Chap 3 Overdetermined Linear System

- Motivation
- The least square problems
- Solutions
  - Optimizing via LA & calculus
  - Normal equations

  } Mathematical viewpoints

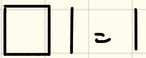
  - QR method for Least squares
  - Householder for QR

  } Numerical viewpoints

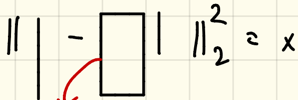
- From linear system to least squares problem

$$Ax = b \qquad\qquad \min \|b - Ax\|_2^2$$

To minimize the sum of the squares of the residual



square linear system $(m=n)$



overdetermined linear system $(m > n)$

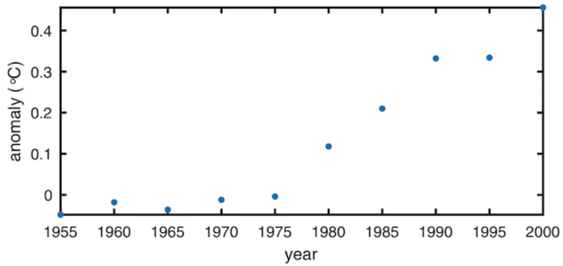
- Three views
  - Math: to minimize the inner product $(\|x\|_2^2 = (x^T \cdot x)^2)$
  - Stat: maximum likelihood
  - Physics: 2-norm coincide energy, minimize in natural systems

§ 3.1 Fitting functions to data

- Example 3.1.1

Here are 5-year averages of the worldwide temperature anomaly as compared to the 1951–1980 average (source: NASA).

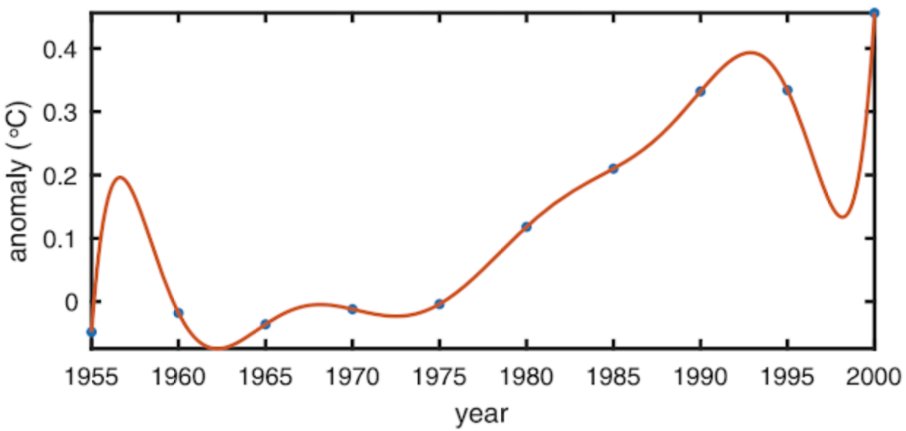
```
t = (1955:5:2000)';
y = [ -0.0480; -0.0180; -0.0360; -0.0120; -0.0040;
    0.1180; 0.2100; 0.3320; 0.3340; 0.4560 ];
plot(t,y,'.')
```



```
t = (t-1950)/10;
V = t.^0;          % vector of ones
n = length(t);
for j = 1:n-1
    V(:,j+1) = t.*V(:,j);
end
c = V\y;
p = @(x) polyval(flipud(c),(x-1950)/10);
hold on, fplot(p,[1955 2000])
```

[HW]: To improve the condition number of the matrix

To construct the Vandermonde matrix



Overfitting!

Try too hard to reproduce the data exactly.

- Sensitivity of higher order polynomial fitting

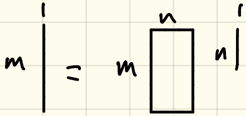
[Think] How to fix the overfitting problem?

[Idea] : Smaller n (lower degree)

$$y \approx f(t) = c_1 + c_2 t + \cdots + c_{n-1} t^{n-2} + c_n t^{n-1}, \qquad (3.1.1)$$

with $n < m$. Just as in (2.1.2), we can express a vector of f values by a matrix-vector multiplication:

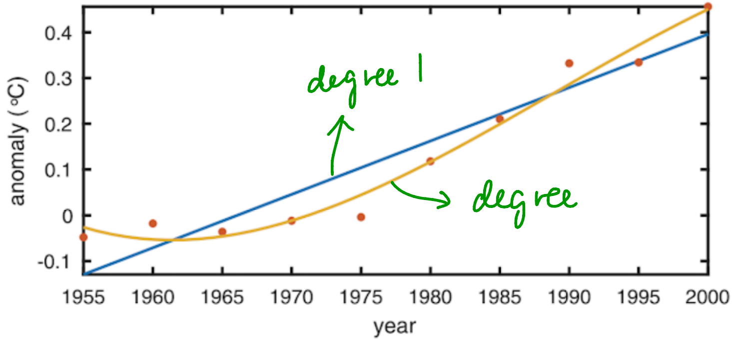
$$\begin{bmatrix} f(t_1) \\ f(t_2) \\ f(t_3) \\ \vdots \\ f(t_m) \end{bmatrix} = \begin{bmatrix} 1 & t_1 & \cdots & t_1^{n-1} \\ 1 & t_2 & \cdots & t_2^{n-1} \\ 1 & t_3 & \cdots & t_3^{n-1} \\ \vdots & \vdots & & \vdots \\ 1 & t_m & \cdots & t_m^{n-1} \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_m \end{bmatrix}. \qquad (3.1.2)$$



$y = Vc$

$\Rightarrow y \approx Vc$

```
V = [ t.^0 t t.^2 t.^3];      % Vandermonde-ish matrix
c = V\y;    f = @(x) polyval(c(end:-1:1),x-1955);
fplot(f,[1955 2000])
```



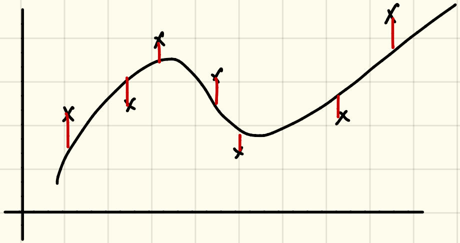
- The linear least squares formulation

$$f(t) = c_1 f_1(t) + c_2 f_2(t) + \cdots + c_n f_n(t)$$

  - Example: $f_1(t) = 1$, $f_2(t) = t$, $\cdots$, $f_n(t) = t^{n-1}$
  - The fit, $f(t)$, depends on the unknown parameters $(c_1, \cdots, c_n)$ <u>linearly</u>
  - <u>Nonlinear</u> example: $f(t) = c_1 + c_2 e^{c_3 t}$

- Optimization: To minimize $R(c_1, \cdots, c_n) = \sum_{i=1}^{m} [y_i - f(t_i)]^2$

Geometric:

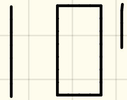


- [HW] why not $|y_i - f(t_i)|$?

Matrix

$$r = \begin{bmatrix} y_1 \\ y_2 \\ \vdots \\ y_{m-1} \\ y_m \end{bmatrix} - \begin{bmatrix} f_1(t_1) & f_2(t_1) & \cdots & f_n(t_1) \\ f_1(t_2) & f_2(t_2) & \cdots & f_n(t_2) \\ & & \vdots & \\ f_1(t_{m-1}) & f_2(t_{m-1}) & \cdots & f_n(t_{m-1}) \\ f_1(t_m) & f_2(t_m) & \cdots & f_n(t_m) \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ \vdots \\ c_n \end{bmatrix}.$$

$\Rightarrow \underset{x \in \mathbb{R}^n}{\operatorname{argmin}} \|b - Ax\|_2^2$



- Note: If $m = n$, to solve $Ax = b$ $\equiv$ $\operatorname{arg\,min} \|b - Ax\|_2^2$

$\because \|b - Ax\|_2^2 \geq 0$

- Change of variables
  - $g(t) = a_1 e^{a_2 t}$ $\Rightarrow \log y \approx \log g(t) = (\log a_1) + a_2 t$
    $\Rightarrow \tilde{y} = c_1 + c_2 t$
  - $h(t) = a_1 t^{a_2 t}$ $\Rightarrow \log y \approx \log h(t) = \log(a_1) + a_2 \log t$
    $\Rightarrow \hat{y} = c_1 + c_2 \hat{t}$
- MATLAB: semilogy, loglog

L7

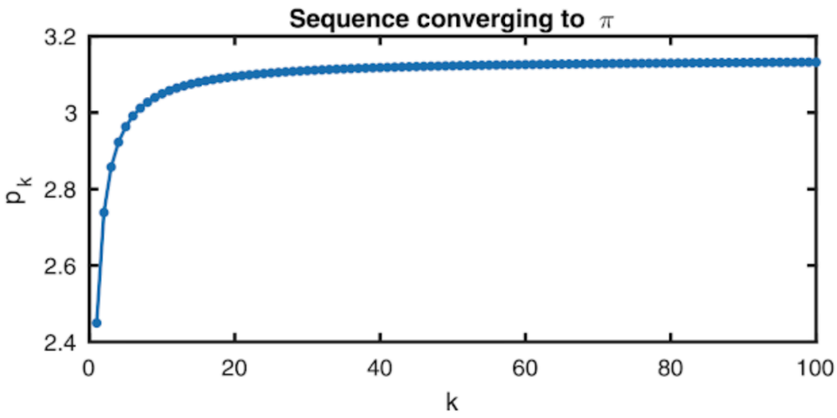
Example 3.1.3

Finding numerical approximations to π has fascinated people for millennia. One famous formula is

$$\frac{\pi^2}{6} = 1 + \frac{1}{2^2} + \frac{1}{3^2} + \cdots.$$

Say s_k is the sum of the first k terms of the series above, and $p_k = \sqrt{6s_k}$. Here is a fancy way to compute these sequences in a compact code.

```
k = (1:100)';
s = cumsum( 1./k.^2 );   % cumulative summation
p = sqrt(6*s);
plot(k,p,'.-')
xlabel('k'), ylabel('p_k')
title('Sequence converging to \pi')
```



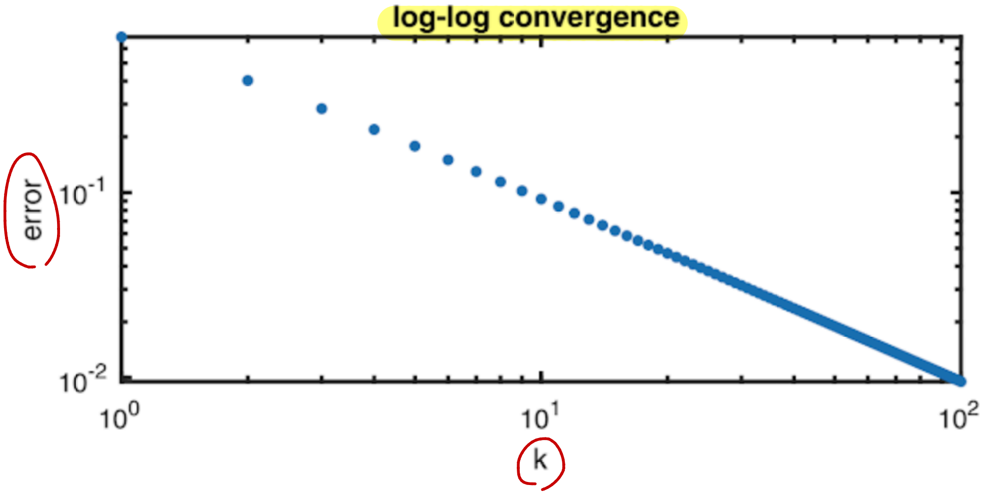
$$\pi^2 \approx 6\left[1 + \cdots + \frac{1}{k^2}\right]$$

$$\Rightarrow \pi \approx \underbrace{\sqrt{6\, s_k}}_{= p_k}$$

p_k: estimate π by k terms

This graph suggests that $p_k \to \pi$ but doesn't give much information about the rate of convergence. Let $\epsilon_k = |\pi - p_k|$ be the sequence of errors. By plotting the error sequence on a log–log scale, we can see a nearly linear relationship.

```
ep = abs(pi-p);      % error sequence
loglog(k,ep,'.'), title('log-log convergence')
xlabel('k'), ylabel('error'),  axis tight
```



This suggests a power-law relationship where $\epsilon_k \approx a k^b$, or $\log \epsilon_k \approx b(\log k) + \log a$.

log

log

Power-law fit

$\varepsilon_k \approx a k^b$

```
V = [ k.^0, log(k) ];      % fitting matrix
c = V \ log(ep)            % coefficients of linear fit
```

```
c =
   -0.1824
   -0.9674
```

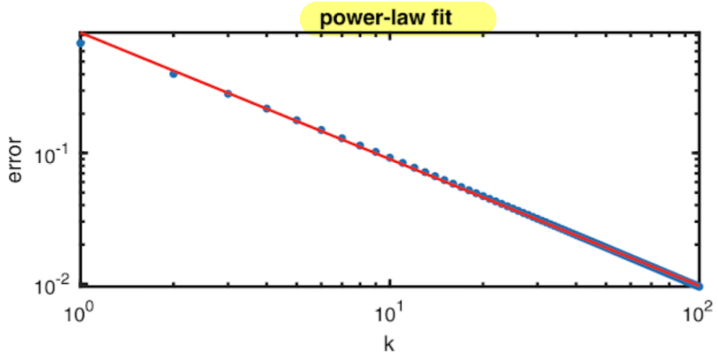
In terms of the parameters a and b used above, we have

```
a = exp(c(1)),  b = c(2),
```

```
a =
    0.8333
b =
   -0.9674
```

It's tempting to conjecture that $b = -1$ asymptotically. Here is how the numerical fit compares to the original convergence curve.

```
hold on, loglog(k,a*k.^b,'r'), title('power-law fit')
```



§ 3.2 The Normal Equations

– Matrix view

Theorem 3.2.1

If x satisfies $A^T(Ax-b)=0$, then x solves the linear least squares problem, i.e., x minimizes $\|b-Ax\|_2$.

Proof. Let $y \in \mathbb{R}^n$ be any vector. Then $A(x+y)-b = Ax-b+Ay$, and

$$\begin{aligned}
\|A(x+y)-b\|_2^2 &= [(Ax-b)+(Ay)]^T[(Ax-b)+(Ay)] \\
&= (Ax-b)^T(Ax-b)+2(Ay)^T(Ax-b)+(Ay)^T(Ay) \\
&= \|Ax-b\|_2^2 + 2y^TA^T(Ax-b)+\|Ay\|_2^2 \\
&= \|Ax-b\|_2^2 + \|Ay\|_2^2 \\
&\geq \|Ax-b\|_2^2.
\end{aligned}$$

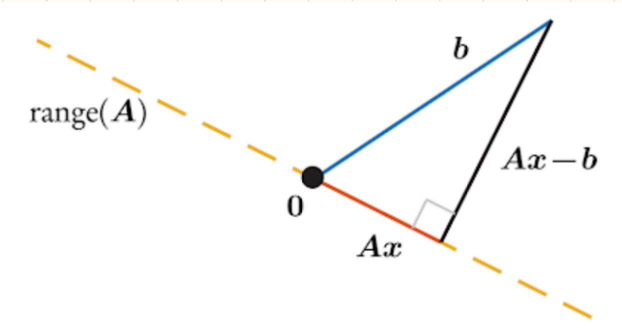
$\square$

$A^T(Ax-b)=0$

$\Rightarrow \underbrace{A^TA}\,x = \underbrace{A^Tb}$

$\Rightarrow \tilde{A}\,x = \tilde{b}$

– Geometric view



- pseudo inverse: $Ax \approx b \Rightarrow x = A^{+} b$

- MATLAB: $x = A \backslash b$ or $x = \mathrm{pinv}(A) * b$

don't use! Just like inv(A)

-

Theorem 3.2.2

For any real $m \times n$ matrix $\boldsymbol{A}$ with $m \geq n$, the following are true:

1. $\boldsymbol{A}^T\boldsymbol{A}$ is symmetric.
2. $\boldsymbol{A}^T\boldsymbol{A}$ is singular if and only if the columns of $\boldsymbol{A}$ are linearly dependent. (Equivalently, the rank of $\boldsymbol{A}$ is less than n.)
3. If $\boldsymbol{A}^T\boldsymbol{A}$ is nonsingular, then it is positive definite.

Proof. The first part is left as an exercise (Exercise 3.2.3). For the second part, suppose that $\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{z} = \boldsymbol{0}$. Note that $\boldsymbol{A}^T\boldsymbol{A}$ is singular if and only if $\boldsymbol{z}$ may be nonzero. Left-multiplying by $\boldsymbol{z}^T$, we find that

$$0 = \boldsymbol{z}^T\boldsymbol{A}^T\boldsymbol{A}\boldsymbol{z} = (\boldsymbol{A}\boldsymbol{z})^T(\boldsymbol{A}\boldsymbol{z}) = \|\boldsymbol{A}\boldsymbol{z}\|_2^2,$$

which is equivalent to $\boldsymbol{A}\boldsymbol{z} = \boldsymbol{0}$. Then $\boldsymbol{z}$ may be nonzero if and only if the columns of $\boldsymbol{A}$ are linearly dependent.

Finally, we can repeat the manipulations above to show that for any nonzero n-vector $\boldsymbol{v}$, $\boldsymbol{v}^T(\boldsymbol{A}^T\boldsymbol{A})\boldsymbol{v} = \|\boldsymbol{A}\boldsymbol{v}\|_2^2 \geq 0$, and equality is not possible thanks to the second part of the theorem. $\square$

Function 3.2.1 (`lsnormal`) Solve linear least squares by normal equations.

```
function x = lsnormal(A,b)
% LSNORMAL   Solve linear least squares by normal equations.
% Input:
%   A      coefficient matrix (m by n, m>n)
%   b      right-hand side (m by 1)
% Output:
%   x      minimizer of || b-Ax ||

N = A'*A;  z = A'*b;
R = chol(N);
w = forwardsub(R',z);                   % solve R'z=c
x = backsub(R,w);                       % solve Rx=z
```

In summary, the steps for solving the linear least squares problem $\boldsymbol{A}\boldsymbol{x} \approx \boldsymbol{b}$ are as follows:

1. Compute $\boldsymbol{N} = \boldsymbol{A}^T\boldsymbol{A}$.
2. Compute $\boldsymbol{z} = \boldsymbol{A}^T\boldsymbol{b}$.
3. Solve the $n \times n$ linear system $\boldsymbol{N}\boldsymbol{x} = \boldsymbol{z}$ for $\boldsymbol{x}$.

- Conditioning and stability
  - To solve $Ax=b$, the condition number is $\kappa(A) = \|A\| \cdot \|A^{-1}\|$
  - To solve the L.S.P., $\kappa(A) = \|A\| \cdot \|A^{+}\|$

    If $\operatorname{rank}(A) < n \Rightarrow \kappa(A) = \infty$

  - To solve the normal equations $(A^TA)x = (A^Tb)$

    $\Rightarrow \kappa(A^TA) = \kappa(A)^2$ (see exercise 7.3.8 for proof)

    Cond. # 变平方倍!

1

Example 3.2.1

Because the functions $\sin^2(t)$, $\cos^2(t)$, and 1 are linearly dependent, we should find that the following matrix is somewhat ill-conditioned.

```
t = linspace(0,3,400)';
A = [ sin(t).^2, cos((1+1e-7)*t).^2, t.^0 ];
kappa = cond(A)
```

```
kappa =
   1.8253e+07
```

2

Now we set up an artificial linear least squares problem with a known exact solution that actually makes the residual zero.

```
x = [1;2;1];
b = A*x;
```

Using backslash to find the solution, we get a relative error that is about κ times machine epsilon.

```
x_BS = A\b;
observed_err = norm(x_BS-x)/norm(x)
max_err = kappa*eps
```

```
observed_err =
   1.3116e-10
max_err =
   4.0530e-09
```

3

If we formulate and solve via the normal equations, we get a much larger relative error. With $\kappa^2 \approx 10^{14}$, we may not be left with more than about 2 accurate digits.

```
N = A'*A;
x_NE = N\(A'*b);
observed_err = norm(x_NE-x)/norm(x)
digits = -log10(observed_err)
```

```
observed_err =
    0.0150
digits =
    1.8226
```

§ 3.3 QR factorization

- Orthogonal: $q_i^T q_j = 0$, $i \neq j$ (e.g. $u \perp v \Leftrightarrow u^T v = 0$)
- Orthonormal: $\begin{cases} q_i^T q_j = 0, \\ q_i^T q_i = 1 \quad (\text{i.e. } \|q_i\|_2 = 1) \end{cases}$

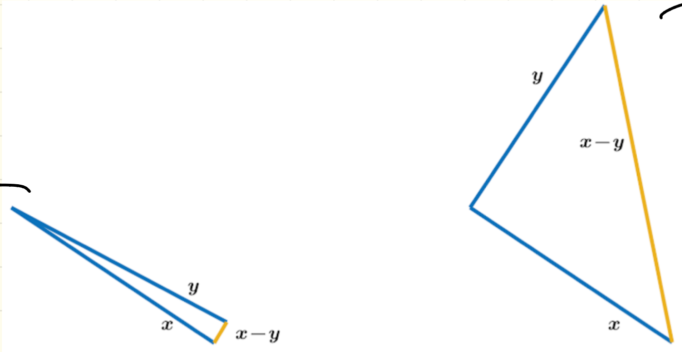


Figure 3.2. *Cancellation and orthogonality.*

- x, y 線性独立
- 若 $x \approx y$, $\|x-y\|$ may have <u>subtractive cancellation</u>

- $x \perp y$
- $x \pm y$ are well conditioned
- 彼此沒有分量 (e.g. $(0,1), (1,0)$)

- Orthgonal matrix, orthonormal matrix

$$Q^TQ = \begin{bmatrix} q_1^T \\ q_2^T \\ \vdots \\ q_k^T \end{bmatrix} \begin{bmatrix} q_1 & q_2 & \cdots & q_k \end{bmatrix} = \begin{bmatrix} q_1^Tq_1 & q_1^Tq_2 & \cdots & q_1^Tq_k \\ q_2^Tq_1 & q_2^Tq_2 & \cdots & q_2^Tq_k \\ \vdots & \vdots & & \vdots \\ q_k^Tq_1 & q_k^Tq_2 & \cdots & q_k^Tq_k \end{bmatrix}. = \begin{bmatrix} \times & & & 0 \\ & \times & & \\ & & \times & \\ 0 & & & \ddots \ \times \end{bmatrix}$$

Theorem 3.3.1

Suppose Q is a real $n \times k$ ONC matrix (matrix with orthonormal columns). Then

1. $Q^TQ = I$ ($k \times k$ identity);
2. $\|Qx\|_2 = \|x\|_2$ for all k-vectors x; ⇒ 保長
3. $\|Q\|_2 = 1$.

Proof. The first part is derived above. The second part follows a pattern that has become well established by now:

$$\|Qx\|_2^2 = (Qx)^T(Qx) = x^TQ^TQx = x^TIx = \|x\|_2^2.$$

The last part of the theorem is left to the exercises. □

Theorem 3.3.2

Suppose Q is an $n \times n$ real orthogonal matrix. Then

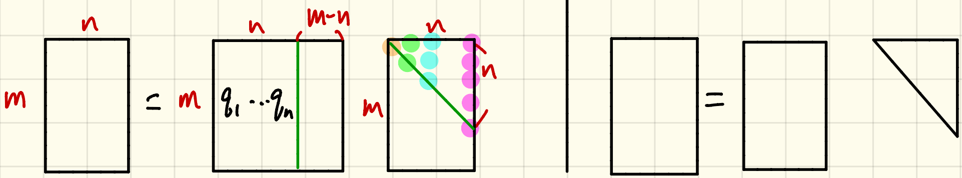
1. Q^T is also an orthogonal matrix;
2. $\kappa(Q) = 1$ in the 2-norm;
3. for any other $n \times n$ matrix A, $\|AQ\|_2 = \|A\|_2$;
4. if U is another $n \times n$ orthogonal matrix, then QU is also orthogonal.

- QR factorization

Theorem 3.3.3

Every real $m \times n$ matrix $\boldsymbol{A}$ $(m \geq n)$ can be written as $\boldsymbol{A} = \boldsymbol{Q}\boldsymbol{R}$, where $\boldsymbol{Q}$ is an $m \times m$ orthogonal matrix and $\boldsymbol{R}$ is an $m \times n$ upper triangular matrix.

- $A = QR$ | $A = \hat{Q}\hat{R}$ (Thin QR factorization)



- $q_1 \cdots q_n$ 是 A 的正交基底

$a_1 \in \text{span}\{q_1\}$ 係數：

$a_2 \in \text{span}\{q_1, q_2\}$ 係數：

$\vdots$

$a_n \in \text{span}\{q_1, q_2, \cdots, q_n\}$ 係數：

[HW] How about LU?

Hint: L forms 線性組合

基底

- Example 3.3.1

Linear system	$Ax = b$	$A = LU$
least squares	$\min \|b - Ax\|_2$	$A = QR$

- Normal equation : $A^T A \, x = A^T b$

① Cholesky factorization

$$(A^T A) = L L^T \qquad \kappa(A^T A) = \kappa(A)^2$$

$$L L^T = \tilde{b}$$

② QR method

$$A = \hat{Q} \hat{R}$$

$$\cancel{\hat{R}^T} \, \cancel{\hat{Q}^T} \cdot \cancel{\hat{Q}} \hat{R} \cdot x = \cancel{\hat{R}^T} Q^T b$$

$$\hat{R} \, x = \bar{b}$$

Function 3.3.1 (`lsqrfact`) Solve linear least squares by QR factorization.

```
function x = lsqrfact(A,b)
% LSQRFACT   Solve linear least squares by QR factorization.
% Input:
%    A      coefficient matrix (m by n, m>n)
%    b      right-hand side (m by 1)
% Output:
%    x      minimizer of || b-Ax ||

[Q,R] = qr(A,0);                        % compressed factorization
c = Q'*b;
x = backsub(R,c);
```

- Note : $x = A \backslash b$ solves the least squares problem

- Note :

importantly, even though we derived (3.3.4) from the normal equations, *the solution of least squares problems via QR factorization does not suffer from the instability seen when the normal equations are solved directly using Cholesky factorization.*

§ 3.4 Computing QR factorization

- Householder reflection

P: orthogonal matrix

$$Px = \begin{bmatrix} \pm\|x\| \\ 0 \\ 0 \\ \vdots \\ 0 \end{bmatrix} = \pm\|x\| \cdot e_1$$

- P is orthogonal

  ⇒ 保長

  $$\|Px\|_2 = \|x\|_2$$

- Px 把 x "所有的重量" 集中在第一個 component

- $$P = \underbrace{I - 2\,\frac{\overset{\text{外積, rank 1}}{vv^T}}{\underset{\text{normalization, } \|v\|_2^2}{v^Tv}}}_{I \text{ 的 rank-1 update}}, \quad \text{where } v = \|z\|e_1 - z$$

Theorem 3.4.1

Let $\boldsymbol{v} = ||\boldsymbol{z}||\boldsymbol{e}_1 - \boldsymbol{z}$ and let $\boldsymbol{P}$ be given by (3.4.3). Then $\boldsymbol{P}$ is symmetric and orthogonal, and $\boldsymbol{P}\boldsymbol{z} = ||\boldsymbol{z}||\boldsymbol{e}_1$.

Proof. The case $\boldsymbol{v} = \boldsymbol{0}$ is obvious. For $\boldsymbol{v} \neq \boldsymbol{0}$, the proofs of symmetry and orthogonality are left to the exercises. As for the last fact, we simply compute

$$\boldsymbol{P}\boldsymbol{z} = \boldsymbol{z} - 2\frac{\boldsymbol{v}\boldsymbol{v}^T\boldsymbol{z}}{\boldsymbol{v}^T\boldsymbol{v}} = \boldsymbol{z} - 2\frac{\boldsymbol{v}^T\boldsymbol{z}}{\boldsymbol{v}^T\boldsymbol{v}}\boldsymbol{v}, \tag{3.4.4}$$

and, since $\boldsymbol{e}_1^T\boldsymbol{z} = z_1$,

$$\begin{aligned}\boldsymbol{v}^T\boldsymbol{v} &= ||\boldsymbol{z}||^2 - 2||\boldsymbol{z}||z_1 + \boldsymbol{z}^T\boldsymbol{z} = 2||\boldsymbol{z}||(||\boldsymbol{z}|| - z_1),\\ \boldsymbol{v}^T\boldsymbol{z} &= ||\boldsymbol{z}||z_1 - \boldsymbol{z}^T\boldsymbol{z} = -||\boldsymbol{z}||(||\boldsymbol{z}|| - z_1),\end{aligned}$$

leading finally to

$$\boldsymbol{P}\boldsymbol{z} = \boldsymbol{z} - 2\cdot\frac{-||\boldsymbol{z}||(||\boldsymbol{z}|| - z_1)}{2||\boldsymbol{z}||(||\boldsymbol{z}|| - z_1)}\boldsymbol{v} = \boldsymbol{z} + \boldsymbol{v} = ||\boldsymbol{z}||\boldsymbol{e}_1. \qquad \square$$

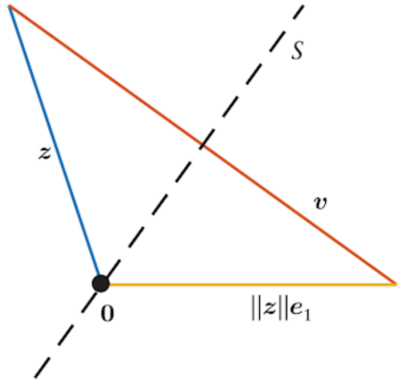


Figure 3.3. *Action of a Householder reflector.*

Step 1:

$$\begin{bmatrix} & & \\ & P_1 & \\ & & \end{bmatrix}\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} = \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \end{bmatrix}$$

Step 2:

$$\begin{bmatrix} 1 & \\ & \boxed{P_2} \end{bmatrix}\begin{bmatrix} & & \\ & P_1 & \\ & & \end{bmatrix}\begin{bmatrix} x & x & x \\ x & x & x \\ x & x & x \\ x & x & x \end{bmatrix} = \begin{bmatrix} 1 & \\ & \boxed{P_2} \end{bmatrix}\begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \\ 0 & \bar{x} & \bar{x} \end{bmatrix}$$

$$= \begin{bmatrix} \bar{x} & \bar{x} & \bar{x} \\ 0 & x' & x' \\ 0 & 0 & x' \\ 0 & 0 & x' \end{bmatrix}$$

Step 3

$$\begin{bmatrix}1&&\\&1&\\&&\boxed{P_3}\end{bmatrix}\underbrace{\begin{bmatrix}1&\\&\boxed{P_2}\end{bmatrix}\begin{bmatrix}&P_1&\end{bmatrix}\begin{bmatrix}x&x&x\\x&x&x\\x&x&x\\x&x&x\end{bmatrix}}$$

$$=\begin{bmatrix}1&&\\&1&\\&&\boxed{P_3}\end{bmatrix}\underbrace{\begin{bmatrix}\bar{x}&\bar{x}&\bar{x}\\0&x'&x'\\0&0&x'\\0&0&x'\end{bmatrix}}$$

$$=\begin{bmatrix}\bar{x}&\bar{x}&\bar{x}\\0&x'&x'\\0&0&x''\\0&0&0\end{bmatrix}$$

$$=\begin{bmatrix}\hat{R}\\\hline 0\end{bmatrix}$$

- $Q_k = \left[\begin{array}{c|c} I_{k-1} & 0 \\ \hline 0 & P_k \end{array}\right]$ is orthogonal!

$$Q_k^T Q_k = \left[\begin{array}{c|c} I_{k-1} & \\ \hline 0 & P_k^T \end{array}\right]\left[\begin{array}{c|c} I_{k-1} & \\ \hline & P_k \end{array}\right] = \left[\begin{array}{c|c} I_{k-1} & \\ \hline & I_k \end{array}\right]$$

- $$Q_3 Q_2 Q_1 A = \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}$$

$$\cancel{Q_1^T}\,\cancel{Q_2^T}\,\cancel{Q_3^T}\,\cancel{Q_3}\,\cancel{Q_2}\,\cancel{Q_1}\, A = \underbrace{Q_1^T Q_2^T Q_3^T}_{Q} \begin{bmatrix} \hat{R} \\ 0 \end{bmatrix}$$

$$A = Q\,R$$

$$Q^T Q = I$$

$$(Q_3 Q_2 Q_1)(Q_1^T Q_2^T Q_3^T) = I$$

- $\text{Cost} \approx 2mn^2 - \frac{n^3}{3}$

Function 3.4.1 (`qrfact`) QR factorization by Householder reflections.

```
function [Q,R] = qrfact(A)
% QRFACT   QR factorization by Householder reflections.
% (demo only--not efficient)
% Input:
%   A       m-by-n matrix
% Output:
%   Q,R     A=QR, Q m-by-m orthogonal, R m-by-n upper triangular

[m,n] = size(A);
Q = eye(m);
for k = 1:n
  z = A(k:m,k);
  v = [ -sign(z(1))*norm(z) - z(1); -z(2:end) ];
  nrmv = norm(v);
  if nrmv < eps, continue, end          % nothing is done in this iteration
  v = v / nrmv;                         % removes v'*v in other formulas
  % Apply the reflection to each relevant column of A and Q
  for j = 1:n
    A(k:m,j) = A(k:m,j) - v*( 2*(v'*A(k:m,j)) );
  end
  for j = 1:m
    Q(k:m,j) = Q(k:m,j) - v*( 2*(v'*Q(k:m,j)) );
  end
end

Q = Q';
R = triu(A);                            % enforce exact triangularity
```