

Linjärisering, Jacobimatris och Newtons metod

1 Inledning

Vi skall lösa system av icke-linjära ekvationer. Som exempel kan vi ta,

$$\begin{cases} x_1(1 + x_2^2) - 1 = 0 \\ x_2(1 + x_1^2) - 2 = 0 \end{cases}$$

som är ett system av två ekvationer i två obekanta (Exempel 1, Adams 13.6). Om vi inför de två funktionerna

$$\begin{aligned} f_1(x_1, x_2) &= x_1(1 + x_2^2) - 1, \\ f_2(x_1, x_2) &= x_2(1 + x_1^2) - 2, \end{aligned}$$

kan ekvationssystemet skrivas

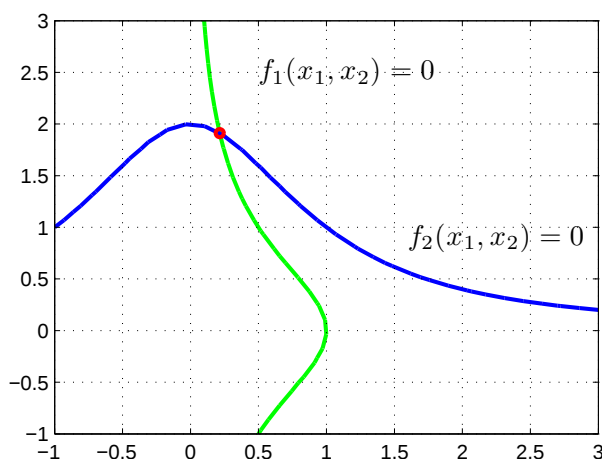
$$\begin{cases} f_1(x_1, x_2) = 0 \\ f_2(x_1, x_2) = 0 \end{cases}$$

Med $\mathbf{f} = (f_1, f_2)$, $\mathbf{x} = (x_1, x_2)$, och $\mathbf{0} = (0, 0)$, ser vi att $\mathbf{f}: \mathbb{R}^2 \rightarrow \mathbb{R}^2$ och vi kan skriva ekvationerna på den kompakta formen

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}.$$

I denna studioövning skall vi använda Newtons metod för att lösa sådana ekvationer och vi skall använda våra kunskaper från tidigare studioövning för att göra grafiska bilder av noll-nivåkurvor.

Vi börjar med att rita upp noll-nivåkurvorna till f_1 respektive f_2 .



Vi ser lösningen till $\mathbf{f}(\mathbf{x}) = \mathbf{0}$ som den punkt där noll-nivåkurvorna till f_1 och f_2 skär varandra.

Vi kan grafiskt läsa av en första approximation av lösningen för att sedan förbättra denna med Newtons metod.

Innan vi kan komma till Newtons metod måste vi dock först se på linjäriseringar av funktioner i flera variabler.

2 Linjärisering och Jacobimatrix

Linjäriseringen av en differentierbar (deriverbar) funktion i en variabel $f: \mathbb{R} \rightarrow \mathbb{R}$ runt punkten a ges av (Definition 8, Adams 4.9)

$$L(x) = f(a) + f'(a)(x - a).$$

Nära punkten a har vi $f(x) \approx L(x)$ och vi påminner oss om att den rätta linjen $y = L(x)$ är tangenten till kurvan $y = f(x)$ vid a .

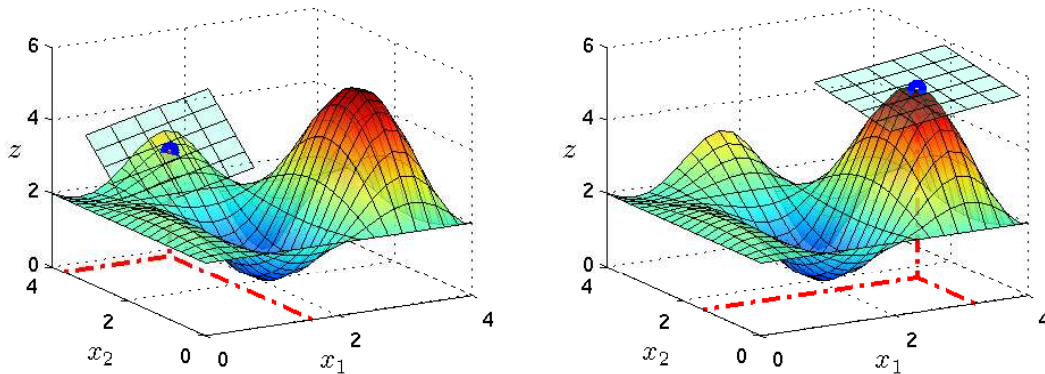
En funktion i två variabler, $f: \mathbb{R}^2 \rightarrow \mathbb{R}$

Låt $f(x_1, x_2)$ vara en differentierbar funktion i två variabler, $f: \mathbb{R}^2 \rightarrow \mathbb{R}$. Linjäriseringen av f runt punkten (a_1, a_2) ges av (Adams 12.6)

$$L(x_1, x_2) = f(a_1, a_2) + \frac{\partial f}{\partial x_1}(a_1, a_2)(x_1 - a_1) + \frac{\partial f}{\partial x_2}(a_1, a_2)(x_2 - a_2)$$

och nära punkten (a_1, a_2) har vi $f(x_1, x_2) \approx L(x_1, x_2)$.

Det rätta planet $z = L(x_1, x_2)$ är tangentplanet till ytan $z = f(x_1, x_2)$ vid punkten (a_1, a_2) .



Vi låter $\mathbf{x} = (x_1, x_2)$, $f(\mathbf{x}) = f(x_1, x_2)$ och $\mathbf{a} = (a_1, a_2)$. Använder vi gradienten

$$\nabla f(\mathbf{x}) = \begin{bmatrix} f'_{x_1}(\mathbf{x}) \\ f'_{x_2}(\mathbf{x}) \end{bmatrix} = \begin{bmatrix} f'_{x_1}(x_1, x_2) \\ f'_{x_2}(x_1, x_2) \end{bmatrix}$$

kan vi skriva linjäriseringen

$$L(\mathbf{x}) = f(\mathbf{a}) + \nabla f(\mathbf{a})^T(\mathbf{x} - \mathbf{a}).$$

Lägg märke till att $\nabla f(\mathbf{a})$ och $\mathbf{x} - \mathbf{a}$ är kolonnvektorer så $\nabla f(\mathbf{a})^T(\mathbf{x} - \mathbf{a})$ är skalärprodukten $\nabla f(\mathbf{a}) \cdot (\mathbf{x} - \mathbf{a})$.

Låt oss som exempel ta $f(\mathbf{x}) = x_1(1 + x_2^2) - 1$. Då gäller

$$\frac{\partial f}{\partial x_1}(\mathbf{x}) = 1 + x_2^2, \quad \frac{\partial f}{\partial x_2}(\mathbf{x}) = 2x_1x_2,$$

och därmed $\nabla f(\mathbf{x})^T = [1 + x_2^2 \quad 2x_1x_2]$. Linjäriseringen vid $\mathbf{a} = (2, 1)$ blir

$$L(\mathbf{x}) = f(\mathbf{a}) + \nabla f(\mathbf{a})^T(\mathbf{x} - \mathbf{a}) = 3 + [2 \quad 4] \begin{bmatrix} x_1 - 2 \\ x_2 - 1 \end{bmatrix} = 3 + 3(x_1 - 2) + 4(x_2 - 1).$$

Uppgift 1. Linjärisera $f(\mathbf{x}) = x_1 \cos(2x_1) \sin(x_2)$ runt $\mathbf{a} = (3, 1)$. Rita upp funktionsytan tillsammans med tangentplanet i $\mathbf{a}$. Även tangeringspunkten skall ritas ut.

Två funktioner i två variabler, $\mathbf{f}: \mathbb{R}^2 \rightarrow \mathbb{R}^2$

Låt $f_1(x_1, x_2)$ och $f_2(x_1, x_2)$ vara två differentierbara funktioner i två variabler, $f_1: \mathbb{R}^2 \rightarrow \mathbb{R}$ och $f_2: \mathbb{R}^2 \rightarrow \mathbb{R}$. Linjäriseringen av f_1 respektive f_2 runt punkten (a_1, a_2) ges av

$$\begin{aligned}L_1(x_1, x_2) &= f_1(a_1, a_2) + \frac{\partial f_1}{\partial x_1}(a_1, a_2)(x_1 - a_1) + \frac{\partial f_1}{\partial x_2}(a_1, a_2)(x_2 - a_2), \\L_2(x_1, x_2) &= f_2(a_1, a_2) + \frac{\partial f_2}{\partial x_1}(a_1, a_2)(x_1 - a_1) + \frac{\partial f_2}{\partial x_2}(a_1, a_2)(x_2 - a_2),\end{aligned}$$

Låter vi $\mathbf{x} = (x_1, x_2)$, $\mathbf{f}(\mathbf{x}) = (f_1(\mathbf{x}), f_2(\mathbf{x}))$ och $\mathbf{a} = (a_1, a_2)$ kan vi skriva

$$\begin{aligned}L_1(\mathbf{x}) &= f_1(\mathbf{a}) + \nabla f_1(\mathbf{a})^T(\mathbf{x} - \mathbf{a}), \\L_2(\mathbf{x}) &= f_2(\mathbf{a}) + \nabla f_2(\mathbf{a})^T(\mathbf{x} - \mathbf{a}),\end{aligned}$$

eller med matrisbeteckningar

$$\mathbf{L}(\mathbf{x}) = \mathbf{f}(\mathbf{a}) + \mathbf{D}\mathbf{f}(\mathbf{a})(\mathbf{x} - \mathbf{a}),$$

där $\mathbf{L}(\mathbf{x}) = (L_1(\mathbf{x}), L_2(\mathbf{x}))$ och

$$\mathbf{D}\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \frac{\partial f_1}{\partial x_2}(\mathbf{x}) \\ \frac{\partial f_2}{\partial x_1}(\mathbf{x}) & \frac{\partial f_2}{\partial x_2}(\mathbf{x}) \end{bmatrix},$$

är Jacobimatrisen.

Låt oss som exempel ta $\mathbf{f}(\mathbf{x}) = (x_1(1 + x_2^2) - 1, x_2(1 + x_1^2) - 2)$. Då gäller

$$\frac{\partial f_1}{\partial x_1}(\mathbf{x}) = 1 + x_2^2, \quad \frac{\partial f_1}{\partial x_2}(\mathbf{x}) = 2x_1x_2,$$

$$\frac{\partial f_2}{\partial x_1}(\mathbf{x}) = 2x_1x_2, \quad \frac{\partial f_2}{\partial x_2}(\mathbf{x}) = 1 + x_1^2,$$

och därmed

$$\mathbf{D}\mathbf{f}(\mathbf{x}) = \begin{bmatrix} 1 + x_2^2 & 2x_1x_2 \\ 2x_1x_2 & 1 + x_1^2 \end{bmatrix}.$$

Linjäriseringen vid $\mathbf{a} = (2, 1)$ blir

$$\mathbf{L}(\mathbf{x}) = \mathbf{f}(\mathbf{a}) + \mathbf{D}\mathbf{f}(\mathbf{a})(\mathbf{x} - \mathbf{a}) = \begin{bmatrix} 3 \\ 3 \end{bmatrix} + \begin{bmatrix} 2 & 4 \\ 4 & 5 \end{bmatrix} \begin{bmatrix} x_1 - 2 \\ x_2 - 1 \end{bmatrix}.$$

Uppgift 2. Linjärisera $\mathbf{f}(\mathbf{x}) = (x_1^3 + x_2^2 - 1, e^{x_1x_2} + x_1 + x_2 - 2)$ runt $\mathbf{a} = (2, 1)$. Ingen idé att försöka rita bilder. Skriv ned på papper och skriv kod i MATLAB. Använd matrisbeteckningar.

Flera funktioner i flera variabler, $\mathbf{f}: \mathbb{R}^n \rightarrow \mathbb{R}^m$

Vi generaliserar nu till godtyckligt antal funktioner i godtyckligt många variabler. Låt f_i beteckna m funktioner i n variabler, dvs. $\mathbf{f}: \mathbb{R}^n \rightarrow \mathbb{R}^m$. Vi låter

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{f}(\mathbf{x}) = \begin{bmatrix} f_1(x_1, \dots, x_n) \\ \vdots \\ f_m(x_1, \dots, x_n) \end{bmatrix},$$

och

$$D\mathbf{f}(\mathbf{x}) = \begin{bmatrix} \frac{\partial f_1}{\partial x_1}(\mathbf{x}) & \dots & \frac{\partial f_1}{\partial x_n}(\mathbf{x}) \\ \vdots & & \vdots \\ \frac{\partial f_m}{\partial x_1}(\mathbf{x}) & \dots & \frac{\partial f_m}{\partial x_n}(\mathbf{x}) \end{bmatrix}.$$

Här är $m \times n$ matrisen $D\mathbf{f}(\mathbf{x})$ Jacobimatrisen av $\mathbf{f}$ i $\mathbf{x}$. Linjäriseringen av $\mathbf{f}$ i punkten $\mathbf{a}$ blir

$$L(\mathbf{x}) = \mathbf{f}(\mathbf{a}) + D\mathbf{f}(\mathbf{a})(\mathbf{x} - \mathbf{a}).$$

Uppgift 3. Linjärisera $\mathbf{f}(t) = (\cos(t), \sin(t), t)$, $0 \leq t \leq 6\pi$, runt $a = 4$. Rita kurvan och tangenten. Även tangeringspunkten skall ritas ut.

3 Newtons metod

Låt $f: \mathbb{R} \rightarrow \mathbb{R}$ vara en deriverbar funktion. Vi skall lösa ekvationen $f(x) = 0$ med Newtons metod.

Antag att vi har en approximativ lösning x_k och vi vill hitta en bättre approximation x_{k+1} . Vi bildar linjäriseringen av f i x_k :

$$L(x) = f(x_k) + f'(x_k)(x - x_k)$$

och löser $L(x) = 0$ istället för $f(x) = 0$.

$$f(x_k) + f'(x_k)(x - x_k) = 0 \tag{1}$$

Lösningen får bli nästa approximation:

$$x_{k+1} = x_k - \frac{f(x_k)}{f'(x_k)}.$$

Detta är Newtons metod. Geometriskt betyder (1) att vi följer tangenten och hittar x_{k+1} där denna skär x -axeln.

Som stoppvillkor för iterationen tar vi

$$|x_{k+1} - x_k| \leq \text{tol.}$$

Det betyder att vi accepterar x_{k+1} om ändringen i sista iteration är mindre än toleransen. Vi tillåter maximalt k_{max} iterationer ($k_{max} = 10$ är rimligt).

Som ett litet exempel tar vi $f(x) = \cos(x) - x$ och vi skall lösa $f(x) = 0$. En graf (rita den gärna) visar att vi har ett nollställe och vi tar $x_0 = 1$ som startapproximation.

```
>> f=@(x)cos(x)-x; Df=@(x)-sin(x)-1;
>> x=1;
>> kmax=10; tol=0.5e-8;
>> for k=1:kmax
    d=-f(x)/Df(x);
    x=x+d;
    disp([x d])
    if abs(d)<tol, break, end
end

0.750363867840244  -0.249636132159756
0.739112890911362  -0.011250976928882
0.739085133385284  -0.000027757526078
0.739085133215161  -0.000000000170123
```

System av ekvationer

Nu är det dags för system av n ekvationer i n obekanta

$$\begin{cases} f_1(x_1, \dots, x_n) = 0 \\ \vdots \\ f_n(x_1, \dots, x_n) = 0 \end{cases}$$

Om vi låter

$$\mathbf{x} = \begin{bmatrix} x_1 \\ \vdots \\ x_n \end{bmatrix}, \quad \mathbf{f} = \begin{bmatrix} f_1 \\ \vdots \\ f_n \end{bmatrix}, \quad \mathbf{0} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix},$$

så har vi $\mathbf{f} : \mathbb{R}^n \rightarrow \mathbb{R}^n$, och vi kan skriva systemet på formen

$$\mathbf{f}(\mathbf{x}) = \mathbf{0}.$$

Antag att vi har en approximativ lösning $\mathbf{x}_k$ och vi vill hitta en bättre approximation $\mathbf{x}_{k+1}$. Vi bildar linjäriseringen av $\mathbf{f}$ i $\mathbf{x}_k$:

$$\mathbf{L}(\mathbf{x}) = \mathbf{f}(\mathbf{x}_k) + \mathbf{Df}(\mathbf{x}_k)(\mathbf{x} - \mathbf{x}_k) = \mathbf{0}.$$

och löser $\mathbf{L}(\mathbf{x}) = \mathbf{0}$ istället för $\mathbf{f}(\mathbf{x}) = \mathbf{0}$.

Om vi låter $\mathbf{d} = \mathbf{x} - \mathbf{x}_k$ så kan $\mathbf{L}(\mathbf{x}) = \mathbf{0}$ skrivas som ekvationen

$$\mathbf{Df}(\mathbf{x}_k)\mathbf{d} = -\mathbf{f}(\mathbf{x}_k).$$

Detta är ett linjärt ekvationssystem, Jacobimatrisen $\mathbf{Df}(\mathbf{x}_k)$ är en $n \times n$ -matris, som vi löser med avseende på $\mathbf{d}$. Nu bildar vi nästa approximation av lösningen till $\mathbf{f}(\mathbf{x}) = \mathbf{0}$:

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \mathbf{d}.$$

Som stoppvillkor för iterationen tar vi

$$\|\mathbf{x}_{k+1} - \mathbf{x}_k\| \leq \text{tol}$$

Vi avbryter när vi har tillräcklig noggrannhet och vi tillåter maximalt $k_{\max}$ iterationer ($k_{\max} = 10$ är rimligt).

Nu ser vi åter på exemplet från inledningen (Exempel 1, Adams 13.6). Vi har alltså

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} x_1(1+x_2^2)-1 \\ x_2(1+x_1^2)-2 \end{bmatrix}, \quad \mathbf{Df}(\mathbf{x}) = \begin{bmatrix} 1+x_2^2 & 2x_1x_2 \\ 2x_1x_2 & 1+x_1^2 \end{bmatrix},$$

och skall lösa $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. Från bilden med noll-nivåkurvorna ser vi att $\mathbf{x}_0 = (0.25, 2)$ nog är en bra startapproximation.

```
>> f=@(x) [x(1)*(1+x(2)^2)-1;x(2)*(1+x(1)^2)-2];
>> Df=@(x) [1+x(2)^2 2*x(1)*x(2);2*x(1)*x(2) 1+x(1)^2];
>> x=[0.25;2];
>> kmax=10; tol=0.5e-8;
>> for k=1:kmax
    d=-Df(x)\f(x);
    x=x+d;
    disp([x' norm(d)])
    if norm(d)<tol, break, end
end

0.217391304347826    1.913043478260870    0.092869605927364
0.214829670172721    1.911781803315968    0.002855484777347
0.214829232694196    1.911768811990568    0.000012998689285
0.214829232680284    1.911768811998807    0.000000000016168
```

Uppgift 4. Låt $\mathbf{f}(\mathbf{x}) = (x_1^3 + x_2^2 - 1, e^{x_1x_2} + x_1 + x_2 - 2)$. Lös ekvationssystemet $\mathbf{f}(\mathbf{x}) = \mathbf{0}$. Rita upp noll-nivåkurvorna till f_1 och f_2 för att se var ungefär lösningarna (skärningspunkterna) ligger. Hur många lösningar finns det? Läs av i grafiken en första approximation av en lösning för att sedan förbättra denna med Newtons metod. Rita ut lösningen med en liten ring. Upprepa tills du beräknat alla lösningar till systemet.

Uppgift 5. Ladda ner funktionen `newton` från studiohemsidan. Studera programkod och hjälptext. Använd `newton` för att lösa ekvationssystemet $\mathbf{f}(\mathbf{x}) = \mathbf{0}$, där $\mathbf{f}(\mathbf{x}) = (\sin(x_1) - x_2, x_1 - \cos(x_2))$.

4 Modifiering av Newtons metod

En dålig startapproximation kan leda till att Newtons metod divergerar (inte konvergerar), då är det lämpligt med dämpad Newton

$$\mathbf{x}_{k+1} = \mathbf{x}_k + \alpha_k \mathbf{d}_k, \quad k = 0, 1, \dots,$$

$$\mathbf{Df}(\mathbf{x}_k) \mathbf{d}_k = -\mathbf{f}(\mathbf{x}_k).$$

Dämpningsfaktorn α_k väljs så att $\|\mathbf{f}(\mathbf{x}_{k+1})\| < \|\mathbf{f}(\mathbf{x}_k)\|$. Man kan t.ex. börja med $\alpha_k = 1$, på försök ta ett steg i iterationen, om vi har fått en minskning av $\|\mathbf{f}\|$ accepterar vi steget. I annat fall halverar vi successivt α_k och gör nya försök, tills vi har en minskning av $\|\mathbf{f}\|$.

Om vi inte vill beräkna derivator, dvs. Jacobimatrisen, så kan vi approximera den kolonnvis med differenskvoter,

$$\frac{\partial \mathbf{f}}{\partial x_j}(\mathbf{x}) \approx \frac{\mathbf{f}(\mathbf{x} + \delta \mathbf{e}_j) - \mathbf{f}(\mathbf{x} - \delta \mathbf{e}_j)}{2h},$$

för lämpligt valt litet positivt h , där $\mathbf{e}_j$ är j :te enhetsvektorn. Gå tillbaka och kontrollera att $\frac{\partial \mathbf{f}}{\partial x_j}(\mathbf{x})$ är den j :te kolonnen i $\mathbf{D}\mathbf{f}(\mathbf{x})$!

På studiohemsidan finns funktionen `jacobi` som beräknar en approximation av Jacobimatrisen enligt ovan. Där finns också en modifierad Newton funktion `modnewton` som använder denna approximation av derivatan.

I MATLAB OPTIMIZATION TOOLBOX finner vi `fsolve` som också använder dessa modifieringar. För en sista gång ser vi på exemplet från inledningen. Vi har alltså

$$\mathbf{f}(\mathbf{x}) = \begin{bmatrix} x_1(1 + x_2^2) - 1 \\ x_2(1 + x_1^2) - 2 \end{bmatrix} = \mathbf{0},$$

och i MATLAB löser vi med (samma startapproximation som tidigare)

```
>> f=@(x) [x(1)*(1+x(2)^2)-1;x(2)*(1+x(1)^2)-2];
>> x0=[0.25;2];
>> x=fsolve(f,x0)
x =
    0.214829232694215
    1.911768811990538
```