Innehåll

1 Introduktion **2**

2 Determinanter **2**

2.1 Determinantens definition 2

2.2 Sarrus regel 5

2.3 2×2 matriser och parallellogram 6

3 Determinanters egenskaper **8**

4 Räkneexempel Determinanter **11**

4.1 Laplaceutveckling 11

4.1.1 Exempel 12

4.2 Gausseliminering 12

4.2.1 Exempel 12

4.3 Bråkfri Triangulering 13

4.3.1 Exempel 14

4.4 Bareiss algoritm 15

4.5 Variant 1 16

4.5.1 3×3 exempel 16

4.5.2 4×4 matris exempel 17

4.6 Variant 2 19

4.6.1 4×4 exempel 20

4.7 Tre fall med Nollor 21

4.7.1 En rad/kolonn med 0 22

4.7.2 Division med 0 22

5 Bevis för Bareiss algoritm **23**

5.1 Bevis intro 23

5.2 Bevis 24

6 Effektivitet och historik **26**

1 Introduktion

Detta är ett arbete på kandidatnivå inom matematik som handlar om olika metoder för att beräkna determinanter med fokus på bråkfria metoder där bland annat Bareiss Algoritm finns med. Arbetet tar upp determinantens definition utifrån Laplaceutvecklingen och räkneexempel med Laplaceutveckling, Gausselimination, Bråkfri triangulering och Bareiss Algoritm. Ett bevis om Bareiss algoritm presenteras där vi visar och förtydligar att Bareiss algoritm fungerar och faktiskt är bråkfri.

2 Determinanter

2.1 Determinantens definition

För kvadratiska matriser kan man beräkna ett tal som kallas determinanten. Determinanten kan skrivas på många olika sätt vilket kan vara ganska förvirrande om man inte har stött på dessa tidigare. De vanligaste skrivsätten av determinanten kan se väldigt olika ut men alla förklarar/beskriver en och samma sak. Här är några exempel på de som är de vanligaste att man stöter på:

$$\begin{vmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{vmatrix}, \left|a_{(ij)}\right|_{(1\leq i,j\leq n)}, |A| \text{ eller } \det A.$$

Determinanten kan defineras på många olika sätt. I detta arbete kommer vi definiera determinanten utifrån Laplaceutveckling efter rad ett vilket var den metod jag stötte på när jag lärde mig om determinanter för första gången [2]. För en 1×1 matris $A = (a)$ defineras determinanten som talet a

$$\det A = a$$

För en 2×2 matris $A = \begin{pmatrix} a & b \\ c & d \end{pmatrix}$ defineras determinanten som talet:

$$\det A = ad - bc$$

Determinanten av en 2×2 matris kan också geometriskt kopplas till arean av en parallellogram då man kan se raderna eller kolonnerna i matrisen som vektorer som spänner upp parallellogramen, hur detta hänger ihop kommer visas senare i kapitlet. När vi ska definiera en 3×3 blir arbetet lite knöligare.
Definitionen för en 3×3 determinant ser ut såhär

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} \\ a_{21} & a_{22} & a_{23} \\ a_{31} & a_{32} & a_{33} \end{pmatrix}$$

$$\det A = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31}$$

För dessa 3×3 matriser finns det en enkel variant för att komma ihåg denna formel. Den metoden kallas för Sarrus regel och är väldigt användbar för att komma ihåg formeln utan att behöva gå tillbaka till definitionen när man ska räkna ut sina determinanter för 3×3 matriser. Sarrus regel återkommer vi till senare i kapitlet då vi ska se hur den fungerar.
Om vi omarrangerar termerna i definitionen ovan så kan vi få ut ett nytt uttryck

$$\begin{aligned} \det A &= +a_{11}(a_{22}a_{33} - a_{23}a_{32}) - a_{12}(a_{21}a_{33} - a_{23}a_{31}) + a_{13}(a_{21}a_{32} - a_{22}a_{31}) \\ &= a_{11}\begin{vmatrix} a_{22} & a_{23} \\ a_{32} & a_{33} \end{vmatrix} - a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{13}\begin{vmatrix} a_{21} & a_{22} \\ a_{31} & a_{32} \end{vmatrix} \end{aligned} \tag{1}$$

Vi kan även skriva om det på detta sätt om vi så önskar:

$$\begin{aligned} \det A &= -a_{12}(a_{21}a_{33} - a_{23}a_{31}) + a_{22}(a_{11}a_{33} - a_{13}a_{31}) - a_{32}(a_{11}a_{23} - a_{13}a_{21}) \\ &= -a_{12}\begin{vmatrix} a_{21} & a_{23} \\ a_{31} & a_{33} \end{vmatrix} + a_{22}\begin{vmatrix} a_{11} & a_{13} \\ a_{31} & a_{33} \end{vmatrix} - a_{32}\begin{vmatrix} a_{11} & a_{13} \\ a_{21} & a_{23} \end{vmatrix} \end{aligned} \tag{2}$$

Det vi nu har gjort kallas för utvecklingen av determinanten efter rad ett i det första fallet och utveckling efter kolonn två i det andra fallet. [2] Vi kan utveckla

efter godtycklig rad eller kolonn vilket kan ge oss många fler omskrivningar men vi nöjer oss med att enbart visa dessa två för stunden.
Tecknena framför talen a_{11},a_{12},a_{13} i faktorisering (1) bestäms efter ett mönster. För att hitta mönstret så följer man platsen enligt beteckningen (ij) som betyder att platsen vi söker finns där rad i möter kolonn j. Detta ges utav $(-1)^{i+j}$ och beroende på om radnummret $i+$ kolonnnummret j blir ett jämt eller udda tal så kommer elementet i raden eller kolonnen man utvecklar efter att multipliceras med ett positiv eller negativ tal $(-1)^{i+j}$. Detsamma gäller för omskrivning (2) men eftersom den börjar utvecklingen efter an annan kolonn så skiljer sig faktoriseringen och tecknena åt. Oavsett vilken rad eller kolonn man utvecklar efter så förändras inte determinanten då man kan se det som att man väljer att göra beräkningarna i en annan ordning men produkten och summan blir densamma. Man kan få en förtydligande bild av hur denna formel fungerar om man skriver ut tecknena i en matris enligt detta mönster:

$$\begin{pmatrix} +1 & -1 & +1 \\ -1 & +1 & -1 \\ +1 & -1 & +1 \end{pmatrix}$$

2×2 matriserna i (1) får man genom att i den ursprungliga 3×3 matrisen stryka rad 1 och kolonn j. Detta kallas Laplaceutveckling eller kofaktorutveckling. Exempel på hur man räknar med Laplace finns under rubrik 3.1.

Nu när vi vet hur vi kan beräkna determinanten för en 2×2 och 3×3 matris, samt hur utveckling efter en rad eller kolonn går till så är vi redo för en godtycklig $n \times n$ matris A. För att definiera determinanten för en godtycklig $n \times n$ matris A använder vi oss av en rekursiv definition. Med det menas att vi utgår ifrån determinanten för en matris av storlek $(n-1) \times (n-1)$ och använder oss av detta i definitionen för determinanten av en $n \times n$ matris.
Antag att

$$A = \begin{pmatrix} a_{11} & a_{12} & a_{13} & \dots & a_{1n} \\ a_{21} & a_{22} & a_{23} & \dots & a_{2n} \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & a_{n3} & \dots & a_{nn} \end{pmatrix}$$

Låt A_{1j} vara matrisen som vi får från A genom att stryka bort rad ett och kolonn j. Då är A_{1j} en mindre matris $(n-1)\times(n-1)$ vars determinant vi redan definierat.
Determinanten för en $n \times n$ matris defineras som

$$\det A = (-1)^{1+1}a_{11}\det A_{11} + (-1)^{1+2}a_{12}\det A_{12} +$$

$$(-1)^{1+3}a_{13}\det A_{13} + \cdots + (-1)^{1+n}a_{1n}\det A_{1n}$$

Detta kan skrivas på ett lite mer formellt sätt med summan

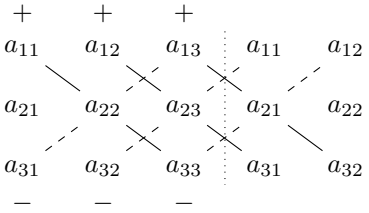
$$\det A = \sum_{j=1}^{n}(-1)^{1+j}a_{1j}\det A_{1j}.$$

Vi kan även utveckla efter en godtycklig rad i eller kolonn j vilket ger samma determinant som att utveckla efter rad 1 t.ex. Nedan ser vi satsen för utvecklingen av en godtycklig rad i. Vi låter A vara en $n \times n$ matris och A_{ij} den determinant man får genom att stryka rad i och kolonn j. $A = (a_{ij})_{1\leq i,j\leq n}$, och vi låter $1 \leq i \leq n$ vara ett heltal, dvs. i är den rad vi väljer att utveckla efter [2].

$$\det A = \sum_{j=1}^{n}(-1)^{i+j}a_{ij}\det A_{ij}$$

2.2 Sarrus regel

Tidigare nämndes Sarrus regel som en snabb metod för att beräkna determinanten på en 3×3 matris utan att behöva memorera eller kolla upp definitionen. Metoden för Sarrus regel är utskriven i figuren nedan och kom ihåg att Sarrus regel fungerar enbart på 3×3 matriser.



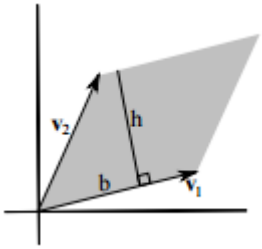
Enligt bilden så expanderar man matrisens ursprungliga tre kolonner till fem genom att lägga till kolonn ett som kolonn fyra och kolonn två som kolonn fem. Sedan multiplicerar man alla element längs ett heldraget streck och adderar det med produkten av det andra heldragna strecket osv. Efter detta subtraherar man med produkten av ett streckat streck med produkten av nästa streckade streck osv. tills man har adderat tre produkter av heldragna streck och subtraherat tre produkter av streckade streck. Detta ser ut såhär

$$\det A = a_{11}a_{22}a_{33} + a_{12}a_{23}a_{31} + a_{13}a_{21}a_{32} - a_{11}a_{23}a_{32} - a_{12}a_{21}a_{33} - a_{13}a_{22}a_{31}$$

och denna känner vi igen sen tidigare ifrån definitionen av en 3×3 matris dvs. att Sarrus regel ger samma resultat som om vi skulle Laplaceutvecklat en 3×3 matris.

2.3 2×2 matriser och parallellogram

Determinanten för 2×2 matriser kan geometriskt kopplas till arean av wn parallellogram som spänns av två vektorer som vi benämner $v_1 = (x_1, y_1)$ och $v_2 = (x_2, y_2)$. Om vi väljer vektorn v_1 som basen så är höjden den vinkelräta linjen från v_1.



[8]

Med hälp av distansformeln kan vi hitta både höjden och basen på parallellogramet:

$$\begin{aligned} b &= \sqrt{(x_1 - 0)^2 + (y_1 - 0)^2} \\ &= \sqrt{x_1^2 + y_1^2} \end{aligned}$$

För att hitta h så blir det en lite längre uträkning, vi börjar med att hitta ekvationen för linjen som går igenom punkterna (x_2, y_2) och $(x_1 + x_2, y_1 + y_2)$:

$$y = \frac{y_1}{x_1}x + y_2 - \frac{y_1 x_2}{x_1}$$

Sedan behöver vi ekvationen för den linje som är vinkelrät mot linjen ovan som går igenom Origo:

$$y = -\frac{x_1}{y_1}x$$

Efter vi har vår vinkelräta linje behöver vi hitta punkten, som vi namnger (x_3, y_3), där de två linjerna korsar varandra:

$$x_3 = \frac{y_1^2 x_2 - y_1 y_2 x_1}{x_1^2 + y_1^2}$$

$$y_3 = \frac{x_1^2 y_2 - y_1 x_2 x_1}{x_1^2 + y_1^2}$$

Sedan använder vi distans formeln för att hitta h:

$$\begin{aligned} h &= \sqrt{(x_3 - 0)^2 + (y_3 - 0)^2} = \\ &= \sqrt{\left[\frac{y_1^2 x_2 - y_1 y_2 x_1}{x_1^2 + y_1^2}\right]^2 + \left[\frac{y_1^2 x_2 - y_1 x_2 x_1}{x_1^2 + y_1^2}\right]^2} = \\ &= \sqrt{\frac{(x_1 y_2 - x_2 y_1)^2}{x_1^2 + y_1^2}} = \\ &= \frac{|x_1 y_2 - x_2 y_1|}{\sqrt{x_1^2 + y_1^2}} \end{aligned}$$

Om vi nu sätter in variablerna b och h i formeln för en parallellogram så kommer vi få ut formeln för arean av en parallellogram med hjälp av vektorerna:

$$\begin{aligned} Area &= bh \\ &= \sqrt{x_1^2 + y_1^2} \cdot \frac{|x_1 y_2 - x_2 y_1|}{\sqrt{x_1^2 + y_1^2}} = \\ &= |x_1 y_2 - x_2 y_1| = Area \end{aligned}$$

$x_1 y_2 - x_2 y_1$ visar sig också vara formeln för determinanten för en 2×2 matris där vektorerna $v_1 = (x_1, y_1)$ och $v_2 = (x_2, y_2)$ är kolonnerna eller raderna i matrisen:

$$A = \begin{pmatrix} x_1 & y_1 \\ x_2 & y_2 \end{pmatrix}$$

$$|\det(A)| = |\det \begin{vmatrix} x_1 & y_1 \\ x_2 & y_2 \end{vmatrix}| = |x_1y_2 - x_2y_1| = Area$$

Absolutbeloppet av determinanten behövs på grund av att om vektorerna byter plats eller placeras annorlunda i matrisen så kan determinanten bli negativ. Detta relaterar till determinanten för en 2×2 som vi tidigare tittade på i definitionen.

3 Determinanters egenskaper

Det finns flera metoder för att beräkna en determinant, med olika fördelar och nackdelar. Laplaceutveckling fungerar väldigt bra som metod men beräkningarna blir snabbt ohanterbara för större matriser. För en 4×4 matris krävs det cirka $4!$ [7] produkter som ska räknas ut. För en 5×5 matris blir det cirka $5!$ multiplikationer dvs. 120 stycken. När vi utvecklar en 5×5 matris med hjälp av Laplaceutveckling får vi fyra stycken 4×4 matriser som vi behöver beräkna determinanten på, fortsätter vi att utveckla så får vi nu 20 stycken 3×3 matriser och efter sista utvecklingen får vi 60 stycken 2×2 matriser att beräkna determinanten på. Enligt Stephen [7] så behövs det $20!$ eller cirka 2.4×10^{18} multiplikationer för att beräkna determinanten på en 20×20 matris. Med en dator som beräknar en miljon multiplikationer per sekund så skulle det ta över 77'000 år för att bestämma det A med Laplaceutveckling. Detta är en extremt lång tid för att beräkna en enda determinant vilket gör det omöjligt att använda just Laplaceutvecklingen för att beräkna determinanten för större matriser och därav behövs andra metoder för att göra dessa beräkningar.

Det finns några knep för att förenkla matriser utan att ändra dess determinant och de kallas för de elementära radoperationerna. De elementära radoperationerna kan användas till att modifiera matriser så att beräkningarna inte blir lika tunga jämfört med om man inte använder dessa. Radoperationerna fungerar så bra att de används i alla utvecklade metoder för determinantberäkningar. De elementära radoperationer som vi använder oss av i detta arbete är de tre som

står listade här [12]:
1. Om två rader eller två kolonner byter plats i en matris A så har den nya matrisen determinanten $-\det A$.
2. När en rad eller kolonn multipliceras med en skalär c i en matris A så blir determinanten för den nya matrisen $c \times \det A$.
3. Om en multipel av en rad eller kolonn adderas till en rad respektive kolonn i en matris A så ändras inte determinantens värde.
Detta kan rätt lätt visas från definitionen men vi utelämnar beviset. Vad är då bra med de elementära radoperationerna? Som tidigare nämnts så kan vi med hjälp av dessa radoperationer göra förändringar i en matris så att vi får en ny enklare matris att beräkna vår determinant på. Det viktigaste med dessa operationer är att de ändrar inte determinantens värde eller ändrar värdet med en känd faktor -1 eller c.

Med hjälp av de tre elementära radoperationerna kan vi modifiera vår matris så att den blir en över- eller undertriangulär matris. När vi har en sådan matris så finns det en genväg som man kan använda sig av och den går till såhär. När man har en över- eller undertriangulär matris så kan man beräkna determinanten genom att multiplicera ihop elementen som ligger på diagonalen i matrisen. Varför fungerar detta då? Om A är en över-/undertriangular $n \times n$ matris så är determinanten produkten av de diagonala elementen i A dvs. $\det A = a_{11}a_{22}a_{33}\dots a_{nn}$. Om vi antar att A är en $n \times n$ övertriangulär matris och använder Laplaceutveckling efter första kolonnen i A så kommer alla tal i den kolonnen vara 0 förutom a_{11}.

$$\det A = \begin{vmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ & a_{22} & \dots & a_{2n} \\ & & \ddots & \vdots \\ & & & a_{nn} \end{vmatrix} = a_{11} \begin{vmatrix} a_{22} & a_{23} & \dots & a_{2n} \\ & a_{33} & \dots & a_{3n} \\ & & \ddots & \vdots \\ & & & a_{nn} \end{vmatrix}$$

Genom att repetera processen till mindre matriser får vi

$$= a_{11} \begin{vmatrix} a_{22} & a_{23} & \dots & a_{2n} \\ & a_{33} & \dots & a_{3n} \\ & & \ddots & \vdots \\ & & & a_{nn} \end{vmatrix} = a_{11}a_{22} \begin{vmatrix} a_{33} & a_{34} & \dots & a_{3n} \\ & a_{44} & \dots & a_{4n} \\ & & \ddots & \vdots \\ & & & a_{nn} \end{vmatrix}$$

$$= a_{11}a_{22}a_{33} \begin{vmatrix} a_{44} & a_{45} & \dots & a_{4n} \\ & a_{55} & \dots & a_{5n} \\ & & \ddots & \vdots \\ & & & a_{nn} \end{vmatrix}$$

$$\vdots$$

$$= a_{11}a_{22}a_{33} \dots a_{nn}.$$

och detta är då vår determinant för matrisen A.
En metod som är väldigt välkänd och använd är Gausselemination som kan användas både för att beräkna determinanter och lösa ekvationssystem. Gausselemination använder sig av de elementära radoperationerna för att reducera matriser till triangulära matriser för att sedan använda sig av diagonalmultiplicering för att beräkna determinanten. Detta är mycket effektivare ($O(n^3)$) än Laplaceutveckling ($O(n!)$) på större matriser [7]. Det stora O:ett eller "Ordo" är ett begrepp inom matematik och datavetenskap som används för att ge ett mått på hur beräkningstung en term är och i datavetenskap används det för att beskriva algoritmers effektivitet, så $O(n^3)$ växer lika fort som n^3 då n ökar.

Vi ska i fortsättningen främst ägna oss åt matriser som enbart innehåller heltal men även om en matris enbart innehåller heltal så kan bråktal ändå uppkomma med hjälp av Gausselemination [8]. Trots att Gausselemination är effektivare så finns det tillfällen då bråktal uppkommer i beräkningarna och datorer precis som vi människor tar längre tid på sig att göra beräkningar då flera räknesätt uppkommer.

Ett exempel på en sådan matris:

$$\begin{pmatrix} 3 & 1 & 7 \\ -2 & 5 & 1 \\ 2 & 1 & 4 \end{pmatrix}$$

Determinanten i denna matris är ett heltal (-17) men bråktal uppkommer i beräk ningarna och detta gör att det tar extra tid för datorn att slutföra beräkningarna. Därför har matematiker hittat metoder som är bråkfria för att minska kraven på datorer så att de kan göra sina beräkningar snabbare. "Bråkfri determinantberäkning är en metod som beräknar determinanter så att alla divisioner som introduceras går jämt ut"[4], notera att detta enbart gäller om matrisen bara innehåller heltal från början. Det finns några metoder just nu som t.ex. bråkfri triangulering, Bareiss algoritm m.fl. som håller på att uvecklas för att optimera datorers beräkningar. Vi återkommer med mer fakta om dem senare.

4 Räkneexempel Determinanter

I detta kapitel finns några exempelberäkningar på hur man beräknar determinanten med olika metoder. Vi utgår ifrån definitionen för laplaceutveckling efter rad ett som finns i kapitel två. Observera att alla beräkningar här görs på bästa möjliga mån på samma matris. Men variationer förekommer.

4.1 Laplaceutveckling

Laplaceutveckling beräknas genom att stryka en godtycklig rad eller kolonn i matrisen. Det vanligaste är att man utvecklar efter rad ett men i vissa fall kan det vara lättare att utveckla efter en annan rad eller någon kolonn. Detaljbeskrivning för hur Laplaceutveckling fungerar finns i determinantens definition för $n \times n$ matriser i kapitel tre.

4.1.1 Exempel

$$\begin{vmatrix} 3 & 2 & 2 \\ 6 & 3 & 5 \\ 1 & 3 & 4 \end{vmatrix} = (-1)^2 \cdot 3 \begin{vmatrix} 3 & 5 \\ 3 & 4 \end{vmatrix} + (-1)^3 \cdot 2 \begin{vmatrix} 6 & 5 \\ 1 & 4 \end{vmatrix} + (-1)^4 \cdot 2 \begin{vmatrix} 6 & 3 \\ 1 & 3 \end{vmatrix} =$$

$$= 3(3 \cdot 4 - 3 \cdot 5) - 2(6 \cdot 4 - 1 \cdot 5) + 2(6 \cdot 3 - 1 \cdot 3)$$

$$= -9 - 38 + 30$$

$$= -17$$

4.2 Gausseliminering

Beräkna determinanten med hjälp av Gausseliminering gör man genom att reducera element i matrisen så att man får en över- eller undertriangulär matris för att sedan lättare kunna beräkna determinanten med hjälp av knepet som står beskrivet i kapitel två för triangulära matriser. Reduceringen gör man genom att använda sig av de elementära radoperationerna som står beskrivna i kapitel tre. Gausseliminering kan också användas till att t.ex. lösa ut linjära ekvationssystem.

4.2.1 Exempel

$$\begin{vmatrix} 3 & 2 & 2 \\ 6 & 3 & 5 \\ 1 & 3 & 4 \end{vmatrix}$$

För att eliminera 6:an i rad två subtraherar vi rad ett ifrån rad två 2 gånger,

$$\begin{vmatrix} 3 & 2 & 2 \\ 0 & -1 & 1 \\ 0 & 3 & 4 \end{vmatrix},$$

sedan subtraheras rad 1 ifrån rad tre 1/3 gånger,

$$\begin{vmatrix} 3 & 2 & 2 \\ 0 & -1 & 1 \\ 0 & 7/3 & 10/3 \end{vmatrix},$$

och slutligen adderas rad 2 till rad tre 7/3 gånger,

$$\begin{vmatrix} 3 & 2 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & 17/3 \end{vmatrix}.$$

Nu när matrisen är övertriangulerad beräknas determinanten genom att multiplicera diagonalernas element med varandra $3 \cdot (-1) \cdot (17/3) = -17$ enligt kapitel tre.

4.3 Bråkfri Triangulering

Nu ska vi titta på Bråkfri triangulering som är ett alternativt sätt att triangulera en matris och som dessutom inte introducerar några bråktal. Denna metod använder sig också av de tre elementära radoperationerna som vi nämnde i kapitel tre och målet är att få en över- eller undertriangulär matris för att utnyttja att $a_{11}a_{22}a_{33}\ldots a_{nm} = \det A$ [8]. För att metoden ska vara bråkfri så kommer vi att använda oss av radoperation två 'När en rad eller kolonn multipliceras med en skalär c' och operation tre 'en multipel av en rad eller kolonn adderas till en rad respektive kolonn'. Vi låter den första matrisen vara $A^{(n)}$ där n är matrisens ordning. Efter steg 1 i algoritmen har vi matrisen $A^{(n-1)}$ osv. Algoritmen slutar då vi har nått matrisen $A^{(1)}$. Den icke bråkfria triangulering går till enligt formeln nedan där vi i steg nr i adderar eller subtraherar olika multiplar av rader för att eliminera det första elementet i kolonnen. Vi kommer att benämna radnummerna efter rad i med k, där $i + 1 \leq k \leq n$, notera att varje hakparentes nedan representerar en rad och $a^i_{k,i}$ är elementet på plats (k, i) och

den i:te beräkningen a^i.

$$\left[a^i_{(k,i)}, a^i_{(k,i+1)}, \ldots, a^i_{(k,n)}\right] \rightarrow$$

$$\left[a^{(i+1)}_{(k,i)}, a^{(i+1)}_{(k,i+1)}, \ldots, a^{(i+1)}_{(k,n)}\right] - \frac{a^{(i+1)}_{(k,i)}}{a^{(i+1)}_{i,i}} \left[a^{(i+1)}_{(i,i)}, a^{(i+1)}_{(i,i+1)}, \ldots, a^{(i+1)}_{(i,n)}\right]$$

Den första delen i subtraktionen är vår rad $i+1$ där det första nollskilda elementet ska elimineras, divisionen som står efter minustecknet är element $(i+1, i)$ dividerat med element (i, i) som oftast krävs för att första elementet på raden ska bli noll. Det här steget i beräkningen är skälet till att bråk kan uppkomma i triangulering. För att förhindra bråktalens uppkomst använder man sig av $d = SGD\left(a^{i+1}_{(i,1)}, a^{i+1}_{(k,i)}\right)$ och då ser bråkfri triangulering ut såhär

$$\left[a^i_{(k,i)}, a^i_{(k,i+1)}, \ldots, a^i_{(k,n)}\right] \rightarrow$$

$$\frac{a^{(i+1)}_{(i,i)}}{d} \left[a^{(i+1)}_{(k,i)}, a^{(i+1)}_{(k,i+1)}, \ldots, a^{(i+1)}_{(k,n)}\right] - \frac{a^{(i+1)}_{(k,i)}}{d} \left[a^{(i+1)}_{(i,i)}, a^{(i+1)}_{(i,i+1)}, \ldots, a^{(i+1)}_{(i,n)}\right]$$

Nu har vi multiplicerat rad $i+1$ med $\frac{a^{(i+1)}_{(i,i)}}{d}$ och eftersom vi gör detta för varje rad $i+1, i+2 \ldots, n$ så behöver vi lagra faktorn $\frac{a^{(i+1)}_{(i,i)}}{d}$ för att sedan kompensera för det vi har lagt till, så i slutet av beräkningen dividerar vi den nya matrisen $A^{(i)}$ med $\frac{a^{(i+1)}_{i,i}}{d}$ för att hitta $\det A$.

4.3.1 Exempel

För att enklare förstå det ovan som kan se ganska komplicerat så tittar vi på ett exempel, detta är vår matris som ska trianguleras.

$$A = \begin{pmatrix} 3 & 2 & 2 \\ 6 & 3 & 5 \\ 1 & 3 & 4 \end{pmatrix}$$

Vi börjar med att multiplicera rad två med 1 och subtrahera rad ett från rad två 2 gånger.

$$rad\ 2 = 1[6\ \ 3\ \ 5] - 2[3\ \ 2\ \ 2]$$

$$= [0 \ -1 \ 1]$$

och för att eliminera rad tre så multiplicerar vi rad tre med 3 och subtraherar rad ett från rad tre 1 gång.

$$rad\ 3 = 3[1 \ 3 \ 4] - 1[3 \ 2 \ 2]$$

$$= [0 \ -1 \ 1]$$

I detta steg så har vi multiplicerat rad två med 1 och rad tre med 3 och behöver därför lagra dessa till slutet av beräkningen för att sedan dividera så att determinantens värde inte ändras. Resultatet av tidigare steg blir

$$A^{(2)} = \begin{pmatrix} 3 & 2 & 2 \\ 0 & -1 & 1 \\ 0 & 7 & 10 \end{pmatrix}$$

där $A^{(2)}$ är den matris som har fått första kolonnen av A triangulariserad.

$$\det A = \frac{\det A^{(2)}}{3 \cdot 1}.$$

För att triangularisera resten av matrisen så multiplicerar vi rad tre med 1 och subtraherar rad tre med rad två -7 gånger. Eftersom SGD mellan 1 och 7 är 1 så behöver inte multipliceringen med 1 skrivas ut då den inte ändrar något i raderna eller värdet på determinanten. Efter detta steg får vi matrisen

$$A^{(1)} = \begin{pmatrix} 3 & 2 & 2 \\ 0 & -1 & 1 \\ 0 & 0 & 17 \end{pmatrix}$$

Nu kan vi beräkna determinanten genom att multiplicera diagonalelementen med varandra och sedan måste vi dividera determinanten med överskotts multipliceringen som vi gjorde under beräkningen

$$\det A = \frac{\det (A^1)}{3 \cdot 1 \cdot 1} = \frac{3 \cdot (-1) \cdot 17}{3} = -17.$$

4.4 Bareiss algoritm

Bareiss Algoritm är en annan bråkfri metod som används inom data för att beräkna determinanter. Bareiss algoritm kan ses som en sofistikerad variant av

radreduktion. Notera att varje steg i Bareiss algoritm är bråkfritt så om något bråk uppkommer i beräkningen har ett fel uppstått.[8]

4.5 Variant 1

Beräkningen enligt Variant 1 använder sig av tre steg. När dessa tre steg är genomförda så är ”ett varv” avklarat. Under det första varvet utgår vi ifrån diagonalelementet a_{11}, det andra varvet blir det elementet a_{22} osv.
De tre stegen i ett varv är:
1. Multiplicera alla rader som ligger under diagonalelementsraden vi utgår från med diagonalelementet i den rad vi utgår från.
2. Addera multiplar av den raden diagonalelementet ligger på till alla rader nedanför så att elementen i kolonnen under diagonalelementet blir 0.
3. Dividera sedan dessa rader med diagonalelementet som vi utgick ifrån i det föregående varvet.
Observera att under första varvet behöver vi inte dividera i steg tre då algoritmen är utformade så att det enbart blir division med 1.

4.5.1 3×3 exempel

Vi börjar med att utgå från det första diagonalelementet i matrisen vilket är a_{11}. I matrisen nedan är $a_{11} = 3$.

$$A^{(3)} = \begin{pmatrix} 3 & 2 & 2 \\ 6 & 3 & 5 \\ 1 & 3 & 4 \end{pmatrix}$$

Alla rader under rad 1 multipliceras med det markerade diagonalelementet.

$$\begin{pmatrix} 3 & 2 & 2 \\ 18 & 9 & 15 \\ 3 & 9 & 12 \end{pmatrix}$$

När vi har multiplicerat alla rader med det första diagonalelementet så ska vi addera multiplar av rad 1 till raderna $2 - 3$ så att alla elementen i kolonn 1

under diagonalelementet blir 0.

$$A^{(2)} = \begin{pmatrix} 3 & 2 & 2 \\ 0 & -3 & 3 \\ 0 & 7 & 10 \end{pmatrix}$$

I det första varvet så behöver vi inte dividera, i och med det så är varv ett klart. I varv två utgår vi ifrån diagonalelementet a_{22}

$$\begin{pmatrix} 3 & 2 & 2 \\ 0 & \text{-3} & 3 \\ 0 & 7 & 10 \end{pmatrix}$$

Vi följer de tre stegen och börjar att multiplicera alla rader under raden som innehåller diagonalelementet på rad två med diagonalelementet på rad två.

$$\begin{pmatrix} 3 & 2 & 2 \\ 0 & \text{-3} & 3 \\ 0 & -21 & -30 \end{pmatrix}$$

Vi adderar ett lämpligt antal multiplar av diagonalelementsraden till alla rader under

$$\begin{pmatrix} 3 & 2 & 2 \\ 0 & \text{-3} & 3 \\ 0 & 0 & -51 \end{pmatrix}$$

och slutligen ska vi dividera rad 3. Vi dividerar rad 3 med diagonalelementet från föregående varv, i detta fall är det $a_{11} = 3$ vilket ger oss

$$A^{(1)} = \begin{pmatrix} 3 & 2 & 2 \\ 0 & -3 & 3 \\ 0 & 0 & -17 \end{pmatrix}$$

där vi nu har $a_{nn}^{(1)} = -17$ vilket är vår determinant.

4.5.2 4×4 matris exempel

$$A^{(4)} = \begin{pmatrix} 3 & 2 & 2 & 1 \\ 6 & 3 & 5 & 2 \\ 1 & 3 & 4 & 3 \\ 1 & 2 & 1 & 1 \end{pmatrix}$$

I första varvet multiplicerar vi rad 2-4 med a_{11} för att sedan addera lämpliga multiplar av rad 1 till rad 2-4.

$$A^{(3)} = \begin{pmatrix} 3 & 2 & 2 & 1 \\ 0 & -3 & 3 & 0 \\ 0 & 7 & 10 & 8 \\ 0 & 4 & 1 & 2 \end{pmatrix}$$

I andra varvet så multiplicerar vi rad 3-4 med det nya diagonalelementet $a_{22} = -3$. Vi adderar sedan de lämpliga multiplarna av rad 1 till rad 3-4 för att få nollor i kolonnen. Sedan dividerar vi rad 3-4 med diagonalelementet i föregående varv dvs. a_{11}

$$A^{(2)} = \begin{pmatrix} 3 & 2 & 2 & 1 \\ 0 & -3 & 3 & 0 \\ 0 & 0 & -17 & -8 \\ 0 & 0 & -5 & -2 \end{pmatrix}$$

Slutligen i varv 3 utgår vi ifrån det nya $a_{33} = -17$ och multiplicerar a_{33} med rad 4. Vi adderar en lämplig multipel av rad 3 till rad 4 och dividerar till sist rad 4 med diagonalelementet i föregående varv 2.

$$A^{(1)} = \begin{pmatrix} 3 & 2 & 2 & 1 \\ 0 & -3 & 3 & 0 \\ 0 & 0 & -17 & -8 \\ 0 & 0 & 0 & 2 \end{pmatrix}$$

och då har vi fått ut vår determinant på plats $a^{(1)}_{nn}$, $\det A = 2$
Ett problem som finns med Bareiss algoritm är att den kan stöta på division med 0 om något av elementen i diagonalen är 0. Vi går igenom ett sådant exempel i slutet av kap 4. på hur man varierar algoritmen med hjälp av radbyten om en nolla uppkommer i diagonalen. I kapitel fem ges även ett bevis för att Bareiss algoritm verkligen ger determinanten och att den är bråkfri.

4.6 Variant 2

Nu ska vi titta på en annan variant av Bareiss algoritm. Denna metod förklarar vi genom att använda oss av upplägget i variant 1. Vad var det vi då gjorde i Variant 1? Vi började med att multiplicera vårt första element a_{11} med raderna $2-4$ så att vi fick detta:

$$\begin{pmatrix} 3 & 2 & 2 & 1 \\ 6 & 3 & 5 & 2 \\ 1 & 3 & 4 & 3 \\ 1 & 2 & 1 & 1 \end{pmatrix} \rightarrow \begin{pmatrix} 3 & 2 & 2 & 1 \\ 3\cdot 6 & 3\cdot 3 & 3\cdot 5 & 3\cdot 2 \\ 3\cdot 1 & 3\cdot 3 & 3\cdot 4 & 3\cdot 3 \\ 3\cdot 1 & 3\cdot 2 & 3\cdot 1 & 3\cdot 1 \end{pmatrix}.$$

Efter att vi multiplicerat med a_{11} så adderar vi våra lämpliga multiplar av rad 1 till rad $2-4$ för att eleminera elementen i kolonn 1.

$$\begin{pmatrix} 3 & 2 & 2 & 1 \\ 3\cdot 6-6\cdot 3 & 3\cdot 3-6\cdot 2 & 3\cdot 5-6\cdot 2 & 3\cdot 2-6\cdot 1 \\ 3\cdot 1-1\cdot 3 & 3\cdot 3-1\cdot 2 & 3\cdot 4-1\cdot 2 & 3\cdot 3-1\cdot 1 \\ 3\cdot 1-1\cdot 3 & 3\cdot 2-1\cdot 2 & 3\cdot 1-1\cdot 2 & 3\cdot 1-1\cdot 1 \end{pmatrix}$$

Om vi nu studerar det ovan markerade elementet $a_{42} = 3\cdot 2-1\cdot 2$ så ser vi att detta motsvarar en 2×2 determinant i vår ursprungliga matris med a_{11} i övre vänstra hörnet och a_{42} i nedre högra hörnet

$$\begin{pmatrix} 3 & 2 & 2 & 1 \\ 6 & 3 & 5 & 2 \\ 1 & 3 & 4 & 3 \\ 1 & 2 & 1 & 1 \end{pmatrix}.$$

Detta stämmer för alla elementen under rad 1 och är grunden till Variant 2 av Bareiss algoritm. Likt variant 1 behöver vi inte dividera under första varvet men under resterande varv så divideras varje element med diagonalelementet som är två steg upp i diagonalen.

4.6.1 4×4 exempel

Vi fortsätter att titta på samma matris som vi började med i 4.6.

$$\left|\begin{array}{cc|ccc} \boxed{3} & 2 & 2 & 1 \\ \hline 6 & 3 & 5 & 2 \\ 1 & 3 & 4 & 3 \\ 1 & 2 & 1 & 1 \end{array}\right|$$

Till varje tal a_{ik} i den avskilda matrisen ovan hör en 2×2-determinant, nämligen den som har 3 i övre vänstra hörnet och a_{ik} i nedre högra. Varje tal a_{ik} skall ersättas av sin determinant. När vi har gjort detta så får vi delmatrisen:

$$\left|\begin{array}{c|ccc} 3 & & & \\ \hline & \boxed{\text{-3}} & 3 & 0 \\ & 7 & 10 & 8 \\ & 4 & 1 & 2 \end{array}\right|$$

Nu har vi -3 längst upp t.v. i delmatrisen. Vi markerar -3 och avdelar en nu ännu mindre matris nedanför och till höger om -3 och utför sedan determinantoperationerna i den nu avdelade matrisen vilket ger oss:

$$\left|\begin{array}{c|ccc} \boxed{3} & & & \\ & -3 & & \\ \hline & & -51 & -24 \\ & & -15 & -6 \end{array}\right|.$$

Eftersom vi nu börjat på andra varvet så behöver vi som tidigare dividera elementen i den avdelade matrisen. Varje element i matrisen divideras med diagonalelementet som är två steg upp i diagonal, i detta fallet är det 3. När vi dividerat får vi matrisen nedan.

$$\left|\begin{array}{c|ccc} 3 & & & \\ & -3 & & \\ \hline & & \boxed{\text{-17}} & -8 \\ & & -5 & -2 \end{array}\right|$$

Algoritmen fortsätter nu på varv tre där den sista determinanten i den avdelade matrisen beräknas

$$\begin{vmatrix} 3 & & & \\ & -3 & & \\ & & -17 & \\ & & & -6 \end{vmatrix}.$$

Det sista elementet ska sedan divideras med -3 vilket är diagonalelementet två steg upp i diagonlen. Detta ger oss

$$\begin{vmatrix} 3 & & & \\ & -3 & & \\ & & -17 & \\ & & & 2 \end{vmatrix}$$

och tvåan som blir längst ner i högra hörnet ($a_{nn}^{(1)}$) i matrisen är då vår determinant.

För fler exempel och möjligheten att se steg för steg hur beräkningarna enligt Bareiss algoritm går till för en valfri matris rekommenderas websidan Matrix Calculator [10].

4.7 Tre fall med Nollor

Vi har tre olika fall i Bareiss algoritm där talet 0 förekommer och kräver extra uppmärksamhet då Bareiss algoritm används. I vårt första fall får vi en hel rad eller kolonn med nollor. I detta fall så vet vi att om en rad eller kolonn består av bara nollor så är determinanten 0. I det andra fallet stöter vi på division med 0. Detta går också att lösa genom att använda sig av en eller flera radbyten. Sedan finns det ett tredje fall där vi inte kan lösa matrisen genom att använda oss av radbyten på grund av att det inte finns någon rad att byta med. I det fallet så blir determinanten 0.

4.7.1 En rad/kolonn med 0

När en rad eller en kolonn blir 0 i en matris så vet vi från tidigare att determinanten blir 0.

$$\det \begin{pmatrix} 2 & 1 & 3 & 4 & 6 \\ 0 & 15 & 7 & 2 & 12 \\ 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & -56 & -91 & -111 \\ 0 & 0 & -80 & -100 & -225 \end{pmatrix} = 0$$

4.7.2 Division med 0

Ibland kan division med 0 uppkomma i Bareiss algoritm. Om man stöter på detta så kan man unvika problemet genom att göra radbyten. Här är ett sådant exempel.

$$A = A^{(4)} = \begin{pmatrix} 1 & -4 & 1 & 2 \\ -1 & 4 & 4 & 1 \\ 3 & 3 & 3 & 4 \\ 2 & 5 & 2 & -1 \end{pmatrix}$$

ett varv ger:

$$A^{(3)} = \begin{pmatrix} 1 & -4 & 1 & 2 \\ 0 & 0 & 5 & 3 \\ 0 & 15 & 0 & -2 \\ 0 & 13 & 0 & -5 \end{pmatrix}$$

Här kommer vi att stöta på division med noll då vi kommer till $A^{(1)}$ då $a_{22}^{(3)}$ är 0. Detta kan vi lösa genom ett enkelt radbyte och genom att komma ihåg att

byta tecken på determinanten i slutet.

$$A^{(3)} = \begin{pmatrix} 1 & -4 & 1 & 2 \\ 0 & 0 & 5 & 3 \\ 0 & 15 & 0 & -2 \\ 0 & 13 & 0 & -5 \end{pmatrix} \; radbyte \rightarrow A'^{(3)} = \begin{pmatrix} 1 & -4 & 1 & 2 \\ 0 & 15 & 0 & -2 \\ 0 & 0 & 5 & 3 \\ 0 & 13 & 0 & -5 \end{pmatrix}$$

$$\rightarrow A'^{(2)} = \begin{pmatrix} \ddots & & & \\ & 75 & 45 & \\ & 0 & -49 & \end{pmatrix}$$

$$\rightarrow A'^{(1)} = \begin{pmatrix} \ddots & & \\ & -245 & \end{pmatrix}$$

$$\rightarrow \det(A) = 245$$

Vid ett fall av matrisen

$$A^{(3)} = \begin{pmatrix} 1 & -4 & 1 & 2 \\ 0 & 0 & 5 & 3 \\ 0 & 0 & 0 & -2 \\ 0 & 0 & 0 & -5 \end{pmatrix}$$

5 Bevis för Bareiss algoritm

5.1 Bevis intro

Detta baseras på vad Bareiss har skrivit i sitt arbete år 1968 om Sylvester's identitet och heltalsbevarande beräkningar[1]. Teorin är att om division med 0 inte förekommer så fungerar algoritmen inom $O(n^3)$ operationer. Beviset vi följer kommer ifrån Leggetts arbete [8] men vi går igenom det mer i detalj och förklarar stegen. Beviset baseras på algoritmen som är skriven nedan av Chee[4]. Chee har skrivit Bareiss algoritm i programeringskod då det är på detta sätt som den oftast stöts på och Chees program stämmer överens med variant 2 av Bareiss algoritm som vi tidigare har tittat på.

Input:

$A \in \mathbb{Z}^{n \cdot n}$, en $n \times n$ matris vars principal-minorer $x_{kk}^{(k)}$ är nollskilda

Output:

det A, som är lagrad i a_{nn}

Do:

-Notera att a_{00} är en speciell variabel

Låt $a_{00} = 1$.

för $k \in \{1, \ldots, n-1\}$ **gör**

för $i \in \{k+1, \ldots, n\}$ **gör**

för $j \in \{k+1, \ldots, n\}$ **gör**

Låt $a_{ij} = \frac{a_{kk} \cdot a_{ij} - a_{ik} \cdot a_{kj}}{a_{k-1,k-1}}$

return a_{nn}

5.2 Bevis

Bareiss algoritm i denna form kräver att alla principal-minorer (determinanten av en mindre kvadrtisk matris som fås av att ta bort rad eller kolonn i matris A) av A är nollskillda för annars kommer vi att stöta på division med 0. Det finns dock lösningar för detta med hjälp av radbyten som vi tidigare visat. För den intresserade så finns det mer om detta i Leggett[8] men för vårt syfte så är alla principal-minorer nollskilda.

Låt A

$$A = \begin{pmatrix} a_{11} & a_{12} & \ldots & a_{1n} \\ a_{21} & a_{22} & \ldots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{n1} & a_{n2} & \ldots & a_{nn} \end{pmatrix}$$

vara så att alla principal-minorer av A är nollskilda. Vi beräknar B från A genom att först multiplicera rad 2 till n med a_{11}. Då har determinanten ökat med en faktor a_{11}^{n-1}. Sedan adderar vi den nödvändiga multipeln av rad ett för

att eliminera $a_{21}, a_{31}, \ldots, a_{n1}$ och då får vi matrisen

$$B = \begin{pmatrix} a_{11} & * * * \\ & A' \end{pmatrix}.$$

Determinanten kan skrivas om på de här olika sätten, notera att notationen $n-1$ syftar till upphöjt och inte principal-minorerna.

$$\det(A) = \frac{1}{a_{11}^{n-1}} \det(B) = \frac{1}{a_{11}^{n-1}} a_{11} \det(A') = \frac{1}{a_{11}^{n-2}} \det(A').$$

Vi beräknar C från A' genom att multiplicera rad två till $n-1$ med a'_{11} och sedan addera en multipel av rad ett för att eliminera $a'_{21}, a'_{31}, \ldots, a'_{n-1,1}$. a'_{ij} står för elementen i den nya matrisen A'. Då får vi matrisen

$$C = \begin{pmatrix} a'_{11} & * * * \\ & A'' \end{pmatrix}.$$

Determinanten kan då skrivas om på de här sätten,

$$\begin{aligned} \det A &= \frac{1}{a_{11}^{n-2}} \cdot \det A' \\ &= \frac{1}{a_{11}^{n-2}} \left(\frac{1}{(a'_{11})^{n-2}} \cdot \det C \right) \\ &= \frac{1}{a_{11}^{n-2}} \left(\frac{1}{(a'_{11})^{n-2}} \cdot a'_{11} \cdot \det A'' \right) \\ &= \frac{1}{a_{11}^{n-2}} \left(\frac{1}{(a'_{11})^{n-3}} \cdot \det A'' \right) \\ &= \frac{1}{(a'_{11})^{n-3}} \left(\frac{1}{(a_{11})^{n-2}} \cdot \det A'' \right) \end{aligned} \tag{3}$$

Från (3) sätter $\left(\frac{1}{(a_{11})^{n-2}} \cdot \det A'' \right) = \det M''$ och då får vi

$$\det A = \frac{1}{(a'_{11})^{n-3}} \det M''$$

där M'' är delmatrisen vi får ifrån Bareiss Algoritm. Efter nästa steg får vi på motsvarande sätt

$$\det A = \frac{1}{(a''_{11})^{n-4}} \det M'''.$$

Efter n steg når man till $M^{(n)}$, det slutliga elementet på plats $a_{(nn)}$. Då fås

$$\det A = \frac{1}{(a_{11}^{n-1})^0} \cdot \det M^{(n)} = \det M^{(n)} = M^{(n)}$$

För att visa att M är bråkfri kollar vi på det j:te elementet i den i:te kolonnen av A'. Ett krav för att det alltid ska bli bråkfria resultat är att alla tal i matrisen är heltal från start.

$$a'_{ij} = a_{11} \cdot a_{i+1,j+1} - a_{i+1,1} \cdot a_{1,j+1}$$

och det motsvarande elementet av A''

$$\begin{aligned}
a''_{ij} &= a'_{11} \cdot a'_{(i+1,j+1)} - a'_{(i+1,1)} \cdot a'_{(1,j+1)} \\
&= (a_{11}a_{22} - a_{21}a_{12})(a_{11}a_{(i+2,j+2)} - a_{(i+2,1)}a_{(1,j+2)}) \\
&- (a_{11}a_{(i+2,2)} - a_{(i+2,1)}a_{12})(a_{11}a_{(2,j+2)} - a_{21}a_{(1,j+2)}) \\
&= a_{11}a_{22}a_{11}a_{(i+2,j+2)} - a_{11}a_{22}a_{(i+2,1)}a_{(1,j+2)} - a_{21}a_{12}a_{11}a_{(i+2,j+2)} \\
&- a_{11}a_{(i+2,2)}a_{11}a_{(2,j+2)} + a_{11}a_{(i+2,2)}a_{21}a_{(1,j+2)} + a_{(i+2,1)}a_{12}a_{11}a_{(2,j+2)}
\end{aligned}$$

Eftersom de enda två termerna som inte innehåller a_{11} tar ut varandra och varje a''_{ij} är dividerbar med a_{11} så är M'' bråkfri. På samma sätt blir övriga matriser $M^{(k)}$ bråkfria vilket bevisar algoritmen. Bareiss använder sig av tre slutna loopar och eftersom varje enskild av dem går upp till $n-1$ beräkningar så tar Bareiss algoritm $O(n^3)$ aritmetiska steg [4]

6 Effektivitet och historik

Bareiss metod baserades på arbetet som Erwin H. Bareiss skrev 1968 med titeln "Sylvester's Identity and Multistep Integer Preserving Gaussian Elimination"[1] där Bareiss utvecklade en metod för att undvika bråktal när man använde sig av Gausseliminering. Beviset använde sig av den s.k. Sylvester's identitet och är därför med i titeln. Bareiss var inte den som uppfann Bareiss algoritm 1968 när han publicerade sitt arbete utan det var René Montante Mario Pardo som år 1973 utvecklade metoden ifrån Bareiss arbete, dvs. 5 år efter Bareiss publicerade sitt arbete. Metoden som Montante då uppfann kom att kallas för Bareiss

algoritm. Namnet Montantes metod förekommer också men det syftar till samma metod. Bareiss algoritm publicerades för första gången 1976 [5]. Det har hänt mycket sen dess med tekniken och nu handlar mycket av arbetet om att optimera algoritmer som Bareiss algoritm, Strassen osv.[9]

Varför krånglar vi med massa bevis och algoritmer då när man vara kan avrunda med 'float' osv. i datorer? Jo för att om man kan undvika att avrunda så kan man undvika fall där tal ligger väldigt nära 0 och när dessa tal uppkommer kan det bli problematist då division med tal nära 0 kan ställa till det. Ett annat problem som kan uppkomma med avrundning i stora beräkningar är att små små avrundningar kan i slutet ha väldigt stora påverkningar på resultatet vilket man uppenbarligen vill undvika så gott som det går.[4]

Gausseleminering och Bareiss algoritm använder sig av $O(n^3)$ beräkningar för att beräkna en determinant av ordning n. Vi kan titta på varför Bareiss algoritm opererar inom $O(n^3)$ operationer. I första steget har vi $(n-1)^2$ stycken 2×2 determinanter som skall beräknas och varje determinant svarar mot två stycken multiplikationer. I nästa steg är det $(n-2)^2$ determinanter osv.
Totalt har vi då:

$$(n-1)^2 \cdot 2 + (n-2)^2 \cdot 2 + \cdots + 2^2 \cdot 2 + 1^2 \cdot 2$$

multiplikationer. För att bestämma detta antal kan vi använda formeln

$$1^2 + 2^2 + \cdots + k^2 = \frac{k(k+1)(2k+1)}{6}$$

Sedan ersätter vi $k = n-1$ och multiplicerar med 2. Vi får då:

$$(n-1)^2 \cdot 2 + (n-2)^2 \cdot 2 + \cdots + 2^2 \cdot 2 + 1^2 \cdot 2 = \frac{n(n-1)(2n-1)}{3} \approx \frac{2n^3}{3}.$$

Detta är de multiplikationer som utförs men vi har även ett antal divisioner, det ger $(n-2)^2$ divisioner osv. Totalt antal divisioner blir:

$$(n-2)^2 + (n-3)^2 + \cdots + 2^2 + 1^2 = \frac{(n+2)(n+3)(2n+6)}{6} \approx \frac{n^3}{3}.$$

I detta skede har cirka $\frac{2n^3}{3}$ multiplikationer och cirka $\frac{n^3}{3}$ divisioner utförts. Totalt blir detta cirka n^3 operationer.

Vid division med 0 så blir Bareiss algoritms beräkningar något längre men den är fortfarande effektiv då man kan stöta på bråktal med många decimaler i Gausseleminationen [4]. Två metoder som är snabbare än Gausselimination och Bareiss algoritm är Strassens algoritm $O(n^{2,807355})$ och Coppersmith Winograds algoritm och det finns fler algoritmer som är ännu snabbare än dessa men de har ingen praktisk nytta ännu då datorer ej kan hantera de stora mängderna med tal[9]. Att jämföra Strassens effektivitet jämtemot $O(n^3)$ metoderna är svårt då Strassen kräver vissa specifika krav och speciell hårdvara för att kunna operera inom $O(n^{2,807355})$. Men tidigare uppskattade man att Strassen's algoritm var effektivare för matriser med ordning mellan 32-128 enheter för optimerade tillämpningar. Dock har en studie från 2010 observerat att inte ens en enstegs beräkning med Strassen är effektivare med vår nuvarande struktur jämfört med de högt optimerade traditionella metoderna. Inte förrän matriserna överstiger ordning 1000 som Strassen har ett övertag. Vid dessa matriser så är effektiviteten marginell då de är endast 10% effektivare i de bästa fallen.
Coppersmith–Winograds algoritm $O(n^{2,37..})$ är den snabbaste algoritmen som vi vet om just nu men är enbart funktionell för matriser som är alldeles för stora för våra nuvarande system. Detta på grund av att den innehåller extremt stora konstanta faktorer som gör det alldeles för svårt för datorerna att hantera.[11]

Varför lägger vi så mycket tid på att hitta lösningar och sätt att beräkna determinanter då, jo för determinanter används t.ex. för att karakterisera matriser och att beskriva explicita lösningar av dess motsvarande linjära ekvationssytem, att avgöra om ett homogent ekvationssystem har icke triviala lösningar, en enda lösning 'den triviala lösningen' eller om ekvationssystem har ingen eller oändligt antal lösningar . Man kan även använda determinanten för att hitta egenvärden för olika matriser och några andra användningsområden är volymer, Jacobian och orienteringen på en bas t.ex. Så determinanten har många användningsområden och är därav av ett intresse att optimera.

Determinanter användes dock långt före matrisealgebran upptäcktes. Ursprungligen var determinanten definerad som en egenskap av linjära ekvationssytem där

determinanten (eng. determines) bestämmer om systemet har en unik lösning. I den meningen så användes determinanter för första gången i en Kinesisk matematik bok "The nine Chapters on the Mathematical Art". Detta var runt tre till två århundraden före Kristus födelse[6]. I Europa pratade Cardano om determinanter för 2×2 matriser i slutet av 1500talet och större matriser nämnde Leibniz. Gauss introducerade namnet "Determinant" år 1801 men använde det mer som en diskriminant (diskriminanten av ett polynom är en funktion av dess koefficienter) istället för vår nutida betydelse av determinanten. 1811/1812 kom nästa stora bidragande faktor ifrån Binet om Cauchy-Binets formel. Binet släppte sin teori kring produkten av två matriser med m kolonner och n rader och specialfallet där $m = n$ till satsen av multiplikationer av matriser. Samtidigt så presenterade Cauchy sitt arbete inom samma område där han använde sig av ordet determinant i sin nutida betydelse. Han förbättrade också notationerna och gav satsen för multiplikationer av matriser ett mer tillfredsställande bevis jämtemot Binets[3].

Referenser

[1] Bareiss, Erwin H, Sylvester's Identity and Multistep Integer Preserving Gaussian Elimination, Mathematics of Computation nr 22 sida 565-578, 1968,

[2] Bøgvad, Rikard och Vaderlind, Paul, Linjär algebra grundkurs, Matematiska institutionen Stockholms universitet, 2014,

[3] Campbell, H, "Linear Algebra With Applications", pages 111–112, Appleton Century Crofts, 1971

[4] Chee-Keng, Yap, Fundamental Problems of Algorithmic Algebra, Oxford University Press, 2000

[5] Enetor-Marzo "Entervista al doctor René Mario Montante Pardo" Ciencia UANL Vol no1. 2002

[6] Eves, H, "An Introduction to the History of Mathematics", pages 405, 493–494, Saunders College Publishing, 1990

[7] Friedberg, Stephen H, A difference equation and operation counts in the computation of determinants, Mathematics Magazine, 61,:295-297, 1988

[8] Leggett, Deanna Richelle, Fraction-Free Methods For Determinants, The University of Southern Mississippi, 2011.

[9] Le Gall, F., "Faster algorithms for rectangular matrix multiplication", Foundations of Computer Science (FOCS 2012), pp. 514–523, 2012

[10] Matrix Caclulator, en online sida för massa olika beräkningar av matriser, https://matrixcalc.org/en/ (2016)

[11] Robinson, Sara, Toward an Optimal Algorithm for Matrix Multiplication, SIAM news, Volume 38, Number 9 2005

[12] Shifrin, Theodore och Adams, Malcolm, Linear Algebra: A Geometric Approach, W. H. Freeman and Company, 2002,