



Forelesning IV

IMAT2150: Høsten 2023

Kent Holing, NTNU (30. august 2023)

LÆRINGSMÅL for dagens forelesning

- PA = LU faktorisering (S: 2.4)
- Feilkilder/Feilforplantning (S: 2.3)

(Merk: motsatt rekkefølge av det som er i SAUER.)

Eksempel på hvorfor naiv GE **må** modifiseres

Gitt LS-et med koeffisientmatrise $A = \begin{pmatrix} \delta & -1 \\ 1 & 2 \end{pmatrix}$ for $\delta > 0$ og venstre side $b = [-1, 8]^T$.

Naiv Gauss eliminasjon (GE) på $\begin{pmatrix} \delta & -1 & -1 \\ 1 & 2 & 8 \end{pmatrix}$ gir $\begin{pmatrix} \delta & -1 & -1 \\ 0 & 2 + 1/\delta & 8 + 1/\delta \end{pmatrix}$.

IMAT2150:

Eksempel på hvorfor naiv GE **må** **modifiseres**

Gitt LS-et med koeffisientmatrise $A = \begin{pmatrix} \delta & -1 \\ 1 & 2 \end{pmatrix}$ for $\delta > 0$ og venstre side $b = [-1, 8]^T$.

Naiv Gauss eliminasjon (GE) på $\begin{pmatrix} \delta & -1 & -1 \\ 1 & 2 & 8 \end{pmatrix}$ gir $\begin{pmatrix} \delta & -1 & -1 \\ 0 & 2 + 1/\delta & 8 + 1/\delta \end{pmatrix}$.

Med $\delta = 2^{-56}$ er $2 + 1/\delta = (1 + 2^{-55}) \times 2^{56}$, som **på datamaskinen lagres som** $1/\delta = 2^{56}$.

Gitt LS-et med koeffisientmatrise $A = \begin{pmatrix} \delta & -1 \\ 1 & 2 \end{pmatrix}$ for $\delta > 0$ og venstre side $b = [-1, 8]^T$.

Naiv Gauss eliminasjon (GE) på $\begin{pmatrix} \delta & -1 & -1 \\ 1 & 2 & 8 \end{pmatrix}$ gir $\begin{pmatrix} \delta & -1 & -1 \\ 0 & 2 + 1/\delta & 8 + 1/\delta \end{pmatrix}$.

Med $\delta = 2^{-56}$ er $2 + 1/\delta = (1 + 2^{-55}) \times 2^{56}$, som på datamaskinen lagres som $1/\delta = 2^{56}$.

Videre er $8 + 1/\delta = (1 + 2^{-53}) \times 2^{56}$, som også lagres som $1/\delta = 2^{56}$.¹

¹For flyttallsaritmetikken i IEEE754 DP er $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$.

IMAT2150:

Eksempel på hvorfor naiv GE må modifiseres

Gitt LS-et med koeffisientmatrise $A = \begin{pmatrix} \delta & -1 \\ 1 & 2 \end{pmatrix}$ for $\delta > 0$ og venstre side $b = [-1, 8]^T$.

Naiv Gauss eliminasjon (GE) på $\begin{pmatrix} \delta & -1 & -1 \\ 1 & 2 & 8 \end{pmatrix}$ gir $\begin{pmatrix} \delta & -1 & -1 \\ 0 & 2 + 1/\delta & 8 + 1/\delta \end{pmatrix}$.

Med $\delta = 2^{-56}$ er $2 + 1/\delta = (1 + 2^{-55}) \times 2^{56}$, som på datamaskinen lagres som $1/\delta = 2^{56}$.

Videre er $8 + 1/\delta = (1 + 2^{-53}) \times 2^{56}$, som også lagres som $1/\delta = 2^{56}$.¹

Vi sier at matrisekoeffisientene 2 og 8 har blitt "oversvømt" under den naive GE-en, og det er det som skaper problemet med tapet av nøyaktighet.

¹For flyttallsaritmetikken i IEEE754 DP er $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$.

IMAT2150:

Eksempel på hvorfor naiv GE må modifiseres

Gitt LS-et med koeffisientmatrise $A = \begin{pmatrix} \delta & -1 \\ 1 & 2 \end{pmatrix}$ for $\delta > 0$ og venstre side $b = [-1, 8]^T$.

Naiv Gauss eliminasjon (GE) på $\begin{pmatrix} \delta & -1 & -1 \\ 1 & 2 & 8 \end{pmatrix}$ gir $\begin{pmatrix} \delta & -1 & -1 \\ 0 & 2 + 1/\delta & 8 + 1/\delta \end{pmatrix}$.

Med $\delta = 2^{-56}$ er $2 + 1/\delta = (1 + 2^{-55}) \times 2^{56}$, som på datamaskinen lagres som $1/\delta = 2^{56}$.

Videre er $8 + 1/\delta = (1 + 2^{-53}) \times 2^{56}$, som også lagres som $1/\delta = 2^{56}$.¹

Vi sier at matrisekoeffisientene 2 og 8 har blitt "oversvømt" under den naive GE-en, og det er det som skaper problemet med tapet av nøyaktighet.

Vi ser at dette fører til løsningen datamaskinen beregner er $x_1 = 0$ og $x_2 = 1$,² noe som er (som vi skal se) veldig unøyaktig for å si det mildt!

¹For flytallsaritmetikken i IEEE754 DP er $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$.

²LS-et blir jo $\delta x_1 - x_2 = -1$ og $(1/\delta) x_2 = 1/\delta$. (Egentlig returnerer datamaskinen $x_1 = 0.0$ og $x_2 = 1.0$).

IMAT2150:

Eksempel på hvorfor naiv GE må modifiseres

Gitt LS-et med koeffisientmatrise $A = \begin{pmatrix} \delta & -1 \\ 1 & 2 \end{pmatrix}$ for $\delta > 0$ og venstre side $b = [-1, 8]^T$.

Naiv Gauss eliminasjon (GE) på $\begin{pmatrix} \delta & -1 & -1 \\ 1 & 2 & 8 \end{pmatrix}$ gir $\begin{pmatrix} \delta & -1 & -1 \\ 0 & 2 + 1/\delta & 8 + 1/\delta \end{pmatrix}$.

Med $\delta = 2^{-56}$ er $2 + 1/\delta = (1 + 2^{-55}) \times 2^{56}$, som på datamaskinen lagres som $1/\delta = 2^{56}$.

Videre er $8 + 1/\delta = (1 + 2^{-53}) \times 2^{56}$, som også lagres som $1/\delta = 2^{56}$.¹

Vi sier at matrisekoeffisientene 2 og 8 har blitt "oversvømt" under den naive GE-en, og det er det som skaper problemet med tapet av nøyaktighet.

Vi ser at dette fører til løsningen datamaskinen beregner er $x_1 = 0$ og $x_2 = 1$,² noe som er (som vi skal se) veldig unøyaktig for å si det mildt!

Vi skal nå bytte om på rekkefølgen av ligningene i LS-et før vi starter med GE-en (dvs. vi velger 1 som pivotelement i stedet for $\delta \ll 1$).

¹For flytallsaritmetikken i IEEE754 DP er $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$.

²LS-et blir jo $\delta x_1 - x_2 = -1$ og $(1/\delta) x_2 = 1/\delta$. (Egentlig returnerer datamaskinen $x_1 = 0.0$ og $x_2 = 1.0$).

Eksempel på hvorfor naiv GE må modifiseres (forts.)

Vi skal altså anvende GE med såkalt **partiell/delvis privotering** (altså ta hensyn til at radombytting kan være nødvendig).

Etter ombytting av radene i LS på forrige slide gir dette LS-et $\begin{pmatrix} 1 & 2 & 8 \\ \delta & -1 & -1 \end{pmatrix}$ med

$$\begin{pmatrix} 1 & 2 & 8 \\ 0 & -(1+2\delta) & -(1+8\delta) \end{pmatrix} \text{ etter GE-en.}$$

Vi skal altså anvende GE med såkalt **partiell/delvis pivoting** (altså ta hensyn til at radombytting kan være nødvendig).

Etter ombytting av radene i LS på forrige slide gir dette LS-et $\begin{pmatrix} 1 & 2 & 8 \\ \delta & -1 & -1 \end{pmatrix}$ med $\begin{pmatrix} 1 & 2 & 8 \\ 0 & -(1+2\delta) & -(1+8\delta) \end{pmatrix}$ etter GE-en.

Dette lagres i datamaskinen som $\begin{pmatrix} 1 & 2 & 8 \\ 0 & -1 & -1 \end{pmatrix}$,¹ som igjen fører til (som vi skal se) den svært nøyaktige løsningen $x_1 = 6.0$ og $x_2 = 1.0$.

¹For flytallsaritmetikken i IEEE754 DP er som sagt tidligere $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$. Og, da $2\delta < 2^{-53}$ og $8\delta = 2^{-53}$, er $1 + 2\delta = 1 + 8\delta = 1$.

IMAT2150:

Eksempel på hvorfor naiv GE må modifiseres (forts.)

Vi skal altså anvende GE med såkalt **partiell/delvis pivoting** (altså ta hensyn til at radombytting kan være nødvendig).

Etter ombytting av radene i LS på forrige slide gir dette LS-et $\begin{pmatrix} 1 & 2 & 8 \\ \delta & -1 & -1 \end{pmatrix}$ med $\begin{pmatrix} 1 & 2 & 8 \\ 0 & -(1+2\delta) & -(1+8\delta) \end{pmatrix}$ etter GE-en.

Dette lagres i datamaskinen som $\begin{pmatrix} 1 & 2 & 8 \\ 0 & -1 & -1 \end{pmatrix}$,¹ som igjen fører til (som vi skal se) den svært nøyaktige løsningen $x_1 = 6.0$ og $x_2 = 1.0$.

Med partiell pivoting (PP) for GE-en har vi altså **unngått det forrige tapet av nøyaktigheten**.

¹For flytallsaritmetikken i IEEE754 DP er som sagt tidligere $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$. Og, da $2\delta < 2^{-53}$ og $8\delta = 2^{-53}$, er $1 + 2\delta = 1 + 8\delta = 1$.

Vi skal altså anvende GE med såkalt **partiell/delvis pivoting** (altså ta hensyn til at radombytting kan være nødvendig).

Etter ombytting av radene i LS på forrige slide gir dette LS-et $\begin{pmatrix} 1 & 2 & 8 \\ \delta & -1 & -1 \end{pmatrix}$ med $\begin{pmatrix} 1 & 2 & 8 \\ 0 & -(1+2\delta) & -(1+8\delta) \end{pmatrix}$ etter GE-en.

Dette lagres i datamaskinen som $\begin{pmatrix} 1 & 2 & 8 \\ 0 & -1 & -1 \end{pmatrix}$,¹ som igjen fører til (som vi skal se) den svært nøyaktige løsningen $x_1 = 6.0$ og $x_2 = 1.0$.

Med partiell pivoting (PP) for GE-en har vi altså **unngått det forrige tapet av nøyaktigheten**.

Ved PP bruker vi maksimale (i tallverdi) pivotelementer for hver kolonne under hele GE-en. Dette hindrer at eliminasjonene "ødelegger" de andre ligningene.

¹For flytallsaritmetikken i IEEE754 DP er som sagt tidligere $1 + \mu = 1$ for alle flyttall $\mu \leq 2^{-53}$. Og, da $2\delta < 2^{-53}$ og $8\delta = 2^{-53}$, er $1 + 2\delta = 1 + 8\delta = 1$.

Merk at den nøyaktige løsningen $x_1 = 6.0$ og $x_2 = 1.0$ på forrige lysark ikke er eksakt!

Merk at den nøyaktige løsningen $x_1 = 6.0$ og $x_2 = 1.0$ på forrige lysark ikke er eksakt!

Python (IEEE754 DP) til venstre og Mathematica® til høyre:¹

```
augmented matrix:
[[ 1.38777878e-17 -1.00000000e+00 -1.00000000e+00]
 [ 1.00000000e+00  2.00000000e+00  8.00000000e+00]]
result of Gauss elimination on augmented matrix:
[[ 1.38777878e-17 -1.00000000e+00 -1.00000000e+00]
 [ 0.00000000e+00  7.20575940e+16  7.20575940e+16]]
solution of LS no pivoting:
[0.0, 1.0]

augmented matrix:
[[ 1.00000000e+00  2.00000000e+00  8.00000000e+00]
 [ 1.38777878e-17 -1.00000000e+00 -1.00000000e+00]]
result of Gauss elimination on augmented matrix:
[[ 1.  2.  8.]
 [ 0. -1. -1.]]
solution of LS with pivoting:
[6.0, 1.0]
```

```
> kernel
Mathematica 13.0.1 Kernel for Linux x86 (64-bit)
Copyright 1988-2022 Wolfram Research, Inc.

In[1]:= m={{1/2^(56),-1},{1,2}};
In[2]:= b={-1,8};
In[3]:= n
Out[3]= {{....., -1}, {1, 2}}
          72057594037927936
In[4]:= sol=LinearSolve[m,b];
In[5]:= sol
Out[5]= {{....., .....}
          72057594037927936  12009599006321324
          12009599006321323  12009599006321323}
In[6]:= N[sol,20]
Out[6]= {5.9999999999999998335, 1.0000000000000000833}
In[7]:= m.sol
Out[7]= {-1, 8}
```

¹Til venstre ovenfor er det gitt to forskjellige numeriske løsninger mens til høyre er det gitt den eksakte løsningen. (Merk hvor god løsningen (6.0, 1.0) er i forhold til den eksakte løsningen.)

Merk at den nøyaktige løsningen $x_1 = 6.0$ og $x_2 = 1.0$ på forrige lysark ikke er eksakt!

Python (IEEE754 DP) til venstre og Mathematica® til høyre:¹

```
augmented matrix:
[[ 1.38777878e-17 -1.00000000e+00 -1.00000000e+00]
 [ 1.00000000e+00 2.00000000e+00 8.00000000e+00]]
result of Gauss elimination on augmented matrix:
[[ 1.38777878e-17 -1.00000000e+00 -1.00000000e+00]
 [ 0.00000000e+00 7.20575940e+16 7.20575940e+16]]
solution of LS no pivoting:
[0.0, 1.0]

augmented matrix:
[[ 1.00000000e+00 2.00000000e+00 8.00000000e+00]
 [ 1.38777878e-17 -1.00000000e+00 -1.00000000e+00]]
result of Gauss elimination on augmented matrix:
[[ 1. 2. 8.]
 [ 0. -1. -1.]]
solution of LS with pivoting:
[6.0, 1.0]
```

```
kernel
Mathematica 13.0.1 Kernel for Linux x86 (64-bit)
Copyright 1988-2022 Wolfram Research, Inc.

In[1]:= m={{1/2^(56),-1},{1,2}};
In[2]:= b={-1,8};
In[3]:= n
Out[3]= {{....., -1}, {1, 2}}
72057594037927936

In[4]:= sol=LinearSolve[m,b];
In[5]:= sol
Out[5]= {{....., .....}
12009599006321323 12009599006321323}

In[6]:= N[sol,20]
Out[6]= {5.9999999999999998335, 1.00000000000000000833}

In[7]:= m.sol
Out[7]= {-1, 8}
```

Bruk av python-funksjonene **gausseliminasjon** og **backsub** bekrefter de løsningene vi allerede har funnet for de to ligningssystemene.

¹Til venstre ovenfor er det gitt to forskjellige numeriske løsninger mens til høyre er det gitt den eksakte løsningen. (Merk hvor god løsningen (6.0, 1.0) er i forhold til den eksakte løsningen.)

PA LU FAKTORISERING [S:2.4]

VALG AV PIVOTELEMENTER

Av kolonneelementene, velg det elementet som har størst tallverdi som pivotelement

This example is written in tableau form as

$$\left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ -1 & 0 & -2 & 1 \\ 2 & 2 & 4 & 0 \end{array} \right].$$

Under partial pivoting we compare $|a_{11}| = 1$ with $|a_{21}| = 1$ and $|a_{31}| = 2$, and choose a_{31} for the new pivot. This is achieved through an exchange of rows 1 and 3:

$$\begin{aligned} \left[\begin{array}{ccc|c} 1 & -1 & 3 & -3 \\ -1 & 0 & -2 & 1 \\ 2 & 2 & 4 & 0 \end{array} \right] &\xrightarrow{\substack{\text{exchange row 1} \\ \text{and row 3}}} \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ -1 & 0 & -2 & 1 \\ 1 & -1 & 3 & -3 \end{array} \right] \\ &\xrightarrow{\substack{\text{subtract } -\frac{1}{2} \times \text{row 1} \\ \text{from row 2}}} \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 1 & -1 & 3 & -3 \end{array} \right] \\ &\xrightarrow{\substack{\text{subtract } \frac{1}{2} \times \text{row 1} \\ \text{from row 3}}} \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & -2 & 1 & -3 \end{array} \right]. \end{aligned}$$

Before eliminating column 2 we must compare the current $|a_{22}|$ with the current $|a_{32}|$. Because the latter is larger, we again switch rows:

$$\begin{aligned} \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & 1 & 0 & 1 \\ 0 & -2 & 1 & -3 \end{array} \right] &\xrightarrow{\substack{\text{exchange row 2} \\ \text{and row 3}}} \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & -2 & 1 & -3 \\ 0 & 1 & 0 & 1 \end{array} \right] \\ &\xrightarrow{\substack{\text{subtract } -\frac{1}{2} \times \text{row 2} \\ \text{from row 3}}} \left[\begin{array}{ccc|c} 2 & 2 & 4 & 0 \\ 0 & -2 & 1 & -3 \\ 0 & 0 & \frac{1}{2} & -\frac{1}{2} \end{array} \right]. \end{aligned}$$

IMAT2150: Permutasjonsmatriser- Definisjon og egenskaper

A **permutation matrix** is an $n \times n$ matrix consisting of all zeros, except for a single 1 in every row and column. $\square$

Equivalently, a permutation matrix P is created by applying arbitrary row exchanges to the $n \times n$ identity matrix (or arbitrary column exchanges). For example,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

are the only 2×2 permutation matrices, and

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix},$$
$$\begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

are the six 3×3 permutation matrices.

IMAT2150: Permutasjonsmatriser- Definisjon og egenskaper

A **permutation matrix** is an $n \times n$ matrix consisting of all zeros, except for a single 1 in every row and column. $\square$

Equivalently, a permutation matrix P is created by applying arbitrary row exchanges to the $n \times n$ identity matrix (or arbitrary column exchanges). For example,

$$\begin{bmatrix} 1 & 0 \\ 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}$$

are the only 2×2 permutation matrices, and

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 1 & 0 & 0 \\ 0 & 0 & 1 \end{bmatrix}, \begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}, \\ \begin{bmatrix} 0 & 0 & 1 \\ 0 & 1 & 0 \\ 1 & 0 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}, \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix}$$

are the six 3×3 permutation matrices.

For example, the permutation matrix

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix}$$

formed by exchanging rows 2 and 3 of the identity matrix. Multiplying an arbitrary matrix A on the left with P has the effect of exchanging rows 2 and 3:

$$\begin{bmatrix} 1 & 0 & 0 \\ 0 & 0 & 1 \\ 0 & 1 & 0 \end{bmatrix} \begin{bmatrix} a & b & c \\ d & e & f \\ g & h & i \end{bmatrix} = \begin{bmatrix} a & b & c \\ g & h & i \\ d & e & f \end{bmatrix}.$$

Når vi skal beregne LU faktoriseringen for PA for en gitt matrise A gjennomfører vi følgende trinn:

Hvordan utføre $PA = LU$ -faktorisering

Lage U Utfør Gauss-eliminasjon med delvis pivotering

Lage P For hver radskift i Gauss-eliminasjonen gjør vi samme radskift på identitetsmatrisen.

Lage L For hver addering av rad i multiplisert med c til rad j skriver vi $-c$ istedet for null på plassen (j, i) i matrisen som skal bli U , (NB! merk disse slik at de kan settes lik null til slutt.)

Eksempel:

Finn $PA = LU$ -faktoriseringen til $A = \begin{bmatrix} 1 & 7 & 2 \\ 2 & 1 & 2 \\ 4 & 1 & -1 \end{bmatrix}$.

$$A = \begin{bmatrix} 1 & 7 & 2 \\ 2 & 1 & 2 \\ 4 & 1 & -1 \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ 2 & 1 & 2 \\ 1 & 7 & 2 \end{bmatrix} \begin{matrix} (-\frac{1}{4}) \\ (-\frac{1}{4}) \end{matrix}$$

$$\begin{aligned}
 A &= \begin{bmatrix} 1 & 7 & 2 \\ 2 & 1 & 2 \\ 4 & 1 & -1 \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ 2 & 1 & 2 \\ 1 & 7 & 2 \end{bmatrix} \begin{matrix} (-\frac{1}{2}) \\ (-\frac{1}{4}) \end{matrix} \\
 &\sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \\ \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \end{bmatrix} \begin{matrix} (-\frac{2}{27}) \end{matrix}
 \end{aligned}$$

$$A = \begin{bmatrix} 1 & 7 & 2 \\ 2 & 1 & 2 \\ 4 & 1 & -1 \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ 2 & 1 & 2 \\ 1 & 7 & 2 \end{bmatrix} \begin{matrix} (-\frac{1}{2}) \\ (-\frac{1}{4}) \end{matrix}$$

$$\sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \\ \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \end{bmatrix} \begin{matrix} (-\frac{2}{27}) \end{matrix}$$

$$\sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \\ \frac{1}{2} & \frac{2}{27} & \frac{7}{3} \end{bmatrix} \Rightarrow L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{4} & 1 & 8 \\ \frac{1}{2} & \frac{2}{27} & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} 4 & 1 & -1 \\ 0 & \frac{27}{4} & \frac{9}{4} \\ 0 & 0 & \frac{7}{3} \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \sim \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

$$A = \begin{bmatrix} 1 & 7 & 2 \\ 2 & 1 & 2 \\ 4 & 1 & -1 \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ 2 & 1 & 2 \\ 1 & 7 & 2 \end{bmatrix} \begin{matrix} (-\frac{1}{2}) \\ (-\frac{1}{4}) \end{matrix}$$

$$\sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \end{bmatrix} \sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \\ \frac{1}{2} & \frac{1}{2} & \frac{5}{2} \end{bmatrix} \begin{matrix} (-\frac{2}{27}) \end{matrix}$$

$$\sim \begin{bmatrix} 4 & 1 & -1 \\ \frac{1}{4} & \frac{27}{4} & \frac{9}{4} \\ \frac{1}{2} & \frac{2}{27} & \frac{7}{3} \end{bmatrix} \Rightarrow L = \begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{4} & 1 & 0 \\ \frac{1}{2} & \frac{2}{27} & 1 \end{bmatrix}$$

$$U = \begin{bmatrix} 4 & 1 & -1 \\ 0 & \frac{27}{4} & \frac{9}{4} \\ 0 & 0 & \frac{7}{3} \end{bmatrix}$$

$$P = \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix} \sim \begin{bmatrix} 0 & 0 & 1 \\ 1 & 0 & 0 \\ 0 & 1 & 0 \end{bmatrix}$$

```
A={{1,7,2},{2,1,2},{4,1,-1}};
L={{1,0,0},{1/4,1,0},{1/2,2/27,1}};
U={{4,1,-1},{0,27/4,9/4},{0,0,7/3}};
P={{0,0,1},{1,0,0},{0,1,0}};
diff=P.A-L.U;
test=diff=={{0,0,0},{0,0,0},{0,0,0}};
test//Print; (** gives True **)
```

Bestem $PA = LU$ faktoriseringen av $A = \begin{pmatrix} 2 & 1 & 5 \\ 4 & 4 & -4 \\ 1 & 3 & 1 \end{pmatrix}$.

Bestem $PA = LU$ faktoriseringen av $A = \begin{pmatrix} 2 & 1 & 5 \\ 4 & 4 & -4 \\ 1 & 3 & 1 \end{pmatrix}$.

Med $P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$, $L = \begin{pmatrix} 1 & 0 & 0 \\ 1/4 & 1 & 0 \\ 1/2 & -1/2 & 1 \end{pmatrix}$ og $U = \begin{pmatrix} 4 & 4 & -4 \\ 0 & 2 & 2 \\ 0 & 0 & 8 \end{pmatrix}$

er $PA = LU$.

Bestem $PA = LU$ faktoriseringen av $A = \begin{pmatrix} 2 & 1 & 5 \\ 4 & 4 & -4 \\ 1 & 3 & 1 \end{pmatrix}$.

Med $P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}$, $L = \begin{pmatrix} 1 & 0 & 0 \\ 1/4 & 1 & 0 \\ 1/2 & -1/2 & 1 \end{pmatrix}$ og $U = \begin{pmatrix} 4 & 4 & -4 \\ 0 & 2 & 2 \\ 0 & 0 & 8 \end{pmatrix}$

er $PA = LU$.

Kontroller selv at $PA = LU$!

Et detaljert beskrivelse på hvordan denne faktoriseringen $PA = LU$ oppnås gis på neste lysark.

Vi utelater beviset for at algoritmen fungerer i det generelle tilfellet.

Vi starter med P og A henholdsvis til venstre og høyre ($\longleftrightarrow$ indikerer radbytte).
Pivotelementene nedenfor er uthevet i **blått**.

$$\text{Bestem } PA = LU \text{ faktoriseringen av } A = \begin{pmatrix} 2 & 1 & 5 \\ 4 & 4 & -4 \\ 1 & 3 & 1 \end{pmatrix}.$$

$$\begin{aligned} & \left[\begin{array}{ccc|ccc} 1 & 0 & 0 & 2 & 1 & 5 \\ 0 & 1 & 0 & \textcolor{blue}{4} & 4 & -4 \\ 0 & 0 & 1 & 1 & 3 & 1 \end{array} \right] & R_1 \longleftrightarrow R_2 \\ \Rightarrow & \left[\begin{array}{ccc|ccc} 0 & 1 & 0 & 4 & 4 & -4 \\ 1 & 0 & 0 & 2 & 1 & 5 \\ 0 & 0 & 1 & 1 & 3 & 1 \end{array} \right] & R_2 - \frac{1}{2}R_1 \text{ (}\textcolor{red}{x} = \textcolor{red}{\frac{1}{2}}\text{)} \\ \Rightarrow & \left[\begin{array}{ccc|ccc} 0 & 1 & 0 & 4 & 4 & -4 \\ 1 & 0 & 0 & x & -1 & 7 \\ 0 & 0 & 1 & 1 & 3 & 1 \end{array} \right] & R_3 - \frac{1}{4}R_1 \text{ (}\textcolor{red}{y} = \textcolor{red}{\frac{1}{4}}\text{)} \\ \Rightarrow & \left[\begin{array}{ccc|ccc} 0 & 1 & 0 & 4 & 4 & -4 \\ 1 & 0 & 0 & x & -1 & -7 \\ 0 & 0 & 1 & y & \textcolor{blue}{2} & 2 \end{array} \right] & R_2 \longleftrightarrow R_3 \\ \Rightarrow & \left[\begin{array}{ccc|ccc} 0 & 1 & 0 & 4 & 4 & -4 \\ 0 & 0 & 1 & y & 2 & 2 \\ 1 & 0 & 0 & x & -1 & 7 \end{array} \right] & R_3 + \frac{1}{2}R_2 \text{ (}\textcolor{red}{z} = \textcolor{red}{-\frac{1}{2}}\text{)} \\ \Rightarrow & \left[\begin{array}{ccc|ccc} 0 & 1 & 0 & 4 & 4 & -4 \\ 0 & 0 & 1 & y & 2 & 2 \\ 1 & 0 & 0 & x & z & 8 \end{array} \right] \end{aligned}$$

$$\text{Med } P = \begin{pmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{pmatrix}, L = \begin{pmatrix} 1 & 0 & 0 \\ 1/4 & 1 & 0 \\ 1/2 & -1/2 & 1 \end{pmatrix} \text{ og } U = \begin{pmatrix} 4 & 4 & -4 \\ 0 & 2 & 2 \\ 0 & 0 & 8 \end{pmatrix}$$

Use the PA=LU factorization to solve the system $Ax = b$, where

$$A = \begin{bmatrix} 2 & 1 & 5 \\ 4 & 4 & -4 \\ 1 & 3 & 1 \end{bmatrix}, \quad b = \begin{bmatrix} 5 \\ 0 \\ 6 \end{bmatrix}.$$

The PA=LU factorization is known from (2.22). It remains to complete the two back substitutions.

1. $Lc = Pb$:

$$\begin{bmatrix} 1 & 0 & 0 \\ \frac{1}{4} & 1 & 0 \\ \frac{1}{2} & -\frac{1}{2} & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ c_2 \\ c_3 \end{bmatrix} = \begin{bmatrix} 0 & 1 & 0 \\ 0 & 0 & 1 \\ 1 & 0 & 0 \end{bmatrix} \begin{bmatrix} 5 \\ 0 \\ 6 \end{bmatrix} = \begin{bmatrix} 0 \\ 6 \\ 5 \end{bmatrix}.$$

Starting at the top, we have

$$c_1 = 0$$

$$\frac{1}{4}(0) + c_2 = 6 \Rightarrow c_2 = 6$$

$$\frac{1}{2}(0) - \frac{1}{2}(6) + c_3 = 5 \Rightarrow c_3 = 8.$$

2. $Ux = c$:

$$\begin{bmatrix} 4 & 4 & -4 \\ 0 & 2 & 2 \\ 0 & 0 & 8 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 0 \\ 6 \\ 8 \end{bmatrix}$$

Starting at the bottom,

$$8x_3 = 8 \Rightarrow x_3 = 1$$

$$2x_2 + 2(1) = 6 \Rightarrow x_2 = 2$$

$$4x_1 + 4(2) - 4(1) = 0 \Rightarrow x_1 = -1.$$

Therefore, the solution is $x = [-1, 2, 1]$.

Bruk av python til å løse LS-et fra eksempel 2.17 ovenfor

Merk at biblioteket **scipy** er brukt.

```
1 import numpy as np
2 from scipy.linalg import solve_triangular
3 #SAUER PA = LU Example 2.17
4 a = np.array([[2, 1, 5], [4, 4, -4], [1, 3, 1]])
5 p = np.array([[0, 1, 0], [0, 0, 1], [1, 0, 0]])
6 l = np.array([[1, 0, 0], [0.25, 1, 0], [0.5, -0.5, 1]])
7 l = np.array([[1, 0, 0], [1/4, 1, 0], [1/2, -1/2, 1]])
8 u = np.array([[4, 4, -4], [0, 2, 2], [0, 0, 8]])
9 b = np.array([5, 0, 6])
10 pb=p.dot(b)
11 print("A:",a)
12 print("b:",b)
13 print("P:",p)
14 print("L:",l)
15 print("U:",u)
16 #Lc=pb step
17 c = solve_triangular(l, pb, lower=True)
18 #Ux=c step
19 x = solve_triangular(u, c, lower=False)
20 print("solution of A x = b:")
21 print(x)
22 print("solution check A x = b:")
23 print(a.dot(x)) #should be b as it is
```

Gir igjen løsningen $[-1, 2, 1]$.

Nedenfor antas A å være en kvadratisk matrise.

Observasjoner

- 1) Ikke alle A har en LU faktorisering, som $\begin{pmatrix} 0 & 1 \\ 1 & 1 \end{pmatrix}$,¹ men $\begin{pmatrix} 1 & 1 \\ 0 & 1 \end{pmatrix}$ har.²
- 2) For alle ikke-singulære A (dvs. $\det(A) \neq 0$) finnes det en permutasjonsmatrise P slik at PA har en LU faktorisering.
- 3) Hvis vi møter et pivotelement som er null for en kolonne under delvis pivotering av A , så har ikke LS-et med A som koeffisientmatrise noen løsning.³

¹Se SAUER eksempel 2-9, s-84.

²Hvorfor?

³ A er jo en singulær matrise.

FEILKILDER/FEILFORPLANTNING [S:2.3]

For et lineært ligningssystem $\mathbf{A} \mathbf{x} = \mathbf{b}$ kan feil og unøyaktigheter oppstå pga.

- 1) en dårlig kondisjonert koeffisientmatrise A og/eller såkalt
- 2) "swamping" (oversvømming av sifre).

Feilkilder ved løsning av lineære ligningssystem

For et lineært ligningssystem $\mathbf{A} \mathbf{x} = \mathbf{b}$ kan feil og unøyaktigheter oppstå pga.

- 1) en dårlig kondisjonert koeffisientmatrise A og/eller såkalt
- 2) "swamping" (oversvømming av sifre).

For å diskutere 1. trenger vi noen definisjoner av begrep som normer og kondisjonstall. For såkalte "ill-conditioned" systemer kan løsningen $\mathbf{x}$ være svært sensitiv for inngangsdata.

For et lineært ligningssystem $\mathbf{A} \mathbf{x} = \mathbf{b}$ kan feil og unøyaktigheter oppstå pga.

- 1) en dårlig kondisjonert koeffisientmatrise $\mathbf{A}$ og/eller såkalt
- 2) "swamping" (oversvømming av sifre).

For å diskutere 1. trenger vi noen definisjoner av begrep som normer og kondisjonstall. For såkalte "ill-conditioned" systemer kan løsningen $\mathbf{x}$ være svært sensitiv for inngangsdata.

Utfordringen under 1. ligger i egenskaper til selve koeffisientmatrisa $\mathbf{A}$ til systemet og ikke i løsningsmetoden. Det er derfor viktig å identifisere denne situasjonen før en går i gang med selve løsningen av systemet.

Feilkilder ved løsning av lineære ligningssystem

For et lineært ligningssystem $\mathbf{A} \mathbf{x} = \mathbf{b}$ kan feil og unøyaktigheter oppstå pga.

- 1) en dårlig kondisjonert koeffisientmatrise $\mathbf{A}$ og/eller såkalt
- 2) "swamping" (oversvømming av sifre).

For å diskutere 1. trenger vi noen definisjoner av begrep som normer og kondisjonstall. For såkalte "ill-conditioned" systemer kan løsningen $\mathbf{x}$ være svært sensitiv for inngangsdata.

Utfordringen under 1. ligger i egenskaper til selve koeffisientmatrisa $\mathbf{A}$ til systemet og ikke i løsningsmetoden. Det er derfor viktig å identifisere denne situasjonen før en går i gang med selve løsningen av systemet.

Utfordringen under 2. er allerede behandlet under pivotering, og kan slik som vi har sett adresseres godt under løsningen av ligningssystemet.

For et lineært ligningssystem $\mathbf{A} \mathbf{x} = \mathbf{b}$ kan feil og unøyaktigheter oppstå pga.

- 1) en dårlig kondisjonert koeffisientmatrise $\mathbf{A}$ og/eller såkalt
- 2) "swamping" (oversvømming av sifre).

For å diskutere 1. trenger vi noen definisjoner av begrep som normer og kondisjonstall. For såkalte "ill-conditioned" systemer kan løsningen $\mathbf{x}$ være svært sensitiv for inngangsdata.

Utfordringen under 1. ligger i egenskaper til selve koeffisientmatrisa $\mathbf{A}$ til systemet og ikke i løsningsmetoden. Det er derfor viktig å identifisere denne situasjonen før en går i gang med selve løsningen av systemet.

Utfordringen under 2. er allerede behandlet under pivotering, og kan slik som vi har sett adresseres godt under løsningen av ligningssystemet.

I læreboka er det definert mange forskjellige normer, men vi skal legge hovedvekt på uendelig-normen. Vi skal ved hjelp av normbegrepet utvikle kriterier (kondisjonstall for $\mathbf{A}$) for å måle hvor sterk feilforsterkningen kan være.

For et lineært ligningssystem $Ax = b$ kan feil og unøyaktigheter oppstå pga.

- 1) en dårlig kondisjonert koeffisientmatrise A og/eller såkalt
- 2) "swamping" (oversvømming av sifre).

For å diskutere 1. trenger vi noen definisjoner av begrep som normer og kondisjonstall. For såkalte "ill-conditioned" systemer kan løsningen x være svært sensitiv for inngangsdata.

Utfordringen under 1. ligger i egenskaper til selve koeffisientmatrisa A til systemet og ikke i løsningsmetoden. Det er derfor viktig å identifisere denne situasjonen før en går i gang med selve løsningen av systemet.

Utfordringen under 2. er allerede behandlet under pivotering, og kan slik som vi har sett adresseres godt under løsningen av ligningssystemet.

I læreboka er det definert mange forskjellige normer, men vi skal legge hovedvekt på uendelig-normen. Vi skal ved hjelp av normbegrepet utvikle kriterier (kondisjonstall for A) for å måle hvor sterk feilforsterkningen kan være.

Normer sier noe om størrelsen av elementene til vektorer og matriser.

¹Merk at $||x||_{\max}$ er det samme som $||x||_{\infty}$.

Vektornormer

La vektoren $[x_1, x_2, x_3, \dots, x_n] \in \mathbb{R}^n$ være gitt.

Vi definerer følgende såkalte vektornormer:¹

$$\|x\|_1 = \sum_i |x_i|$$

$$\|x\|_2 = \sqrt{\sum_i |x_i|^2}$$

$$\|x\|_p = \left(\sum_i |x_i|^p \right)^{\frac{1}{p}}$$

$$\|x\|_{\max} = \max_{1 \leq i \leq n} |x_i|$$

¹Merk at $\|x\|_{\max}$ er det samme som $\|x\|_{\infty}$.

Vektornormer

La vektoren $[x_1, x_2, x_3, \dots, x_n] \in \mathbb{R}^n$ være gitt.

Vi definerer følgende såkalte vektornormer:¹

$$\|x\|_1 = \sum_i |x_i|$$

$$\|x\|_2 = \sqrt{\sum_i |x_i|^2}$$

$$\|x\|_p = \left(\sum_i |x_i|^p \right)^{\frac{1}{p}}$$

$$\|x\|_{\max} = \max_{1 \leq i \leq n} |x_i|$$

Matrisenormer

For en $n \times n$ matrise A definerer vi følgende normer:

$$\|A\|_{\infty} = \text{maksimal absolutt radsum}$$

$$\|A\|_1 = \text{maksimal absolutt kolonnesum}$$

$$\|A\|_2 = \text{maksimal absolutt egenverdi}$$

¹Merk at $\|x\|_{\max}$ er det samme som $\|x\|_{\infty}$.

$\|A\|_{\infty}$ = maksimal absolutt radsum

$$\text{Eks: } A = \begin{bmatrix} 1 & -2 & 3 \\ 1 & 7 & 6 \\ 2 & 8 & -3 \end{bmatrix} \rightarrow \begin{aligned} & \rightarrow (1+|-2|)+3 = 6 \\ & \rightarrow 1+7+6 = 14 \\ & \rightarrow 2+8+|-3| = 13 \end{aligned}$$

$$\|A\|_{\infty} = 14$$

 $\|A\|_1$ = maksimal absolutt søylesum

$$A = \begin{bmatrix} -3 & 4 & 3 \\ 2 & -2 & -5 \\ 4 & 1 & 2 \end{bmatrix}$$

$$\begin{array}{ccc} \downarrow & \downarrow & \downarrow \\ 9 & 7 & 10 \end{array}$$

$$\|A\|_1 = 10$$

 $\|A\|_2$ = maksimal absolutt egenverdi til A

$$A = \begin{bmatrix} 1 & 3 \\ 2 & 2 \end{bmatrix} \quad |A - \lambda I| = \begin{vmatrix} 1-\lambda & 3 \\ 2 & 2-\lambda \end{vmatrix} = 0$$

$$= (1-\lambda)(2-\lambda) - 3 \cdot 2 = 2 - 3\lambda + \lambda^2 - 6 = 0$$

$$\lambda^2 - 3\lambda - 4 = 0$$

$$(\lambda - 4)(\lambda + 1) = 0$$

$$\lambda = 4, \lambda = -1$$

Definisjonene her er analoge med de tilsvarende definisjoner i kapittel 1.

I øvingene blir dere bedt om å regne eksempler på disse feiltypene.

Let x_a be an approximate solution of the linear system $Ax = b$. The **residual** is the vector $r = b - Ax_a$. The **backward error** is the norm of the residual $\|b - Ax_a\|_\infty$, and the **forward error** is $\|x - x_a\|_\infty$. $\square$

Denote the residual by $r = b - Ax_a$. The **relative backward error** of system $Ax = b$ is defined to be

$$\frac{\|r\|_\infty}{\|b\|_\infty},$$

and the **relative forward error** is

$$\frac{\|x - x_a\|_\infty}{\|x\|_\infty}.$$

The **error magnification factor** for $Ax = b$ is the ratio of the two, or

$$\text{error magnification factor} = \frac{\text{relative forward error}}{\text{relative backward error}} = \frac{\frac{\|x - x_a\|_\infty}{\|x\|_\infty}}{\frac{\|r\|_\infty}{\|b\|_\infty}}.$$

The **condition number** of a square matrix A , $\text{cond}(A)$, is the maximum possible error magnification factor for solving $Ax = b$, over all right-hand sides b . $\square$

Surprisingly, there is a compact formula for the condition number of a square matrix. Analogous to the norm of a vector, define the **matrix norm** of an $n \times n$ matrix A as

$$\|A\|_{\infty} = \text{maximum absolute row sum}, \quad (2.19)$$

that is, total the absolute values of each row, and assign the maximum of these n numbers to be the norm of A .

The condition number of the $n \times n$ matrix A is

$$\text{cond}(A) = \|A\| \cdot \|A^{-1}\|. \quad \blacksquare$$

The **condition number** of a square matrix A , $\text{cond}(A)$, is the maximum possible error magnification factor for solving $Ax = b$, over all right-hand sides b . $\square$

Surprisingly, there is a compact formula for the condition number of a square matrix. Analogous to the norm of a vector, define the **matrix norm** of an $n \times n$ matrix A as

$$\|A\|_{\infty} = \text{maximum absolute row sum}, \quad (2.19)$$

that is, total the absolute values of each row, and assign the maximum of these n numbers to be the norm of A .

The condition number of the $n \times n$ matrix A is

$$\text{cond}(A) = \|A\| \cdot \|A^{-1}\|. \quad \blacksquare$$

Den siste påstanden ovenfor teorem 2.6 fra SAUER. Vi utelater beviset.

Merk at vi skal maksimum når vi betrakter **alle mulige høyresider** og allikevel ender opp med den enkle formelen for $\text{cond}(A)$ ovenfor!

Matrisenormer

For hver p -norm for vektorer finnes en norm for matriser.

Definisjon:

$$\|A\|_p = \max_v \frac{\|Av\|_p}{\|v\|_p}$$

Hva er en norm (Orienteringsstoff)

En vektornorm $\|\cdot\|$ må tilfredstille:

- $\|\mathbf{x}\| \geq 0$.
- $\|\mathbf{x}\| = 0 \Leftrightarrow \mathbf{x} = \mathbf{0}$.
- $\|a\mathbf{x}\| = |a| \|\mathbf{x}\|$ for alle a og $\mathbf{x}$.
- $\|\mathbf{x} + \mathbf{y}\| \leq \|\mathbf{x}\| + \|\mathbf{y}\|$ (Trekantulikheten).

Eksempel på beregning av kondisjonstall

Bestem kondisjonstallet til $A = \begin{pmatrix} 1 & 1 \\ 1 + \delta & 1 \end{pmatrix}$ for $\delta > 0$.

¹Bruk at at determinanten til A er lik $-\delta$ og at for $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ er $M^{-1} = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} / (ad - bc)$ for determinanten $\lambda = ad - bc$ til M . (Vi har at M er inverterbar hvis og bare hvis $\lambda \neq 0$.)

Eksempel på beregning av kondisjonstall

Bestem kondisjonstallet til $A = \begin{pmatrix} 1 & 1 \\ 1 + \delta & 1 \end{pmatrix}$ for $\delta > 0$.

Vi trenger A^{-1} .

Kondisjonstallet (med hensyn på uendelignormen) er $\|A\|_{\infty} \times \|A^{-1}\|_{\infty}$.

¹Bruk at at determinanten til A er lik $-\delta$ og at for $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ er $M^{-1} = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} / (ad - bc)$ for determinanten $\lambda = ad - bc$ til M . (Vi har at M er inverterbar hvis og bare hvis $\lambda \neq 0$.)

Eksempel på beregning av kondisjonstall

Bestem kondisjonstallet til $A = \begin{pmatrix} 1 & 1 \\ 1 + \delta & 1 \end{pmatrix}$ for $\delta > 0$.

Vi trenger A^{-1} .

Kondisjonstallet (med hensyn på uendelignormen) er $\|A\|_{\infty} \times \|A^{-1}\|_{\infty}$.

Vi har at

$$A^{-1} = \begin{pmatrix} -1/\delta & 1/\delta \\ (1 + \delta)/\delta & -1/\delta \end{pmatrix}, \text{ s\aa } \|A\|_{\infty} = 2 + \delta \text{ og } \|A^{-1}\|_{\infty} =$$

$1 + 2/\delta$. Vis dette!¹

¹Bruk at at determinanten til A er lik $-\delta$ og at for $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ er $M^{-1} = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} / (ad - bc)$ for determinanten $\lambda = ad - bc$ til M . (Vi har at M er inverterbar hvis og bare hvis $\lambda \neq 0$.)

Eksempel på beregning av kondisjonstall

Bestem kondisjonstallet til $A = \begin{pmatrix} 1 & 1 \\ 1 + \delta & 1 \end{pmatrix}$ for $\delta > 0$.

Vi trenger A^{-1} .

Kondisjonstallet (med hensyn på uendelignormen) er $\|A\|_{\infty} \times \|A^{-1}\|_{\infty}$.

Vi har at

$$A^{-1} = \begin{pmatrix} -1/\delta & 1/\delta \\ (1 + \delta)/\delta & -1/\delta \end{pmatrix}, \text{ s\aa } \|A\|_{\infty} = 2 + \delta \text{ og } \|A^{-1}\|_{\infty} = 1 + 2/\delta. \text{ Vis dette!}^1$$

M.a.o. s\aa er $\text{cond}_{\infty} = (\delta + 2)^2/\delta \approx 4/\delta$ hvis $\delta \ll 1$.

Vi skal bruke dette p\aa eksemplet p\aa neste lysark.

¹Bruk at at determinanten til A er lik $-\delta$ og at for $M = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ er $M^{-1} = \begin{pmatrix} d & -b \\ -c & a \end{pmatrix} / (ad - bc)$ for determinanten $\lambda = ad - bc$ til M . (Vi har at M er inverterbar hvis og bare hvis $\lambda \neq 0$.)

Eksempel 2.11 SAUER (spesialtilfelle av forrige eksempel)

Her gir SAUER kondisjonstallet $\approx 4 \times 10^4$ (som vi også får fra forrige lysark¹) og vi kan derfor forvente med IEEE754 DP $16 - 4 = 12$ korrekte signifikante sifre i løsningen av LS.

¹Her er $\delta = 10^{-4}$, så kondisjonstallet etter forrige lysark er tilnærmet lik $4/\delta = 4 \times 10^4$.

Her gir SAUER kondisjonstallet $\approx 4 \times 10^4$ (som vi også får fra forrige lysark¹) og vi kan derfor forvente med IEEE754 DP 16 – 4 = 12 korrekte signifikante sifre i løsningen av LS.

MATLAB (SAUER s. 89) og Python (nedenfor) gir samme resultat.²

```

1 import numpy as np
2 from numpy import linalg as LA
3 A = np.array([[1,1],[1.0001,1]])
4 b=np.array([2.0000, 2.0001])
5 x=LA.solve(A,b)
6 cond=LA.cond(A,np.inf) ### using infinity matrix norm
7 print("A, b ",A,b)
8 print("solution of LS:")
9 print("%.14f" %x[0])
10 print("%.14f" %x[1])
11 print("condition number of A:", cond)
12
13 A, b  [[ 1.      1.    ]
14        [ 1.0001  1.    ]] [ 2.      2.0001]
15 solution of LS:
16 1.000000000000222
17 0.999999999999778
18 condition number of A: 40004.0001

```

¹Her er $\delta = 10^{-4}$, så kondisjonstallet etter forrige lysark er tilnærmet lik $4/\delta = 4 \times 10^4$.

²Eksakt løsning av LS-et med koeffisientmatrise A og høyreside $(2, 2 + \delta)^T$ er $x_1 = x_2 = 1$.

Hilbert-matrisa er den alltid inverterbare $n \times n$ matrisa $H = (h_{ij})$ der $h_{ij} = 1/(i + j - 1)$ for $j = 1 \cdots (1) \cdots n$.

Kondisjonstallet for H vokser svært fort med økende n .

Hilbert-matrisa er den alltid inverterbare $n \times n$ matrisa $H = (h_{ij})$ der $h_{ij} = 1/(i + j - 1)$ for $j = 1 \cdots (1) \cdots n$.

Kondisjonstallet for H vokser svært fort med økende n .

Anta at vi har et LS med koeffisientmatrise H og en venstreside valgt slik at LS-et har den eksakte løsningen er $x_1 = 1, x_2 = 1, \cdots, x_n = 1$.¹

¹Velg venstresiden lik $H \times [1, \cdots, 1]^T$.

Hilbert-matrisa er den alltid inverterbare $n \times n$ matrisa $H = (h_{ij})$ der $h_{ij} = 1/(i + j - 1)$ for $j = 1 \cdots (1) \cdots n$.

Kondisjonstallet for H vokser svært fort med økende n .

Anta at vi har et LS med koeffisientmatrise H og en venstreside valgt slik at LS-et har den eksakte løsningen $x_1 = 1, x_2 = 1, \dots, x_n = 1$.¹

Allerede for $n = 13$ vil vi da ikke kunne forvente (garantere) noen korrekte sifre i løsningen med IEEE754 DP presisjon. (Se neste lysark.)

For $n = 10$ kan vi kun forvente 3 korrekt sifre.²

¹Velg venstresiden lik $H \times [1, \dots, 1]^T$.

²For $n = 10$ er $\text{cond}_{\infty}(H) \approx 10^{13}$ og for $n = 13$ er $\text{cond}_{\infty}(H) \approx 10^{17}$.

Hilbert-matrisa er den alltid inverterbare $n \times n$ matrisa $H = (h_{ij})$ der $h_{ij} = 1/(i + j - 1)$ for $j = 1 \cdots (1) \cdots n$.

Kondisjonstallet for H vokser svært fort med økende n .

Anta at vi har et LS med koeffisientmatrise H og en venstreside valgt slik at LS-et har den eksakte løsningen er $x_1 = 1, x_2 = 1, \dots, x_n = 1$.¹

Allerede for $n = 13$ vil vi da ikke kunne forvente (garantere) noen korrekte sifre i løsningen med IEEE754 DP presisjon. (Se neste lysark.)

For $n = 10$ kan vi kun forvente 3 korrekt sifre.²

Det skal sies at H er langt i fra representativ for de koeffisientmatriser vi vil vanligvis støte på, men det er vel å merke seg at LS med dårlig kondisjonerte koeffisientmatriser forekommer, og da er det viktig å være oppmerksom.

¹Velg venstresiden lik $H \times [1, \dots, 1]^T$.

²For $n = 10$ er $\text{cond}_{\infty}(H) \approx 10^{13}$ og for $n = 13$ er $\text{cond}_{\infty}(H) \approx 10^{17}$.

Hilbert-matrisa er den alltid inverterbare $n \times n$ matrisa $H = (h_{ij})$ der $h_{ij} = 1/(i + j - 1)$ for $j = 1 \cdots (1) \cdots n$.

Kondisjonstallet for H vokser svært fort med økende n .

Anta at vi har et LS med koeffisientmatrise H og en venstreside valgt slik at LS-et har den eksakte løsningen $x_1 = 1, x_2 = 1, \dots, x_n = 1$.¹

Allerede for $n = 13$ vil vi da ikke kunne forvente (garantere) noen korrekte sifre i løsningen med IEEE754 DP presisjon. (Se neste lysark.)

For $n = 10$ kan vi kun forvente 3 korrekt sifre.²

Det skal sies at H er langt i fra representativ for de koeffisientmatriser vi vil vanligvis støte på, men det er vel å merke seg at LS med dårlig kondisjonerte koeffisientmatriser forekommer, og da er det viktig å være oppmerksom.

(H er diskutert i eksempel 2.12, s. 89 i SAUER.)

¹Velg venstresiden lik $H \times [1, \dots, 1]^T$.

²For $n = 10$ er $\text{cond}_\infty(H) \approx 10^{13}$ og for $n = 13$ er $\text{cond}_\infty(H) \approx 10^{17}$.

IMAT2150:

Et eksempel for LS $Hx = [1, \dots, 1]^T$

Python-kode og løsninger for LS-et $Hx = [1, \dots, 1]^T$ for $n = 13$ med kondisjonstall $\approx 10^{17}$.

LS-et er løst med IEEE754 DP.

IMAT2150:

Et eksempel for LS $Hx = [1, \dots, 1]^T$

Python-kode og løsninger for LS-et $Hx = [1, \dots, 1]^T$ for $n = 13$ med kondisjonstall $\approx 10^{17}$.

LS-et er løst med IEEE754 DP.

Merk **fraværet** av korrekte sifre i noen av løsningene.
(Eksakt løsning er $x_1 = 1, x_2 = 1, \dots, x_{13} = 1$.)

1	<code>import scipy.linalg</code>	1	7.78165815188e+17
2	<code>import numpy as np</code>	2	13
3	<code>n=13</code>	3	1.000000011716654
4	<code>H=scipy.linalg.hilbert(n)</code>	4	0.99998156061712
5	<code>ones=np.repeat(1,n)</code>	5	1.00071381910135
6	<code>b=np.dot(H,ones)</code>	6	0.98806220069330
7	<code>cond=np.linalg.cond(H,np.inf)</code>	7	1.10766598848194
8	<code>print(cond)</code>	8	0.41358020351546
9	<code>print(n)</code>	9	3.05376789589736
10	<code>x=np.linalg.solve(H,b)</code>	10	-3.77924830434696
11	<code>for i in range(n):</code>	11	8.46721034001142
12	<code>print("%.14f" %x[i])</code>	12	-6.74190866265439
		13	6.10775391634939
		14	-0.94089176797249
		15	1.32331280896201