MA2501, Vårsemestre 2007, Numeriske metoder for lineær algebra

Elena Celledoni

1 Introduksjon

Vi vil approksimere løsningen av lineære systemet av n ligningene og n ukjente:

$$\begin{array}{rcl} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + a_{1,n}x_n &=& b_1 \\ a_{2,1}x_1 + a_{2,2}x_2 + \cdots + a_{2,n}x_n &=& b_2 \\ \vdots & \vdots & \vdots \\ a_{n,1}x_1 + a_{n,2}x_2 + \cdots + a_{n,n}x_n &=& b_n \end{array} \tag{1.1}$$

hvor $(x_1, \cdots, x_n)$ er de ukjente. Gitt $(a_{i,j})_{1\leq i,j\leq n}$, og b_i, $i = 1, \cdots, n$.

Vi kan skrive (1.1) i matrise form ved å definere:

$$A := \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ a_{2,1} & a_{2,2} & \cdots & \cdots & a_{2,n} \\ \vdots & \vdots & \cdots & \cdots & \vdots \\ a_{n,1} & a_{n,2} & \cdots & \cdots & a_{n,n} \end{bmatrix}, \quad x := \begin{bmatrix} x_1 \\ x_2 \\ \vdots \\ x_n \end{bmatrix}, \quad b := \begin{bmatrix} b_1 \\ b_2 \\ \vdots \\ b_n \end{bmatrix}$$

så skriver vi (1.1) i følgende måte:

$$A \cdot x = b. \tag{1.2}$$

Lineære systemer kan løses veldig enkelt når A har en spesiell form, for eksempel når A er en diagonal matrise eller en triangulærmatrise og $a_{i,i} \neq 0$ for $i = 1, \ldots, n$:

- A diagonalmatrise

$$A = \begin{bmatrix} a_{1,1} & 0 & \cdots & \cdots & 0 \\ 0 & a_{2,2} & 0 & \cdots & 0 \\ \vdots & \vdots & \ddots & \cdots & \vdots \\ \vdots & \vdots & \ddots & \ddots & \vdots \\ 0 & 0 & \cdots & \cdots & a_{n,n} \end{bmatrix} \Rightarrow \begin{array}{rcl|rcl} a_{1,1}x_1 & = & b_1 & x_1 & = & \frac{b_1}{a_{1,1}} \\ a_{2,2}x_2 & = & b_2 & x_2 & = & \frac{b_2}{a_{2,2}} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ a_{n,n}x_n & = & b_n & x_n & = & \frac{b_n}{a_{n,n}} \end{array}$$

- A triangulærmatrise

$$A = \begin{bmatrix} a_{1,1} & a_{1,2} & \cdots & \cdots & a_{1,n} \\ 0 & a_{2,2} & a_{2,3} & \cdots & a_{2,n} \\ \vdots & \ddots & \ddots & \ddots & \vdots \\ 0 & \ddots & 0 & a_{n-1,n-1} & a_{n-1,n} \\ 0 & 0 & \cdots & 0 & a_{n,n} \end{bmatrix} \Rightarrow$$

$$\begin{array}{rcl} a_{1,1}x_1 + a_{1,2}x_2 + \cdots + \cdots + a_{1,n}x_n & = & b_1 \\ a_{2,2}x_2 + \cdots + \cdots + a_{2,n}x_n & = & b_2 \\ \cdots \quad \vdots \quad \vdots & & \Rightarrow \\ a_{n-1,n-1}x_{n-1} + a_{n-1,n}x_n & = & b_{n-1} \\ a_{n,n}x_n & = & b_n \end{array}$$

$$\begin{array}{rcl} x_1 & = & \frac{b_1 - \sum_{j=2}^{n} a_{1,j}x_j}{a_{1,1}} \\ x_2 & = & \frac{b_2 - \sum_{j=3}^{n} a_{2,j}x_j}{a_{2,2}} \\ \vdots & \vdots & \vdots \\ x_{n-1} & = & \frac{b_{n-1} - a_{n-1,n}x_n}{a_{n-1,n-1}} \\ x_n & = & \frac{b_n}{a_{n,n}} \end{array} \tag{1.3}$$

Formlene (1.3) kalles innsettingsalgoritmen.

2 Gauss eliminasjon

Gauss eliminasjon er en prosess der man reduserer system (1.1) til et triangulær system med den samme løsningen som (1.1). Som sagt triangulære systemer løses veldig lett ved å bruke formler (1.3).

Teknikken som brukes for å få den triengulære formen er delt i $n-1$ deler:

$$Ax = b \to A^{(1)}x = b^{(1)} \to \cdots \to A^{(k)}x = b^{(k)} \to \cdots \to A^{(n-1)}x = b^{(n-1)}$$

alle systemer $A^{(k)}x = b^{(k)}$ $k = 1, \ldots, n-1$ har den samme løsningen. Den siste, $A^{(n-1)}x = b^{(n-1)}$, er i triangulær formen og den k. er

$$A^{(k)} = \begin{bmatrix} \tilde{a}_{1,1} & \cdots & \tilde{a}_{1,k} & \tilde{a}_{1,k+1} & \cdots & \tilde{a}_{1,n} \\ 0 & \ddots & \vdots & \vdots & \cdots & \vdots \\ \vdots & \ddots & \tilde{a}_{k,k} & \tilde{a}_{k,k+1} & \cdots & \tilde{a}_{k,n} \\ \vdots & \vdots & 0 & \tilde{a}_{k+1,k+1} & \cdots & \tilde{a}_{k+1,n} \\ \vdots & \vdots & \vdots & \vdots & \vdots & \vdots \\ 0 & \cdots & 0 & \tilde{a}_{n,k+1} & \cdots & \tilde{a}_{n,n} \end{bmatrix}.$$

Man finner $A^{(1)}x = b^{(1)}$ fra $Ax = b$ ved å substituere den 2., 3., n. ligningen i $Ax = b$ ved tilsværende lineære kombinasjoner av den 1. ligningen med den 2., 3. o.s.v. For å finne $A^{(2)}x = b^{(2)}$ (og de følgende) den samme prosessen er repetert med å ta hensyn til kolonnene fra 2. til n. og radene fra 2. til n.

2.1 Eksempel

Gitt:

$$\begin{array}{rcl} x_1 + 4x_2 + x_3 &=& 6 \\ 2x_1 - x_2 - 2x_3 &=& 3 \\ x_1 + 3x_2 + 2x_3 &=& 5 \end{array} \qquad A = \begin{bmatrix} 1 & 4 & 1 \\ 2 & -1 & -2 \\ 1 & 3 & 2 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 6 \\ 3 \\ 5 \end{bmatrix} \tag{2.4}$$

vi vil redusere det i triangulær formen.

Vi begynne med å substituere den andre ligningen med en lineær kombinasjon av de første to ligningene, d.v.s. vi substituere
$2x_1 - x_2 - 2x_3 = 3$ med

$$(2x_1 - x_2 - 2x_3) + (-2) \cdot (x_1 + 4x_2 + x_3) = 3 + (-2) \cdot 6, \Rightarrow -9x_2 - 4x_3 = -9.$$

Vi substituere den 3. ligningen med en lineær kombinasjon av den 3. og den 1. ligningen:

$$(x_1+3x_2+2x_3)+(-1)\cdot(x_1+4x_2-x_3)=5+(-1)\cdot 6, \Rightarrow -x_2+x_3=-1.$$

Så får vi en nytt system

$$\begin{array}{rcl} x_1+4x_2+x_3 &=& 6 \\ -9x_2-4x_3 &=& -9 \\ -x_2+x_3 &=& -1 \end{array} \qquad A^{(1)}=\begin{bmatrix} 1 & 4 & 1 \\ 0 & -9 & -4 \\ 0 & -1 & 1 \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=\begin{bmatrix} 6 \\ -9 \\ -1 \end{bmatrix} \tag{2.5}$$

Koeffisienter i lineære kombinasjoner av ligningene er $\begin{bmatrix} 2 \\ 1 \end{bmatrix}$, og de er valgt slik at i det nye systemet får vi den andre og det tredje elementet lik null, på denne måten har vi "eliminert" to koeffisienter i systemet.

Nå jobber vi bare med de to siste ligningene:

$$\begin{array}{rcl} -9x_2-4x_3 &=& -9 \\ -x_2+x_3 &=& -1 \end{array} \tag{2.6}$$

Vi substituere den siste ligning med

$$(-x_2+x_3)+(-\frac{1}{9})\cdot(-9x_2-4x_3)=-1+\left(-\frac{1}{9}\right)\cdot(-9)\Rightarrow \frac{13}{9}x_3=0.$$

Koeffisienten av lineære kombinasjonen er $\frac{1}{9}$.

Til slutt får vi det følgende lineære systemet:

$$\begin{array}{rcl} x_1+4x_2+x_3 &=& 6 \\ -9x_2-4x_3 &=& -9 \\ \frac{13}{9}x_3 &=& 0 \end{array} \qquad A^{(2)}=\begin{bmatrix} 1 & 4 & 1 \\ 0 & -9 & -4 \\ 0 & 0 & \frac{13}{9} \end{bmatrix}\begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix}=\begin{bmatrix} 6 \\ -9 \\ 0 \end{bmatrix} \tag{2.7}$$

og løsning kan beregnes ved å bruke insettingsalgoritmen og man starter fra den siste linningen $x_3=0$, og videre oppover $-9x_2=-9\Rightarrow x_2=1$ og $x_1+4=6\Rightarrow x_1=2$ og man får

$$x=\begin{bmatrix} 2 \\ 1 \\ 0 \end{bmatrix}.$$

Merk nå at for å "eliminere" den første kolonnen brukte vi de to koeffisienter:

$$\begin{bmatrix} 2 \\ 1 \end{bmatrix},$$

og for å "eliminere" den andre kolonnen brukte vi

$$\frac{1}{9}.$$

Ved å bruke disse koeffisientene bygger vi nå en triangulære matrise L:

$$L := \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & \frac{1}{9} & 1 \end{bmatrix}.$$

Hvis fra (2.7) tar vi nå matrisa $U := A^{(2)}$ og beregner vi $L \cdot U$, får vi A tilbake.

Generelt når vi beregner Gauss eliminasjon, beregner vi samtidig en faktorisering av matrisa $A = LU$,hvor L og U er to triangulære matriser.

Dette er veldig praktisk hvis man vil beregne løsningen av to (eller flere) systemer med den samme koeffisienter-matrisen A og forskjellige vektorer b og $\hat{b}$ på høyre side av ligningen. I dette tilfellet kan man bruke den samme faktorisering $LU = A$ to ganger og beregne to forskjellige innsettigsalgoritmer, den første med b og den andre med $\tilde{b}$.

Dette er også brukt for å beregne determinanten til A, siden $\det(A) = \prod_{i=1}^{n} u_{i,i}$, og for å finne A^{-1}. I vårt tilfelle har vi

$$\det(A) = 1 \cdot (-9) \cdot \frac{13}{9} = -13$$

og

$$A^{-1} = \begin{bmatrix} 1 & 4 & 1 \\ 0 & -9 & -4 \\ 0 & 0 & \frac{13}{9} \end{bmatrix}^{-1} \begin{bmatrix} 1 & 0 & 0 \\ 2 & 1 & 0 \\ 1 & \frac{1}{9} & 1 \end{bmatrix}^{-1} = \begin{bmatrix} 1 & \frac{4}{9} & \frac{7}{13} \\ 0 & -\frac{1}{9} & -\frac{4}{13} \\ 0 & 0 & \frac{9}{13} \end{bmatrix} \begin{bmatrix} 1 & 0 & 0 \\ -2 & 1 & 0 \\ -\frac{7}{9} & -\frac{1}{9} & 1 \end{bmatrix}$$

Merk at det er mye lettere å finne den inverse av en triangulære matrise enn av en generell matrise.

2.2 Gauss eliminasjon: generell algoritme

For den generelle matrisen (1.1) har vi:

$$Ax = b \rightarrow A^{(1)}x = b^{(1)}$$

hvis $a_{1,1} \neq 0$

$$\left.\begin{array}{lllllllll} l_{2,1} & := & \frac{a_{2,1}}{a_{1,1}} & a^{(1)}_{2,p} & := & a_{2,p} - l_{2,1}a_{1,p} & b^{(1)}_2 & := & b_2 - l_{2,1}b_1 \\ l_{3,1} & := & \frac{a_{3,1}}{a_{1,1}} & a^{(1)}_{3,p} & := & a_{3,p} - l_{3,1}a_{1,p} & b^{(1)}_3 & := & b_3 - l_{3,1}b_1 \\ \vdots & & & & & & & & \\ l_{n,1} & := & \frac{a_{n,1}}{a_{1,1}} & a^{(1)}_{n,p} & := & a_{n,p} - l_{n,1}a_{1,p} & b^{(1)}_n & := & b_n - l_{n,1}b_1 \end{array}\right\} p = 2, \ldots, n.$$

Nå setter vi

$$A := A^{(1)}, b := b^{(1)}$$

og vi fortsetter med

$$Ax = b \rightarrow A^{(2)}x = b^{(2)}.$$

Hvis $a_{2,2} \neq 0$, får vi, for $j = 3, \ldots, n$,

$$l_{j,2} := \frac{a_{j,2}}{a_{2,2}} \quad a^{(2)}_{j,p} := a_{j,p} - l_{j,2}a_{2,p} \quad b^{(2)}_j := b_j - l_{j,2}b_2 \quad p = 3, \ldots, n,$$

og vi setter

$$A := A^{(2)}, b := b^{(2)}.$$

Generelt for $A^{(k)}$ har vi,

$$A := A^{(k-1)}, b := b^{(k-1)}$$

og

$$Ax = b \rightarrow A^{(k)}x = b^{(k)}$$

med, for $j = k + 1, \ldots n$, hvis $a_{k,k} \neq 0$

$$l_{j,k} := \frac{a_{j,k}}{a_{k,k}} \quad a^{(k)}_{j,p} := a_{j,p} - l_{j,k}a_{k,p} \quad b^{(k)}_j := b_j - l_{j,k}b_k \quad p = k + 1, \ldots, n.$$

Til slutt får vi den følgende algoritmen:

Gauss eliminasjon

For $k = 1, \ldots, n-1$

For $j = k+1, \ldots, n$

$l_{j,k} := \frac{a_{j,k}}{a_{k,k}}$

For $p = k+1, \ldots, n+1$,

$a_{j,p} := a_{j,p} - l_{j,k} a_{k,p}$

End

End

End

2.3 Gauss eliminasjon med partiell pivotering

I Gauss eliminasjon (G-E), som beskrevet i den generell algoritme, må vi alltid dividere for elementet $a_{k,k}$ (pivot elementet). Selv sagt får vi problemer når dette er enten null eller veldig liten i absoluttverdi. Man kan unngå slike problemer ved å bytte ligningene av systemet med hverandre. Prosedyren heter partiell pivotering.

2.4 To eksempler: Gauss eliminasjon med partiell pivotering

La oss ta det følgende systemet:

$$\begin{array}{rcl} x_1 + x_2 + x_3 + x_4 &=& 1 \\ x_1 + x_2 + 2x_3 - x_4 &=& 1 \\ x_1 + 2x_2 - x_3 - x_4 &=& 1 \\ x_1 - x_2 + x_3 - x_4 &=& 1 \end{array} \qquad \begin{bmatrix} 1 & 1 & 1 & 1 \\ 1 & 1 & 2 & -1 \\ 1 & 2 & -1 & -1 \\ 1 & -1 & 1 & -1 \end{bmatrix} \tag{2.8}$$

Først "eliminere" vi den 1. kolonnen ved å subtrahere den 1. ligningen fra de følgende tre. Og får vi

$$\begin{array}{rcl} x_1 + x_2 + x_3 + x_4 &=& 1 \\ x_3 - 2x_4 &=& 0 \\ x_2 - 2x_3 - 2x_4 &=& 0 \\ -2x_2 - 2x_4 &=& 0 \end{array} \qquad \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 0 & 0 & 1 & -2 \\ 0 & 1 & -2 & -2 \\ 0 & -2 & 0 & -2 \end{array}\right] \tag{2.9}$$

nå er det umulig å fortsette fordi i lineære kombinasjon av den 2. ligningen med den 3. det fins ikke noen parameter α som vi kan bruke for å eliminere variabelen x_2

$$(x_2 - 2x_3 - 2x_4) - \alpha \cdot (x_3 - 2x_4) = 0.$$

Det eneste som kan gjøres er a bytte ligninene. Av hensyn til avrundingsfeil er det lurt å bytte den 2. ligningen med denne som har størst koeffisient (i absoluttverdi) for x_2, d.v.s. den 3. Så får vi

$$\begin{array}{rcl} x_1 + x_2 + x_3 + x_4 &=& 1 \\ -2x_2 - 2x_4 &=& 0 \\ x_2 - 2x_3 - 2x_4 &=& 0 \\ x_3 - 2x_4 &=& 0 \end{array} \qquad \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 0 & -2 & 0 & -2 \\ 0 & 1 & -2 & -2 \\ 0 & 0 & 1 & -2 \end{array}\right] \tag{2.10}$$

og nå kan vi fortsette med Gauss eliminasjonen som vanlig. Og vi får:

$$\begin{array}{rcl} x_1 + x_2 + x_3 + x_4 &=& 1 \\ -2x_2 - 2x_4 &=& 0 \\ -2x_3 - 3x_4 &=& 0 \\ x_3 - 2x_4 &=& 0 \end{array} \qquad \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 0 & -2 & 0 & -2 \\ 0 & 0 & -2 & -3 \\ 0 & 0 & 1 & -2 \end{array}\right] \tag{2.11}$$

og til slutt:

$$\begin{array}{rcl} x_1 + x_2 + x_3 + x_4 &=& 1 \\ -2x_2 - 2x_4 &=& 0 \\ -2x_3 - 3x_4 &=& 0 \\ -\frac{7}{2}x_4 &=& 0 \end{array} \qquad \left[\begin{array}{cccc} 1 & 1 & 1 & 1 \\ 0 & -2 & 0 & -2 \\ 0 & 0 & -2 & -3 \\ 0 & 0 & 0 & -\frac{7}{2} \end{array}\right] \tag{2.12}$$

som med insettingsalgoritmen gir løsning

$$x = \left[\begin{array}{c} 1 \\ 0 \\ 0 \\ 0 \end{array}\right].$$

I den andre eksempel gitt:

$$\begin{array}{rcl} x_1 + 3x_2 + 2x_3 &=& 5 \\ 2x_1 - x_2 - 2x_3 &=& 3 \\ x_1 + 4x_2 + x_3 &=& 6 \end{array} \qquad A = \begin{bmatrix} 1 & 3 & 2 \\ 2 & -1 & -2 \\ 1 & 4 & 1 \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \\ x_3 \end{bmatrix} = \begin{bmatrix} 5 \\ 3 \\ 6 \end{bmatrix} \tag{2.13}$$

vi vil redusere det i triangulær formen ved å bruke Gauss-eliminasjon med partiell pivotering.

Først bytter vi ligninger:

$$\begin{array}{rcl} 2x_1 - x_2 - 2x_3 &=& 3 \\ x_1 + 3x_2 + 2x_3 &=& 5 \\ x_1 + 4x_2 + x_3 &=& 6 \end{array} \tag{2.14}$$

så eliminerer vi x_1 fra den første kolonnen,

$$\begin{array}{rcl} 2x_1 - x_2 - 2x_3 &=& 3 \\ 3.5x_2 + 3x_3 &=& 3.5 \\ 4.5x_2 + 2x_3 &=& 4.5 \end{array} \tag{2.15}$$

da bytter vi ligningene en gang til:

$$\begin{array}{rcl} 2x_1 - x_2 - 2x_3 &=& 3 \\ 4.5x_2 + 2x_3 &=& 4.5 \\ 3.5x_2 + 3x_3 &=& 3.5 \end{array} \tag{2.16}$$

og eliminere x_2 fra den siste ligningen,

$$\begin{array}{rcl} 2x_1 - x_2 - 2x_3 &=& 3 \\ 4.5x_2 + 2x_3 &=& 4.5 \\ 1.4444x_3 &=& 0 \end{array} \tag{2.17}$$

med innsettingalgoritme for vi løsningen $x_3 = 0$, $x_2 = 1$, $x_1 = 2$.

2.5 G-E med partiell pivotering: generell algoritme

Generelt man får den følgende algoritme for Gauss eliminasjon med partiell pivotering: først initialiserer man vektoren π (pivot vektor) slik at $\pi_i := i$ for $i = 1, \ldots n$,

Gauss eliminasjon med partiell pivotering

For $k = 1, \ldots, n-1$

$a := |a_{k,k}|$

For $j = k+1, \ldots, n$

if ($a < |a_{j,k}|$)

$a := |a_{j,k}|$

$\pi_k := j$

End

End

if ($\pi_k \neq k$)

$s := \pi_k$

For $p = k, \ldots, n$

$r := a_{k,p}$

$a_{k,p} := a_{s,p}$

$a_{s,p} := r$

End

End

For $j = k+1, \ldots, n$

$l_{j,k} := \frac{a_{j,k}}{a_{k,k}}$

For $p = k+1, \ldots, n+1,$

$a_{j,p} := a_{j,p} - l_{j,k} a_{k,p}$

End

End

End

2.6 Kompleksitet av Gauss-eliminasjon

Man kan bevise at for en $n \times n$ matrise er kompleksitet av Gauss eliminasjon er

$$\frac{1}{3}n^3 - \frac{1}{3}n,$$

operasjoner (addisjoner og multiplikasjoner). For store n er det $\frac{1}{3}n^3$ som dominerer. Vi sier at Gauss-eliminasjon har kompleksitet $\mathcal{O}(n^3)$. Insettingsalgoritmen (beskrevet med (1.3)) er billigere, $\mathcal{O}(n^2)$.

3 Løsning av lineære systemer ved iterasjon

For å approksimere x, løsningen til $Ax = b$, gitt $x^{(0)}$, vil vi konstruere en sekvens av vektorer $x^{(1)}, \ldots, x^{(n)}, \ldots$. En måte å gjøre det på er å bruke fikspunkt iterasjon.

Vi vil forsøke å finne en ekvivalent formulering av $Ax = b$ som fikspunkt ligning. For eksempel:

$$Ax = b \Leftrightarrow x = (I - A)x + b$$

hvor I er $n \times n$ identitets matrise. Gitt x_0, fører dette til iterasjonen:

$$x^{(n+1)} = (I - A)x^{(n)} + b.$$

Generelt, kan man lage en iterasjon i den følgende måte:

- skriv A som en summe av to ledd: $A = M - N$, velg M inverterbar;
- fra $(M - N)x = b$ man får $Mx = Nx + b$ og

$$x = M^{-1}Nx + M^{-1}b;$$

- så gitt x_0 kan man konstruere iterasjonen:

$$x^{(n+1)} = M^{-1}Nx^{(n)} + M^{-1}b.$$

Vanligvis M er valgt slik at M^{-1} er lett å beregne, for eksempel kan M være diagonal eller triangulær.

3.1 Eksempel

Vi skal løse ved fikspunktiterasjon systemet

$$\begin{array}{rcl} x_1 - x_2 & = & 1 \\ -x_1 + 2x_2 & = & -1 \end{array} \quad A := \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix}, \quad \begin{bmatrix} 1 \\ -1 \end{bmatrix}, \tag{3.18}$$

og starte med

$$x^{(0)} = \begin{bmatrix} 0 \\ 0 \end{bmatrix}.$$

Løsningen er

$$x = \begin{bmatrix} 1 \\ 0 \end{bmatrix}.$$

Vi tar

$$M := \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix} \quad N = M - A = \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix}.$$

Vi har $Mx = Nx + b$, og $x = M^{-1}Nx + M^{-1}b$, d.v.s.

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

Ved å bruke verdier av $x^{(0)}$ beregner vi

$$x^{(1)} = M^{-1}Nx^{(0)} + M^{-1}b,$$

$$\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ -\frac{1}{2} \end{bmatrix}.$$

Vi fortsetter $x^{(2)} = M^{-1}Nx^{(1)} + M^{-1}b$:

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

og videre

$$\begin{bmatrix} x_1^{(3)} \\ x_2^{(3)} \end{bmatrix} = \begin{bmatrix} 1 \\ -0.2500 \end{bmatrix}, \quad \begin{bmatrix} x_1^{(4)} \\ x_2^{(4)} \end{bmatrix} = \begin{bmatrix} 0.7500 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} x_1^{(5)} \\ x_2^{(5)} \end{bmatrix} = \begin{bmatrix} 1 \\ -0.1250 \end{bmatrix},$$

$$\begin{bmatrix} x_1^{(6)} \\ x_2^{(6)} \end{bmatrix} = \begin{bmatrix} 0.8750 \\ 0 \end{bmatrix}, \quad \begin{bmatrix} x_1^{(28)} \\ x_2^{(28)} \end{bmatrix} = \begin{bmatrix} 1.0000 \\ -0.0000 \end{bmatrix}.$$

3.2 Eksempel

I systemet (3.18) tar vi nå

$$M := \begin{bmatrix} 1 & 0 \\ -1 & 2 \end{bmatrix} \quad N = M - A = \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix}.$$

og med den samme $x^{(0)}$ beregner vi $x^{(1)} = M^{-1}Nx^{(0)} + M^{-1}b$:

$$\begin{bmatrix} x_1^{(1)} \\ x_2^{(1)} \end{bmatrix} = \begin{bmatrix} 1 & 0 \\ -1 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 1 \\ -1 \end{bmatrix} = \begin{bmatrix} 1 \\ 0 \end{bmatrix},$$

og får vi eksakt løsningen i den 1. iterasjonen.

Generelt tar vi M lik diagonalen til A, d.v.s

$$m_{i,i} = a_{i,i}, \quad i = 1, \ldots, n, \quad m_{i,j} = 0, \quad i \neq j,$$

(hvor $m_{i,j}$ er elementene til M, og $a_{i,j}$ er elementene til A), så har vi en iterasjon metode som heter **Jacobi metode**.

Når tar vi M lik den lavere triangulære delen av A, d.v.s.

$$m_{i,j} = a_{i,j}, \quad i = 1, \ldots, n, \; j = 1, \ldots, i, \quad m_{i,j} = 0, \quad i = 1, \ldots, n, \; j = i{+}1, \ldots, n,$$

så får vi en iterasjon metode som heter **Gauss-Seidel metode**.

Se på de generelle formlene til Jacobi's og Gauss-Seidel's metoder i notat om Newton's metode og tabell 18.2 Kreyszig s. 902.

3.3 Konvergens

Siden løsningen x og iterasjons beregnet verdi $x^{(n)}$ er to vektorer, for å måle hvor nær $x^{(n)}$ er til x må vi bruke et norm: $\|x - x^{(n)}\|$.

Hvis u er en vektor med n komponenter, eksempler av normer av u er:

- $\|u\|_1 := |u_1| + \ldots + |u_n|$, norm-1;
- $\|u\|_{\max} := \max_{1 \leq i \leq n} |u_i|$, max-norm;
- $\|u\|_2 := (|u_1|^2 + \ldots + |u_n|^2)^{\frac{1}{2}} = \left(u^T u\right)^{\frac{1}{2}}$, norm-2.

I konvergensanalyse bruker vi også matrise-norm. Eksempler av matrise-norm til en matrise U (med elementer $u_{i,j}$) er:

- $\|U\|_F = \sqrt{\sum_{i=1}^n \sum_{j=1}^n u_{i,j}^2}$, Frobenius norm;
- $\|U\|_{\max} := \max_i \sum_{j=1}^n |u_{i,j}|$ max-norm;
- $\|U\|_1 := \max_j \sum_{i=1}^n |u_{i,j}|$ norm-1.

For eksemper kan vi beregne norm av

$$U := \begin{bmatrix} 3 & 2 \\ -1 & 0 \end{bmatrix},$$

vi har

$$\begin{array}{rcc} \|U\|_F & = & \sqrt{3^2+2^2+(-1)^2} = \sqrt{14} = 3.74165738677394, \\ \|U\|_{\max} & = & \max\{|3|+|2|, |-1|+|0|\} = 5, \\ \|U\|_1 & = & \max\{|3|+|-1|, |2|+|0|\} = 4. \end{array}$$

La oss skrive iterasjonsmetoden vår i den følgende generelle måten:

$$x^{(n+1)} = M^{-1}Nx^{(n)} + M^{-1}b,$$

og ved å definere $C := M^{-1}N$ og $g := M^{-1}b$ kan vi skrive:

$$x^{(n+1)} = Cx^{(n)} + g. \tag{3.19}$$

Theorem 1 *Hvis det fins et norm $\|\cdot\|$ slik at $\|C\| < 1$ da konvergerer iterasjonen (3.19) for alle $x^{(0)}$.*

3.4 Eksempel

La oss ta

$$B = \begin{bmatrix} 3 & 1 \\ -1 & 2.5 \end{bmatrix}, M = \begin{bmatrix} 3 & 0 \\ 0 & 2.5 \end{bmatrix}, N = \begin{bmatrix} 0 & -1 \\ 1 & 0 \end{bmatrix}$$

da har vi at

$$C_J = M^{-1}N = \begin{bmatrix} 0 & -0.3333 \\ 0.4000 & 0 \end{bmatrix}$$

og $\|C_J\|_F = 0.5207$, $\|C_J\|_1 = \|C_J\|_{\max} = 0.4000$, dermed, gitt f, konvergerer Jacobi metode for $Bx = f$ for hver $x^{(0)}$.

For Gauss-Seidel metode har vi

$$C_{GS} = \begin{bmatrix} 3 & 0 \\ -1 & 2.5 \end{bmatrix}^{-1} \begin{bmatrix} 0 & -1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & -0.3333 \\ 0 & -0.1333 \end{bmatrix}$$

og $\|C_{GS}\|_F = 0.3590$, $\|C_{GS}\|_1 = 0.4667$ og $\|C_{GS}\|_{\max} = 0.3333$, dermed konvergerer Gauss-Seidel metode for $Bx = f$ for hver $x^{(0)}$.

3.5 Eksempel

La oss ta matrisen fra eksempel (3.18):

$$A := \begin{bmatrix} 1 & -1 \\ -1 & 2 \end{bmatrix},$$

vi har verifisert at både Jacobi og Gauss-Seidel konvergerer. For Jacobi metode har vi:

$$C_J = \begin{bmatrix} 1 & 0 \\ 0 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 0 & 1 \\ 1 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ \frac{1}{2} & 0 \end{bmatrix}$$

med normer $\|C_J\|_F \geq 1$ $\|C_J\|_{\max} \geq 1$ $\|C_J\|_1 \geq 1$. For Gauss-Seidel har vi

$$C_{GS} = \begin{bmatrix} 1 & 0 \\ -1 & 2 \end{bmatrix}^{-1} \begin{bmatrix} 0 & 1 \\ 0 & 0 \end{bmatrix} = \begin{bmatrix} 0 & 1 \\ 0 & \frac{1}{2} \end{bmatrix}$$

og $\|C_{GS}\|_F \geq 1$ $\|C_{GS}\|_{\max} \geq 1$ $\|C_{GS}\|_1 \geq 1$.

Her hypotesene av forrige teorem er ikke oppfylt, selv om vi vet fra numeriske beregninger at iterasjonene konvergerer. Vi må bruke en annen teknikk hvis vi vil forkaste (bevise) konvergensen teoretisk.

Husk at et tall reell eller kompleks λ slik at det fins u vektoren slik at

$$Cu = \lambda u$$

kalles egenverdi av C og u kalles egenvektor. La $\sigma(C)$ være mengden av alle egenverdier av C (det fins ikke mer enn n distinkte egenverdier for en $n \times n$ matrise), så kan man definere

$$\rho(C) := \max_{\lambda \in \sigma(C)} |\lambda|$$

$\rho(C)$ heter spektral radius av C og det er en positivt tall som brukes mye for å estimere konvergens av iterasjonsmetoder i lineære systemer.

Man kan bevise:

Theorem 2 *Hvis $\rho(C) < 1$ iterasjonsmetode (3.19) konvergerer for alle $x^{(0)}$.*

(Merk at det holder det motsatt også d.v.s, hvis iterasjonen konvergerer for alle x^0, $\rho(C) < 1$.)

I eksemplet våre vi skal finne egenverdier til C_J og C_{GS}. Merk at

$$Cu = \lambda u \Leftrightarrow (C - \lambda I)u = 0 \Leftrightarrow \det(C - \lambda I) = 0,$$

hvor I er identitet matrise og

$$\det(U) = \det\left(\begin{bmatrix} u_{1,1} & u_{1,2} \\ u_{2,1} & u_{2,2} \end{bmatrix}\right) = u_{1,1} \cdot u_{2,2} - u_{1,2} \cdot u_{2,1}.$$

Så har vi at

$$\det(C_J - \lambda I) = \det\left(\begin{bmatrix} 0 & 1 \\ \frac{1}{2} & 0 \end{bmatrix} - \begin{bmatrix} \lambda & 0 \\ 0 & \lambda \end{bmatrix}\right) = \det\left(\begin{bmatrix} -\lambda & 1 \\ \frac{1}{2} & -\lambda \end{bmatrix}\right) = \lambda^2 - \frac{1}{2},$$

og vi har $\lambda_{1,2}^J = \pm\sqrt{\frac{1}{2}} \approx \pm 0.7071$. Så $\rho(C) = 0.7071$ og dermed $\rho(C) < 1$.

For C_{GS} har vi

$$\det(C_J - \lambda I) = \det\left(\begin{bmatrix} -\lambda & 1 \\ 0 & \frac{1}{2} - \lambda \end{bmatrix}\right) = \lambda\left(\frac{1}{2} - \lambda\right),$$

og vi har $\lambda_1^{GS} = 0$ og $\lambda_2^{GS} = \frac{1}{2}$ og $\rho(C_{GS}) = \frac{1}{2}$.

Grunnen til at Gauss-Seidel metode konvergerer raskere enn Jacobi metode er at $\rho(C_{GS}) < \rho(C_J)$.