8.14 QR-faktorisering

En ortogonal matris Q av dimension n är en matris som satisfierar

$$Q^TQ = QQ^T = I;$$

matrisens kolonner (och rader) är alltså ortogonala och normerade till euklidisk längd 1. Om man multiplicerar en vektor x med Q, så förändras inte den euklidiska längden:

$$\|y\|_2^2 = \|Qx\|_2^2 = x^TQ^TQx = x^Tx = \|x\|_2^2.$$

Man säger att den euklidiska normen är *invariant under ortogonala transformationer*. Just denna egenskap gör dessa matriser så användbara för matrisberäkningar, t ex lösning av minsta kvadratproblem. Vi skall strax bevisa följande sats.

Sats 8.14.1 QR-faktorisering *Varje $m \times n$-matris A, med $m \geq n$, kan faktoriseras*

$$A = Q\begin{pmatrix} R \\ 0 \end{pmatrix},$$

där Q är en ortogonal $m \times m$-matris, och R är högertriangulär av dimension n. Om vi partitionerar Q enligt $Q = (Q_1\ Q_2)$, där Q_1 har n kolonner, så kan faktoriseringen skrivas på formen

$$A = Q_1R.$$

Om kolonnerna i A är lineär oberoende så är matrisen R ickesingulär.

Beviset för satsen är konstruktivt, dvs vi skall ge en algoritm, som beräknar faktoriseringen.

Med hjälp av QR-faktoriseringen kan vi skriva normen av residualen i minsta kvadratproblemet

$$\begin{aligned}
\|r\|_2^2 &= \|Ax - b\|_2^2 = \left\|Q\begin{pmatrix} R \\ 0 \end{pmatrix}x - b\right\|_2^2 = \left\|Q\left(\begin{pmatrix} R \\ 0 \end{pmatrix}x - Q^Tb\right)\right\|_2^2 \\
&= \left\|\begin{pmatrix} R \\ 0 \end{pmatrix}x - \begin{pmatrix} Q_1^Tb \\ Q_2^Tb \end{pmatrix}\right\|_2^2 = \|Rx - Q_1^Tb\|_2^2 + \|Q_2^Tb\|_2^2.
\end{aligned}$$

(Vi har använt definitionen av en ortogonal matris, normens invarians och partitioneringen av Q). Under förutsättning att kolonnerna i A är lineärt oberoende, så blir R ickesingulär, och vi kan minimera residualen (göra första termen i sista högerledet lika med noll) genom att lösa systemet

$$Rx = Q_1^T b.$$

Man kan alltså *lösa minsta kvadratproblemet utan att bilda normalekvationerna*. Vi skall nu visa hur QR-faktoriseringen kan beräknas numeriskt. Våra verktyg för detta är enkla ortogonala matriser[3] P, som kan konstrueras så att multiplikationen PA nollställer specificerade element i matrisen A (jfr Gausstransformationer, Avsnitten 8.5 och 8.6). Vi börjar med ett exempel på ortogonala matriser av dimension 2.

Exempel En rotationsmatris

$$P = \begin{pmatrix} \cos\theta & \sin\theta \\ -\sin\theta & \cos\theta \end{pmatrix}$$

är ortogonal (visa det!), och multiplikationen Px vrider vektorn x medurs vinkeln θ. Antag, att vi vill nollställa andra elementet i x,

$$Px = \begin{pmatrix} \kappa \\ 0 \end{pmatrix};$$

detta kan åstadkommas genom att vi väljer vinkeln θ så att $\cos\theta = x_1/\sqrt{x_1^2 + x_2^2}$ och $\sin\theta = x_2/\sqrt{x_1^2 + x_2^2}$, eftersom

$$\frac{1}{\sqrt{x_1^2 + x_2^2}} \begin{pmatrix} x_1 & x_2 \\ -x_2 & x_1 \end{pmatrix} \begin{pmatrix} x_1 \\ x_2 \end{pmatrix} = \begin{pmatrix} \sqrt{x_1^2 + x_2^2} \\ 0 \end{pmatrix}.$$

Notera att vi inte behöver välja vinkeln θ explicit. I numerisk lineär algebra kallas rotationsmatriser ofta **Givenstransformationer**.

Givenstransformationer kan bäddas in i enhetsmatriser av större dimension och användas för att nollställa element i vektorer och matriser av större dimension än 2. Vi skall nu introducera en annan typ av ortogonala transformationer i R^n, som kan konstrueras så att de samtidigt nollställer en hel följd element i en vektor eller matris.

Låt $v \neq 0$ var en godtycklig vektor, och sätt

$$P = I - \frac{2}{v^T v} v v^T;$$

[3] Observera, att vi nu använder beteckningen P för ortogonala matriser, som är betydligt mer komplicerade än permutationsmatriserna i Avsnitt 8.5.

P är symmetrisk och ortogonal (verifieras genom en enkel uträkning). Sådana matriser kallas reflektionsmatriser eller **Householdertransformationer**. Låt x och y vara givna vektorer av samma längd, $\|x\|_2 = \|y\|_2$, och ställ frågan "Kan vi bestämma en Householdertransformation P sådan att $Px = y$?"

Ekvationen $Px = y$ kan skrivas

$$x - \frac{2v^T x}{v^T v} v = y,$$

vilket är av formen $\beta v = x - y$. Eftersom v ingår i P på ett sådant sätt att en faktor β kan förkortas bort, kan vi välja $\beta = 1$. Med $v = x - y$ får vi

$$v^T v = x^T x + y^T y - 2x^T y = 2(x^T x - x^T y),$$

eftersom $x^T x = y^T y$. Vidare blir

$$v^T x = x^T x - y^T x = \frac{1}{2} v^T v.$$

Alltså gäller

$$Px = x - \frac{2v^T x}{v^T v} v = x - v = y,$$

som vi ville. I matrisberäkningar brukar vi vilja nollställa element i en vektor, så vi väljer nu $y = \kappa e_1$, där $\kappa = \pm\|x\|_2$, och $e_1^T = (1 \quad 0 \quad \cdots \quad 0)$. Vektorn v ska alltså tas lika med

$$v = x - \kappa e_1.$$

För att undvika kancellation väljer vi $\operatorname{sign}(\kappa) = -\operatorname{sign}(x_1)$. När vi nu har beräknat v, kan vi, för att förenkla, skriva

$$P = I - \frac{2}{v^T v} v v^T = I - 2uu^T, \qquad u = \frac{1}{\|v\|_2} v;$$

Householdervektorn u har alltså längden 1. Beräkningen av Householdervektorn kan implementeras i följande Matlab-kod.

```
function u=househ(x)
% Beräkna Householdervektorn u så att
% (I - 2 u * u')x = k*e_1, där
% |k| är lika med euklidiska normen av x
% och e_1 är första enhetsvektorn
n=length(x);      % Antal komponenter i x
```

```
kap=norm(x); v=zeros(n,1);
v(1)=x(1)+sign(x(1))*kap;
v(2:n)=x(2:n);
u=(1/norm(v))*v;
```

Man bör undvika att bilda Householdermatrisen P explicit, eftersom matrisen kan representeras mycket kompaktare av vektorn u enbart. Multiplikation med P bör ske enligt $Px = x - (2u^T x)u$, varvid alltså matris-vektormultiplikationen kräver $4n$ flyttalsoperationer (i stället för $O(n^2)$ om man skulle ha bildat P explicit). Matrismultiplikationen PX sker analogt

$$PX = A - 2u(u^T X).$$

Multiplikation med en Householdertransformation implementeras i följande kod.

```
function Y=apphouse(u,X);
% Multiplicera matrisen X med Householdermatris
% Y = (I - 2 * u * u') * X
Y=X-2*u*(u'*X);
```

Exempel De första elementen i vektorn $x = (1 \quad 2 \quad 3 \quad 4)^T$ nollställs:

```
>>    x=[1; 2; 3; 4];
>>    u=househ(x);
>>    y=apphouse(u,x)

    y = -5.4772
              0
              0
              0
```

Vi skall nu beskriva hur man kan reducera en $m \times n$-matris A, där $m \geq n$, till högertriangulär form med hjälp av n Householdertransformationer. Bestäm först en Householdermatris P_1 av dimension n, som nollställer alla element utom det första i första kolonnen av A. Resultatet när vi multiplicerar A med P_1 blir

$$A^{(2)} = P_1 A^{(1)} = P_1 A = \begin{pmatrix} r_{11} & r_1^T \\ 0 & \hat{A}^{(2)} \end{pmatrix}.$$

Bestäm därefter en Householdermatris $\hat{P}_2$ av dimension n, som nollställer alla element utom det första i första kolonnen av $\hat{A}^{(2)}$. Sätt in $\hat{P}_2$ i en utökad ortogonal matris av dimension n,

$$P_2 = \begin{pmatrix} 1 & 0 \\ 0 & \hat{P}_2 \end{pmatrix},$$

och multiplicera

$$\begin{aligned} A^{(3)} &= P_2 A^{(2)} = \begin{pmatrix} 1 & 0 \\ 0 & \hat{P}_2 \end{pmatrix} \begin{pmatrix} r_{11} & r_1^T \\ 0 & \hat{A}^{(2)} \end{pmatrix} \\ &= \begin{pmatrix} r_{11} & r_1^T \\ 0 & \hat{P}_2 \hat{A}^{(2)} \end{pmatrix} = \begin{pmatrix} r_{11} & r_{12} & \cdots \\ 0 & r_{22} & r_2^T \\ 0 & 0 & \hat{A}^{(3)} \end{pmatrix}; \end{aligned}$$

observera att P_2 inte påverkar elementen i första raden, och inte heller nollorna i första kolonnen av $A^{(2)}$. Efter ytterligare $n-2$ analoga steg har vi överfört matrisen A på högertriangulär form,

$$A^{(n)} = P_n P_{n-1} \cdots P_2 P_1 A = \begin{pmatrix} R \\ 0 \end{pmatrix}.$$

Om vi kallar produkten av de ortogonala matriserna Q^T, så har vi beräknat QR-faktoriseringen

$$A = Q \begin{pmatrix} R \\ 0 \end{pmatrix}, \qquad Q = P_1 P_2 \cdots P_{n-1} P_n.$$

Om A har lineärt oberoende kolonner, så är enligt Sats 8.13.1 matrisen $A^T A$ positivt definit, vilket innebär att den har en Cholesky-faktorisering (se sidan 216) $A^T A = U^T U$, där U är högertriangulär och ickesingulär. Med QR-faktoriseringen får vi

$$A^T A = \begin{pmatrix} R^T & 0 \end{pmatrix} Q^T Q \begin{pmatrix} R \\ 0 \end{pmatrix} = \begin{pmatrix} R^T & 0 \end{pmatrix} \begin{pmatrix} R \\ 0 \end{pmatrix} = R^T R,$$

vilket visar att U i Cholesky-faktoriseringen av $A^T A$ är identisk med R i QR-faktoriseringen av A.

Beviset av Sats 8.14.1 är nu fullständigt.

Följande Matlab-funktion implementerar beräkning av QR-faktorisering med Householdertransformationer. Vi har valt att som utdata ge den "kompakta" formen

$$A = Q_1 R,$$

där Q_1 har dimension $m \times n$, med ortogonala kolonner.

```
function [Q1,R]=qrh(A);
% Beräkna QR-faktoriseringen av A med
% Householdertransformationer
[m,n]=size(A); qt=eye(m); % Initiera q' till enhetsmatris
R=A;
for i=1:n
  u=househ(R(i:m,i));
  R(i:m,i:n)=apphouse(u,R(i:m,i:n));
  qt(i:m,:)=apphouse(u,qt(i:m,:));
end
R=triu(R(1:n,1:n));   % Högertriangulär och n x n
q=qt'; Q1=q(:,1:n);
```

Exempel Vi löser inledningsexemplet i detta avsnitt (sidan 241).

```
>>    A=[1   1
         1   2
         1   3
         1   4
         1   5 ];
>>    b=[7.97; 10.2; 14.2; 16.0; 21.2];

>>    [Q1,R]=qrh(A)        % Beräkna Q1 och R
```

```
    Q1 =  -0.4472   -0.6325
          -0.4472   -0.3162
          -0.4472    0.0000
          -0.4472    0.3162
          -0.4472    0.6325

    R =   -2.2361   -6.7082
                0    3.1623

>>    x=R\(Q1'*b)          % Lös minsta kvadratproblemet

    x =  4.2360
         3.2260
```

Det är inte nödvändigt att beräkna matrisen Q (eller Q_1) explicit; i stället kan man spara Householdervektorerna. Om man inte beräknar Q kräver QR-faktoriseringen $2n^2(m - n/3)$ flyttalsoperationer.

Om man har ett överbestämt ekvationssystem, kan man i Matlab lösa det i minsta kvadratmening med operationen `x=A\b`. Detta innebär att matrisen QR-faktoriseras med Householdertransformationer, och att lösningen beräknas med hjälp av faktoriseringen (analogt till fallet då A är kvadratisk, då samma Matlaboperation innebär LR-faktorisering).

Ortogonala transformationer har väsentligt bättre stabilitetsegenskaper är Gauss-transformationer: eftersom multiplikation med en ortogonal matris bevarar vektorlängden, så kan ingen tillväxt av matriselement ske (det är ju detta som leder till noggrannhetsförlust vid Gausselimination). QR-faktorisering för att lösa ett vanligt lineärt ekvationssystem kräver dock dubbelt så mycket arbete som Gausselimination.

Arbetet för att lösa ett minsta kvadratproblem med hjälp av normalekvationer är jämförbart med QR-varianten om $m \approx n$, men normalekvationerna kan bildas och lösas dubbelt så snabbt då m är mycket större än n. Det är dock viktigt att påpeka att information förloras då man bildar normalekvationerna.

Exempel För matrisen

$$a = \begin{pmatrix} 1 & 1 \\ \epsilon & 0 \\ 0 & \epsilon \end{pmatrix}$$