6.3. Direkta sökmetoder

Förutom de nyss nämnda metoderna för att uppsöka ett minimum av en funktion av en variabel finns det en enkel metod som baserar sig på polynomapproximation av funktionen.

Om vi vet att funktionen $f(x)$ är unimodal inom ett givet intervall, så kan vi söka upp ett intervall som innehåller minimet på följande sätt (kallas bracketing): Vi beräknar först funktionsvärdet $f_0 = f(x_0)$, och ökar därpå x_0 med h till $x_1 = x_0 + h$. Om funktionsvärdet $f_1 = f(x_1)$ i punkten x_1 är större än f_0, så kastas sökriktningen om, och funktionens värde beräknas istället i punkten $x_1 = x_0 - h$. Skulle det nya funktionsvärdet $f_1 = f(x_1)$ vara större än f_0, så har minimet passerats, och detta ingår alltså i intervallet $[x_0, x_1]$.

Om $f_1 < f_0$ upprepas proceduren genom upprepad fördubbling av steglängden h: $x_k = x_{k-1} + 2^{k-1}h$, $k = 1, 2, \ldots$ ända tills det nya funktionsvärdet f_k är större än det närmast föregående f_{k-1}. Minimet kommer då alltså att ingå i intervallet $[x_{k-1}, x_k]$.

När vi på detta sätt avgränsat minimet med hjälp av tre punkter, som vi betecknar x_0, x_1, x_2, så kan vi interpolera funktionen i dessa punkter med ett kvadratiskt polynom. Om vi uttrycker polynomet i formen $p_2(x) = a_0 + a_1 x + a_2 x^2$ och sätter $f(x_0) = p_2(x_0)$, $f(x_1) = p_2(x_1)$ och $f(x_2) = p_2(x_2)$, så kan vi beräkna polynomets koefficienter.

Minimet för detta polynom, som är en approximation för funktionsminimet, fås ur villkoret $p_2'(x) = 0$ (x är alltså stationär), dvs

$$\tilde{x} = -\frac{a_1}{2a_2} = \frac{(x_1 + x_0)f[x_0, x_2] - (x_2 + x_0)f[x_0, x_1]}{2(f[x_0, x_2] - f[x_0, x_1])} = \frac{1}{2}\left(x_0 + x_2 + \frac{x_1 - x_2}{1 - \frac{f[x_0,x_1]}{f[x_0,x_2]}} \right)$$

Här har vi använt beteckningen

$$f[x_0, x_i] = \frac{f(x_i) - f(x_0)}{x_i - x_0} = a_1 + a_2(x_i + x_0)$$

Genom att jämföra funktionsvärdena i punkterna $x_0, x_1, x_2, \tilde{x}$, kan vi välja ut tre nya punkter, som begränsar minimet. Det är lätt skriva ett enkelt MATLAB–program, som approximerar en given funktion med ett kvadratiskt polynom i tre punkter, och konstruerar ett intervall, som begränsar minimet med önskad tolerans:

```
function xmin = kvadmin(funk, x0,x1,x2,tol)
% Minimimering av funk med kvadratisk interpolation
% Begynnelsepunkter : x0 < x1 < x2, f(x1) < f(x0),f(x2)
% tol : toleransen
f0 = feval(funk,x0); f1 = feval(funk,x1); f2 = feval(funk,x2);
while x2-x0 > tol
    f01 = (f1-f0)/(x1-x0); f02 = (f2-f0)/(x2-x0);
    xm = (x0+x2 + (x1-x2)/(1-f01/f02))/2; fm = feval(funk,xm);
    if xm < x1      % nytt intervall
       if fm <= f1
          x2 = x1 ; x1 = xm; f2 = f1; f1 = fm;
       else
          x0 = xm ; f0 = fm;
       end
    else
       if fm <= f1
          x0 = x1 ; x1 = xm; f0 = f1; f1 = fm;
       else
          x2 = xm ; f2 = fm;
       end
    end
end
xmin=x1;
```

Vi kan tillämpa programmet på funktionen $x + 1/x^2$, som har ett minimum i punkten $x_{\min} = \sqrt[3]{2} \approx 1.25992104989$. Resultatet av beräkningen visas nedan:

```
>> xmin=kvadmin('tstkvad',0.5,1,1.5,1e-4)
 1  xm=1.261363636363636  x0= 1.0000000000 x1= 1.2613636364 x2= 1.5000000000
 2  xm=1.292738511488512  x0= 1.0000000000 x1= 1.2613636364 x2= 1.2927385115
 3  xm=1.264529583697077  x0= 1.0000000000 x1= 1.2613636364 x2= 1.2645295837
 4  xm=1.260734401122526  x0= 1.0000000000 x1= 1.2607344011 x2= 1.2613636364
 5  xm=1.260224716848444  x0= 1.0000000000 x1= 1.2602247168 x2= 1.2607344011
 6  xm=1.260071528628276  x0= 1.0000000000 x1= 1.2600715286 x2= 1.2602247168
 7  xm=1.259982248812828  x0= 1.0000000000 x1= 1.2599822488 x2= 1.2600715286
 8  xm=1.259949579035346  x0= 1.0000000000 x1= 1.2599495790 x2= 1.2599822488
 9  xm=1.259933143863138  x0= 1.0000000000 x1= 1.2599331439 x2= 1.2599495790
10  xm=1.259926525426100  x0= 1.0000000000 x1= 1.2599265254 x2= 1.2599331439
11  xm=1.259923418091240  x0= 1.0000000000 x1= 1.2599234181 x2= 1.2599265254
12  xm=1.259922107156190  x0= 1.0000000000 x1= 1.2599221072 x2= 1.2599234181
13  xm=1.259921511628524  x0= 1.0000000000 x1= 1.2599215116 x2= 1.2599221072
14  xm=1.259921254665266  x0= 1.0000000000 x1= 1.2599212547 x2= 1.2599215116
15  xm=1.259921139538329  x0= 1.0000000000 x1= 1.2599211395 x2= 1.2599212547
16  xm=1.259921090038450  x0= 1.0000000000 x1= 1.2599210900 x2= 1.2599211395
17  xm=1.259921066570770  x0= 1.0000000000 x1= 1.2599210666 x2= 1.2599210900
18  xm=1.259921057959724  x0= 1.0000000000 x1= 1.2599210580 x2= 1.2599210666
19  xm=1.259921046449707  x0= 1.0000000000 x1= 1.2599210464 x2= 1.2599210580
20  xm=1.259921051815368  x0= 1.2599210464 x1= 1.2599210518 x2= 1.2599210580
```

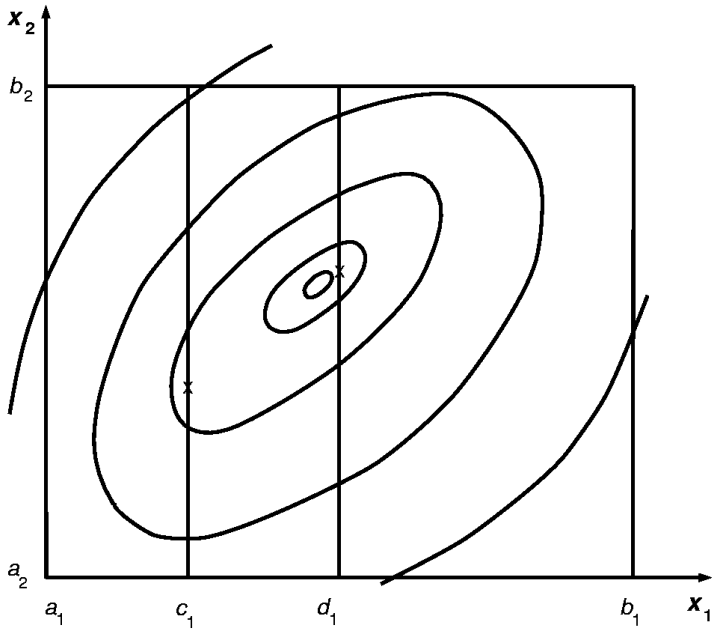
```
xmin = 1.25992105181537
```

Som vi ser, konvergerar programmet efter 20 iterationer, men noggrannheten är faktiskt betydligt större än fyra decimaler. Orsaken är att den vänstra ändpunkten av intervallet är konstant ($= 1$) under 19 iterationer, medan de två andra punkterna alltmer närmar sig varandra, och övergår i det rätta minimet. Konvergensen skulle kunna förbättras genom att man ersätter x_0 med en punkt som ligger närmare x_1 då t.ex. $x_1 - x_0 > 10(x_2 - x_1)$.

En annan möjlighet är att interpolera med ett polynom av högre gradtal. Men ofta används den enkla kvadratiska interpolationen i kombination med någon annan metod, som t.ex. gyllene snitt–metoden, som vi beskrev i det förra avsnittet. Denna strategi används t.ex. i MATLAB–funktionen `fminbnd`.

Metoderna för att söka ett minimum av en funktion av en variabel kan lätt generaliseras till funktioner av flere variabler. Låt oss t.ex. anta att vi önskar mimimera en funktion av två variabler, x_1 och x_2 för $a_1 \leq x_1 \leq b_1$ samt $a_2 \leq x_2 \leq b_2$. Vi väljer därpå två punkter $x_1 = c_1$ och $x_1 = d_1$ samt beräknar minimerna av de två endimensionella restriktionerna av f:

$$\min_{a_2 \leq x_2 \leq b_2} f(c_1, x_2) \text{ samt } \min_{a_2 \leq x_2 \leq b_2} f(d_1, x_2).$$



Dessa två funktionsvärden kan associeras med $x_1 = c_1$ och $x_1 = d_1$ (se ovanstående bild). Av figuren framgår, att vi kan begränsa området där minimet skall sökas, eftersom funktionsvärdet, som associeras med $x_1 = c_1$ är större än det som hör ihop med $x_1 = d_1$. Vi kan alltså upprepa proceduren på intervallet (c_1, b_1). Metoden kan lätt generaliseras till högre dimensioner, men är inte särskilt praktisk, utom då funktionen har väldefinierade minimer.

En mera praktisk metod är den s.k. alternerande variabelmetoden, där man minimerar med avseende på varje variabel i *tur och ordning*. Sökproceduren fortskrider alltså parallellt med varje koordinataxel i tur och ordning, och är mycket effektiv för funktioner, som har elliptiska konturer med axlar, som är parallella med koordinataxlarna. I allmänhet påverkar dock variablerna varandra (dvs de är korrelerade), så att funktionskonturerna är sneda i förhållande till axlarna. I detta fall måste proceduren upprepas, och den är inte särskilt effektiv, isynnerhet då funktionen beror av många variabler.

En av de effektivaste metoderna för direkt uppsökning av minimer av funktioner av flere variabler som man för närvarande känner, är simplexmetoden. Ett reguljärt simplex i n dimensioner är $n+1$ punkter på lika avstånd ifrån varandra, dvs för $n=2$ har vi en liksidig triangel, för $n=3$ en reguljär tetraeder etc. Metoden går ut på att man ställer upp ett reguljärt simplex i den rymd som spännes av de oberoende variablerna, och beräknar funktionsvärdena i varje vertex. Om $x_0^{(1)}$ är en approximation till minimet, så kan man konstruera ett simplex med kantlängden 1 genom att välja punkterna $x_0^{(1)}$,$x_1^{(1)}$, $\ldots$, $x_n^{(1)}$ på följande sätt:

$$
\begin{aligned}
x_0^{(1)} =& (x_{01}^{(1)}, x_{02}^{(1)}, \ldots, x_{0n}^{(1)}) \\
& \ldots \\
x_i^{(1)} =& (x_{01}^{(1)} + \delta_1, x_{02}^{(1)} + \delta_1, \ldots, x_{0,i-1}^{(1)} + \delta_1, x_{0i}^{(1)} + \delta_2, \\
& \quad x_{0,i+1}^{(1)} + \delta_1, \ldots, x_{0n}^{(1)} + \delta_1) \qquad (i = 1, 2, \ldots, n),
\end{aligned}
$$

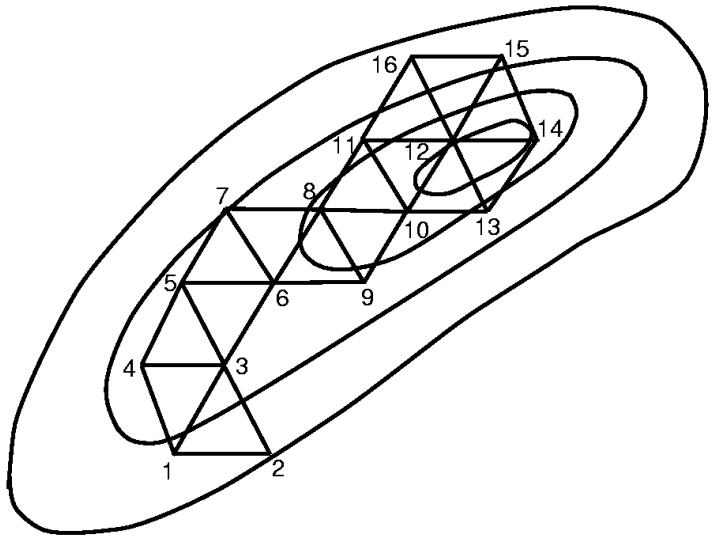
där $\delta_1 = (\sqrt{n+1} - 1)/(n\sqrt{2})$ och $\delta_2 = (\sqrt{n+1} + n - 1)/(n\sqrt{2})$.

Den k:te iterationen i denna procedur innebär helt enkelt, att en av hörnpunkterna i simplexet, t.ex. $x_j^{(k)}$, ersätts av sin spegelbild med avseende på de återstående n hörnpunkternas massmedelpunkt $\bar{x}$, dvs.

$$x_j^{(k+1)} = 2(\bar{x} - x_j^{(k)}) + x_j^{(k)} = \frac{2}{n}(x_0^{(k)} + x_1^{(k)} + \ldots + x_{j-1}^{(k)} + x_{j+1}^{(k)} + \ldots x_n) - x_j^{(k)}.$$

Den förkastade punkten väljs vanligen så, att den svarar mot det största funktionsvärdet. Man inser ganska lätt, att en reflektion av punkten $x_j^{(k)}$ kommer att alstra en punkt $x_j^{(k+1)}$ som sammanfaller med $x_j^{(k-1)}$, ifall $x_j^{(k-1)}$ skulle ge det högsta funktionsvärdet vid den $(k-1)$:a iterationen. Processen skulle då stagnera, och endast oskillera mellan de två senaste simplexen. För att undvika detta speglar man istället den punkt, som gav upphov till det *nästhögsta* funktionsvärdet.

Proceduren åskådliggörs i nedanstående figur, där sifforna anger den ordning, i vilken hörnpunkterna konstruerades.



Då en av hörnpunkterna i ett simplex befinner sig i närheten av det verkliga minimet, kommer denna hörnpunkt alltid att ingå i ett av de simplex, som alstras i fortsättningen, och simplexen kommer att *rotera* kring denna punkt. Då detta inträffar, brukar man *halvera* simplexens kantlängder, bevara den gemensamma hörnpunkten, och fortsätta processen ända till dess att kantlängden efter ett tillräckligt antal rotationer och halveringar underskrider en given gräns.

En modifikation av simplexmetoden, där man inte behöver använda reguljära simplex, infördes av *Nelder* och *Mead* år 1965[1].

Låt oss anta, att hörnpunkterna i ett simplex vid den k:te iterationen är $x_0^{(k)}, x_1^{(k)}, \ldots, x_n^{(k)}$, och att de motsvarande funktionsvärdena $f_0, f_1, \ldots, f_n$ är ordnade så, att $f_n > f_{n-1} > \ldots > f_1 > f_0$. Funktionen har alltså sitt bästa värde i punkten $x_0^{(k)}$ och det sämsta värdet i punkten $x_n^{(k)}$. Låt oss beteckna massmedelpunkten av de $n-1$ hörnpunkterna $c_i^{(k)} = \frac{1}{n}\sum_{j=0}^{n-1} x_j^{(k)}$. Liksom i den ursprungliga simplex–metoden försöker man först ersätta den sämsta hörnpunkten $x_n^{(k)}$ med sin spegelbild $x_r^{(k)} = c^{(k)} + \alpha(c^{(k)} - x_n^{(k)})$, där α är en reflektionskoefficient (vanligen 1).

Vi har nu tre möjligheter: $x_r^{(k)}$ kan vara en sådan punkt att $f_0 < f_r < f_{n-1}$, i vilket fall vi ersätter $x_n^{(k)}$ med den nya punkten $x_r^{(k)}$. Eller så är $f_r < f_0$, och då är $x_r^{(k)}$ en ny bästa punkt. Slutligen kan $f_r > f_{n-1}$ gälla, varvid $x_r^{(k)}$ är en ny sämsta punkt.

Om reflektionen leder till en ny bästa punkt, så kan reflektionsriktningen vara gynnsam, och vi kan definiera en ny punkt $x_e^{(k)} = c^{(k)} + \beta(x_r^{(k)} - c^{(k)})$, där β är en utvidgningskoefficient (t.ex. 2). Om $f_e < f_0$ så har utvidgningen lyckats, och $x_n^{(k)}$ ersätts med $x_e^{(k)}$, i annat fall har utvidgningen misslyckats, och $x_n^{(k)}$ ersätts med $x_r^{(k)}$.

[1] J.A. Nelder och R. Mead: *A simplex method for function minimization*, Comput. J. **7**, 308-313 (1965)

Om reflektionen leder till en ny sämsta punkt, så försöker man sammandra simplexet genom att definiera en ny punkt $x_c^{(k)}$ med ekvationerna

$$\begin{cases} x_c^{(k)} = c^{(k)} + \gamma(x_n^{(k)} - c^{(k)}) & \text{om } f_n < f_r \\ x_c^{(k)} = c^{(k)} + \gamma(x_r^{(k)} - c^{(k)}) & \text{om } f_n > f_r, \end{cases}$$

där γ ($0 < \gamma < 1$, t.ex. 0.5) är en kontraktionskoefficient. Om $f_c < \min(f_n, f_r)$ så har sammandragningen lyckats och $x_n^{(k)}$ ersätts med $x_c^{(k)}$, i annat fall halveras avstånden till den bästa punkten $x_0^{(k)}$ av alla de övriga hörnpunkterna i simplexet. Simplexmetoden anses ha konvergerat, när standardavvikelsen av funktionsvärdena i hörnpunkterna understiger ett givet tröskelvärde. MATLAB–funktionen `fminsearch` utnyttjar simplex–metoden för att bestämma ett minimum av en funktion av flera variabler.

6.4. Gradientmetoder

Gradientmetoderna bygger på funktionens Taylor–utveckling:

$$f(x + \Delta x) \approx f(x) + g^T \Delta x + \tfrac{1}{2} \Delta x^T H \Delta x.$$

Minimeringsmetoder, som endast använder gradientvektorn g för att beräkna korrektioner till $f(x)$, kallas metoder av första ordningen, medan metoder som även använder andra derivatorna kallas metoder av andra ordningen.

I allmänhet är gradientmetoderna effektivare än de direkta metoderna, när det gäller att minimera en analytisk funktion, vars derivator lätt kan beräknas. Det finns dock fall, då derivatorna inte kan beräknas lätt eller t.o.m. är diskontinuerliga, i vilket fall gradientmetoderna inte är så användbara.

I den enklaste metoden av första ordningen, som kallas steepest descent–metoden (som användes första gången av Cauchy år 1847 för att lösa ett olinjärt ekvationssystem[2]), utnyttjas gradientvektorn för att beräkna den riktning, i vilken funktionen avtar snabbast. Härvid uttrycks funktionsändringen Δf i formen:

$$\Delta f = f(x + \Delta x) - f(x) \approx g^T \Delta x = \sum_i \frac{\partial f(x)}{\partial x_i} \Delta x_i,$$

då parametervektorn x antas få ett litet tillskott Δx.

På grund av att uttrycket $g^T \Delta x$ kan uppfattas som skalärprodukten av gradientvektorn g och tillskottsvektorn Δx, så kan vi också skriva Δf i formen

$$\Delta f \approx \|g\| \cdot \|\Delta x\| \cdot \cos(g, \Delta x).$$

Funktionsändringen Δf beror alltså av vinkeln mellan gradientvektorn och tillskottsvektorn och varierar mellan gränserna $+\|g\| \cdot \|\Delta x\|$ och $-\|g\| \cdot \|\Delta x\|$, då vinkeln varierar mellan 0 och π. Funktionen f avtar alltså *mest*, om vinkeln mellan g och Δx är π, dvs funktionen f *avtar snabbast i riktningen* $-g$.

[2] Augustin L. Cauchy: *Méthode générale pour la résolution des systèmes d'équations simultanées*, C.R. Acad. Sci. Paris **25**, 536-538 (1847)

Eftersom en enhetsvektor u i $-g$:s riktning kan skrivas $u = -g/\|g\|$, så kan parametertillskottet Δx väljas proportionellt mot u:

$$\Delta x = \lambda u,$$

där λ är en parameter som bestäms med en linjär sökmetod.

Ofta följer proceduren en sicksack–kurva, som småningom närmar sig minimet ($x_{\min}$). Konvergensen är (tyvärr) ofta långsam, beroende på att riktningen av den negativa gradienten $-g$ kan vara nästan vinkelrät mot riktningen till funktionens minimum.

Det bästa värdet av λ kan bestämmas genom att man minimerar $f(x + \lambda u)$ i avseende på λ. Emedan den riktning, som anges av u i allmänhet inte direkt leder till ett lokalt minimum, så måste proceduren upprepas flera gånger.

I sådana fall att funktionen har elliptiska konturer, löper kurvan mellan två räta linjer, som skär varandra i minimet. Det finns en minimeringsmetod (partanmetoden) som utnyttjar denna egenskap. Vi kan dessutom se, att successiva riktningar vid minimeringsprocessen alltid står vinkelrätt mot varandra, emedan $\frac{\partial f(x+\lambda u)}{\partial \lambda} = g(x + \lambda u) \cdot u = 0$ gäller för ett optimalt värde av λ.

I närheten av ett minimum är processen långsam, emedan funktionsändringarna avtar och flere iterationer behövs. För att avgöra när det är bäst att avbryta processen, brukar man använda ett konvergenstest som bygger på detta faktum. Man kan t.ex. välja en faktor $K,\ 0 < K < 1$, och avbryta minimeringen, då $\|\Delta x_k\| = |\lambda_k|\|u_k\| \leq |K\lambda_0|\|u_0\|$, dvs $\lambda_k \leq K\lambda_0$, eller alltså en bråkdel av det första steget.

Gradientmetoden kan förbättras, om man medtar andra ordningens termer i funktionens Taylor–utveckling kring en approximation till minimet:

$$f(x_{\min}) \approx f(x) + g^T \Delta x + \tfrac{1}{2}\Delta x^T H \Delta x,$$

där $x_{\min} = x + \Delta x$ och gradientvektorn g samt Hesses matris H beräknas i punkten x.

I koordinatform kan denna ekvation utryckas

$$f(x_{\min}) \approx f(x) + \sum_{i=1}^{n} \frac{\partial f(x)}{\partial x_i}\Delta x_i + \tfrac{1}{2}\sum_{i=1}^{n}\sum_{j=1}^{n} \frac{\partial^2 f(x)}{\partial x_i \partial x_j}\Delta x_i \Delta x_j.$$

För att beräkna Δx, antar vi att g och H är konstanta, och beräknar gradienten av $f(x_{\min})$ (dvs deriverar partiellt) i avseende på Δx_i ur ovanstående formel. Vi finner då Δx ur ekvationssystemet

$$\frac{\partial f(x)}{\partial x_i} + \sum_{j=1}^{n} \frac{\partial^2 f(x)}{\partial x_i \partial x_j} \Delta x_j = 0 \ (i = 1, 2, \ldots, n),$$

som i matrisform kan skrivas

$$g = -H\Delta x.$$

Om Hesses matris är positivt definit, så erhålls korrektionen till minimet således ur ekvationen

$$\Delta x = -H^{-1} g.$$

Denna ekvation, som är exakt för en kvadratisk funktion, är endast approximativt giltig för en godtycklig olinjär funktion. Den kan användas för att bestämma successiva approximationer till ett minimum och brukar då kallas för Newton-Raphsons metod, ett namn som är bekant från tidigare. Eftersom H inte beräknas exakt i minimet, behöver den inte vara positivt definit. Denna procedur behöver därför inte heller konvergera på stora avstånd från minimet.

Ett sätt att tolka korrektionen $\Delta x = -H^{-1}g$ är, att den ger upphov till ett nytt system av kroklinjiga koordinater i parameterrummet, i avseende på vilka funktionens nivåkurvor är *sfäriska*. Gradienten kommer då alltid att peka i krökningsradiens riktning, och den negativa gradienten är riktad mot minimet. Denna tolkning infördes av *Davidon* år 1959 (den variabla metrikmetoden[3]).

I Davidons metod erhålls parametervektorn $x^{(k+1)}$ medels formeln

$$x^{(k+1)} = x^{(k)} - \lambda^{(k)} H^{(k)} g^{(k)},$$

där $H^{(k)}$ är en positiv definit matris, som korrigerar rummets metrik och $\lambda^{(k)}$ är ett positivt tal, vars värde bestäms genom att minimera funktionen

$$\bar{f}(\lambda) \equiv f\left(x^{(k)} - \lambda H^{(k)} g^{(k)}\right).$$

Vanligen utgår man ifrån $H^{(1)} = I$ ("steepest descent"), men man kan också utgå från någon annan positivt definit matris. Då funktionsvärdena närmar sig minimet bör $H^{(k)}$ närma sig H^{-1}, så att metoden gradvis övergår från en steepest descent--metod till en Newton-Raphson–metod (därav kommer även benämningen kvasi–Newton för denna typ av metoder).

[3] W.C. Davidon: *Variable metric method for minimization*, AEC Research and Development Report, ANL-5990 (1959)

Transformationsmatrisen $H^{(k)}$ beräknas numera enligt en ny metod, som infördes av *R. Fletcher* och *M.J.D. Powell* år 1963, varför den modifierade metoden också brukar kallas Davidon–Fletcher–Powells metod (DFP).

Enligt denna metod modifieras $H^{(k)}$ med hjälp av formeln

$$H^{(k+1)} = H^{(k)} - \frac{H^{(k)} \Delta g^{(k)} \Delta g^{(k)T} H^{(k)}}{\Delta g^{(k)T} H^{(k)} \Delta g^{(k)}} + \frac{\Delta x^{(k)} \Delta x^{(k)T}}{\Delta x^{(k)T} \Delta g^{(k)}},$$

där $\Delta g^{(k)} = g^{(k+1)} - g^{(k)}$ och $\Delta x^{(k)} = x^{(k+1)} - x^{(k)}$. Genom substitution ser vi att härav följer

$$\Delta x^{(k)} = H^{(k+1)} \Delta g^{(k)}.$$

Rekursionsformeln brukar betecknas som en rang-två–formel, emedan korrektionerna till $H^{(k)}$ är av andra ordningen. Formeln visar, att om $H^{(k)}$ är positivt definit, så är även $H^{(k+1)}$ positivt definit, varför denna egenskap kommer att bevaras. Härav följer, att parameterändringens riktning *aldrig* kan bli ortogonal mot gradientens riktning, dvs. $-g^{(k)T} H^{(k)} g^{(k)} \neq 0$. Detta betyder, att nämnaren i uttrycket för $H^{(k+1)}$ aldrig kan försvinna, och funktionens värde kommer därför (i princip) alltid att minska vid varje iteration, dvs. proceduren är *stabil*.