maya Brandi

Självständigt arbete i matematik 15 högskolepoäng, grundnivå

Handledare: Yishao Zhou

2009

Sammanfattning

Direkta sökmetoder är en grupp optimeringsmetoder som karaktäriseras av att de aldrig använder någon gradientbaserad information i sökandet efter en optimal punkt. Vi kommer att koncentrera oss på optimeringsproblemet att utan bivillkor hitta ett lokalt minimum till en funktion f i $\mathbb{R}^n$. Inom den gradientbaserade optimeringen löser man den här typen av problem genom att utifrån en punkt x_0 med gradientens hjälp hitta avtagande riktningar att successivt söka längs med. En direkt sökmetod testar istället längs (som minst) n linjärt oberoende riktningar för att se om någon av dessa reducerar värdet på målfunktionen. Vi kommer att titta på ev gren av de direkta sökmetoderna som kallas mönstersökning och som i huvudsak karaktäriserar av att sökriktningar och steglängd är konstruerade på ett sådant sätt att varje genererad punkt hamnar på en, av f oberoende lattice.

Frånvaron av gradientinformation har gjort det svårt att utveckla konvergensbevis för de direkta sökmetoderna. 1997 kom ett generellt bevis för mönstersökningsmetoderna och nyckeln till beviset är just deras egenskap att generera punkter på en av f oberoende lattice. Vi kommer att studera mönstersökningsmetodernas struktur, dels genom några exempel och dels på generaliserad form och vi kommer att gå igenom konvergensbeviset för den generaliserade mönstersökningsmetoden utvecklat 1997 av Virginia Torczon.

Jag vill tacka min handledare Yishao Zhou som genom stort engagemang, uppmuntran och support har varit ett viktigt stöd under arbetets gång. Jag vill också tacka min granskare Rikard Bøgvad som kommit med många bra anmärkningar. Tack också till Erik Svensson, Thomas Kvorning, Jens Forsgård och Måns Magnusson som på olika sätt har hjälpt mig med mitt arbete. Framför allt vill jag rikta ett stort tack till min pojkvän Oscar Karlsson som alltid har funnits där när jag har behövt hjälp eller stöd.

Innehåll

1 Introduktion **1**
- 1.1 Direkta sökmetoder. 1
- 1.2 Populära i praktiken. 2
- 1.3 Mönstersökning- en typ av direkt sökmetod. 3

2 Mönstersökning - tre exempel. **4**
- 2.1 Cykliska Koordinat-metoden 5
- 2.2 Hook och Jeeves Metod. 8
- 2.3 MDS-Metoden. 11

3 Enkelt avtagande och Tillräckligt avtagande **14**
- 3.1 Armijos regel 16

4 Global konvergens **18**

5 $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$ **19**
- 5.1 Den breda definitionen av en mönstersökningsalgoritm 19
  - 5.1.1 Hook och Jeeves och den generella modellen 23
- 5.2 Konvergensteori och beviset av $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$ . . 25

6 $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$ **29**
- 6.1 Tre nya krav på GMA 29
- 6.2 Konvergensteori och beviset av $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$ . . . 30
- 6.3 Bevis av Proposition 5.3 38

7 I praktiken **39**
- 7.1 Om f är diskontinuelig. 39
- 7.2 Om ∇f är diskontinuelig. 40
- 7.3 Brus. 41

1 Introduktion

Låt f vara en kontinuerligt differentierbar funktion av n variabler. Vår uppgift är att utan bivillkor hitta ett lokalt minimum till f.

Tidigt fick vi lära oss att ett nödvändigt villkor för ett lokalt minimum är att gradienten försvinner i denna punkt och för att lösa problemet att minimera f ligger det närmast till hands att på något sätt använda sig av detta villkor. De gradientbaserade metoderna är många och välutvecklade, men hur löser man problemet om gradienten eller approximationer av denna är otillgängliga? Den här uppsatsen kommer att behandla en del av svaret på denna fråga.

1.1 Direkta sökmetoder.

Metoder för att lösa problem av ovanstående karaktär utan att beräkna eller approximera derivatan, kallas med ett gemensamt namn för *Direkta sökmetoder*[1]. En direkt sökmetod är en algoritm som utifrån en punkt x_k testar sig fram längs n linjärt oberoende riktningar för att se om någon av riktningarna reducerar värdet på målfunktionen. På så sätt genererar den en serie punkter $\{x_k\}$ som iteration för iteration närmar sig ett minimum. Vad som karakteriserar en direkt sökmetod kan därför också sägas vara att den *endast* genom att jämföra värdet av målfunktionen i olika riktningar skapar sig en bild av funktionens utseende och utifrån denna bild lyckas identifiera en avtagande riktning.

Men även om direkta sökmetoder aldrig beräknar eller approximerar derivatan så kan man ändå prata om dem som gradientrelaterade metoder. Det beror på det faktum att en metod som lyckas identifiera en avtagande riktning indirekt också ligger inne med information om gradienten eftersom definitionen av en avtagande riktning är att dess minsta mellanliggande vinkel till gradienten är mindre än $\pi/2$.

Eftersom direkta sökmetoder helt och hållet bara bygger på beräkningar av målfunktionen f kallas de också för "zeroth-order methods" där zeroth-order refererar till derivatan. Man ska dock inte blanda ihop direkta sökmetoder med de metoder som utan att beräkna gradienten ändå använder approximationer av denna. Dessa metoder är också "zeroth-order methods" i den mening att de bara använder sig av beräkningar av målfunktionen, men de hör inte till de direkta sökmetoderna. Varför? Den huvudsakliga skillnaden grundar sig i distinktionen mellan *tillräckligt avtagande*[2] och *enkelt avtagande*[3]. I stora drag är distinktionen mellan tillräckligt och enkelt avtagande, distinktionen mellan "ett lagom stort steg" i en "tillräckligt" avtagande riktning och bara ett steg i en avtagande riktning. Att sätta kravet

[1] Min översättning. Den vedertagna engelska terminologin är *Direct search methods.*
[2] Min översättning. Den vedertagna engelska terminologin är *Sufficient decrease*
[3] Min översättning. Den vedertagna engelska terminologin är *Simple decrease.*

tillräckligt avtagande på ett steg innebär klassiskt sett, som vi ska se, att man använder sig av gradienten eller approximationer av denna. Direkta sökmetoder till skillnad från gradientbaserade optimeringsmetoder ställer aldrig något krav på tillräckligt avtagande steg. Men vi ska se att de på ett lite annorlunda sätt ändå uppfyller idén om tillräckligt avtagande och hur detta används i beviset av deras konvergens.

Ett annat krav som ställs på en metod för att den ska få gå under namnet direkt sökmetod är att *antalet* möjliga sökriktningar i varje iteration är begränsat och bestämt på förhand. Eftersom det är bestämt på förhand är det också oberoende av målfunktionen.

Direkta sökmetoder har funnits sedan 50-talet [2] men begreppet myntades först 1961 av Robert Hook och T.A. Jeeves [3]. Metoderna har sedan dess använts mycket i praktiken, men under tidigt 70-tal började de få dåligt rykte och togs i stort sett helt bort ifrån litteraturen [2]. Förutom att metoderna ibland är långsamma och osäkra ansågs deras konvergensförmåga vara heuristiskt grundad, dvs utan bevisbar grund, baserad på gissningar snarare än fakta. Trots utbredd tillämpning av metoderna har bilden av direkta sökmetoder som teoretiskt suspekta funnits kvar ända fram till mitten på 90-talet då det generella konvergensbeviset för *Mönstersökningsmetoder*[4], en gren av de direkta sökmetoderna, utvecklades av Virginia Torczon[5]. Då ökade återigen intresset för dessa metoder[3].

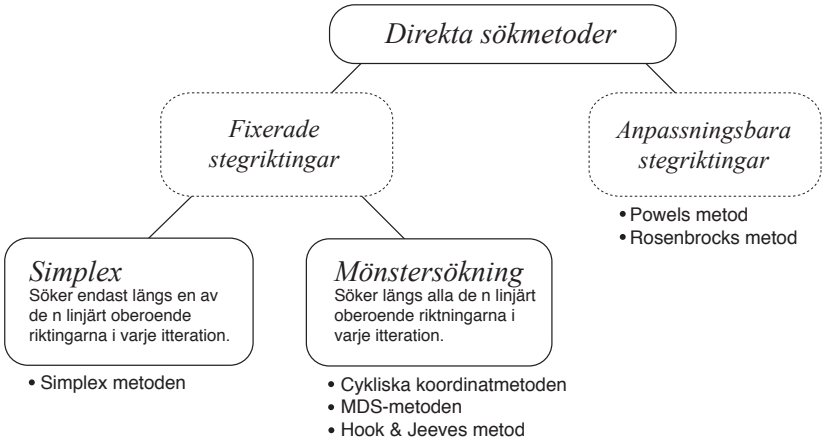
1.2 Populära i praktiken.

Direkta sökmetoder har bestått som verktyg av flera anledningar. För det första är de ibland det enda alternativet till vissa optimeringsproblem av särskilt svår karaktär, t.ex. inom simuleringsbaserad optimering och optimering av ickenumeriska funktioner [3]. För det andra är de ofta mycket användarvänliga i den meningen att det krävs ytterst lite förarbete för att köra en algoritm. Detta till skillnad mot många gradientbaserade metoder som tex kan kräva ett stort maskineri för att beräkna eller approximera gradienten [6]. Så även om algoritmen i sig tar lång tid på sig att få fram en lösning så kan det ändå i vissa fall vara en mer effektiv metod än någon gradientbaserad metod. Dessutom har direkta sökmetoder egenskapen att med rätt val av steglängdsparameter vara robusta på så sätt att de inte är lika känsliga för små variationer hos målfunktionen som de gradientbaserade metoderna är [2]. Dessa praktiskt goda egenskaper tillsammans med det faktum att deras konvergens numera i många fall är bevisad, har gjort att metoderna har återfått sin status.

[4]Min översättning. Den vedertagna engelska terminologin är *Pattern search methods.*

1.3 Mönstersökning- en typ av direkt sökmetod.

De olika direkta sökmetoderna skiljer sig åt både i valet av de n sökriktningarna och i hur de väljer att ta steg längs dessa. Metoderna har delats in i olika grenar efter hur de är strukturerade. Indelningen och namnvigningen har inte varit konsekvent litteraturen igenom, vilket har att göra med den utveckling som metoderna gick igenom under 70, 80 och 90 talet. (Jämför tex. definitionen av mönstersökning i [8] med Torczons definition [5].) Den senare hållningen är att dela in direkta sökmetoder i de metoder som *anpassar sökriktningarna* efter den information om målfunktionen som kommer fram under sökandets gång, och de metoder som har *fixerade sökriktningar* algoritmen igenom. Under den förra gruppen ingår metoder som Powells metod [8] och Rosenbrooks metod[1]. Den senare gruppen delas i sin tur in i två grupper, Mönstersöknings metoder som i varje iteration söker längs alla n riktningar, och simplex metoder [8] som bara söker längs en av riktningarna under en iteration.



Figur 1: Ett vanligt sätt att kategorisera de olika direkta sökmetoderna.

Utvecklingen av konvergensbevis för de olika metoderna har varit olika framgångsrik. För metoderna med anpassningsbara stegriktningar finns det individuella konvergensbevis, där t.ex. beviset för Powells metod bygger på att det är en metod med konjugat-riktningar. [8]

Simplexmetoderna hör till de mest populära av direkta sökmetoder, men för deras konvergens finns ännu inte något känt bevis.

Vad mönstersökningsmetoderna gäller så har det funnits individuella konvergensbevis för några av metoderna redan på tidigt 70 tal, dvs under den tid då direkta sökmetoder var som mest förtalade. Dessa bevis fick på

grund av metodernas dåliga ryckte under denna tid aldrig någon ordentlig plats i litteraturen [3]. Det generella konvergensbeviset för hela gruppen av mönstersökningsmetoder kom 1997 [5] och en av byggstenarna i beviset är just det faktum att man söker i alla n riktningar i varje iteration. Det finns anledning att tro att orsaken till att man misslyckats bevisa simplexmetodernas konvergens är att de saknar denna egenskap [6].

Förutom att mönstersökningsmetoderna söker längs alla n fixerade riktningar i varje iteration så är det hur steglängden väljs som är nyckeln till beviset av deras konvergens. Det kanske enklaste sättet att få en insikt i deras struktur är att gå rakt på exemplen som presenteras i nästa avsnitt. Vi kan i alla fall förbereda med att rikta uppmärksamheten mot det faktum att steglängden i de kommande exemplen alltid ökar eller minskar med någon potens av en och samma rationella faktor och att detta tillsammans med de fixerade stegriktningarna innebär att varje punkt genererad av en algoritm ligger på ett symmetriskt gitter. Mer specifikt, om algoritmen har genererat en sekvens av punkter $\{x_0, x_1, ..., x_N\}$ så kommer varje steg $x_{k+1} - x_k$ att ligga på en lattice $\Phi_N T$ där $\Phi_N \in \mathbb{Q}$ och T är en heltalslattice. Denna generella egenskap hos en mönstersökningsmetod är essentiell i beviset av den globala konvergensen.

Vi kommer att börja i avsnitt 2 med att titta på tre exempel på Mönstersöknings algoritmer och på hur de kan tillämpas. Där efter pratar vi i avsnitt 3 lite mer om distinktionen mellan enkelt och tillräckligt avtagande. I avsnitt 5 tittar vi först på den breda generella strukturen hos en mönstersökningsalgoritm vart efter vi visar att en sådan algoritm garanterar att $\lim_{k\to\infty} \inf \|\nabla f(x_k)\| = 0$ om målfunktionen är kontinuerligt differentierbar I avsnitt 6 lägger vi först till de nya krav som behövs för att kunna visa det starkare resultatet $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$. Där efter visas detta resultat för den nya snävare varianten av en mönstersöknings metod. Avslutningsvis pratar vi i avsnitt 7 om de fallgropar som kan uppstå om man inte håller sig till de teoretiska ramar som sattes upp inför bevisen i avsnitt 5 och 6, och om varför man i praktiken oftast struntar i dessa ramar.

2 Mönstersökning - tre exempel.

Vi kommer här att ge en mer eller mindre geometrisk bild av tre olika mönstersökningsmetoder och titta på enkla exempel i två dimensioner. Metoderna vi kommer att titta på är *Cykliska koordinatmetoden* [1], *Hook och Jeeves metod* [1] och *MDS* [5] *- metoden* [4]. Hook och Jeeves metod och Cykliska koordinatmetoden finns i flera varianter. Vi kommer att titta på de vars struktur faller in under Torczons definition av en mönstersökningsmetod.

[5] Förkortning för *Multi Directional Search*

2.1 Cykliska Koordinat-metoden

En mönstersökningsmetod söker som sagt längs minst n linjärt oberoende riktningar i varje iteration och alltid längs samma riktningar. Vi ska se ett enkelt exempel på detta i en av de äldsta av metoderna cykliska koordinatmetoden. Den är inte särskilt effektiv men många andra metoder, (både mönstersökningsmetoder och andra direkta sökmetoder) är varianter eller förbättringar av denna och därför är den ändå intressant att titta på i stora drag. Den går tillväga såhär:

Utifrån en första baspunkt $\mathbf{x}_0$ och n linjärt oberoende koordinatriktningar $\mathbf{d}_1 ... \mathbf{d}_n$ som är bestämda på förhand, gör cykliska koordinat-metoden sökningar parallellt med koordinatriktningarna, en i taget med en steglängdsparameter Δ_k där k står för den iteration man är i. I iteration k testar algoritmen stegen $\pm\Delta_k \mathbf{d}_i$ där $i = 1...n$ och accepterar bara de steg som reducerar målfunktionsvärdet. När alla riktningar är testade så kallar man den punkt som man står i för x_{k+1}. Om ingen av teststegen varit lyckade så är $x_{k+1} = x_k$ men om någon av stegen varit lyckade så är x_{k+1} en ny punkt. På så sätt närmar sig cykliska koordinatmetoden iteration för iteration en lokal minimipunkt om en sådan existerar. Då man har kommit så pass nära ett minimum att inget steg i någon riktning reducerar värdet på f så minskar man steglängden med en på förhand bestämd rationell faktor θ för att sedan upprepa iterationen och se om man får ett bättre resultat denna gång. På samma sätt fortsätter man till dess att steglängden blivit så liten att man är nöjd. Den stationära punkten ligger inom en radie lika stor som längden av de senast accepterade teststegen. Mer detaljerat kan cykliska koordinatmetoden beskrivas såhär.

Cykliska koordinatmetoden

Välj stopplängd $\epsilon > 0$, startpunkt $\mathbf{x}_0$, en första steglängsdparameter $\Delta_0 > \epsilon$ och en faktor för att minska steglängden $\theta \in \mathbb{Q}$. Låt $\mathbf{y}_1 = \mathbf{x}_0$, låt $j = 1$ och låt $k = 0$. Gå till (i).

(i) Utifrån punkten $\mathbf{y}_j$ söks ett mindre värde på f längs riktning $\mathbf{d}_j$ med steglängd Δ_k eller $-\Delta_k$.
Låt

- $\mathbf{y}_{j+1} = \mathbf{y}_j + \Delta_k \mathbf{d}_j$ om $f(\mathbf{y}_j + \Delta_k \mathbf{d}_j) < f(\mathbf{y}_j)$
- $\mathbf{y}_{j+1} = \mathbf{y}_j - \Delta_k \mathbf{d}_j$ om $f(\mathbf{y}_j - \Delta_k \mathbf{d}_j) < f(\mathbf{y}_j) < f(\mathbf{y}_j + \Delta_k \mathbf{d}_j)$
- $\mathbf{y}_{j+1} = \mathbf{y}_j$ annars

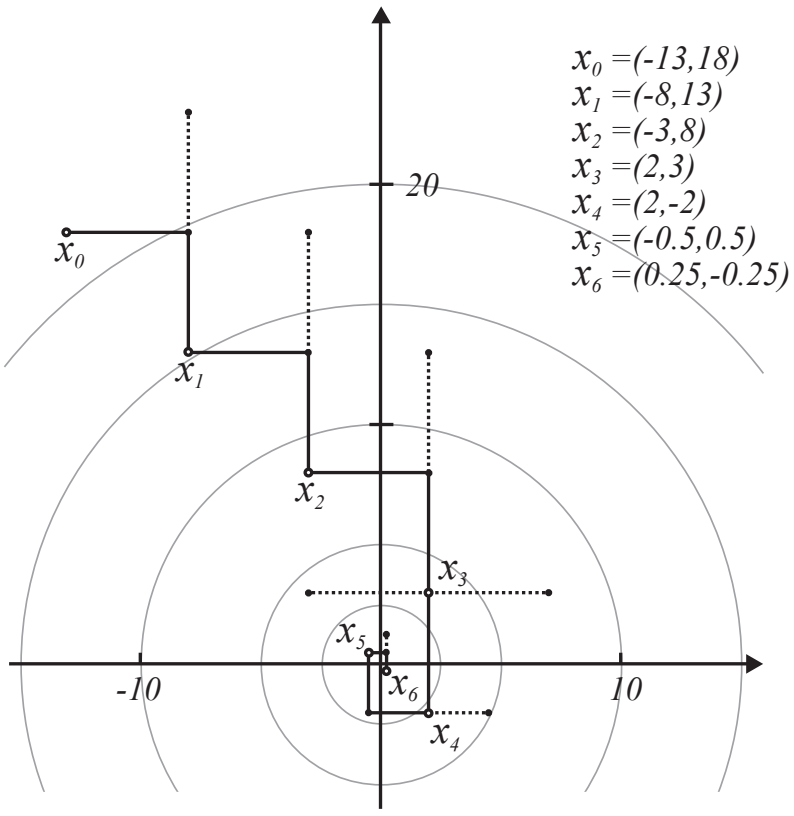
Om alla riktningar ännu inte är undersökta, dvs om $j < n$ låt $j = j+1$ och upprepa steg (i).
Om alla riktningar är undersökta, gå till steg (ii).

(ii) Om $f(\mathbf{y}_{n+1}) < f(\mathbf{x})$ så är iterationen lyckad. Låt $\mathbf{x}_{k+1} = \mathbf{y}_{n+1}$ och gå till steg (iii). Om $f(\mathbf{y}_{n+1}) = f(\mathbf{x}_k)$ så är iterationen misslyckad. Gå till steg (iv).

(iii) Låt $\mathbf{y}_1 = \mathbf{x}_{k+1}$, $\Delta_{k+1} = \Delta_k$, byt ut k mot $k+1$, låt $j = 1$ och gå till steg (i).

(iv) Om $\Delta_k \leq \epsilon$ avsluta algoritmen. $\mathbf{x}_k$ är vår optimala punkt. Om $\Delta_k > \epsilon$ så minskar vi steglängden. Låt $\Delta_{k+1} = \theta\Delta_k$ och låt $\mathbf{y}_1 = \mathbf{x}_k$, $\mathbf{x}_{k+1} = \mathbf{x}_k$, byt ut k mot $k+1$, låt $j = 1$ och upprepa steg (i) med den nya steglängden.

Exempel 2.1 *Bilden illustrerar cykliska koordinatmetoden i två dimmentioner med* $\boldsymbol{d}_1$ *och* $\boldsymbol{d}_2$ *som standardbasvektorerna* $(1,0),(0,1)$, $f = x_1^2 + x_2^2$, $\boldsymbol{x}_0 = (-13, 18)$, $\theta = 1/2$ *och* $\Delta_0 = 5$.



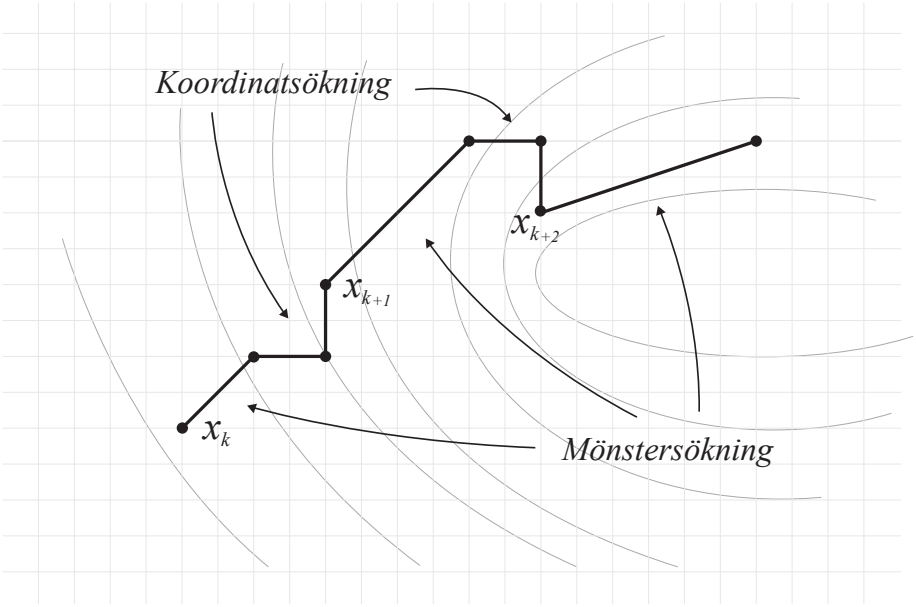
Figur 2: Cykliska koordinatmetoden i $\mathbb{R}^2$.

Man inser att om man börjar med att söka med relativt stora steg så kommer man nära ett minimum ganska snabbt. Problemet är förstås att det är svårt

att veta hur stora steg man ska börja med. Ett annat problem som gör att cykliska koordinatmetoden och de andra mönstersökningsmetoderna kan vara långsamma är att de snabbt kan konvergera mot ett minimum utan att vara medvetna om det eftersom det enda sättet för dem att veta om de är i en stationär punkt är att de har reducerat steglängden tillräckligt många gånger. Dvs om man börjar sökningen väldigt nära en stationär punkt, men söker med en väldigt stor steglängd, så tar det ganska många iterationer innan man kan konstatera att man är i ett minimum.

Beviset för mönstersökningsmetodernas globala konvergens rör bara kontinuerligt differentierbara funktioner. Detta beror på att om f inte är differentierbar finns det en risk att algoritmen konvergerar för tidigt i någon icke stationär punkt. Anledningen till detta är att algoritmen kan fastna i en "vass ränna" där om man har otur, ingen av de n sökriktningarna är avtagande.

Detta problem kan i viss mån undvikas genom något som viss i litteratur (och förvirrade nog), kallas mönstersökning[6]. Mönstersökning innebär att man utifrån de senast bildade baspunkterna $\mathbf{x}_k$ och $\mathbf{x}_{k-1}$ skapar ett nytt steg av längd $\|\mathbf{x}_k - \mathbf{x}_{k-1}\|$ och riktning riktning $(\mathbf{x}_k - \mathbf{x}_{k-1})$.



Figur 3: Mönstersökning.

Nästa metod börjar varje iteration med att göra en sådan mönstersökning.

[6]Svenska för Patternsearch

2.2 Hook och Jeeves Metod.

Hook och Jeeves metod tar, precis som cykliska koordinatmetoden teststeg med förutbestämd längd längs riktningarna $\mathbf{d}_1, ..., \mathbf{d}_n$ en i taget, vartefter den värderar punkten för att sedan förkasta den eller behålla den. Ett teststeg accepteras om det ger ett mindre värde på f och vi kallar det då för ett *lyckat* teststeg. Om teststeget inte ger ett mindre värde på f så förkastas det. Teststeget är då *misslyckat*. När ingen riktning resulterar i ett bättre värde av f minskas steglängden och sökningen upprepas.

Hook och Jeeves Metod
Innan sökningen börjar väljs en stopplängd $\epsilon > 0$ som säger hur små stegen ska vara för att sökningen ska avslutas, en första steglängdsparameter $\Delta_0 \geq \epsilon$, en startpunkt $\mathbf{x}_0$ och en minskande faktor $0 < \theta < 1$ med $\theta \in \mathbb{Q}$. Punkten $\mathbf{y}_1$ och indexen j och k introduceras där $\mathbf{y}_1 = \mathbf{x}_0$, $j = 1$ och $k = 0$. Förloppet kan nu beskrivas steg för steg.

(i) Utifrån punkten $\mathbf{y}_j$ söks ett mindre värde på f längs riktning $\mathbf{d}_j$ med steglängdsparametern Δ_k eller $-\Delta_k$.
Låt

- $\mathbf{y}_{j+1} = \mathbf{y}_j + \Delta_k \mathbf{d}_j$ om $f(\mathbf{y}_j + \Delta_k \mathbf{d}_j) < f(\mathbf{y}_j)$
- $\mathbf{y}_{j+1} = \mathbf{y}_j - \Delta_k \mathbf{d}_j$ om $f(\mathbf{y}_j - \Delta_k \mathbf{d}_j) < f(\mathbf{y}_j) < f(\mathbf{y}_j + \Delta_k \mathbf{d}_j)$
- $\mathbf{y}_{j+1} = \mathbf{y}_j$ annars

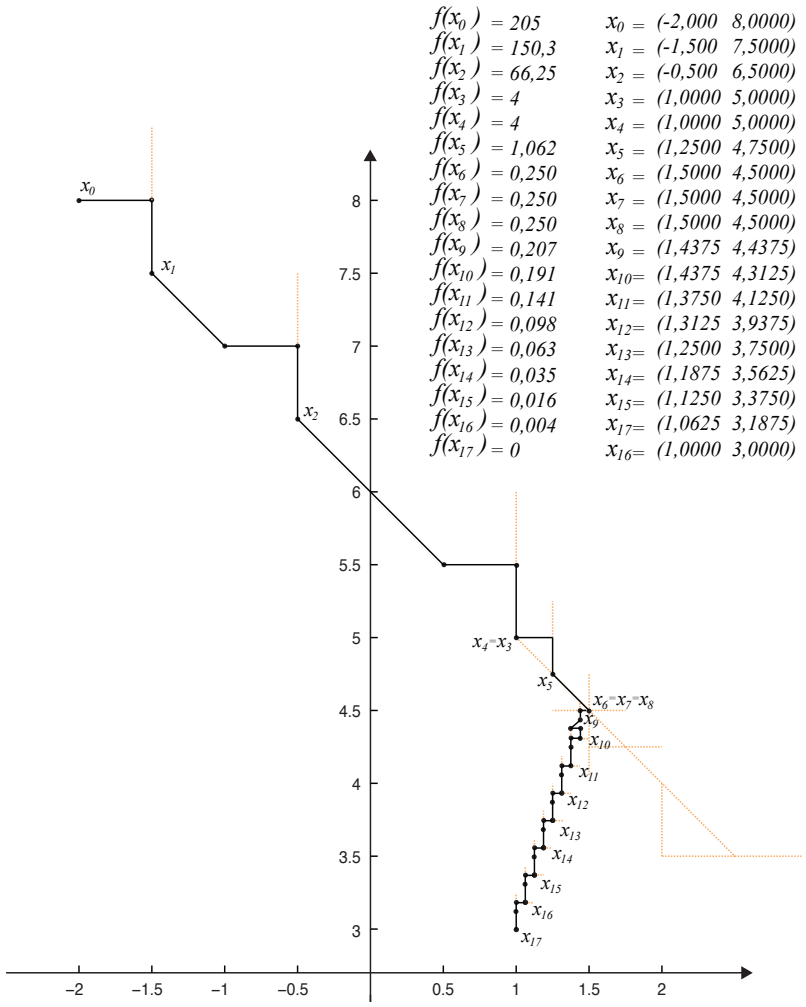
Om alla riktningar ännu inte är undersökta, dvs om $j < n$ låt $j = j+1$ och upprepa steg (i).
Om alla riktningar är undersökta, gå till steg (ii).

(ii) Om $f(\mathbf{y}_{n+1}) < f(\mathbf{x}_k)$ så är $\mathbf{y}_{n+1}$ den nya baspunkten. Låt $\mathbf{x}_{k+1} = \mathbf{y}_{n+1}$. Gå till steg (iii).
Annars, om $f(\mathbf{y}_{n+1}) \geq f(\mathbf{x}_k)$, gå till steg (iv)

(iii) Det är dags för mönstersökning. Bilda den nya riktningen $\mathbf{x}_{k+1} - \mathbf{x}_k$. Låt $\mathbf{y}_1 = \mathbf{x}_{k+1} + (\mathbf{x}_{k+1} - \mathbf{x}_k)$, låt $\Delta_{k+1} = \Delta_k$, ersätt k med $k + 1$, låt $j = 1$ och gå till steg (i)

(iv) om $\Delta_k \leq \epsilon$ avsluta algoritmen. Annars låt $\Delta_{k+1} = \theta \Delta_k$. Låt $\mathbf{y}_1 = \mathbf{x}_k$, $\mathbf{x}_{k+1} = \mathbf{x}_k$ byt k mot $k + 1$, låt $j = 1$, och upprepa (i).

Exempel 2.2 *Vi låter* $f = (x_1 - 1)^2 + (3x_1 - x_2)^2$ *vara vår målfunktion,* $\boldsymbol{x}_0 = (-2, 8)$, $\Delta_0 = 0,5$ *och* $\epsilon = 0,1$. *Vi kommer att nå minimipunkten* $(1, 3)$ *efter 17 iterationer. Bilden visar hur vi närmar oss minimum iteration för iteration.*



Figur 4: Hook och Jeeves metod i $\mathbb{R}^2$.

Tabellen nedan visar de beräkningar som krävs för att genomföra de tio första iterationerna. I tabellen står m för misslyckat teststeg.

$x_0 = (-2, 8)$	$f(-2, 8) = 205$		
$y_1 = (-2, 8)$	$f(-1.5, 8) = 162.5$		
	$f(-1.5, 8.5) = 175.25$	m	
	$f(-1.5, 7.5) = 150.25$		$150.25 < 205 \Rightarrow$
$x_1 = (-1.5, 7.5)$	$f(-1, 7) = 104$		
$y_1 = (-1, 7)$	$f(-0.5, 7) = 74.5$		
	$f(-0.5, 7.5) = 83.25$	m	
	$f(-0.5, 6.5) = 66.25$		$66.25 < 150.25 \Rightarrow$

$x_2 = (-0.5, 6.5)$ $y_1 = (0.5, 5.5)$	$f(0.5, 5.5) = 16.25$ $f(1, 5.5) = 6.25$ $f(1, 6) = 9$ $f(1, 5) = 4$	 m 	 $4 < 66.25 \Rightarrow$
$x_3 = (1, 5)$ $y_1 = (2.5, 3.5)$	$f(2.5, 3.5) = 18.25$ $f(3, 3.5) = 34.25$ $f(2, 3.5) = 7.25$ $f(2, 4) = 5$	 m 	 $5 > 4 \Rightarrow$
$x_4 = x_3$ $\Delta_4 = 0.25$ $y_1 = (1, 5)$	$f(1, 5) = 4$ $f(1.25, 5) = 1.625$ $f(1.25, 5.25) = 2.3125$ $f(1.25, 4.75) = 1.0625$	 m 	 $1,0625 < 4 \Rightarrow$
$x_5 = (1.25, 4.75)$ $y_1 = (1.5, 4.5)$	$f(1.5, 4.5) = 0.25$ $f(1.7, 4.5) = 1.250$ $f(1.25, 4.5) = 0.625$ $f(1.5, 4.75) = 1.0625$ $f(1.5, 4.25) = 0.3125$	 m m m m	 $0,25 < 1.0625 \Rightarrow$
$x_6 = (1.5, 4.5)$ $y_1 = (1.75, 4.25)$	$f(1.75, 4.25) = 1.5625$ $f(2, 4.25) = 4.0625$ $f(1.5, 4.25) = 0.3125$ $f(1.5, 4.5) = 0.25$	 m 	 $0.25 \geq 0.25 \Rightarrow$
$x_7 = x_6$ $\Delta_7 = 0.125$ $y_1 = (1.5, 4.5)$	$f(1.5, 4.5) = 0.25$ $f(1.625, 4.5) = 0.5313$ $f(1.375, 4.5) = 0.2813$ $f(1.5, 4.625) = 0.2656$ $f(1.5, 4.375) = 0.2656$	 m m m m	 $0.25 \geq 0.25 \Rightarrow$
$x_8 = x_7$ $\Delta_8 = 0.0625$ $y_1 = (1.5, 4.5)$	$f(1.5, 4.5) = 0.25$ $f(1.5625, 4.5) = 0.2516$ $f(1.4375, 4.5) = 0.2266$ $f(1.4375, 4.5625) = 0.2539$ $f(1.4375, 4.4375) = 0.2070$	 m m 	 $0.2070 < 0.25 \Rightarrow$
$x_9 = (1.4375, 4.4375)$ $y_1 = (1.375, 4.375)$	$f(1.375, 4.375) = 0.2031$ $f(1.4375, 4.375) = 0.1953$ $f(1.4375, 4.4375) = 0.2070$ $f(1.4375, 4.5625) = 0.2539$ $f(1.4375, 4.3125) = 0.2359$	 m m m	 $0.1953 < 0.207 \Rightarrow$
$x_{10} = (1.4375, 4.375)$			

2.3 MDS-Metoden.

MDS-metoden skiljer sig strukturellt från de metoder vi sett hittills. Den är relativt ny, framtagen på 90-talet av Virginia Torczon och är en utveckling av simplexmetoderna även om den har egenskaper som gör att den faller in under grenen mönstersökning. MDS-metoden söker minimum genom att reflektera, kontrahera och expandera ett simplex om $n+1$ hörn.

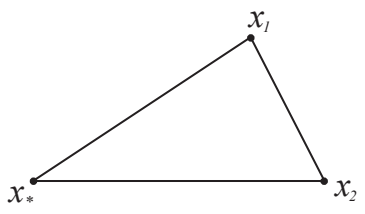
Om vi har k punkter $\mathbf{x}_1, \mathbf{x}_2, ..., \mathbf{x}_k$ i $\mathbb{R}^n$ som är sådana att $\mathbf{x}_2 - \mathbf{x}_1, \mathbf{x}_3 - \mathbf{x}_1, ..., \mathbf{x}_k - \mathbf{x}_1$ alla är linjärt oberoende, så kallas det konvexa höljet av $\mathbf{x}_1, \mathbf{x}_2 ... \mathbf{x}_k$ för ett *simplex*. Ett simplex i $\mathbb{R}^n$ kan förstås aldrig bestå av fler än $n+1$ hörn eftersom det som mest finns n linjärt oberoende vektorer i $\mathbb{R}^n$. Ett simplex i $\mathbb{R}^2$ kan således som mest vara en triangel, i $\mathbb{R}^3$ som mest en tetraeder etc.

I MDS-metoden börjar varje iteration med att beräkna funktionsvärdena i vart och ett av hörnen i ett simplex med $n+1$ hörn för att välja den punkt $\mathbf{x}_*$ som ger det minsta funktionsvärdet. Det första man måste göra innan algoritmen startar är således att välja $n+1$ startpunkter som tillsammans bildar ett simplex i $\mathbb{R}^n$. I och med att detta är gjort har man även valt sina sökriktningar och deras respektive steglängder, som definieras av simplexets kanter.

Algoritmen består sedan av tre sorters steg. *Reflektionssteget*, *expansionssteget* och *kontraktionssteget*. En iteration kan bestå av endast en reflektion, av en reflektion och en expansion, eller av en kontraktion, alla utgående från $\mathbf{x}_*$. Vi ska titta på hur stegen ser ut och hur algoritmen väljer sina steg.

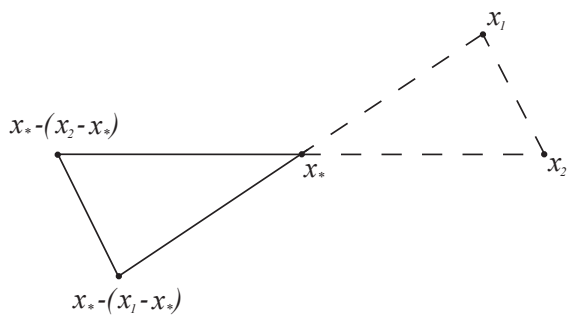
MDS-Metoden.
Välj ett första simplex S_0, en expansionsfaktor $\mu \in (1, +\infty)$ och en kontraktionsfaktor $\theta \in (0, 1)$. Ofta låter man $\mu = 2$ och $\theta = \mu^{-1} = 1/2$. För att MDS-metoden ska passa in i det kommande konvergensbeviset krävs att $\mu, \theta \in \mathbb{Q}$.

(i) Beräkna f i varje hörn av simplexet S_k, låt $\mathbf{x}_*$ vara den punkt som ger det minsta funktionsvärdet, numrera de övriga punkterna från 1 till n och gå till steg (ii).



Figur 5: Ett första simplex i $\mathbb{R}^2$.

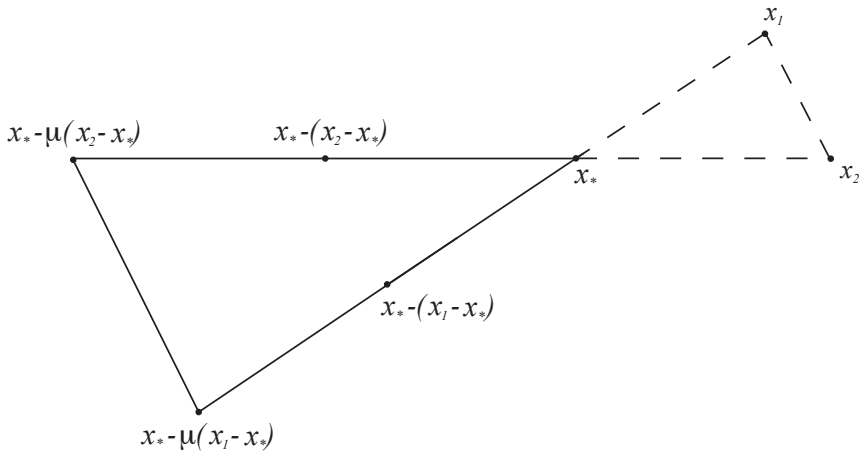
(ii) *Reflektion.* Undersök om reflektion av simplexet genom $\mathbf{x}_*$ kan leda till ett bättre resultat. Dvs beräkna f i reflektionspunkterna $\mathbf{x}_* - (\mathbf{x}_1 - \mathbf{x}_*), \mathbf{x}_* - (\mathbf{x}_2 - \mathbf{x}_*), ..., \mathbf{x}_* - (\mathbf{x}_n - \mathbf{x}_*)$ för att se om någon eller några av dessa ger ett mindre värde än $f(\mathbf{x}_*)$. Om detta är fallet anses reflektionen vara lyckad. Välj den av reflektionspunkterna $\mathbf{x}_{*_r}$ som ger det minsta värdet på f och gå till steg (iii). Om ingen av de



Figur 6: Reflektion.

nya punkterna resulterar i ett bättre värde är reflektionen misslyckad. Gå till steg (iv).

(iii) *Expansion.* Eftersom något av stegen $\mathbf{x}_* - (\mathbf{x}_1 - \mathbf{x}_*), \mathbf{x}_* - (\mathbf{x}_2 - \mathbf{x}_*)$, $..., \mathbf{x}_* - (\mathbf{x}_n - \mathbf{x}_*)$ ledde till ett bättre resultat är det rimligt att undersöka om ett ännu bättre resultat kan nås genom att fortsätta söka i samma riktning.

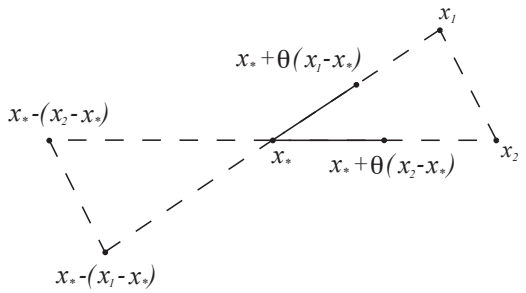


Figur 7: Expansion.

Detta gör du genom att förlänga kanterna på det reflekterade simplexet med $\mathbf{x}_*$ som utgångspunkt för att se om ett ännu bättre resultat är att finna. Med andra ord, beräkna f i expansionspunkterna $\mathbf{x}_* - \mu(\mathbf{x}_1 - \mathbf{x}_*), \mathbf{x}_* - \mu(\mathbf{x}_2 - \mathbf{x}_*), ..., \mathbf{x}_* - \mu(\mathbf{x}_n - \mathbf{x}_*)$ och se om någon av dessa ger ett mindre värde än $f(\mathbf{x}_{*_r})$. Om detta är fallet är expansionen lyckad. Låt $S_{k+1} = conv(\mathbf{x}_*, \mathbf{x}_* - \mu(\mathbf{x}_1 - \mathbf{x}_*), \mathbf{x}_* - \mu(\mathbf{x}_2 - \mathbf{x}_*), ..., \mathbf{x}_* - \mu(\mathbf{x}_n - \mathbf{x}_*))$ och gå till steg (i).

Om ingen av expansionspunkterna gav ett mindre värde än $f(\mathbf{x}_{*_r})$ anses expansionen misslyckad. Låt $S_{k+1} = conv(\mathbf{x}_*, \mathbf{x}_* - (\mathbf{x}_1 - \mathbf{x}_*), \mathbf{x}_* - (\mathbf{x}_2 - \mathbf{x}_*), ..., \mathbf{x}_* - (\mathbf{x}_n - \mathbf{x}_*))$ och gå till (i).

(iv) *Kontraktion.* Av alla punkter undersökta runt omkring $\mathbf{x}_*$ (punkterna i simplexet S_k och punkterna i reflektionssimplexet), ger $\mathbf{x}_*$ det minsta värdet på f. Det finns alltså anledning att tro att stegen i varje riktning tagna utifrån x_* har varit för stora. Kontraktionssteget minskar därför kanterna på S_k utifrån $\mathbf{x}_*$ med parametern θ. Dvs Låt $S_{k+1} = conv(\mathbf{x}_*, \mathbf{x}_* - \theta(\mathbf{x}_1 + \mathbf{x}_*), \mathbf{x}_* + \theta(\mathbf{x}_2 - \mathbf{x}_*), ..., \mathbf{x}_* + \theta(\mathbf{x}_n - \mathbf{x}_*))$ och
gå till Steg (i).



Figur 8: kontraktion

Vi har nu sett tre exempel på mönstersökning och kan konstatera att de uppfyller följande ungefärliga definition av en mönstersökningsmetod.

En mönstersökningsmetod är

1. en direkt sökmetod som bara tittar på målfunktionsvärden i ett geometriskt mönster av punkter eller för att vara mer precis bara i punkter liggande på en av f oberoende lattice (se sats 5.1 för en mer precis definition av detta) och

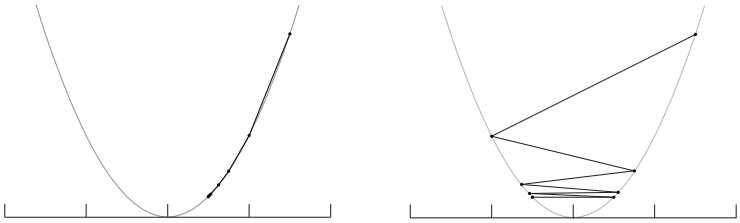
2. en direkt sökmetod som *bara* minskar steglängdsparametern när det är nödvändigt för att reducera målfunktionsvärdet.

För att förstå varför villkoren på en mönstersöknings metod är så viktiga i beviset av deras konvergensförmåga ska vi titta närmare på distinktionen mellan enkelt och tillräckligt avtagande steg och hur denna används av de derivatorbaserade sök metoderna.

3 Enkelt avtagande och Tillräckligt avtagande

De partiella derivatorna av en funktion f med avseende på var och en av de n variablerna kallas tillsammans för gradienten av f. Gradienten ∇f är en kolumnvektor med n komponenter och dess geometriska tolkning är att den utifrån en viss punkt x_k pekar i den riktning dit f växer som snabbast. Detta är ekvivalent med att $-\nabla f$ pekar i den riktning dit f är som mest avtagande. De gradientbaserade minimeringsmetoderna använder sig just av denna trevliga egenskap. Utifrån en startpunkt x_0 vet de ungefär i vilken riktning de ska röra sig för att komma närmare ett minimum. Problemet är bara att den mest avtagande riktningen är en lokal egenskap hos en funktion. Detta innebär att om man är i en viss punkt x_k och tar ett steg längs $-\nabla f(x_k)$ så är det inte alls säkert att man i nästa steg ska fortsätta i samma riktning. Man kan tycka att så länge som man bara accepterar steg som ger ett minskat värde på målfunktionen och så länge man beräknar och använder gradientriktningen i varje iteration så borde man ändå förr eller senare komma till ett minimum om ett sådant existerar. Detta är dock inte fallet. Formar man en algoritm på fel sätt så kan man ta steg i avtagande riktning som minskar värdet på f utan att någonsin konvergera mot en stationär punkt.

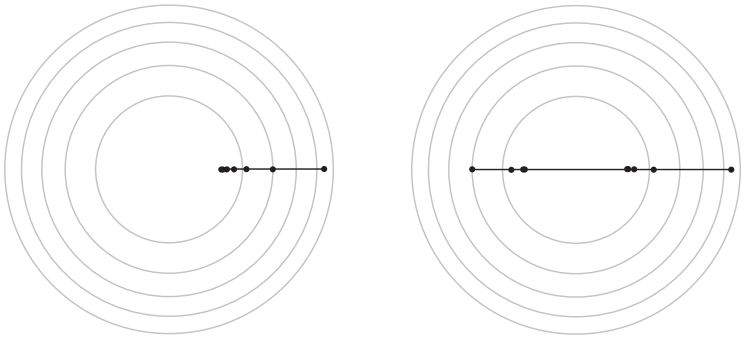
Ett enkelt exempel på detta är sekvensen genererad av $x_k = (-1)^k(0.5 + 2^{-k})$ för att lösa: min $f(x) = x^2$. Sekvensen tar hela tiden steg i avtagande riktning och för varje k gäller att $f(x_{k+1}) < f(x_k)$, men ändå så konvergerar serien mot punkterna ± 0.5.



Figur 9: För korta respektive för långa steg. Stegen är inte tillräckligt avtagande.

Ett annat enkelt exempel är sekvensen genererad av $x_k = 0.5 + 2^{-k}$ som

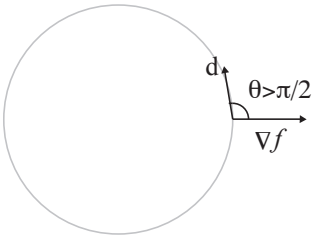
också rör sig i en avtagande riktning med $f(x_{k+1}) < f(x_k)$ för alla k, men konvergerar mot 0.5. Man kan säga att problemet med de två sekvenserna är att de hela tiden tar "för stora", respektive "för små" steg. De två problemen illustreras i bilderna ovan och två dimensioner kan dessa problem översättas till situationerna illustrerade i nästa bild.



Figur 10: Samma steg som i figur 9 fast sedda ovanifrån.

Eftersom egenskapen av mest avtagande riktning är lokal så är det inte alls säkert att den snabbaste vägen till ett minimum är att följa $-\nabla f$ i varje iteration. Men om man väljer att söka längs någon annan avtagande riktning så kan ett tredje problem uppstå. För även om algoritmen tar lagom långa steg i en avtagande riktning så blir det problem om riktningen inte är "tillräckligt avtagande".

Om målfunktionen är differentierbar sägs ju en riktning d vara avtagande i en punkt x, om den uppfyller $\nabla f(x)^t d < 0$ vilket innebär att den minsta vinkeln θ mellan gradienten i punkten x och riktningen d är större än $\pi/2$.



Figur 11: Sökriktningen d bildar nädtan rät vinkel med ∇f.

Att en riktning inte är "tillräckligt avtagande" innebär att θ fortfarande uppfyller $\pi/2 < \theta$, men att den ligger mycket nära gränsen. Konsekvensen

av detta är att algoritmen måste ta väldigt små steg för att kunna reducera målfunktionsvärdet och det i sin tur kan leda till att algoritmen konvergerar för tidigt i någon ickestationär punkt.

Det är utifrån dessa tre problem som man har infört distinktion mellan tillräckligt avtagande och enkelt avtagande eller ett "lagom långt steg" i en "bra avtagande riktning" och bara ett steg i en avtagande riktning.

För att undvika att en derivatorbaserad metod stannar i en helt felaktig lösning, finns vissa regler man kan hålla sig till. En metod som håller sig till en sådan regel garanterar att varje steg är tillräckligt avtagande. Ett enkelt avtagande steg är helt enkelt bara ett avtagande steg. Det finns olika sätt att testa tillräckligt avtagande.

För att undvika problemet med avtagande sökriktningar som nästan är vinkelräta mot gradienten brukar man använda "vinkelvillkoret"

$$\frac{-\nabla f(x_k)^t d_k}{\|\nabla f(x_k)\| \, \|d_k\|} \geq c > 0 \tag{1}$$

för något c oberoende av k. Villkoret är ekvivalent med att $\cos\theta \geq c > 0$ där θ är den minsta mellanliggande vinkeln mellan den avtagande riktningen d_k och $-\nabla f(x_k)$ och det sätter alltså en övre gräns för hur stor denna vinkel (som ju alltid är mindre än $\pi/2$) får vara.

Några klassiska exempel på verktyg som testar om steglängden är lagom lång är Armijos Regel [7], Goldsteins test [7] och Wolfes test [7]. Vi ska titta närmare på en av dessa.

3.1 Armijos regel

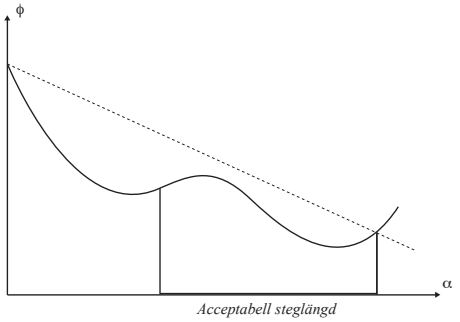
Armijos regel förhindrar en algoritm att ta för stora och för små steg förutsatt att stegen tas i en avtagande riktning d_k. Antag att vi är i punkten x_k och att vi ska ta ett steg i riktningen d_k med en steglängdsparameter α så att längden av steget är $\alpha \|d_k\|$. Vi definierar funktionen $\phi(\alpha) = f(x_k + \alpha d_k)$ och vi ska välja α lagom stort. För att göra detta införs först funktionen $\psi(\alpha) = f(x_k) + \epsilon\alpha d_k^t \nabla f(x_k) = f(x_k) + \epsilon\alpha \|d_k\| \frac{d_k^t}{\|d_k\|} \nabla f(x_k)$ med något fixt $0 < \epsilon < 1$. Vi ser att $\frac{d_k^t}{\|d_k\|} \nabla f(x_k)$ är riktningsderivatan av f i riktningen d_k och vi vet att denna är mindre än noll eftersom d_k är en avtagande riktning. Geometriskt är grafen till funktionen $\psi(\alpha)$ alltså en linje som skär målfunktionen i punkten x_k och som har en lutning som är mindre än riktningsderivatan i punkten x_k och riktningen d_k. Ett α som inte är för stort, är nu vilket tal som helst som ger ett funktionsvärde som ligger under linjen. Dvs som uppfyller

$$\phi(\alpha) \leq \psi(\alpha) \tag{2}$$

För att undvika allt för små steg införs en ny fix faktor $\eta > 1$, och α sägs vara tillräckligt stort om det uppfyller

$$\phi(\eta\alpha) > \psi(\eta\alpha)$$

Detta kan tolkas som att om man har valt en steglängd α och sedan ökar den med faktorn η, så ska detta leda till att (2) inte längre är uppfyllt.



Figur 12: Armijos regel.

Oftast används inte det senare villkoret i parktiken. För att undvika för små steg brukar man istället använda en teknik som kallas *bactracking* på engelska. Denna teknik innebär att man börjar algoritmen med relativt stora steg, för att sedan minska steglängden endast då det är nödvändigt för att reducera värdet på f eller endast då det är nödvändigt för att uppfylla (2) tex.

I de derivatorbaserade sökmetoderna är test så som dessa inbyggda på ett eller annat sätt och därigenom kan de garantera konvergens mot ett minimum. Men som ni märker är testen ofta baserade på att man kan beräkna derivatan av målfunktionen, och det var ju just detta som en derivatafri sökmetod inte skulle göra. Därför har man trott att uttrycket tillräckligt avtagande bara är applicerbart på derivatorbaserade sökmetoder. Det var denna felaktiga idé som motbevisades av Torczon 1997. Vi ska se hur Torczon visar att mönstersökningsmetodernas egenskaper garanterar att tillräckligt avtagande steg tas i varje iteration utan att beräkna eller approximera derivatan och hur detta blir nyckeln till beviset av mönstersökningsmetodernas globala konvergens.

4 Global konvergens

Vi har fått se några exempel och förhoppningsvis fått en bild av hur mönstersökningsmetoderna fungerar och vi har satt oss in i distinktionen mellan enkelt och tillräckligt avtagande. Vi ska strax gå in på teorin som garanterar mönstersökningsmetodernas globala konvergens. Men först något om själva begreppet. För inom *nonlinear programming* innebär inte global konvergens vad man kanske skulle förvänta sig, konvergens mot ett globalt minimum. Efter att ha läst föregående avsnitt är det förhoppningsvis också tydligt att metoderna inte garanterar att lösningen är global i den meningen om det inte är så att det endast finns en optimal lösning. Istället innebär global konvergens i detta sammanhang *första ordningens konvergens från en godtyckligt vald startpunkt*. Första ordningen refererar till att förstaderivatan går mot noll, så med global konvergens menas här, konvergens mot någon stationär punkt oavsett startpunkt. Detta till skillnad från lokal konvergens som i sammanhanget innebär första ordningens konvergens förutsatt att startpunkten är "tillräckligt nära" den aktuella minimipunkten. Även om mönstersökningsmetoderna är derivatorfria i den meningen att de inte använder sig av derivatan för att hitta en minpunkt så kan man förstås ändå prata om att derivatan i denna punkt är lika med noll om funktionen är kontinuerligt differentierbar.

Beviset som vi ska titta på visar att om målfunktionen är kontinuerligt differentierbar så gäller för en mönstersökningsmetod med $k = 0, 1, 2...$ att

$$\lim_{k \to +\infty} \inf \| \nabla f(x_k) \| = 0.$$

Dvs att åtminstone en delsekvens av punkterna genererade av en mönstersökningsalgoritm konvergerar mot en stationär punkt. Med bara några restriktioner kommer därefter beviset av det starkare resultatet

$$\lim_{k \to +\infty} \| \nabla f(x_k) \| = 0$$

som säger att varje gränspunkt av $\{x_k\}$ är en stationär punkt.

Att en metod är gradientbaserad innebär i regel att den garanterar tillräckligt avtagande steg i varje iteration och det är standard för konvergensbevisen av dessa metoder att utnyttja just detta. Eftersom direkta sökmetoder per definition aldrig beräknar eller approximerar derivatan så kan iden om tillräckligt avtagande steg aldrig användas på samma sätt som i de gradientbaserade metoderna, utan att samtidigt ändra på definitionen av vad en direkt sökmetod är. Sådana ansatser har gjorts, där man har formulerat om en mönstersökningsmetod så att ett krav på tillräckligt avtagande steg helt enkelt ingår i algoritmen [5]. Vad Torczon gör är att visa att mönstersökningsmetoderna faktiskt alltid tar tillräckligt avtagande steg,

men på ett helt annat sätt än via den klassiska vägen. Beviset som ska presenteras här är det första som utan att ändra på definitionen fångar in alla mönstersökningsmetoder på en gång.

Beviset bygger på tre viktiga egenskaper hos mönstersökningsmetoderna.

1. Varje iteration garanterar åtminstone en tillräckligt avtagande riktning så länge algoritmen inte har nått ett minimum. Detta krav uppfylls utan hänsyn till derivatan, genom att man i varje iteration söker parallellt med *tillräckligt många* linjärt oberoende riktningar.

2. Varje punkt genererad av algoritmen ligger på en heltalslattice multiplicerat med något rationellt tal. Detta uppfylls genom att alltid ha fixerade stegriktningar och genom att steglängdsparametern alltid är någon potens av ett enda rationellt tal. Att varje genererad punkt ligger på en heltalslattice innebär att det finns ändligt många möjliga punkter inom ett kompakt område som man i en iteration kan hamna på. Detta tillsammans med villkoret att ett teststeg bara accepteras om det är avtagande spelar en essentiell roll i konvergensbeviset.

3. Steglängdsparametern reduceras bara då varje riktning genererar ett misslyckat teststeg och man därmed har tillräckligt mycket information om kurvans utseende för att vara säker på att man är relativt nära ett minimum. Detta krav innebär att en mönstersökningsalgoritm inte minskar steglängden för tidigt och förebygger därmed konvergens mot någon ickestationär punkt.

För att kunna genomföra bevisen krävs en strikt ram av egenskaper som vi kan använda för att definiera en generell mönstersöknings metod. Denna ram ska vara så bred som möjligt med så få restriktioner som möjligt för att fånga in alla mönstersöknings metoder, samtidigt som den fortfarande ska fylla sitt syfte att garantera de egenskaper som grävs för att genomföra beviset. Vi kommer först att titta på definitionen för den bredare ramen som är tillräcklig för att visa $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$. Efter att ha genomfört beviset lägger vi till de restriktioner som krävs för att visa
$\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$.

5 $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$

5.1 Den breda definitionen av en mönstersökningsalgoritm

Som vi har sett i exemplen består mönstersöknings-algoritmerna av flera komponenter och man kan faktiskt generellt se att en mönstersökningsalgoritm består av flera delalgoritmer. För att ge en idé av den generella formen och underlätta läsningen ges en lite förenklad bild av hur den generaliserade mönstersöknings-algoritmen ser ut redan nu.

GMA - Generaliserad Mönstersöknings-Algoritm.
Bestäm en startpunkt x_0 och en första steglängdsparameter.

(i) Beräkna $f(x_k)$.

(ii) Algoritm A: Väljer ett steg s_k.

(iii) beräkna $\rho_k = f(x_k) - f(x_k + s_k)$.

(iv) är $f(x_k) - f(x_k + s_k) > 0$ låt $x_{k+1} = x_k + s_k$. Annars låt $x_{k+1} = x_k$.

(v) Algoritm B: Uppdaterar steglängdsparametern.
Algoritm C: Uppdaterar stegriktningarna.

Efter att ha gått igenom Algoritm A, B och C kommer en mer detaljerad beskrivning.

I de metoder som beskrivits har vi sett att det i varje iteration görs ett antal teststeg innan det slutgiltiga steget väljs. Hur teststegen har valts har bestämts på förhand och har inte berott på målfunktionens utseende. För att ge ett exempel så gick sökningen i Hook och Jeeves metod först längs mönstersökningsriktningen och sedan längs koordinatriktningarna, medans man i cykliska koordinatmetoden bara testade längs koordinatriktningarna. Denna egenskap, att söka minimum genom att följa ett förutbestämt mönster som valts oberoende av målfunktionen, är som sagt en generell egenskap hos en mönstersöknings-metod. För att kunna genomföra beviset för alla metoder samtidigt beskriver vi den generella formen för hur ett sådant mönster ser ut.

Ett mönster är definierat av två komponenter; en basmatris B och en genererande matris C_k. I de metoder som beskrivits har man genomgående börjat med att bestämma koordinataxlarna, eller basen; $d_1 ... d_n$. Basmatrisens funktion är att spänna upp rummet. Den måste alltså vara inverterbar och ligga i $\mathbb{R}^{n\times n}$. Basen bestäms alltid på förhand och vanligast är att man låter enhetsmatrisen vara basmatris, men det behöver inte vara så. Det finns situationer då det är mer lämpligt att välja en annan bas.

Den genererande matrisen genererar teststegen. Dess kolumner utgörs av varje möjlig testriktning i en iteration. Dessa riktningar kan ändras mellan iterationerna och matrisen betecknas därför C_k där indexet k står för iteration nr k. C_k är en matris i $\mathbb{Z}^{n\times p}$ där $p > 2n$. Att $p > 2n$ innebär att det finns minst $2n+1$ möjliga testriktningar som alla är representerade i kolumnerna i C_k. Att de är minst $2n + 1$ beror på att alla mönstersöknings-metoder någon gång i varje iteration gör koordinatsökning, dvs söker parallellt med koordinataxlarna definierade av B i båda riktningarna. Att inte ta något steg alls, dvs. att $x_k = x_{k+1}$ är också ett möjligt teststeg gemensamt för alla mönstersöknings-metoder, därav är $p > 2n$. Om en algoritm även gör det vi kallat mönstersökning blir de möjliga testriktningarna fler.

Man kan på ett naturligt sätt dela upp C_k på följande sätt.

$$C_k = \left[\begin{array}{cccc} M_k & -M_k & L_k \end{array} \right] = \left[\begin{array}{cc} \Gamma_k & L_k \end{array} \right], \tag{3}$$

där M_k är en inverterbar $n \times n$-matris som representerar de n möjliga testriktningarna parallella med axlarna i B, dvs $M_k \in \mathbb{Z}^{n \times n}$ och $-M_k$ är förstås samma sak i motsatt riktning. Om $\mathbf{M}$ är mängden av alla sådana matriser M_k så kräver vi att $\mathbf{M}$ är en ändlig mängd. L_k är $n \times (p-2n)$-matrisen som minst består av en kolumn av nollor (då inget steg görs), och, i fall av mönstersökning, av kolumnerna som utgör varje möjligt koordinatsteg kombinerat med den bestämda mönstersökningsriktningen. Alltså gäller $L_k \in \mathbb{Z}^{n \times (p-2n)}$.

Ett mönster P_k definieras nu som

$$P_k = BC_k = \left[\begin{array}{cccc} BM_k & -BM_k & BL_k \end{array} \right] = \left[\begin{array}{cc} B\Gamma_k & BL_k \end{array} \right] \tag{4}$$

Ett mönster utgör alltså alla möjliga testriktningar i basen B. Men ett teststeg är inte bara en riktning, utan också en steglängd. Som vi har sett i exemplen så har längden på det första steget bestämts på förhand. Efterhand man närmat sig minimum har steglängden minskats. Den varierar alltså mellan iterationerna, så i iteration k betecknar vi steglängdsparametern med Δ_k. För Δ_k gäller att $\Delta_k \in \mathbb{R}, \Delta_k > 0$.

Vi har nu vad som behövs för att definiera ett teststeg s_k^i.

$$s_k^i = \Delta_k B c_k^i \tag{5}$$

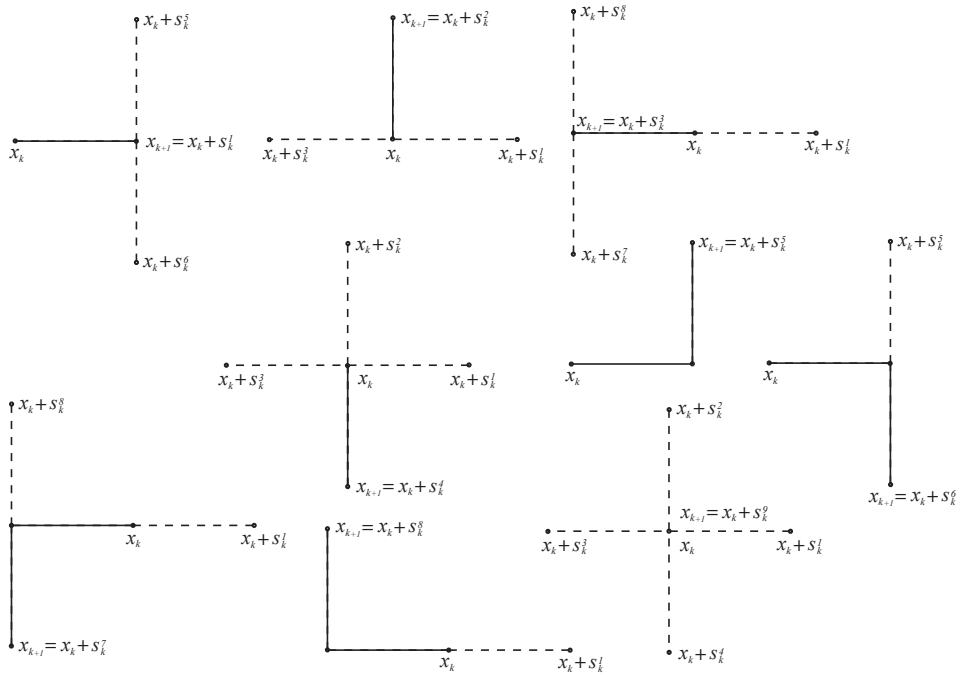
där Bc_k^i är någon av kolumnerna i P_k. Under en iteration genomförs endast ett av de möjliga teststegen. Vi kallar detta teststeg för *det accepterade teststeget* och betecknar det med s_k.

Innan vi går vidare ska vi titta på ett exempel som visar på hur cykliska koordinatmetoden passar in i mönstret. Låt oss anta att f är en tvåvariabelfunktion, att vi har valt $B = I$ och att vi är i iteration nr k i någon punkt x_k. Bilden på nästa sida visar vilka olika situationer som kan uppstå. De streckade linjerna står för misslyckade teststeg och de heldragna för lyckade. Vi ser hur alla de möjliga stegen representeras i C_k nedan.

$$C_k = \begin{pmatrix} 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\ 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0 \end{pmatrix}$$

Eftersom man i cykliska koordinatmetoden alltid söker längs axlarna så uppdateras aldrig C_k. Detta är speciellt för denna metod.

Vi har nu sett hur P_k visar vilka möjligheter som finns i en iteration k, men notera att P_k inte säger något om villkoren för att acceptera ett visst teststeg och inte heller något om vad som bestämmer om det valda teststeget ska leda till vår nya punkt. Det är Algoritm A i GMA som väljer vilket av teststegen i $\Delta_k P_k$ som ska accepteras. Följande krav ställs på Algoritm A.



Figur 13: De möjliga stegen i en itteration.

Algoritm A genererar bara teststeg s_k som uppfyller

(i) $s_k \in \Delta_k P_k$,

(ii) $\min\left\{f(x_k + s_k^i), s_k^i \in \Delta_k B\Gamma_k\right\} < f(x_k) \Rightarrow f(x_k + s_k) < f(x_k)$.

Detta innebär att om man kan hitta ett teststeg parallellt med koordinaterna som ger ett mindre funktionsvärde än punkten vi är i, så kommer det accepterade teststeget s_k också att ge ett mindre funktionsvärde. Observera att det inte ställs något krav på att $f(x_k+s_k) \leq \min\{f(x_k + y), y \in \Delta_k B\Gamma_k\}$ ska gälla. I praktiken innebär det att det är tillåtet för GMA att ta "för stora" steg, eller, för att återkoppla till vår tidigare diskussion, det ställs inget krav på att teststeget ska vara tillräckligt avtagande för att det ska accepteras.

Algoritm A är nu vilken algoritm som helst som uppfyller ovan nämnda villkor

I de olika exemplen på mönstersöknings-algoritmer som vi har tittat på så har vi sett att om inget av teststegen i iteration k var avtagande, så lät algoritmen $x_{k+1} = x_k$ vartefter den minskade steglängdsparametern. I MDS-metoden såg vi även hur vi kunde få en större steglängdsparameter i och med att vi expanderade simplexet då $\rho_k > 0$. Algoritm B nedan ger de generella reglerna för hur GMA uppdaterar Δ_k.

ALGORITM B.
Givet $\tau \in \mathbf{Q}$, $\tau > 1$
låt $\{w_0, w_1, ..., w_L\} \subset \mathbf{Z}$ med $w_0 < 0$ och $w_i \geq 0, i = 1, ..., L$,
låt $\theta = \tau^{w_0}$ och låt $\lambda_k \in \Lambda = \{\tau^{w_1}, ..., \tau^{w_L}\}$ med $L \equiv |\Lambda| < +\infty$.

(i) Om $\rho_k \leq 0$, låt $\Delta_{k+1} = \theta\Delta_k$.

(ii) Om $\rho_k > 0$, låt $\Delta_{k+1} = \lambda_k\Delta_k$.

Detta innebär att om teststeget genererat i Algoritm A ger ett större värde på målfunktionen så minskas steglängden, medan den förblir densamma eller ökar om teststeget ger ett mindre värde.

Algoritm C ska ju vara en algoritm vars funktion är att uppdatera testriktningarna, dvs uppdatera C_k. Det enda krav som ställs på denna algoritm är att den ska generera kolumner i C_k som uppfyller (3) med de villkor som är satta på M_k och L_k där.

Den generaliserade mönstersöknings-algoritmen för optimering utan bivillkor blir nu:

GMA - GENERALISERAD MÖNSTERSÖKNINGS-ALGORITM.
Bestäm bas B, genererande matris C_k, startpunkt $x_0 \in \mathbf{R}^n$ och en första steglängd $\Delta_0 > 0$. Välj algoritmer A, B och C.
För $k = 0, 1, ...$

(i) Beräkna $f(x_k)$.

(ii) ALGORITM A: Bestämmer ett steg s_k.

(iii) Beräkna $\rho = f(x_k) - f(x_k + s_k)$,

(iv) Om $\rho > 0$ låt $x_{k+1} = x_k + s_k$. Annars låt $x_{k+1} = x_k$.

(v) ALGORITM B: Uppdaterar Δ_k.
ALGORITM C: Uppdaterar C_k.

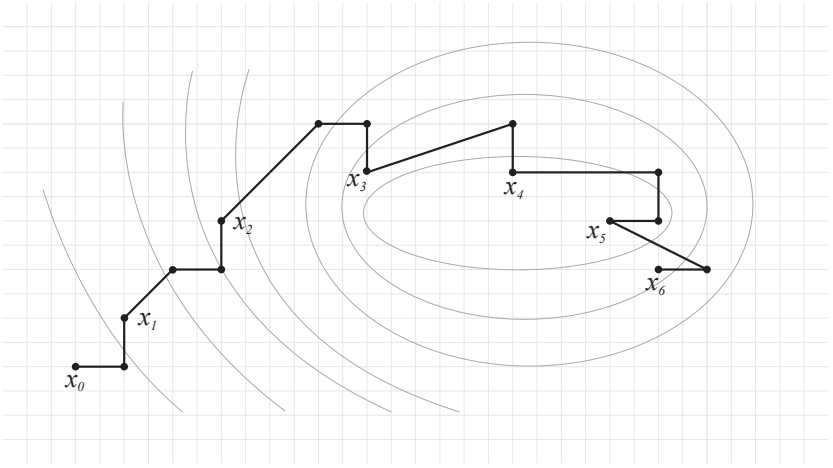
Med $s_k = 0$ i ett misslyckat steg ser vi hur algoritmen genererar punkter på formen $x_N = x_0 + \sum_{k=0}^{N-1} s_k$ som närmar sig minimum då $k \to \infty$. Vi har sett villkoren för att en algoritm ska ha detta beteende. För att få djupare förståelse i innebörden av detta illustrerar jag hur dessa villkor uppfylls av Hook och Jeeves metod.

5.1.1 Hook och Jeeves och den generella modellen

Innan vi börjar ska vi snabbt påminna oss om strukturen i Hook och Jeeves metod. Basen B är oftast definierad till enhetsmatrisen. Antag att vi är i

punkten x_k. Det sista som hände i föregående iteration var att vi gjorde en koordinatsökning för att hamna i denna punkt. För att komma vidare till punkten x_{k+1} börjar vi med en mönstersökning som är ett steg av samma riktning och längd som hela steget $x_k - x_{k-1}$ vartefter vi gör en ny koordinatsökning. Vi är nu i punkten x_{k+1}. Så ett steg s_k består först av en mönstersökning som beror av föregående steg och sedan en koordinatsökning. Hur uttrycks då detta i vår genererande matris C_k?

I Hook & Jeeves metod består C_k av $2 \cdot 3^n$ kolumner där de första 3^n kolumnerna består av alla möjliga koordinatsökningar dvs. alla sätt att med repetition välja ut n ur $\{-1, 0, 1\}$. De återstående 3^n kolumnerna består av alla möjliga koordinatriktningar adderat med den bestämda mönstersökningsriktningen. Exemplet nedan tillsammans med Algoritm H&J som visar hur de olika matriserna har uppstått, förtydligar förhoppningsvis något.



Figur 14: Exemplet är i två dimensioner. I iteration k har mönstersteget har alltid samma längd och riktning som vektorn $(x_k - x_{k-1})$.

De feta kolumnerna representerar steget som tas, de understrukna representerar koordinatsökningen.

$$
\mathrm{C}_0 = \left(\begin{array}{ccccccccc|ccccccccc}
1 & 0 & -1 & 0 & \underline{\mathbf{1}} & 1 & -1 & -1 & 0 & 1 & 0 & -1 & 0 & 1 & 1 & -1 & -1 & 0 \\
0 & 1 & 0 & -1 & \underline{\mathbf{1}} & -1 & -1 & 1 & 0 & 0 & 1 & 0 & -1 & 1 & -1 & -1 & 1 & 0
\end{array}\right)
$$

$$
\mathrm{C}_1 = \left(\begin{array}{ccccccccc|ccccccccc}
1 & 0 & -1 & 0 & \underline{1} & 1 & -1 & -1 & 0 & 2 & 1 & 0 & 1 & \mathbf{2} & 2 & 0 & 0 & 1 \\
0 & 1 & 0 & -1 & \underline{1} & -1 & -1 & 1 & 0 & 1 & 2 & 1 & 0 & \mathbf{2} & 0 & 0 & 2 & 1
\end{array}\right)
$$

$$
\mathrm{C}_2 = \left(\begin{array}{ccccccccc|ccccccccc}
1 & 0 & -1 & 0 & 1 & \underline{1} & -1 & -1 & 0 & 3 & 2 & 1 & 2 & 3 & \mathbf{3} & 1 & 1 & 2 \\
0 & 1 & 0 & -1 & 1 & \underline{-1} & -1 & 1 & 0 & 2 & 3 & 2 & 1 & 3 & \mathbf{1} & 1 & 3 & 2
\end{array}\right)
$$

$$
\mathrm{C}_3 = \left(\begin{array}{ccccccccc|ccccccccc}
1 & 0 & -1 & \underline{0} & 1 & 1 & -1 & -1 & 0 & 4 & 3 & 2 & \mathbf{3} & 4 & 4 & 2 & 2 & 3 \\
0 & 1 & 0 & \underline{-1} & 1 & -1 & -1 & 1 & 0 & 1 & 2 & 1 & \mathbf{0} & 2 & 0 & 0 & 2 & 1
\end{array}\right)
$$

$$
\mathrm{C}_4 = \left(\begin{array}{ccccccccc|ccccccccc}
1 & 0 & -1 & 0 & 1 & 1 & \underline{-1} & -1 & 0 & 4 & 3 & 2 & 3 & 4 & 4 & \mathbf{2} & 2 & 3 \\
0 & 1 & 0 & -1 & 1 & -1 & \underline{-1} & 1 & 0 & 0 & 1 & 0 & -1 & 1 & -1 & \mathbf{-1} & 1 & 0
\end{array}\right)
$$

Som ni ser förändras inte de första $3^2 = 9$ kolumnerna.

I Algoritm H&J är $c_k^p = x_k - x_{k-1}$, dvs c_k^p representerar mönstersteget i iteration k som alltså är detsamma som det accepterade teststeget i iteration $k-1$. c_k^p är den sista kolumnen i C_k. Bc_k är teststeget accepterat i iteration k dvs $x_{k+1} - x_k$ som alltså kommer att bli mönstersteg i iteration $k+1$.

Algoritm H&J: Uppdaterar C_k.
För $i = 3^n + 1, ..., 2 \cdot 3^n$ låt

$$c_{k+1}^i = c_k^i + Bc_k - c_k^p$$

Det är uppenbart att kolumnerna i C_k alltid består av heltal och Algoritm H&J uppfyller därmed villkoren för den generaliserade Algoritm C.

Algoritm A i vår generaliserade modell, består av steg (iii),(i) och (ii) i Hook och Jeeves metod i avsnitt 2.2. Man ser att teststeget $x_{k+1} - x_k$ alltid är representerat bland de sista 3^n kolumnerna i matrisen $\Delta_k P_k = \Delta_k B C_k$, där C_k är matrisen genererad av Algoritm H&J ovan. Vi ser också hur steg (ii) och (iv) i Hook och Jeeves metod garanterar att varje accepterat teststeg är avtagande. Så villkoren på den generella Algoritm A är uppfyllda

Det är enkelt att se hur Algoritm B uppfylls av steg (iv) i Hook & Jeeves metod. Om det accepterade teststeget gett ett större värde på målfunktionen så minskas steglängden med en halv, annars förblir den oförändrad. Detta innebär i termer av Algoritm B att $\theta = 1/2$ och $w_1, ..., w_L$ alla är lika med 0. Vi har i stora drag sett hur Hook & Jeeves metod uppfyller kraven för GMA.

5.2 Konvergensteori och beviset av $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$

Vi använder nu reglerna för hur en generaliserad mönstersöknings-metod är konstruerad för att visa att en algoritm som uppfyller dessa regler genererar en sekvens $\{x_k\}$ som är sådan att $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$. Det första steget är att visa att varje punkt genererad av GMA ligger på en heltals lattice. Detta innebär att det på ett kompakt område endast finns ett ändligt antal platser som en genererad punkt kan hamna på. Tillsammans med villkoret om avtagande steg i Algoritm A ger detta resultat att GMA kommer att stå och stampa på en och samma punkt om $k \to \infty$. Tillsamman med villkoren på Algoritm B användas detta sedan i beviset av Sats 5.2 som säger att någon delsekvens av Δ_k kommer att gå mot noll då k går mot oändligheten. Det slutliga beviset är sedan ett motsägelsebevis som använder ett resultat som kommer att visas senare, men som i princip säger att det finns en undre gräns $\Delta_{LB} > 0$ sådan att $\Delta_k > \Delta_{LB}$ om $\lim_{k\to\infty} \inf \|\nabla f(x_k)\| \neq 0$.

Sats 5.1 *Varje punkt x_N genererad av en generaliserad mönstersökningsalgoritm kan uttryckas på formen*

$$x_N = x_0 + (\beta^{r_{LB}}\alpha^{-r_{UB}})\Delta_0 B \sum_{k=0}^{N-1} z_k \tag{6}$$

där B är basmatris, x_0 är startpunkt och Δ_0 är första valet av steglängdsparametern.
$\beta/\alpha \equiv \tau$, med $\alpha, \beta \in \mathbb{N}$ som är relativt prima och τ är som i Algoritm B, r_{LB} och r_{UB} beror på N och är som i beviset nedan och $z_k \in \mathbb{Z}^n$ för $k = 0, ..., N-1$.

Bevis. Vi har sett att den generaliserade mönstersökningsalgorittmen genererar punkter på formen

$$x_N = x_0 + \sum_{k=0}^{N-1} s_k \tag{7}$$

där villkoren på Algoritm A ger att s_k är något av teststegen $s_k^i = \Delta_k B c_k^i$, $i = 1, ..., p$.
Från Algoritm B har vi att Δ_k kan skrivas som

$$\Delta_k = \theta^{q_k^0}\lambda_1^{q_k^1}\lambda_2^{q_k^2}...\lambda_L^{q_L^1}\Delta_0, \tag{8}$$

där vi kan förstå q_k^i på följande sätt. Efter k iterationer har man återanvänt λ_i q_k^i gånger för att uppdatera steglängden. För q_k^i gäller alltså $q_k^i \in \mathbb{Z}, q_k^i \geq 0$.

Vi påminner om restriktionerna på formen för θ och λ_i.
Med $\tau \in \mathbb{Q}, \tau > 1$ och $\{w_0, w_1, ..., w_L\} \in \mathbb{Z}$ hade vi

$$\theta = \tau^{w_0}$$

där $w_0 < 0$ och

$$\lambda_i = \tau^{w_i}$$

med $w_i \geq 0, i = 1, 2, ..., L$

Vi kan alltså skriva om (8):

$$\Delta_k = (\tau^{w_0})^{q_k^0}(\tau^{w_1})^{q_k^1}(\tau^{w_2})^{q_k^2}...(\tau^{w_L})^{q_k^L}\Delta_0 = \tau^{r_k}\Delta_0 \tag{9}$$

där $r_k = \sum_{i=0}^{L} w_i q_k^i$ och $r_k \in \mathbb{Z}$ eftersom $w_i, q_k^i \in \mathbb{Z}$.
Låt nu

$$r_{LB} = \min_{0 \leq k < N}\{r_k\} \qquad r_{UB} = \max_{0 \leq k < N}\{r_k\} \tag{10}$$

(7) och (9) ger

$$x_N = x_0 + \sum_{k=0}^{N-1} s_k = x_0 + \sum_{k=0}^{N-1} \Delta_k B c_k = x_0 + \sum_{k=0}^{N-1} \tau^{r_k} \Delta_0 B c_k. \qquad (11)$$

Att $\tau \in \mathbb{Q}$ medför att τ kan skrivas $\tau = \beta/\alpha$ där $\beta, \alpha \in \mathbf{N}$ är relativt prima. Med beteckningarna införda i (10) kan vi nu skriva

$$\tau^{r_k} = \frac{\beta^{r_k}}{\alpha^{r_k}} = \frac{\beta^{r_{LB}+(r_k - r_{LB})}}{\alpha^{r_{UB}+(r_k - r_{UB})}} = \beta^{r_{LB}} \alpha^{-r_{UB}} \gamma_k$$

med $\gamma_k \in \mathbb{Z}$. Där den sista likheten följer av att $r_k - r_{LB} \geq 0$ och $r_k - r_{UB} \leq 0$, samt att $\alpha, \beta \in \mathbb{Z}$. Byter vi ut τ^{r_k} mot $\beta^{r_{LB}} \alpha^{-r_{UB}} \gamma_k$ i (11) och låter $z_k = \gamma_k c_k$ så gäller $z_k \in \mathbb{Z}^n$ och vi får (6). □

Kontentan av nästa sats är att om målfunktionen och startpunkten är sådana att mängden av alla x som tar f till värden mindre än eller lika med startvärdet är kompakt, så kommer någon delsekvens av $\{\Delta_k\}$ att gå mot noll då k går mot oändligheten. I beviset kommer vi att använda den algebraiska struktur hos en punkt genererad av GMA som vi just visat. Vi kommer också att använda villkor (iv) i GMA och villkoren för Algoritm B. Följande definition kommer att behövas:

$$L(y) = \{x; f(x) \leq f(y)\}$$

Sats 5.2 *Om $L(x_0)$ är kompakt så är $\Delta_{LB} := \liminf_{k \to +\infty} \Delta_k = 0$.*

Bevis. Satsen visas genom motsägelse, så antag att den minsta steglängden är större än noll, dvs att $0 < \Delta_{LB} \leq \Delta_k$ för alla k.

Av villkor (iv) i GMA följer att alla punkter genererade av denna ligger inom det kompakta området $L(x_0)$. Eftersom $L(x_0)$ är kompakt så vet vi att det finns ett tal $s > 0$ sådant att $\|x_{k+1} - x_k\| = \|s_k\| \leq s$ för alla k. Låt σ vara det minsta singulärvärdet till basen B definierad i avsnitt 5.1.
För alla steg s_k sådana att $\|s_k\| \neq 0$ kan vi nu skriva

$$s \geq \|s_k\| = \Delta_k \|B c_k\| \geq \Delta_k \sigma \|c_k\| \geq \Delta_k \sigma,$$

där den sista olikheten följer av att $c_k \in \mathbb{Z}^n$ och av att $s_k \neq 0 \Rightarrow c_k \neq 0$.
Så för Δ_k gäller alltså att $0 < \Delta_{LB} \leq \Delta_k \leq \frac{s}{\sigma} < \infty$. Vi vet från (9) att Δ_k kan skrivas $\Delta_k = \tau^{r_k} \Delta_0$ med $r_k \in \mathbb{Z}$ och $\Delta_0 > 0$ så vi kan skriva $0 < \frac{\Delta_{LB}}{\Delta_0} \leq \tau^{r_k} \leq \frac{s}{\sigma \Delta_0} < \infty$. Men eftersom $r_k \in \mathbb{Z}$ så måste det finnas tal r_{UB} och r_{LB} sådana att

$$r_{LB} = \min_{0 \leq k < +\infty} \{r_k\} \qquad r_{UB} = \max_{0 \leq k < +\infty} \{r_k\} \qquad (12)$$

vilket gör att (6) nu kan sägas vara sann för (12) istället för (10).

Eftersom alla punkter genererade av GMA är på formen $x_N = x_0 + (\beta^{r_{LB}}\alpha^{-r_{UB}})\Delta_0 B\sum_{k=0}^{N-1} z_k$ där $z_k \in \mathbb{Z}^n$ och det bara finns ett ändligt antal heltalspunkter på en kompakt mängd så måste mängden av olika element i följden vara en ändlig mängd. Så för $k \to \infty$ finns det *minst* en punkt x_* sådan att $x_k = x_*$ för oändligt många k. Villkor (iv) i GMA säger att ett nytt steg accepteras om och endast om den nya punkten ger ett mindre värde på målfunktionen. Detta innebär att om algoritmen en gång har lämnat en punkt så kommer den aldrig att återvända till densamma. Därför måste punkten x_* vara unik i den meningen att för ett tillräckligt stort tal N så är $x_k = x_*$ för *alla* $k \geq N$. Men att algoritmen står och hoppar på punkten x_* i oändligheten är detsamma som att $\rho_k = 0$ i oändligt många, på varandra följande, iterationer. Tillsammans med villkor (i) i Algoritm B medför detta att $\Delta_k \to 0$ och vi har en motsägelse. □

Notera att det tack vare de genererade punkternas algebraiska struktur räcker med att vi kräver att det accepterade teststeget är avtagande i varje iteration för att visa att någon delsekvens av $\{\Delta_k\}$ går mot noll. En proposition återstår för att det generella konvergensbeviset ska kunna slutföras. I beviset av Proposition 5.3 används flera av resultat som kommer fram i avsnitt 6.2 och därför ligger beviset i slutet av detta avsnitt. Det ska poängteras att vi dock inte använder några av de starkare kraven på GMA när vi bevisar Proposition 5.3.

Proposition 5.3 *Om $L(x_0)$ är kompakt, f är kontinuerligt differentierbar runt $L(x_0)$ och om $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| \neq 0$ så finns en konstant $\Delta_{LB} > 0$ sådan att $\Delta_k > \Delta_{LB}$ för alla k.*

Sats 5.4 *Antag att $L(x_0)$ är kompakt och att f är kontinuerligt differentierbar runt $L(x_0)$. Då gäller för sekvensen $\{x_k\}$ genererad av GMA att*

$$\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$$

Bevis. Antag motsatsen att $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| \neq 0$. Från Proposition 5.3 vet vi att det finns en undre gräns $\Delta_{LB} > 0$ sådan att $\Delta_k > \Delta_{LB}$ för alla k vilket motsäger Sats 5.2. □

Sammanfattningsvis. Vi har förutsatt att $L(x_0)$ är en kompakt mängd och sedan sett hur punkterna genererade av GMA givet denna förutsättning endast har ett ändligt antal platser att hamna på. Eftersom en ny punkt bara accepteras om den reducerar värdet på f så visste vi att algoritmen aldrig återvänder till en plats som den en gång har lämnat. Detta ledde till vår slutledning att någon delsekvens av steglängdsparametern måste konvergera mot noll. Vi använde sedan faktumet att om GMA aldrig konvergerar mot en stationär punkt så kommer ingen delsekvens av Δ_k att konvergera mot noll. Detta ledde till vår motsägelse och vår första viktiga sats var därmed bevisad.

6 $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$

6.1 Tre nya krav på GMA

Den breda generella modellen av GMA som definierades i 5.1 var konstruerad så att den passar in på alla mönstersökningsalgoritmer. De villkor som sattes på modellen var tillräckliga för att visa att de algoritmer som faller under den uppfyller $\lim_{k\to+\infty} \inf \|\nabla f(x_k)\| = 0$. För att visa det starkare resultatet $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$ krävs som sagt ytterligare restriktioner på GMA. Detta gör att vissa av de ursprungliga algoritmerna inte passar in i denna snävare modell utan att några ändringar måste göras.

De nya kraven på GMA är

1. Vi kräver existensen av en konstant $C > 0$, sådan att för varje k så är $C > \left\|c_k^i\right\|$ med $i = 1, ..., p$. Givet en sådan konstant kan vi visa följande Lemma

   Lemma 6.1 *Antag att det existerar en konstant $C > 0$ sådan att i varje iteration k så är $C > \left\|c_k^i\right\|$ för alla $i = 1, ..., p$. I så fall finns det en konstant $\psi_* > 0$ oberoende av k sådan att*

   $$\Delta_k \geq \psi_* \left\|s_k^i\right\|$$

   för alla teststeg s_k^i genererade av GMA.

   Bevis. Per definition så gäller $s_k^i = \Delta_k B c_k^i$. Så alltså har vi

   $$\left\|s_k^i\right\| = \left\|\Delta_k B c_k^i\right\| \leq \Delta_k \|B\| \left\|c_k^i\right\| \leq \Delta_k C \|B\| .$$

   Där den första olikheten följer av Cauchy-Schwarz olikhet och det faktum att $\Delta_k \geq 0$. Och den andra olikheten följer av vårt antagande. B är en basmatris som spänner upp $\mathbb{R}^n$ och har därför norm strikt större än noll. Sätt $\psi_* = \frac{1}{C\|B\|}$. □

2. Vi måste skärpa kraven på Algoritm A. Vi kräver nu att ett accepterat teststeg s_k ska ge ett funktionsvärde som är mindre än eller lika med det minsta av funktionsvärdena av teststegen $s_k^i \in \Delta_k B\Gamma_k$. Eller med andra ord.

   Algoritm A (Starkare krav)

   (i) $s_k \in \Delta_k P_k \equiv \Delta_k BC_k \equiv \Delta_k \left[\begin{array}{cc} B\Gamma_k & BL_k \end{array}\right]$.

   (ii) Om $\min\{f(x_k + y), y \in \Delta_k B\Gamma_k\} < f(x_k)$
   så är $f(x_k + s_k) \leq \min\{f(x_k + y), y \in \Delta_k B\Gamma_k\}$.

   Detta innebär att en mönstersöknings algoritm måste testa alla koordinatriktningar innan den bestämmer sig för att acceptera ett teststeg.

3. Vi kräver att $\lim_{k\to+\infty} \Delta_k = 0$. Detta kan tex garanteras genom att aldrig låta algoritmen öka steglängdsparametern, dvs genom att låta $w_i = 0$, $i = 1, ..., L$ i Algoritm B.

Inga av de nya kraven är orimliga att ställa på en mönstersöknings algoritm. Tex uppfyller både Hook och Jeeves Metod och Cykliska koordinat metoden redan dessa villkor.

6.2 Konvergensteori och beviset av $\lim_{k\to+\infty} \|\nabla f(x_k)\| = 0$

Följande Lemma säger att om vi i iteration nr k inte har nått ett minimum så kommer en mönstersöknings algoritm generera ett avtagande steg inom ett ändligt antal iterationer. Detta innebär dels att algoritmen genererar avtagande riktningar och dels att den ger en steglängd som förr eller senare blir tillräckligt liten.

Lemma 6.2 *Om f är kontinuerligt differentierbar i ett område runt $L(x_0)$ och $\nabla f(x_k) \neq 0$ så finns det ett heltal $q \geq 0$ sådant att $\rho_{k+q} > 0$.*

Bevis. Eftersom kolumnerna i M_k är linjärt oberoende och B bildar bas i $\mathbb{R}^n$ per definition, så vet vi att även kolumnerna BM_k är linjärt oberoende. Detta innebär att det i punkten x_k finns minst en testriktning $d_k^i = Bc_k^i - x_k$, där $c_k^i \in M_k$ sådan att $\nabla f(x_k)^t d_k^i \neq 0$. Eftersom $\Gamma_k = \begin{bmatrix} M_k & -M_k \end{bmatrix}$ vet vi dessutom att det för något $c_k^i \in \Gamma_k$ gäller att $\nabla f(x_k)^t d_k^i < 0$. Så för ett tillräckligt litet $h > 0$ finns det en riktning bland kolumnerna i Γ_k som ger $f(x_k + hd_k^i) < f(x_k)$.

Antag att vi är i iteration k och GMA har genererat ett misslyckat teststeg. Eftersom det alltid finns minst en avtagande riktning att välja bland i Γ_k, så vet vi att det misslyckade teststeget är orsakat av att steglängden varit för stor. Men i varje misslyckad iteration låter GMA $x_{k+1} = x_k$ och minskar därefter Δ_k med parametern $\theta < 1$. Det finns alltså ett heltal $q \geq 0$ sådant att $\theta^q \Delta_k$ är tillräckligt litet och ger $\rho_{k+q} > 0$. □

Vi vet nu att så länge en mönstersökningsalgoritm inte har nått ett minimum så kommer matrisen P_k alltid att bestå av åtminstone någon avtagande riktning. Vi vet också att detta tillsammans med Algoritm B garanterar att GMA fortsätter att förflytta sig i avtagande riktning så länge den inte stöter på en minimipunkt. Men detta är, som vi vet från avsnitt 3 inte tillräckligt för att garantera konvergens mot en minimipunkt.

I beviset av Lemma 6.4 får vi se hur GMA faktiskt uppfyller vinkelvillkoret (1) i varje iteration. Mer specifikt ska vi få se att GMA genererar något avtagande teststeg vars mellanliggande vinkel till $-\nabla f(x_k)$ har en övre gräns som endast beror av n, M_k och B. Detta innebär att man genom att anpassa matriserna M_k och B kan forma en mönstersökningsalgoritm så att den undviker att ta steg som nästan är vinkelräta mot $-\nabla f(x_k)$. Istället för att direkt visa existensen av en övre gräns på en sådan vinkel θ visas existensen av en undre gräns för $\cos\theta$. Men för att visa det generella fallet tittar

vi först på ett specialfall i Lemma 6.3 där $B = I$ och $P_k = \begin{bmatrix} I & -I & 0 \end{bmatrix}$. Dvs specialfallet där GMA bara genererar teststeg av längd 1 parallellt med axlarna i I. Lemma 6.3 gör följande. Till en normerad vektor y i rummet definieras $\theta(y)$ som den minsta av vinklarna mellan y och $e_1, e_2 ... e_n$, vektorerna i $B = I$. Lemmat säger sedan att för alla normerade vektorer y i rummet så kan $\theta(y)$ aldrig bli större än att $\cos\theta(y) = \frac{1}{\sqrt{n}} > 0$ med $\theta(y) \in [0, \pi/2]$. Vi ser direkt att $\theta(y)$ ökar med antalet dimensioner n.

Lemma 6.3 *Antag att* $\|y\| = 1$ *och definiera* $\theta_j(y) \in [0, \pi/2]$ *som vinkeln mellan en vektor* y *och* e_j *där* $e_1, e_2, ..., e_n$ *är standardbasvektorerna. Låt* $\theta(y)$ *vara sådan att*

$$\cos\theta(y) = \max_{1 \le j \le n} \{|\cos\theta_j(y)|\}.$$

Då gäller

$$\min_{y \in \boldsymbol{R}^n} \cos\theta(y) = \frac{1}{\sqrt{n}}. \tag{13}$$

*Bevis.*Vi vet att $|y_j| = \left|y^t e_j\right| = |\cos\theta_j(y)|$ där $y = (y_1, y_2, ..., y_n)^t$. Vi vet också att $\|y\| = 1 \Leftrightarrow \sum_{j=1}^{n} |y_j|^2 = 1$ vilket medför att det för åtminstone något j gäller att $|y_j| \ge 1/\sqrt{n} \Leftrightarrow \left\|y^t e_j\right\| = \|\cos\theta_j(y)\| \ge 1/\sqrt{n}$. Dvs för varje vektor y finns åtminstone någon vektor e_j sådan att $\|\cos\theta_j(y)\| \ge 1/\sqrt{n}$ så det är klart att det också för varje vektor y gäller att $\cos\theta(y) \ge 1/\sqrt{n}$. Speciellt när $y = \alpha_1 e_1 + \alpha_2 e_2 + ... + \alpha_n e_n$ med $\alpha_j = \pm 1/\sqrt{n}$ så har vi att $\cos\theta(y) = 1/\sqrt{n}$. □

Eftersom Lemma 6.3 gäller för alla riktningar y i rummet så gäller det även för godtycklig gradientriktning. Så så länge teststegen är av längd 1 parallella med enhetskoordinaterna så finns det oavsett gradientens riktning alltid något avtagande teststeg vars mellanliggande vinkel till gradienten är bunden enligt ovan. Den undre gränsen för $\cos\theta(y)$ kommer nu att användas för att visa ett liknande resultat men där teststegen s_k^i inte har några begränsningar annat än de som ligger i den ursprungliga definitionen av s_k^i i (5).

Lemma 6.4 *Till varje mönstersökningsalgoritm finns en konstant* $\xi > 0$ *som beror av matriserna* M *och* B*, och antalet dimensioner* n*, sådan att för alla* $k \ge 0$ *och vektorer* $x \ne 0$ *så gäller*

$$\max\left\{\frac{\left|x^t s_k^i\right|}{\|x\|\left\|x^t s_k^i\right\|}, i = 1...p\right\} \ge \xi$$

Bevis. Eftersom vi inte bryr oss om längden av teststegen $s_k^i = \Delta_k B c_k^i$ i det här beviset räcker det med att visa att det gäller för vektorerna i

$BC_k = \begin{bmatrix} BM_k & -BM_k & BL_k \end{bmatrix}$ för varje k. Och om vi lyckas hitta någon kolumn i matrisen BM_k som uppfyller olikheten så har vi visat lemmat, så vi tar en titt på BM_k.

Låt B och $M_k \in \mathbf{M}$ vara godtyckligt valda enligt reglerna för GMA. Vi skriver $BM_k = \begin{bmatrix} y_k^1 & y_k^2 & ... & y_k^n \end{bmatrix}$. Då gäller för varje nollskiljd vektor x och *något* j

$$\frac{\left|x^t y_k^j\right|}{\|x\| \left\|y_k^j\right\|} = \frac{\left|x^t BM_k e_j\right|}{\|x\| \, \|BM_k e_j\|} = \frac{\left|((BM_k)^t x)^t e_j\right|}{\|x\| \, \|BM_k e_j\|}.$$

Om vi låter $w = (BM_k)^t x$ så kan vi skriva $x = (BM_k)^{-t} w$ och vi får

$$\frac{\left|x^t y_k^j\right|}{\|x\| \left\|y_k^j\right\|} = \frac{\left|w^t e_j\right|}{\|(BM_k)^{-t} w\| \, \|BM_k e_j\|} \geq \frac{\left|w^t e_j\right|}{\|(BM_k)^{-t}\| \, \|w\| \, \|BM_k\| \, \|e_j\|} =$$

$$\frac{1}{\|(BM_k)^{-t}\| \, \|BM_k\|} \frac{\left|w^t e_j\right|}{\|w\| \, \|e_j\|} = \frac{1}{\|(BM_k)^{-1}\| \, \|BM_k\|} \frac{\left|w^t e_j\right|}{\|w\| \, \|e_j\|} \geq$$

$$\frac{1}{\|(BM_k)^{-1}\| \, \|BM_k\|} \frac{1}{\sqrt{n}} > 0.$$

där den första olikheten följer av Cauchy-Schwarz olikhet och den andra olikheten följer av (13) för *något* j. Eftersom $\mathbf{M}$ en ändlig mängd så kan vi välja den matris $M_k \in \mathbf{M}$ som ger det största värdet på $\left\|(BM_k)^{-1}\right\| \|BM_k\|$ för att bestämma ξ så att

$$\xi = \min_{M_k \in \mathbf{M}} \left\{ \frac{1}{\|(BM_k)^{-1}\| \, \|BM_k\| \, \sqrt{n}} \right\}. \tag{14}$$

Så om y_k^j är den av alla vektorer i BM_k som har minst mellanliggande vinkel θ_k till x så gäller alltså i varje iteration k och för alla x att

$$|\cos \theta_k| = \frac{\left|x^t y_k^j\right|}{\|x\| \left\|y_k^j\right\|} \geq \xi \tag{15}$$

och speciellt så gäller detta förstås då $x = \nabla f(x_k)$. □

Vi ser alltså hur den övre gränsen för vinkeln mellan teststegen och gradienten ökar, dels med antalet dimensioner och dels med $\left\|(BM_k)^{-1}\right\| \|BM_k\|$. Det är alltså avgörande för effektiviteten hos en mönstersökningsmetod hur matriserna M_k och B är bestämda.

Följande lemma visar hur steglängdsparametern förhindrar algoritmen från att ta allt för små steg. Genom att välja Δ_0 relativt stor så använder alltså GMA tekniken backtracing som nämndes i avsnitt 3.

Lemma 6.5 *Det existerar en konstant $\zeta_* > 0$ oberoende av k, sådan att, för varje teststeg $s_k^i \neq 0$ definierat som i* (5)*, så gäller*

$$\left\|s_k^i\right\| \geq \zeta_* \Delta_k.$$

Bevis. Per definition gäller att $s_k^i = \Delta_k B c_k^i$ där $c_k^i \in C_k$. Villkoren på C_k satta i (3) ger $c_k^i \in \mathbf{Z}^n$. Låt ζ_* vara det minsta singulärvärdet till B. Då gäller

$$\left\|s_k^i\right\| = \Delta_k \left\|Bc_k^i\right\| \geq \Delta_k \zeta_* \left\|c_k^i\right\| \geq \Delta_k \zeta_*.$$

Första likheten följer av att $\Delta_k > 0$. Sista olikheten följer av antagandet att $s_k^i \neq 0$ vilket medför att $c_k^i \neq 0$, samt det faktum att $c_k^i \in \mathbf{Z}^n$, vilket tillsammans ger $\left\|c_k^i\right\| \geq 1$. □

Vi vet nu från Lemma 6.2 att GMA förr eller senare genererar något lyckat teststeg, vi vet från Lemma 6.4 att något eller några av de lyckade teststegen i en iteration uppfyller vinkelvillkoret (1) och från Lemma 6.5 vet vi i någon mån att Δ_k gör att GMA inte tar för små steg. För att kunna visa $\lim_{k\to\infty} \|\nabla f(x_k)\| = 0$ behöver vi veta lite mer om det accepterade teststeget, bland annat att det inte är för långt. Proposition 6.6 säger att GMA genererar åtminstone något lyckat teststeg om Δ_k är mindre än en viss, av k oberoende konstant δ. Och till följd av detta och av Algoritm B vet vi att om $\Delta_k < \delta$ så är $\Delta_{k+1} \geq \Delta_k$. Denna egenskap används i beviset av Sats 6.8.

Men i beviset av Proposition 6.6 kommer en annan viktig egenskap fram. Där binds nämligen graden av avtagande hos något av de lyckade teststegen (inte nödvändigtvis det accepterade teststeget.) i P_k till gradienten av f i x_k. Vad som kommer fram är faktiskt något så trevligt som varje iteration kommer att generera något lyckat teststeg som uppfyller (2), Armijos mått på tillräckligt avtagande.

Inför beviset av nästa resultat behöver vi följande definitioner.

$$X_* = \{x : \nabla f(x) = 0\}$$
$$dist(y, X_*) = \inf\{\|y - x\| ; x \in X_*\}$$

Eller med andra ord $dist(y, X_*)$ är det kortaste avståndet mellan en punkt y och mängden av stationära punkter X_*.

Proposition 6.6 *Antag att $L(x_0)$ är kompakt och f kontinuerligt differentierbar runt $L(x_0)$ och låt*

$$\Omega_\epsilon = \{x \in L(x_0) : dist(x, X_*) \geq \epsilon\}$$

med $\epsilon \geq 0$. Antag också att $x_0 \in \Omega_\epsilon$. Då vet vi att det finns ett $\delta > 0$ oberoende av k sådant att om $x_k \in \Omega_\epsilon$ och $\Delta_k \leq \delta$, så kommer iteration nr k att ge ett lyckat teststeg. Dvs. $\rho_k > 0$ och därmed $\Delta_{k+1} \geq \Delta_k$.

Bevis. Det är tillräckligt att titta på vektorerna i $\Delta_k B\Gamma_k$ eftersom om något av dessa steg är enkelt avtagande så är det accepterade teststeget s_k det också enligt villkoren på Algoritm A. Om s_k^i med $i = 1, ..., 2n$ är någon av vektorerna i $\Delta_k B\Gamma_k$, så ser vi att det finns ett tal ζ^* större än noll och oberoende av k så att $\left\|s_k^i\right\| \leq \zeta^* \Delta_k$ för $i = 1, 2, ..., 2n$ eftersom

$$\left\|s_k^i\right\| = \left\|\Delta_k B c_k^i\right\| \leq \|B\| \left\|c_k^i\right\| \Delta_k \leq \zeta^* \Delta_k \tag{16}$$

Den första olikheten är Cauchy-Schwarz olikhet. Den andra följer av att $M_k \in \mathbf{M}$ där $\mathbf{M}$ är en ändlig mängd. Från Lemma 6.5 har vi nu

$$\zeta_* \Delta_k \leq \left\|s_k^i\right\| \leq \zeta^* \Delta_k$$

för $i = 1, ..., 2n$.

Eftersom $x_0 \in \Omega_\epsilon$ så är $\nabla f(x_0) \neq 0$. Enligt Lemma 6.2 kommer därför ett ändligt antal iterationer att generera ett lyckat teststeg och vi kan därför skriva $N = \min\{k : x_k \neq x_0\}$.

Låt $C(y) = \{x : f(x) = f(y)\}$ och definiera $d = dist(L(x_N), C(x_0))$. Eftersom $f(x_N) < f(x_0)$ så är $L(x_N)$ och $C(x_0)$ disjunkta och därmed $d > 0$.

Vi söker en övre gräns för Δ_k som oberoende av k garanterar ett lyckat teststeg och vi testar med $d/2\zeta^*$. Antag alltså att $\Delta_k < d/2\zeta^*$. Då är $\left\|s_k^i\right\| \leq \zeta^* \Delta_k < d/2$ för $i = 1, ..., 2n$. I så fall ligger alla testpunkter x_k^i inom ett område med radien $d/2$. Kalla området för $B(x_k, d/2)$. Så om $k \geq N$ och $\Delta_k < d/2\zeta^*$ så har x_k som närmast avståndet d till randen av $L(x_0)$ och därmed gäller det att $B(x_k, d/2) \subset L(x_0)$.

Definiera nu $\alpha = \min_{x \in \Omega_\epsilon} \|\nabla f(x)\|$. Från definitionen av Ω_ϵ vet vi att $\alpha > 0$.

Eftersom ∇f är kontinuerlig i en omgivning av $L(x_0)$ så är ∇f även likformigt kontinuerlig i någon omgivning av $L(x_0)$. Enligt definitionen av likformig kontinuitet vet vi därför att det till varje tal $\beta > 0$ existerar ett tal $r > 0$ sådant att om $x, x_k \in L(x_0)$ så gäller

$$\|x - x_k\| \leq r \Rightarrow \|\nabla f(x) - \nabla f(x_k)\| \leq \beta. \tag{17}$$

Vi låter r vara sådan att (17) gäller för $\beta = \frac{\xi\alpha}{2}$. Med ξ som i Lemma 6.4. Om vi nu definierar $\delta = \frac{1}{\zeta^*} \min\left\{\frac{d}{2}, r\right\}$ och låter $\Delta_k < \delta$ med k fortfarande sådan att $k \geq N$, så är vi dels garanterade att

$$x_k^i \in B(x_k, \frac{d}{2}) \subset L(x_0), i = 1, ..., 2n \tag{18}$$

och dels att

$$\left\|\nabla f(x_k^i) - \nabla f(x_k)\right\| \leq \frac{\xi\alpha}{2}, i = 1, ..., 2n. \tag{19}$$

Vi ska se att om vi är i iteration $k \geq N$ och $x_k \in \Omega_\epsilon$ och om $\Delta_k < \delta$ så kommer vi att finna ett lyckat teststeg i matrisen P_k.

Välj något av teststegen $s_k^i = x_k^i - x_k$ ur $\Delta_k B\Gamma_k$, som uppfyller både

$$\nabla f(x_k)^t s_k^i < 0 \tag{20}$$

och

$$\frac{\left|\nabla f(x_k)^t s_k^i\right|}{\|\nabla f(x_k)\| \left\|x_k^i - x_k\right\|} \geq \xi \tag{21}$$

dvs något teststeg som är avtagande och vars mellanliggande vinkel till $-\nabla f(x_k)$ är uppåt begränsad enligt (21). Eftersom $x_k \in \Omega_\epsilon$ och därmed $\nabla f(x_k) \neq 0$ så finns enligt Lemma 6.4 och beviset av Lemma 6.2 ett sådant teststeg i $\Delta_k B\Gamma_k$.

Om $k \geq N$ och $\Delta_k < \delta$ så gäller nu alltså för vårt utvalda teststeg s_k^i (18), (19), (20) och (21). Vi kan nu använda medelvärdessatsen på detta teststeg som säger att det existerar ett $w \in (x_k, x_k^i)$ så att $f(x_k^i) - f(x_k) = \nabla f(w)^t(x_k^i - x_k)$ gäller. Detta kan skrivas om

$$f(x_k^i) - f(x_k) = \nabla f(x_k)^t(x_k^i - x_k) + (\nabla f(w) - \nabla f(x_k))^t(x_k^i - x_k) \tag{22}$$

Om den vänstra termen i högerledet i (22) vet vi fårn (21) att

$$\left|\nabla f(x_k)^t(x_k^i - x_k)\right| \geq \xi \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\|$$

och (20) ger sedan

$$\nabla f(x_k)^t(x_k^i - x_k) \leq -\xi \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\|.$$

Vi kan därför skriva om (22) till en olikhet som vi sedan utvecklar

$$f(x_k^i) - f(x_k) \leq -\xi \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\| + (\nabla f(w) - \nabla f(x_k))^t(x_k^i - x_k) \leq$$

$$-\xi \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\| + \left\|(\nabla f(w) - \nabla f(x_k))^t(x_k^i - x_k)\right\| \leq$$

$$(-\xi \|\nabla f(x_k)\| + \|\nabla f(w) - \nabla f(x_k)\|) \left\|x_k^i - x_k\right\| \leq$$

$$(-\xi \|\nabla f(x_k)\| + \frac{\xi}{2} \|\nabla f(x_k)\|) \left\|x_k^i - x_k\right\| =$$

$$-\frac{\xi}{2} \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\| < 0.$$

Den tredje olikheten följer av Cauchy-Schwarz. Den fjärde olikheten följer av att $w \in (x_k, x_i^i) \Rightarrow \|x_k - w\| < r \Rightarrow \|\nabla f(x_k) - \nabla f(w)\| \leq \frac{\xi\alpha}{2} \leq \frac{\xi}{2} \|\nabla f(x_k)\|$ eftersom $\alpha = \min_{x \in \Omega_\epsilon} \|\nabla f(x)\|$.

Så om $k \geq N$ och $\Delta_k < \delta$ så är $f(x_k^i) < f(x_k)$ för åtminstone något teststeg $s_k^i \in \Delta_k B\Gamma_k$. Punkt (ii) i Algoritm A garanterar därmed att det accepterade teststeget s_k också är enkelt avtagande och från Algoritm B vet vi därmed att $\Delta_{k+1} \geq \Delta_k$. □

För en viss konstant δ gäller alltså i varje iteration att om $\Delta_k < \delta$ så kommer man att hitta åtminstone ett teststeg s_k^i bland kolumnerna i $\Delta_k B\Gamma_k$ som är lyckat. Från Algoritm A vet vi därmed att GMA kommer att generera ett lyckat steg s_k och att $\Delta_{k+1} \geq \Delta_k$. I slutet av beviset ser vi också hur vi får olikheten $f(x_k^i) - f(x_k) \leq -\frac{\xi}{2} \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\|$ för detta teststeg s_k^i. Vi kan skriva om olikheten på följande sätt

$$f(x_k^i) - f(x_k) \leq -\frac{\xi}{2} \|\nabla f(x_k)\| \left\|x_k^i - x_k\right\| \leq -\frac{\xi}{2} \left\|\nabla f(x_k)(x_k^i - x_k)\right\| =$$

$$\frac{\xi}{2} \nabla f(x_k)(x_k^i - x_k)$$

där den sista likheten följer av att $(x_k^i - x_k)$ är en avtagande riktning vilket innebär att $\nabla f(x_k)(x_k^i - x_k) < 0$. Vi ser att detta är formen för testet av ett tillräckligt avtagande steg formulerad i (2) med $\epsilon = \xi/2$. Att $0 < \xi/2 < 1$ gäller vet från (14) och (15) där vi ser att $0 < \xi \leq |\cos\theta_k|$.

Vi vet alltså nu att det alltid finns något teststeg i $\Delta_k P_k$ som uppfyller en himla massa bra egenskaper, men vi vet fortfarande inte särskilt mycket om det teststeg som algoritmen faktiskt väljer förutom att det är just lyckat, dvs uppfyller $\rho_k > 0$. För att kunna garantera att det teststeg som väljs är just ett sådant teststeg som är tillräckligt avtagande så måste vi använda oss av villkor (1) och (2) av de tre nya villkoren på GMA.

Följdsats 6.7 säger, givet de starkare kraven på GMA, att om Δ_k är mindre än en av k oberoende konstant så kommer det accepterade teststeget s_k att vara tillräckligt avtagande.

Följdsats 6.7 . *Antag att $L(x_0)$ är kompakt och att f är kontinuerligt differentierbar runt $L(x_0)$. Antag också att de två första av de nya kraven på GMA är sanna.*

Givet $\epsilon > 0$, låt $\Omega_\epsilon = \{x \in L(x_0); dist(x, X_*) \geq \epsilon\}$. Antag också att $x_0 \in \Omega_\epsilon$. Då existerar ett $\delta > 0$ och ett $\sigma > 0$ oberoende av k, sådana att det för alla, men ändligt många k, om $x_k \in \Omega_\epsilon$ och $\Delta_k < \delta$, gäller att

$$f(x_{k+1}) \leq f(x_k) - \sigma \|\nabla f(x_k)\| \|s_k\| < f(x_k)$$

Bevis. I slutet av beviset av Proposition 6.6 såg vi att det finns åtminstone ett teststeg $s_k^i \in \Delta_k B\Gamma_k$ som uppfyller

$$f(x_k^i) - f(x_k) \leq -\frac{\xi}{2} \|\nabla f(x_k)\| \left\|s_k^i\right\| < 0$$

förutsatt att $k \geq N = \min\{k : x_k \neq x_0\}$ och $\Delta_k < \delta$. De nya kravet på Algoritm A som säger att ett accepterat teststeg s_k måste ge ett funktionsvärde som är mindre än eller lika med funktionsvärdet av den minsta av teststegen $s_k^i \in \Delta_k B\Gamma_k$ leder till att

$$f(x_{k+1}) - f(x_k) \leq -\frac{\xi}{2} \|\nabla f(x_k)\| \left\|s_k^i\right\| < 0.$$

Lemma 6.5 som garanterar existensen av en av k oberoende konstant $\zeta_* > 0$ sådan att $\left\|s_k^i\right\| \geq \zeta_* \Delta_k$ ger oss sedan

$$f(x_{k+1}) - f(x_k) \leq -\frac{\xi}{2}\zeta_* \Delta_k \left\|\nabla f(x_k)\right\| < 0$$

Vi använder nu Lemma 6.1 från det första av de tre nya villkoren på GMA. Från Lemma 6.1 vet vi att det finns en konstant $\psi_* > 0$ oberoende av k sådan att $\Delta_k \geq \psi_* \left\|s_k^i\right\|$ för alla teststeg s_k^i genererade av GMA. Särskilt gäller att $\Delta_k \geq \psi_* \left\|s_k\right\|$ för varje accepterat teststeg s_k och vi får

$$f(x_{k+1}) - f(x_k) \leq -\frac{\xi}{2}\zeta_* \psi_* \left\|\nabla f(x_k)\right\| \left\|s_k\right\| < 0$$

Sätt $\sigma = \frac{\xi}{2}\zeta_* \psi_*$. □

Vi har sett hur man med de nya kraven på GMA har kunnat överföra garantin om ett tillräckligt avtagande teststeg $s_k^i \in \Delta_k B\Gamma_k$ till att det accepterade teststeget $s_k \in P_k$ är tillräckligt avtagande. I beviset av vår sista sats används just detta faktum att GMA uppfyller Armijos regel.

Sats 6.8 *Antag att $L(x_0)$ är kompakt och att f är kontinuerligt differentierbar runt $L(x_0)$ och att vi har antagit de tre nya villkoren på GMA. Då gäller för sekvensen $\{x_k\}$ genererad av GMA att*

$$\lim_{k \to +\infty} \left\|\nabla f(x_k)\right\| = 0.$$

Bevis. Detta är ett motsägelsebevis så vi antar att $\limsup_{k \to +\infty} \left\|\nabla f(x_k)\right\| \neq 0$. Förutsatt detta vet vi att det finns ett $\epsilon > 0$ sådant att $\left\|\nabla f(x_{m_i})\right\| \geq \epsilon$ för någon delsekvens $\{x_{m_i}\}$ av $\{x_k\}$.

Låt η vara något tal sådant att $0 < \eta < \epsilon$. Eftersom

$$\lim_{k \to +\infty} \inf \nabla f(x_k) = 0,$$

så kan vi till varje punkt x_{m_i} hitta en punkt x_{l_i} som är sådan att $\left\|\nabla f(x_{l_i})\right\| < \eta$ och $\left\|\nabla f(x_k)\right\| > \eta$ för alla k sådana att $m_i \leq k < l_i$.

Vi har antagit att $\Delta_k \to 0$ så från följdsats 6.7 har vi för tillräckligt stora k att

$$f(x_k) - f(x_{k+1}) \geq \sigma \left\|\nabla f(x_k)\right\| \left\|s_k\right\| \tag{23}$$

med $\sigma > 0$. Speciellt gäller (23) för $m_i \leq k < l_i$ när i är tillräckligt stort och vi får olikheten

$$f(x_k) - f(x_{k+1}) \geq \sigma\eta \left\|s_k\right\|.$$

Summerar vi över alla k, $m_i \leq k < l_i$ så får vi den teleskopiska summan

$$(f(x_{m_i}) - f(x_{m_{i+1}})) + (f(x_{m_{i+1}}) - f(x_{m_{i+2}})) + ... + (f(x_{l_i-1}) - f(x_{l_i})) \geq$$

$$\sum_{k=m_i}^{l_i} \sigma\eta \|s_k\|$$

som vi kan skriva om

$$f(x_{m_i}) - f(x_{l_i}) \geq \sum_{k=m_i}^{l_i} \sigma\eta \|s_k\| \geq \sigma\eta \|x_{m_i} - x_{l_i}\|. \tag{24}$$

Där den sista olikheten följer av att $\|x_{m_i} - x_{l_i}\|$ är den kortaste vägen mellan x_{m_i} och x_{l_i}.

Eftersom $L(x_0)$ är kompakt så vet vi att f är nedåtbegränsad. Vi vet också att $\{f(x_k)\}$ är en strikt avtagande serie, alltså finns ett värde A sådant att $\lim_{k\to\infty} f(x_k) = A$. För delsekvenserna $\{f(x_{l_i})\}$ och $\{f(x_{m_i})\}$ gäller därmed $\lim_{i\to\infty} f(x_{m_i}) - f(x_{l_i}) = A - A = 0$. Men att $f(x_{m_i}) - f(x_{l_i}) \to 0$ då $i \to \infty$ ger tillsammans med (24) att $\|x_{m_i} - x_{l_i}\| \to 0$ när $i \to +\infty$.

Eftersom ∇f är likformigt kontinuerlig och $\|x_{m_i} - x_{l_i}\| \to 0$ så gäller för tillräckligt stora i att

$$\|\nabla f(x_{m_i}) - \nabla f(x_{l_i})\| < \eta. \tag{25}$$

Men vi har även att $\|\nabla f(x_{l_i})\| < \eta$ vilket tillsammans med (25) ger

$$\|\nabla f(x_{m_i})\| \leq \|\nabla f(x_{m_i}) - \nabla f(x_{l_i})\| + \|\nabla f(x_{l_i})\| \leq 2\eta \tag{26}$$

där den första olikheten följer av triangelolikheten.

Men (26) gäller för alla η, $0 < \eta < \epsilon$ speciellt gäller det till exempel för $\eta = \frac{\epsilon}{4}$ vilket ger $\|\nabla f(x_{m_i})\| \leq \frac{\epsilon}{2}$. Men detta är en motsägelse eftersom $\{x_{m_i}\}$ per definition var sådan att $\|\nabla f(x_{m_i})\| \geq \epsilon$. □

6.3 Bevis av Proposition 5.3

Vi väntade med att visa Proposition 5.3 eftersom vi i beviset behöver använda resultat som kommit fram i beviset av 6.8. Notera dock att vi aldrig använder några av de nya kraven på GMA.

Bevis. Eftersom $\|\liminf_{k\to+\infty} \nabla f(x_k)\| \neq 0$ så vet vi att det finns ett $\epsilon > 0$ sådant att det för alla k gäller att $x_k \in \Omega_\epsilon = \{x \in L(x_0) : dist(x, X_*) \geq \epsilon\}$. Proposition 6.6 garanterar därmed existensen av ett tal $\delta > 0$ sådant att en iteration k alltid blir lyckad om $\Delta_k \leq \delta$ och därmed blir $\Delta_{k+1} \geq \Delta_k$.

Givet en första steglängdsparameter $\Delta_0 > 0$ går det att hitta ett tal $q \in \mathbf{Z}$, $q > 0$ så att $\Delta_0\theta^q \leq \delta$, där $0 < \theta < 1$ och θ är som i Algoritm B. För alla k gäller alltså att Δ_k som minst kan ha storleken $\Delta_0\theta^q \leq \delta$. Låter vi $\Delta_{LB} = \Delta_0\theta^{q+1} \leq \delta$ så har vi vår undre gräns och satsen är visad. □

7 I praktiken

I beviset av mönstersöknings metodernas globala konvergens var vi tvungna att göra två ganska begränsande antaganden om f. Dels var vi tvungna att anta att nivåmängden $L(x_0)$ skulle vara en kompakt mängd och dels att f skulle vara kontinuerligt differentierbar i en omgivning av $L(x_0)$. Men hur går dessa begränsande antaganden ihop med vårt inledande påstående att mönstersökningsmetoderna är robusta och särskilt användbara på komplexa problem där de gradientbaserade metoderna inte duger. Svaret på frågan är enkelt. I praktiken kan man ofta strunta både i villkoret att $L(x_0)$ ska vara kompakt och att f ska vara kontinuerligt differentierbar eftersom mönstersökningsmetoderna oftast fungerar bra ändå. Bra är ett relativt begrepp, men mönstersökningsmetoderna fungerar "bra ändå" i flera avseenden. Vi ska titta på de möjliga fallgroparna som kan uppstå då de två villkoren inte är uppfyllda och i vilken mån mönstersökningsmetoderna fungerar "bra ändå".

7.1 Om f är diskontinuelig.

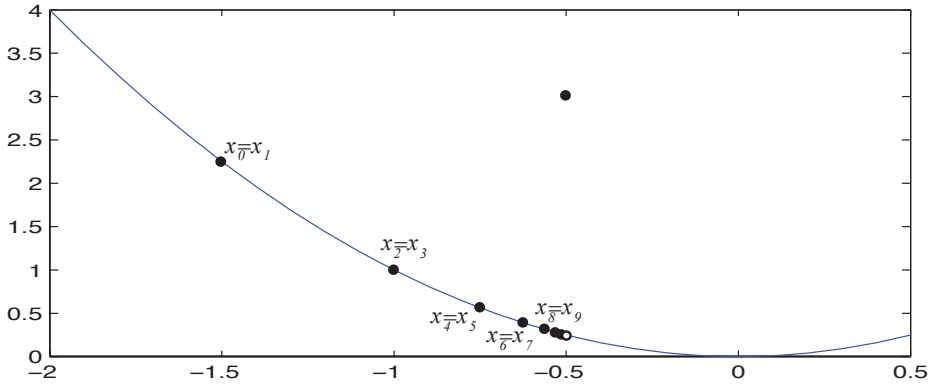
Ett praktiskt exempel på när mönstersökningsmetoderna används på diskontinuerliga funktioner är inom den simuleringsbaserade optimeringen. För mer information om detta hänvisar jag till [2].

Eftersom en mönstersökningsmetod aldrig beräknar gradienten så kan den användas på en diskontinuerlig funktion i den mening att den alltid kommer att vara definierad, även i punkter av diskontinuitet. Detta kan förstås vara en fördel. I alla fall om man nöjer sig med en förbättring eftersom det inte finns någon garanti på att man kommer att konvergera mot en stationär punkt. Vi ska se ett exempel på hur diskontinuitet hos målfunktionen kan få cykliska koordinatmetoden att konvergera mot en ickestationär punkt.

Exempel 7.1 *Bilden illustrerar cykliska koordinatmetoden i en dimension med* $x_0 = -1.5$, $\theta = 1/2$, $\Delta_0 = 1$ *och*

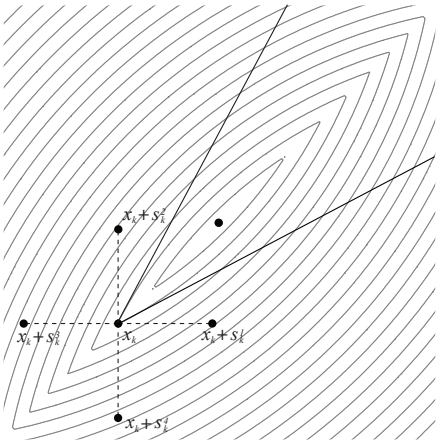
$$f(x) = \begin{cases} x^2 & \text{om } x \neq -0.5, \\ 3 & \text{om } x = -0.5. \end{cases}$$

Exemplet är en typisk situation där sannolikheten är ganska liten att man ska välja x_0, Δ_0 och θ just på ett sådant sätt att man lyckas fastna i fel punkt och där det alltså kan vara bra att använda en mönstersökningsmetod trots diskontinuiteten hos f. Men det finns förstås mer komplexa situationer än denna där sannolikheten att hamna fel är betydligt större och då är det mer fråga om vad det är för resultat man vill uppnå. En förbättring eller det bästa.



7.2 Om ∇f är diskontinuelig.

I beviset så använder vi antagandet om att ∇f existerar och är kontinuerlig för att kunna garantera att någon av de n sökriktningarna är en avtagande riktning och vi har redan pratat lite om problemet som kan uppstå med en diskontinuerlig gradient när vi introducerade begreppet mönstersökning i avsnitt 2.1. Vi sa att det som kan hända är att man kan hamna i en så kallad ”vass ränna” där om man har otur ingen av de n riktningarna är avtagande. Även i ett sådant fall är en mönstersökningsmetod väldefinierad, även om det inte är säkert att man konvergerar mot någon stationär punkt. Men, åter igen. Om man nöjer sig med ett bättre resultat så är en mönstersökningsmetod ett bra verktyg även i detta fall.



Figur 15: Ingen av de fyra sökriktningarna är avtagande.

Med några justeringar kan nämligen konvergensbeviset för mönstersökning modifieras så att det gäller även för icke kontinuerligt differentierbara funk-

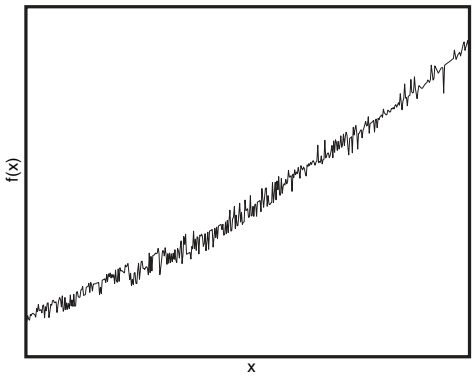
tioner. Det man då visar är att en algoritm konvergerar antingen mot en stationär punkt, mot en punkt där gradienten är diskontinuerlig eller mot en punkt där gradienten inte existerar.

Figur 15 visar nivåkurvorna till en variant av Dennis-Woods funktion och hur man kan fastna i en punkt där gradienten inte existerar. Exemplet är hämtat ur [2].

Eftersom mönstersökningsmetoderna alltid söker längs alla n riktningar i varje iteration (som minst), så har de större chans att ta sig ur en "vass ränna" än tex. en Simplex-metod som bara söker längs en riktning i varje iteration.

7.3 Brus.

Det första av våra antaganden var att $L(x_0)$ skulle vara kompakt och vi behövde det antagandet för att kunna kontrollera steglängdsparameterns beteende då k gick mot oändligheten. I praktiken är det dock vanligast att man använder mönstersökningsmetoder just på funktioner med väldigt många lokala minimum och där det alltså inte alls är säkert att $L(x_0)$ är kompakt. Anledningen är att mönstersökningsmetoderna är okänsliga för *brus*. Med brus menas ungefär små variationer som kan vara störande vid mätningar. Det kan tex i vårat fall vara högfrekventa förändringar med liten amplitud hos en funktion, som bilden visar.



Figur 16: En funktion med numeriskt brus. [2]

Att sätta en gradientbaserad metod på att hitta minimum till en funktion som den i bild 16 hade varit dömt att misslyckas eftersom gradientbaserade metoder bara söker i avtagande riktning. En sådan metod hade alltså fastnat i första bästa lokala minimipunkt som i princip skulle kunna vara hur långt från det bästa värdet som hällts.

En mönstersökningsmetod däremot, hade med tillräckligt stor steglängdsparameter, varit helt oberörd av alla de lokala minimipunkterna. Det är mönstersökningsmetodens egenskap av att "stickprova" sig fram som gör att den är så överlägsen vid den här typen av problem. Man kan säga att den bildar sig en mindre brusig approximativ bild av målfunktionen genom att bara titta på den punktvis med ganska stort mellanrum. Varför antar man då att $L(x_0)$ är kompakt? Svaret är förstås att det inte finns någon garanti på att man lyckas hitta den optimala lösningen om $L(x_0)$ inte är kompakt. Om teststegen är mindre än våglängden hos de små förändringarna så beter sig en mönstersökningsmetod precis lika dåligt som en derivatorbaserad metod hade gjort, dvs den kan fastna i vilket "dåligt" lokalt minimum som helst.

Referenser

[1] Bazaraa, M.S., Sherali, H.D., Shetty, C.M.: Nonlinear Programming, Theory and Algorithms, Whiley-Interscience, 3, 2006.

[2] Kolda, T. G, Lewis, R. M, Torczon, V, Optimization by Direct Search: New Perspectives on Some Classical and Modern Merhods, SIAM Review. 45, (2003), pp. 385–482.

[3] Lewis, R.M., Torczon, V, Trosset, M.W., Direct search methods: then and now, Comput. J. 124 , (2000), pp. 191–207.

[4] Torczon, V, Multi.Directional Search: A direct search algorithm for parallel machines, Ph.D. Theisis, Department of Mathematical Sciences, Rice University, Huston, Texas, (1989), pp. 1–18.

[5] Torczon, V, On the convergence of pattern search algorithms, SIAM J. Optim. 7, (1997), pp. 1–25.

[6] Torczon, V, On the convergence of the multidirectional search algorithm, Department of Mathematical Sciences, Rice University, Huston, Texas, (1990), pp. 1–5.

[7] Luenberger, D.G.: Linear and Nonlinear Programming, Addison-Wesley, 2, 1973.

[8] RAO, S.S.: Optimization theory and applications, Wiley Eastern Limited, 2, 1978.