Slump och sannolikheter

JOHAN WÄSTLUND, LINKÖPINGS UNIVERSITET

När man nämner sannolikhetsteori, tänker väl de flesta på metoder för att besvara frågor om just sannolikheter. Med hjälp av statistik och sannolikhetsteoretiska modeller kan man förutsäga komplicerade och till synes slumpmässiga fenomen. Slump förknippas ju med kaos och osäkerhet, och sannolikhetsteorins roll som vetenskap borde då vara att bringa ordning i detta kaos.

Något som jag tycker är fascinerande är att man ibland kan vända upp och ner på det hela och introducera slumpmässighet för att lösa problem som inte från början har att göra med slump och sannolikheter. En av pionjärerna när det gäller sådana *probabilistiska metoder* var ungraren Paul Erdös (1913–1996). På 1940-talet studerade han ett problem inom grafteorin [2]. För att bevisa att det existerar grafer med vissa till synes mycket speciella egenskaper, visade han att en tillräckligt stor slumpmässigt vald graf har dessa egenskaper med sannolikhet större än noll. Detta bevisade indirekt att sådana grafer måste existera, utan att ge ett enda exempel på en sådan graf.

Sedan dess har sådana metoder blivit vanliga både i matematik och datalogi (se till exempel [1]). Inom datorprogrammering använder man till exempel rutinmässigt så kallade *hashtabeller*, där man ibland lagrar data på ett slumpmässigt valt ställe i minnet, för att minimera den tid det tar att sedan komma åt dem. I spelteorin har det visat sig att slumpmässiga strategier kan vara optimala i vissa spel. Att göra någonting på ett slumpmässigt sätt behöver inte vara irrationellt och ogenomtänkt, utan kan tvärtom vara mycket effektivt.

För några år sedan studerade jag och två andra matematiker, Johan Håstad och Svante Linusson, ett geometriskt problem som kan beskrivas så här: Antag att vi låter någon böja till en ståltråd med given längd (och försumbar tjocklek) på godtyckligt sätt. Antag vidare att vi vill kunna stoppa in ståltrådsfiguren i en byrålåda med höjden 1 dm (och tillräckligt stor längd och bredd). Hur lång får ståltråden vara om vi i förväg ska kunna vara säkra på att det går att få in ståltrådsfiguren i byrålådan oavsett hur den ser ut?

Självklart går det bra om ståltråden bara är 1 dm lång. Då kan man ju placera den hur som helst. Det står också klart att tråden kan få vara lite längre än 1 dm. Den som experimenterar lite med en ståltråd som är till exempel 3 dm lång, kommer snart att inse att det aldrig är några problem att få in den i en låda med höjden 1 dm. Hur man än böjer den, kommer den alltid att bli ganska platt i någon riktning. Men hur *bevisar* man att det alltid går? Det duger inte att bara ge några exempel på hur ståltrådsfiguren kan se ut, och hävda att dessa figurer lätt får plats. Det skulle ju kunna finnas något annat sätt att böja tråden, som man inte har tänkt på. Svårigheten ligger i att det finns så många olika sätt att böja en ståltråd i tre dimensioner. Det är svårt att ange en metod för hur figuren ska placeras, som verkligen täcker in alla fall.

Det krävdes en hel del tankearbete för att komma på följande resonemang, som visar att en figur av längd 3 dm alltid går att få in i lådan: Vi tänker oss att ståltråden delas in i tre segment, vardera av längd 1 dm. Vi markerar mittpunkterna på dessa tre segment, dvs. de två punkter som ligger på avstånd 5 cm från respektive ändpunkt, samt mittpunkten på hela tråden. När tråden har böjts till en tredimensionell figur, tänker vi oss en triangel med hörn i de tre markerade punkterna. Eftersom en triangel är en plan figur, kan vi placera ståltrådsfiguren så att de tre markerade punkterna alla ligger på höjden 5 cm. Eftersom varje punkt på ståltråden ligger på avstånd högst 5 cm från någon av de tre markerade punkterna, måste hela figuren då ligga mellan höjd 0 och höjd 10 cm, vilket bevisar att den får plats i lådan.

Detta resonemang är konstruktivt i den meningen att det anger precis hur man ska göra för att placera figuren i lådan. Till vår förvåning kom vi på ett *probabilistiskt* resonemang som ger ett ännu bättre resultat, nämligen att ståltråden kan få ha längden π dm och alltid få plats i lådan. Vi tänker oss att vi hänger upp ståltrådsfiguren i de två ändpunkterna, så att dessa hamnar på samma höjd. Sedan snurrar vi den som ett chokladhjul så att den stannar efter att ha snurrat en slumpmässigt vald vinkel. En beräkning visade sedan att den *genomsnittliga* höjdskillnaden mellan figurens högsta och lägsta punkt då är högst $1/\pi$ gånger ståltrådens längd. Av detta följer att det måste finnas *någon* vinkel för vilken höjdskillnaden blir högst 1 dm. Figuren måste därför få plats i lådan.

Resultatet är till synes paradoxalt. Vi ansträngde oss för att hitta en snillrik metod att placera in ståltrådsfiguren, och så visar det sig att man får ett bättre resultat om man lämnar vissa detaljer åt slumpen! Nu frågar kanske ”vän av ordning” om man alltså måste ha tur för att den här metoden ska fungera. Svaret är att om någon ger dig en ståltrådsfigur av längd π dm och ber dig lägga den i lådan, kan du säkert titta på den och hitta ett bättre sätt än att snurra den slumpmässigt. Vad resonemanget visar är att det alltid går att få in den i lådan på något sätt, och denna slutsats har ingenting med tur eller sannolikheter att göra.

I [3] visade vi ett något bättre resultat, samt en generalisering till högre dimensioner, men den maximala tillåtna längden på tråden är så vitt jag vet fortfarande okänd. Om någon läsare kan bevisa att tråden kan få vara 33 cm lång, vill jag gärna höra om det!

Övningar

Följande problem var uppgift nummer 6 i kvalomgången av Skolornas Matematiktävling den 2 oktober 2002:

1. ”Två cirklar med samma radie klipps ut ur en kartong. På randen till varje cirkel markeras hörnen i en regelbunden $2n$-hörning. Hälften av hörnen på varje cirkel målas gula, hälften målas blå. Cirklarna placeras ovanpå varandra så att hörnen sammanfaller. Vi får då $2n$ par av hörn. Om ett blått hörn på den ena cirkeln sammanfaller med ett blått hörn på den andra cirkeln, eller om ett gult hörn sammanfaller med ett gult, har vi ett matchande par. Visa att cirklarna kan placeras på ett sådant sätt att vi har minst n matchande par.”

Försök inte hitta något strategiskt och väl genomtänkt sätt att placera cirklarna, utan lägg dem bara slumpmässigt på varandra, och beräkna hur många matchande par det blir i genomsnitt!

2. Fundera över hur vi bär oss åt för att välja namn åt våra barn. Ange för- och nackdelar med följande alternativ:

(a) Alla familjer ger sitt äldsta barn namnet Cecilia (oavsett kön), nästa barn namnet Per, och så vidare enligt en bestämd lista.

(b) Man ser till att varje barn får ett unikt namn, genom att låta namnet inkludera information tillräcklig för att identifiera barnet, till exempel ”Hansomföddeskvartöversjudentjugosjättefebruarinittonhundrasjuttioettihudik”.

(c) Man ser till att varje barn får ett unikt namn, genom att kontrollera varje namnförslag, till exempel ”Jojje769” mot en databas över redan upptagna namn.

(d) Man har en lista med 500 tillåtna namn, och varje barn får ett slumpmässigt valt namn ur listan.

Referenser

[1] Noga Alon & Joel Spencer, *The probabilistic method*, Wiley-Interscience 1992.
[2] Paul Erdös, *Some remarks on the theory of graphs*, Bull. Amer. Math. Soc. 53 (1947), 292–294.
[3] Johan Håstad, Svante Linusson & Johan Wästlund, *A smaller sleeping bag for a baby snake*, Discrete Comput. Geom. 26 (2001) 173–181.

Spelteori och sociala kontrakt:

Ska man betala för sig eller smita?

KIMMO ERIKSSON, MÄLARDALENS HÖGSKOLA

På 1940-talet uppfanns *spelteori* av John von Neumann. Spelteori är en gren av matematik där man med matematik beskriver och räknar på socialt samspel. Många olika typer av socialt samspel förekommer bland både människor och djur. Tillämpningar finns därför dels inom vetenskaper som studerar hur människor beter sig i grupp (ekonomi, statsvetenskap, psykologi och sociologi), dels inom evolutionsbiologi där man studerar djurs sociala beteenden och deras uppkomst.

Fångarnas dilemma

Det mest kända exemplet på en spelteoretisk situation är det så kallade *fångarnas dilemma*. Här är spelarna två personer, ”A” och ”B”, som har blivit arresterade som medbrottslingar till ett grovt brott där straffsatsen är upp till tio års fängelse. Polisen har dock än så länge bevisning bara för ett mindre brott där straffsatsen är ett års fängelse. För att locka fram en bekännelse håller polisen brottslingarna åtskilda i var sitt förhörsrum och ger var och en av dem följande erbjudande:

”Om din kompis tjallar på dig så får du tio års fängelse – fast inte om du också tjallar, för då får ni bara fem års fängelse var. Om din kompis inte tjallar på dig så får du ändå ett års fängelse för ett mindre brott – fast inte om du tjallar på henne, för då låter vi dig gå fri!”

Varje brottsling har två *strategier* att välja bland, antingen att tjalla eller att hålla tyst. Sätt dig nu in i den ena brottslingens situation. Vilken strategi bör hon välja om hon vill sitta så lite i fängelse som möjligt? Hon bör naturligtvis tjalla! Det är ju alltid bättre för henne att tjalla, oavsett vilken strategi medbrottslingen väljer.

För att göra situationer med två spelare mer överskådliga brukar man inom spelteori rita upp en matris där man bokför vad som händer vid olika val av strategier. I varje ruta står två tal, de så kallade *utbetalningarna* för spelare A respektive B. I det här fallet består utbetalningen av ett antal år i fängelse.

	B tjallar	*B tjallar inte*
A tjallar	5,5	0,10
A tjallar inte	10,0	1,1

A borde som sagt välja att tjalla. Om B tjallar på henne får A fem år i fängelse (vilket är bättre än tio år) och om B inte tjallar på henne får A noll år i fängelse (vilket är bättre än ett år). Resultatet av att båda tjallar blir att båda får fem år i fängelse, vilket är vad polisen är ute efter. För brottslingarna skulle det ju vara bättre för båda två om ingen av dem tjallade.

För att undvika att hamna i fångarnas dilemma bör medbrottslingar försöka upprätta ett *kontrakt* med varandra i förväg, där de kommer överens om att aldrig tjalla på varandra. Straffet för att bryta kontraktet måste då självfallet vara större än den belöning man får av polisen för att bryta kontraktet. Exempelvis kanske brottslingarna säger till varandra: ”Om du tjallar på mig kommer jag att förfölja dig resten av livet.”

Det kan dock vara svårt att göra sådana kontrakt trovärdiga; när man väl kommer ur fängelset efter tio år har man ofta mer påträngande behov (till exempel av försörjning eller tak över huvudet) än att förfölja den som tjallade. En del av spelteori handlar just om denna svårighet att göra hot trovärdiga, vilket krävs för att upprätta bindande kontrakt.

Betala skatt och andra sociala dilemman

Kärnan i fångarnas dilemma är att varje spelare vinner på att tjalla, men både skulle ändå föredra att ingen tjallar framför att båda tjallar. I min forskargrupp studerar vi *sociala dilemman* vilket är motsvarigheten till fångarnas dilemma när det är många personer inblandade. I ett socialt dilemma kan varje spelare välja mellan att *bidra* eller att *smita*. Att bidra kostar en del för varje individ, men om alla bidrar så skapas ett mervärde som mer än väl uppväger kostnaden.

Ett exempel på ett socialt dilemma är huruvida man ska betala sina skatter till staten. Varje individ skulle kanske helst slippa betala skatt, men om tillräckligt många smiter ifrån skatten så brakar staten ihop (så att centrala funktioner som polis, rättsväsende, utbildningsväsende mm slutar fungera) och då blir det ett sämre samhälle för alla. För att få människor att betala sina skatter används många metoder, såsom *normbildning* (”det är häftigt att betala skatt” sa Mona Sahlin), *konfiskation* (arbetsgivaren betalar in preliminärskatt direkt så att du inte har någon möjlighet att smita), och *straff* för skattesmitning. Av dessa metoder är straff den mest allmängiltiga; i alla sociala dilemman försöker de som bidrar att straffa dem som smiter.

En enkel matematisk modell för ett socialt dilemma med N spelare kan se ut så här:

- Låt oss anta att varje spelare har två strategier att välja emellan: att bidra (kostar b kr) eller att smita (kostar inget).
- Låt oss vidare anta att bidragen skapar ett mervärde som är lika med tio gånger det sammanlagda bidraget. Detta mervärde delas sedan lika på alla spelare. Även de som inte bidragit får vara med och dela.

Först ska vi se när den här modellen verkligen ger ett socialt dilemma. Det kostar b kr att bidra, men i gengäld ökas ju den del man själv kommer att få av det totala mervärdet med $10b/N$. Därmed kan vi räkna ut nettoförlusten av att bidra som differensen $b - 10b/N = b(1 - 10/N)$. När antalet spelare är större än tio blir det alltså en nettoförlust för varje spelare att bidra. Samtidigt är det mycket sämre för varje spelare om ingen bidrar (vinst = 0) än om alla bidrar (vinst = $10b - b = 9b$). När antalet spelare är stort har alltså detta spel den för sociala dilemman karaktäristiska egenskapen att ingen har lust att bidra själv men att alla ändå tycker att alla borde bidra.

Låt oss se hur straff kan påverka dilemmat. En möjlighet är att gruppen tar en del av mervärdet för att inrätta en polismyndighet som straffar smitare, och att straffet för en upptäckt smitare är att polisen konfiskerar smitarens andel av mervärdet.

I vår modell kan vi till exempel säga att 20 procent av mervärdet går åt för att betala polisen, och att polisens effektivitet är sådan att den upptäcker tre av fyra smitare. Det mervärde som alla kan dela på är nu alltså bara 80 procent av det totala mervärdet, och om man smiter är det 75 procents risk att man blir upptäckt och blir av med sin andel.

I denna modell är nu *kostnaden för att bidra* lika med $b - 0{,}80 \cdot 10b/N$, vilket vi kan approximera med b om det finns väldigt många spelare, kanske miljontals. Vi ska jämföra denna kostnad med *kostnaden för att smita*, men den beror på om man blir upptäckt eller inte. Om smitaren inte blir upptäckt är kostnaden förstås noll, men om hon blir upptäckt blir hon av med sin andel av mervärdet – och hur stort mervärdet är beror på hur stor andel av spelarna som *inte* smiter!

Vi inför därför en variabel a för *andelen* spelare som inte smiter, så att aN är *antalet* spelare som inte smiter. Var och en av dessa spelare bidrar med b kr, så det sammanlagda bidraget blir baN och då skapas mervärdet $10baN$. Efter att polisen har tagit sina 20 procent återstår $0{,}80 \cdot 10baN$ och spelarens egen andel av det blir $0{,}80 \cdot 10ba$. Det är 75 procents chans att en smitare blir upptäckt och blir av med sin andel, så i genomsnitt är kostnaden för att smita $0{,}75 \cdot 0{,}80 \cdot 10ba$.

Nå, ska man smita eller inte? Om man bara ser till sin egen vinning ska man smita om det kostar mindre än om man inte smiter, dvs. om

$0{,}75 \cdot 0{,}80 \cdot 10ba < b$, vilket kan förenklas till $a < 1/6$.

Med andra ord, om det är fler än fem sjättedelar av spelarna som smiter så vill jag också smita. Är det fler än en sjättedel som inte smiter så vill jag inte heller smita.

I spelteori talar man om jämvikt när alla spelare har valt en strategi och ingen har lust att byta strategi om inte någon annan byter först. I det sociala dilemmat ovan finns det två jämvikter: om *alla smiter* så kommer ingen att vilja ändra sig och börja bidra istället, och om *ingen smiter* så kommer ingen att vilja ändra sig och börja smita i stället.

Övning

Vissa spel har *blandad jämvikt*, vilket innebär att en viss andel bidrar och resten smiter. Ett sådant spel kan vi få om polisens skicklighet att upptäcka smitare beror på hur många smitare som finns att träna på. Låt oss exempelvis anta att upptäcktsrisken är $0{,}75 \cdot (1 - a)$. I så fall ska man smita om

$$0{,}75 \cdot (1 - a) \cdot 0{,}80 \cdot 10ba < b,$$ vilket kan förenklas till $(1 - a)a < 1/6$.

Förklara varför det i detta spel inte är en jämvikt att alla bidrar, medan vi i stället har en blandad jämvikt där andelen $1/2 \cdot \left(1 + 1/\sqrt{3}\right)$ bidrar, dvs. ungefär 79 procent av spelarna.

Så fel räknar våra datorer

WARWICK TUCKER , UPPSALA UNIVERSITET

Datorer finns överallt runt om oss i dagens samhälle. De används till vitt skilda uppgifter, varav flertalet faller inom ramarna för lagring, visualisering och bearbetning av data. En stor del av dessa uppgifter involverar mer eller mindre komplicerade beräkningar av matematisk natur.

Inom forskningsvärlden är datortillämpningar ofta mycket beräkningsintensiva. Det kan röra sig om datorsimuleringar för molekylstrukturer, galaxhopars utveckling, eller hur en flygplansvinge bäst bör utformas. På senare tid har även finansiella beräkningar blivit väldigt komplicerade. Flertalet börser har numera helt datoriserade, vilket leder till mycket snabba transaktioner som samtidigt skapar ett behov av att ögonblickligen kunna räkna om värdet av aktier och optioner.

I och med den ökade komplexiteten av beräkningarna minskar vår förmåga att överblicka processerna och att avgöra vad som utgör ett rimligt resultat. Med andra ord blir insmugna fel i beräkningarna näst intill omöjliga att upptäcka. Detta kan naturligtvis få ödesdigra konsekvenser.

En vanlig bakomliggande orsak till felberäkningar är det oundvikliga faktumet att våra datorer nästan *alltid* räknar fel. Anledningen till detta är att en dator endast har ett ändligt antal ettor och nollor (s.k. binära siffror eller *bitar*) till sitt förfogande i minnet. Således kan en dator inte exakt representera de flesta tal vi tar för givet i matematiken. Som ett exempel kan nämnas att, trots att alla datorer kan representera talen 1 och 10 exakt, så kan kvoten 1=10 endast lagrar approximativt, fastän det bara ser ut att krävas en decimal. 1/10 = 0,1. Anledningen att just denna kvot orsakar problem är att de flesta datorer använder sig av en *binär* talrepresentation, vilket innebär att siffrorna i talet står för potenser av 2 istället för 10:

$$(110.01)_2 = 1\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 + 0\cdot 2^{-1} + 1\cdot 2^{-2}$$

$$= 4 + 2 + 1/4 = (6.25)_{10}.$$

$$(110.01)_2 = 1\cdot 2^2 + 1\cdot 2^1 + 0\cdot 2^0 + 0\cdot 2^{-1} + 1\cdot 2^{-2}$$

Notera att vi anger vilken bas som avses genom ett index. Vi kan enkelt översätta binära tal till vår decimala form:

Nu är det lätt att se varför datorer med en binär talrepresentation bara kan lagra 1/10 approximativt. Om vi omvandlar talet till binär bas, erhåller vi följande binära utveckling:

$$(0.1)_{10} = (0.0001100110011001100\ldots)_2 .$$

Vi ser att ett tal som har en ändlig representation i en bas kan ha en oändlig representation i en annan bas. Eftersom en dator bara lagrar ett ändligt antal bitar måste vi klippa av talet och går således miste om information. Felet som uppstår kallas för *avrundningsfel*.

Ett av de vanligaste dataformaten som används i dagens datorer består av 64 bitar, varav 53 är avsedda för att lagra den binära utvecklingen av ett tal. Trots att detta resulterar i ett väldigt litet avrundningsfel, finns risken att det fortplantar sig i alla fortsatta beräkningar. T.ex. kan nämnas att, enligt en modern dator, så gäller likheterna

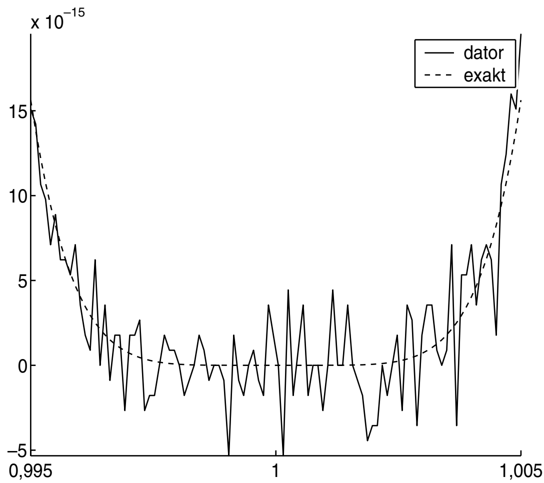
$$\underbrace{0.1+0.1+\ldots+0.1}_{1000\text{ termer}}=99.9999999999998593$$

$$\underbrace{0.1+0.1+\ldots+0.1}_{10000\text{ termer}}=1000.0000000001588$$

Det korrekta svaret i det första fallet vet vi ju är exakt 100, vilket bevisar att datorn räknat fel, omän med ganska lite. I det andra fallet har datorn överskattat resultatet (som skall vara exakt 1000).

Som du kanske redan gissat är det inget speciellt med talet 1/10: samma slags avrundningsfel sker för nästan alla kvoter. Än väarre blir det när man försöker att beräkna mer komplicerade uttryck. I många datorsimuleringar ingår deluppgiften att evaluera polynom. Ett exempel skulle kunna vara att, givet ett tal x, beräkna

$$p(x)=x^6-6x^5+15x^4-20x^3+15x^2-6x+1$$



Figur 1: Avrundningsfel för ett polynom.

Om man beräknar värden av $p(x)$ för olika x-värden kan man konstruera en graf som ger en ”bild” av polynomet. I figuren som visas ser vi hur fel det kan bli!

Den matematiskt korrekta bilden av polynomet ges av den streckade grafen, medan datorns bild ges av den heldragna. Skillnaden är slående: den streckade grafen är slät medan den heldragna oscillerar vilt. Dessa diskrepanser kan naturligtvis leda till allvarliga fel, och ju större uträkningarna är desto svårare blir felen att hitta.

För en matematiker är det extra viktigt att hålla reda på avrundningsfelen i en datorberäkning. När det gäller matematiskt stringenta bevis så får inget lämnas åt slumpen! Därför har man uppfunnit ett sätt att räkna med ”osäkra” tal. Ett sådant tal kan man tänka på som ett intervall, vars bredd motsvarar graden av osäkerhet. Men hur räknar man med intervall? Det visar sig vara ganska enkelt: Låt oss anta notationen $\left[a\right]=\left[\underline{a}, \overline{a}\right]$ för ett slutet intervall.

Då kan vi definiera aritmetiken för intervall enligt:

$$\left[a\right]+\left[b\right]=\left[\underline{a}+\underline{b}, \overline{a}+\overline{b}\right]$$

$$\left[a\right]-\left[b\right]=\left[\underline{a}-\overline{b}, \overline{a}-\underline{b}\right]$$

$$\left[a\right]\times\left[b\right]=\left[\min\left(\underline{a}\underline{b}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\overline{b}\right), \max\left(\underline{a}\underline{b}, \underline{a}\overline{b}, \overline{a}\underline{b}, \overline{a}\underline{b}\right)\right]$$

$$\left[a\right]\div\left[b\right]=\left[a\right]\times\left[1/\overline{b}, 1/\underline{b}\right]$$

med undantaget att $\left[a\right]\div\left[b\right]$ är ogiltigt om intervallet [b] innehåller talet noll (kan du förklara varför?). Lägg märke till att om vi enbart studerar tunna intervall, dvs. intervall $\left[a\right]$ som har bredd noll $(\underline{a}=\overline{a})$, så får vi tillbaka de vanliga räknereglerna. Intervallaritmetiken är således en utvidgning av den reella aritmetiken.

Nu kan vi enkelt räkna med intervall:

$[1; 2] + [3; 4] = [4; 6]$ $\qquad$ $[1; 2] - [3; 4] = [-3; -1]$

$[1; 2] \times [3; 4] = [3; 8]$ $\qquad$ $[1; 2] \div [3; 4] = \left[\frac{1}{4}, \frac{2}{3}\right]$.

För att visa ett användningsområde för intervallaritmetiken behöver vi inte ens lämna klassrummet. Tänk dig att du skall bevisa att funktionen

$$f(x)=\frac{x^2-2x+3}{x^3-2x^2+8}$$

inte antar några negativa värden för $1 \le x \le 2$. Istället för att derivera f eller utföra teckenstudier kan vi helt sonika undersöka hela intervallet [1; 2] i ett enda svep:

$$f([1,2]) = \frac{[1,2]^2 - 2[1,2] + 3}{[1,2]^3 - 2[1,2]^2 + 8}$$

$$= \frac{[1,2]^2 - [2,2][1,2] + [3,3]}{[1,2]^3 - [2,2][1,2]^2 + [8,8]}$$

$$= \frac{[1,4] - [2,4] + [3,3]}{[1,8] - [2,2][1,4] + [8,8]}$$

$$= \frac{[4,7] - [2,4]}{[9,16] - [2,8]} = \frac{[0,5]}{[1,14]} = [0,5]$$

Intervallaritmetiken säger oss nu att, för vilket x som helst, taget från intervallet [1; 2], så gäller att $0 \le f(x) \le 5$. Därmed har vi löst uppgiften!

Tack vare intervallanalysen kan matematiker bevisa komplicerade påståenden med hjälp av datorer, och detta utan att behöva oroa sig för insmugna avrundningsfel. Utan datorns hjälp hade dessa problem varit helt utom räckhåll för vad en människa orkar med att räkna.

Övningar

(1) Kan du skriva om polynomet $p(x)$ som ett enklare uttryck?

(2) Hur många nollställen har $p(x)$? Hur många hittade datorn?

(3) Beräkna $f([x]) = [x]^3 - 5[x] + 1$ för intervallen $[x] = [1, 2]$ och $[x] = [-2, 1]$.

(4) Gäller likheten $[x] - [x] = 0$? Prova med några intervall. Vad kan du säga om $[x] \div [x]$?

(5) Hur skulle du definiera $\sin[x]$?

Svar

(1) Polynomet kan faktoriseras som $p(x)=(1-x)^6$.

(2) Polynomet $p(x)$ har ett nollställe av grad sex. Den heldragna linjen visar över 30 nollställen.

(3) $f([1,\ 2])=[-8,\ 4]$. $f([-2,\ 1])=[-12,\ 15]$.

(4) Likheten $[x]-[x]=0$ gäller bara för tunna intervall. Övriga intervall ger en inneslutning av noll. På samma sätt gäller likheten $[x]\div[x]=1$ bara för tunna intervall. Övriga intervall ger en inneslutning av talet ett.

(5) *Ledtråd*: $\sin[x]=[\sin \underline{x},\ \sin \overline{x}]$ är inte korrekt lösning. Dela upp definitionen i fyra fall.

Ett förvånande pris

– lite finansiell matematik

JOHAN TYSK, UPPSALA UNIVERSITET

Under senare tid har matematik börjat användas i ökad utsträckning inom olika finansiella sektorer i samhället. En bank eller annat finansiellt institut sysslar med olika *finansiella instrument*, t.ex. aktier, fonder eller kapital, eller på senare tid, s.k. *optioner*.

En finansiell option ger innehavaren rätten, men inte skyldigheten, att köpa eller sälja en viss tillgång, t.ex. en aktie, till en givet pris, kallat *lösenpris*. I fallet med *säljoption* kan optionen ses som ett slags försäkring. Hur mycket tillgången än sjunker, kan vi under optionens löptid, sälja tillgången för lösenpriset. Å andra sidan kan vi fortfarande vara med på en eventuell uppgång hos tillgången. Vad är en sådan option värd? Eller kanske bättre formulerat, vad beror priset på?

Låt oss titta på en mycket enkel modell. Vi börjar med en tillgång, t.ex. en aktie med priset 100 (kronor om man så vill) och vi låter lösenpriset också vara 100. *Efter ett år* är aktien, i vår förenklade modell, värd 80 eller 120 (kronor). För att kunna hantera pengar bör vi också ett sparkonto i vår marknadsmodell. Låt oss anta att räntan på kontot är 2 %. Vad är då denna säljoption värd? Ett svar som ligger nära tillhands är måhända att något entydigt pris inte finns (i *modellen*) utan att detta får bestämmas genom aktörerna på en marknad. Tror många på en nedgång i aktien blir säljoptionen attraktiv och därför dyr, medan det omvända gäller om många tror på en uppgång.

För att få ett pris inom modellen kanske man då lägger in ett antagande att aktien ökar till 120 med sannolikheten 30 %, medan den sjunker till 80 med sannolikheten 70 %. Med sannolikheten 70 % fås då ett värde för optionen om ett år om 20 kr, ty värdet av att kunna sälja en aktie som är värd 80 kronor för 100 kronor är 20 kr. Å andra sidan, stiger aktien till 120 blir optionen värdelös, ty i detta fall är det ointressant att kunna sälja den för 100. Den *förväntade utbetalningen* är således 70 % av 20 kronor, vilket ju är 14 kr. Nuvärdet av 14 kronor är 14/1,02 kr ≈ 13,7 kr eftersom vårt bankkonto ger en ränta om 2 %. Således borde priset i denna modell vara 13,7. *Detta är, förvånande nog, fel*! En sådan prissättning skulle i vår modell vara ett så kallat *arbitrage*, vilket betyder att med den prissättningen kan en investerare få en riskfri vinst utan att satsa några egna pengar.

Vi ska nu se hur ett pris som inte ger upphov till något arbitrage kan bestämmas. En idé som visat sig mycket fruktbar är då att försöka konstruera en portfölj, bestående av tillgångarna i marknadsmodellen, nämligen aktien och pengar på kontot, som ger samma värde som optionen efter ett år oavsett hur aktien har gått. En sådan portfölj sägs vara en *replikerande* portfölj. För att se hur en sådan portfölj ska se ut låt oss anta att vi sätter in b kronor på bankkonton och köper aktier för s kronor. För att replikera optionen måste då följande gälla:

$$\begin{cases} b \cdot 1{,}02 + s \cdot 1{,}20 = 0 \\ b \cdot 1{,}02 + s \cdot 0{,}80 = 20 \end{cases}$$

Den första ekvationen säger att portföljens värde ska var noll vid året slut om aktien går upp, medan, enligt den andra ekvationen, portföljens värde ska vara 20 kronor om aktien går ner, enligt den andra ekvationen. Subtraherar vi den första ekvationen från den andra finner vi att $s \cdot 0{,}40 = -20$, vilket ju ger $s = -50$. Sätter vi in detta resultat i någon av de övriga ekvationerna fås

$$b = \frac{60}{1{,}02} \approx 59$$

Ekvationssystemet är således lösbart och vi kan därför konstruera en replikerande portfölj! Vi ska tydligen sätta in 59 kr i sparkontot. Det negativa värdet för investeringen i aktier ska tolkas som att vi ”*blankar*” aktier för 50 kr. Detta innebär att vi lånar ut aktier för detta belopp (i vårt fall blir det faktiskt en halv aktie eftersom aktiens initiala värde i vår modell är 100). När vi lånar ut dessa aktier får vi 50 kronor. Efter ett år måste dessa aktier återköpas till det då gällande marknadsvärdet. Kostnaden för denna portfölj är 59 – 50 = 9kr, ty 50 kr erhålls då vi lånar ut aktier. Vi behöver därför endast lägga till 9 kr av egna medel för att få de erforderliga 59 kronorna på bankkontot. Då portföljen är värd lika mycket som optionen vid periodens slut, både vid uppgång och nedgång för aktien, måste optionens pris vara lika med portföljens pris vid årets början, i vårt fall således 9 kr. Detta är det enda pris som inte leder till arbitrage i vår modell! Det pris vi beräknade tidigare är således för högt ty en investerare skulle annars kunna köpa den replikerande portföljen för 9 kr och sedan sälja optionen för 13,70 kr och på så sätt göra ett arbitrage. Observera att investeraren då inte tar någon risk eftersom portföljens värde är lika med optionens värde efter ett år, oavsett aktiens rörelser. Investeraren kan därför täcka de kostnader som kan uppstå efter ett år då innehavaren av optionen vill lösa in sitt instrument.

Den förväntade tillväxten hos aktien är alltså inte relevant vid optionsprissättning, vilket kan tyckas vara mycket förvånande. Detta gäller även i mer sofistikerade optionsprissättningsmodeller. I den berömda formeln för prissättning av optioner, som publicerades av Fischer Black och Myron Scholes och oberoende av Robert Merton år 1973, är den viktiga parametern den underliggande aktiens *volatilitet*. Denna mäter aktieprisets *tendens* att röra sig upp och ned. Den förväntade tillväxten hos aktien är dock irrelevant i deras formel. Deras resultat kan sägas vara startpunkten för den livaktiga forskning som idag bedrivs för att studera modeller för att prissätta allt mer avancerade finansiella instrument. Den matematiska delen av denna forskningsverksamhet brukar kallas finansiell matematik.

Kedjekod –

ett sätt att beskriva former i digitala bilder

GUNILLA BORGEFORS, CENTRUM FÖR BILDANALYS, SLU, UPPSALA

Datoriserad bildanalys används för att räkna, mäta, känna igen och kontrollera objekt i digitala bilder. Vardagliga enkla exempel är att räkna blodkroppar, läsa streckkoder, läsa blanketter och kontrollera om flaskor som ska fyllas med läsk är tomma och hela. Mer komplicerade uppgifter är att avgöra om vävnadär normal eller är angripen av cancer, kontrollera kvalitén på plankor, känna igen ansikten eller att fåfotbolls-spelande robotar att ”se” bollen och varandra. Ämnet bildanalys uppstod på 1960-talet, när man insåg att en bild kunde representeras i en dator som ett antal punkter med olika ljushetsvärden. I början kunde man bara hantera mycket små bilder – 64×64 punkter var stort! – men mycket teori utvecklades förvånansvärt fort. Många grundläggande begrepp och metoder skapades redan då. Idag är det inga problem att lagra och göra beräkningar i stora bilder i datorer – men teoretiskt finns fortfarande mycket kvar att göra.

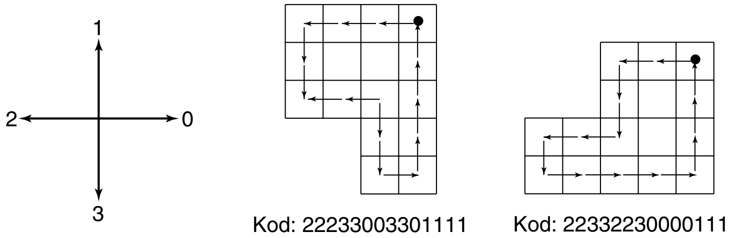
En bild i en dator består alltså av ett antal punkter, kallade pixlar (pixel = picture element), där varje pixel har ett värde. Pixeln ritas som en liten kvadrat. Värdet är oftast ett heltal mellan 0 och 255, där 0 betyder svart och 255 vitt. I en färgbild har varje pixel tre värden, ett för rött, ett för grönt och ett för blått. Genom att blanda dessa tre färger kan man skapa (nästan) alla andra färger. En mörkblå pixel kan till exempel ha värdet (0, 0, 50) och en gul (200, 200, 0). Här ska jag dock inte prata om färg, utan om form.

Form är en mycket viktig egenskap hos olika objekt i bilder. Former används för att känna igen olika föremål och bedöma om de är korrekta eller felaktiga. En bild där man enbart är intresserad av formbrukar vara *binär*, vilket betyder att pixlarna bara kan ha två olika värden: 1 för pixlar som tillhör objektet och 0 för pixlar i bakgrunden. Det finns många olika sätt att skapa en binär bild från en svartvitbild eller en färgbild. Processen kallas *segmentering*. Ett mycket enkelt fall är om man har fotograferat ett mörkt objekt mot en vit bakgrund. Då kan man segmentera fram objektet genom att bestämma att alla pixlar med ett tillräckligt högt värde är bakgrund (0) och alla andra är objekt (1). Tyvärr är det sällan så enkelt – i själva verket är problemet att segmentera en bild i intressant objekt och ointressantbakgrund oftast det allra svåraste i bildanalysprocessen! Men nu förutsätter vi att segmenteringen är gjord och att vi har en binär bild.

Eftersom en digital bild består av ett antal enskilda punkter, inte ett kontinuerligt plan, blir den matematik man använder för att analysera och beskriva form ganska olika jämfört med den ”vanliga” geometrin. I den digitala geometrin måste alla vanliga begrepp som längd, area, räta linjer, cirklar, konvexitet, osv. definieras på lämpliga sätt, så att de får de egenskaper man förväntar sig. En del av detta arbete gjordes redan på 1960-talet, men det är fortfarande ett mycket aktivt forskningsområde, både i Sverige och internationellt.

När man ska se hur pixlar hänger ihop utgår man från den 3×3 pixel stora omgivning till varje pixel som består av dess åtta grannar: fyra som delar en kant med centrumpixeln och fyra som delar en punkt. Ett antal 1-pixlar bildar ett sammanhängade objekt om de ”sitter ihop”. En formell definition är att man ska kunna gå från varje pixel i objektet till alla andra pixlar i det genom att hoppa från granne till granne. Om man alltid kan hitta en väg med bara kantgrannar talar man om ett 4-sammanhängande objekt(fyra grannar). Om man dessutom behöver använda punktgrannar är objektet 8-sammanhängande (åtta grannar). I fortsättningen menar jag med objekt en grupp 4-sammanhängande pixlar och jag förutsätter att det bara finns ett objekt i bilden.

Att räkna upp alla positioner i bilden där det finns en objektpixel är ett dåligt sätt att beskriva formen på objektet. Dels blir den otymplig, dels får samma objektform helt olika beskrivningar om objektet flyttas i bilden. Vad vi vill ha är en kort beskrivning som är enkel att hantera och som är oberoende av läget i bilden (och helst också hur objektet är roterat.) En sådan beskrivning, som skapades i bildanalysens barndom, är *kedjekod*. Kedjekoder kan vara a flera olika slag. Grundidén är att vi går runt objektets kontur och i varje steg antecknar riktningen till nästa pixel. Vi går från kantgranne tillkantgranne (objektet är ju 4-sammanhängande), så vi behöver bara fyra riktningar. Dessa betecknas med talen 0, 1, 2, 3, se Fig. 1 till vänster. I mitten av Fig. 1 finns ett mycket litet objekt. Vi startar rundvandringen i övre högra pixeln, markerad med en fylld cirkel, och går moturs. Nästa konturpixel ligger åt vänster, dvs. i riktning 2. Det gör även nästa två konturpixlar, men sedan blir riktningen neråt, riktning 3. Om vi följer objektet runt till startpunkten får vi kedjan av riktningar: 22233003301111. Detta tal, *riktningskoden*, utgör en fullständig beskrivning av formen på det lilla objektet. Det är lätt att återskapa objektet från koden: vi startar i en punkt och går sedan steg för steg i den angivna riktningen.



Figur 1: Till vänster de fyra riktningarna i kedjekoden. I mitten ett litet objekt och dess riktningskod. Till höger samma objekt roterat. Startpunkterna markeras med •.

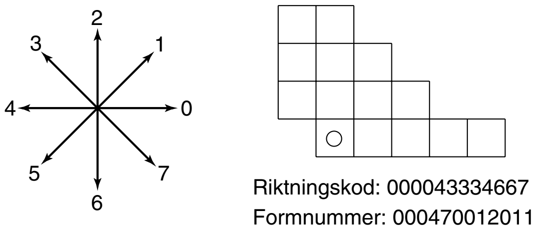
Riktningskoden för ett objekt är oberoende av objektets position i bilden, men den beror förstås påvar längs objektets kontur vi startar. Detta är inte bra – vi vill att beskrivningen ska vara så invariant som möjligt, dvs. bara bero på själva formen och inte på ett antal ovidkommande saker. Den är redan invariant för läget i bilden. Dessbättre är det enkelt att göra riktningskoden invariant för startpunkten. Olika startpunkt innebär bara att kedjan av tal startar på ett annat ställe, ordningen på talen är alltid densamma. Därför kan vi välja att starta kedjan så att motsvarande tal blir så litet som möjligt. I fallet i Fig. 1 blir den invarianta koden 00330111122233. Om vi har två former kan vi nu skapa deras riktningskod och i stället för att jämföra formerna jämföra koderna.

Koden är fortfarande beroende på hur objektet är roterat i bilden. Till höger i Fig. 1 har formen i mitten vridits ett kvarts varv medurs. Med start i övre högra hörnet blir riktningskoden 22332230000111. Skrivet som minsta tal blir det 00001112233223. Koderna för objektet i mitten och till höger i Fig. 1 är olika, trots att det fortfarande är samma form. Kan vi göra något åt detta? Svaret är att det går bra, om vi inser att det inte är riktningsvärdena i sig som är viktiga utan skillnaden mellan dem. I stället för att tänka ”gå vänster”, ”gå vänster”, ”gå nedåt” kan vi tänka ”fortsätt rakt fram”, ”fortsätt rakt fram”, ”sväng ett kvarts varv moturs”. Vi kan räkna fram den rotationsoberoende koden ur den ursprungliga genom att titta på skillnaden mellan på varandra följande tal. I koden i Fig. 1, 00330111122233, drar vi första talet från andra, andra från tredje, osv. tills sista talet dras från det första. Den nya koden blir 0 – 0, 3 – 0, 3 – 3, 0 – 3, 1 – 0, osv., dvs. 0 3 0 – 3 1 osv. Talen 0, 1, 2, 3 i skillnadskoden betyder då inte längre absoluta riktningar, utan 0 = fortsätt i samma riktning, 1 = sväng ett kvarts varv moturs, 2 = sväng ett halvt varv moturs, 3 = sväng ett trekvarts varv moturs (dvs. ett kvarts varv medurs). Men vad betyder – 3? Enligt ovan bör det betyda sväng trekvarts varv bakåt, dvs. medurs. Detta är samma som ett kvartsvarv moturs, vilket betecknas 1. Vi har alltså – 3 = 1. För att inse att detta är helt korrekt och för att kunna räkna skillnader behöver vi använda *modulär aritmetik*.

När man räknar modulo ett visst *n* räknar man helt enkelt endast med den rest man får när man heltalsdividerar med *n*. I vårt fall behöver vi räkna modulo 4, betecknat ”mod 4”. Några exempel: 5 mod 4 = 1 ($5 = 1 \times 4 + 1$), 8 mod 4 = 0 ($8 = 2 \times 4 + 0$), 157 mod 4 = 1 ($157 = 39 \times 4 + 1$). Moduloräkningen fungerar lika bra för negativa tal: – 3 mod 4 = 1 ($-3 = -1 \times 4 + 1$), – 49 mod 4 = 3 ($-49 = -1 \times 13 + 3$). Man har nytta av modulär aritmetik i många sammanhang – här utnyttjar vi den för att hantera att vi efter att ha roterat ett kvarts varv fyra gånger är tillbaka där vi började.

Om vi räknar mod 4 kan vi återföra de negativa talen i den riktningsoberoende koden till 0, 1, 2, 3. Vårt exempel 00330111122233 blir då 03011000100101. Naturligtvis bör vi även här ange den ordning på koden som ger det minsta talet, i detta fall 00010010103011. Riktningskoden för samma form när den vridits ett kvarts varv medurs var 00001112233223. Dess rotationsoberoende kod blir, genom subtraktion av intilliggande tal, 00010010103011, som dessutom direkt är minsta talet, dvs. samma tal som förut! Detta tal, som beskriver vår form i Fig. 1 på mest invariant sätt, kallas *formnummer*.

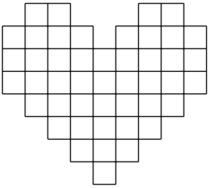
Kedjekoder kan även baseras på åtta riktningar. Allt kommer att fungera på samma sätt, men naturligtvis måste man räkna modulo 8 i formnumret. Ett litet exempel finns i Fig. 2. Lägg även märke till att i den smala delen till höger måste koden vända tillbaka genom samma pixel. Det kan förstås hända även om vi använder fyra riktningar.



Figur 2: Till vänster de åtta riktningarna i kedjekoden. Till höger ett litet objekt och dess riktningskod och formnummer. Startpunkten för minsta riktningskoden markeras med med ○.

Övningsuppgifter

1. Räkna fram både riktningskod och formnummer med både fyra och åtta riktningar för hjärtat i Fig. 3.



Figur 3: Digitalt hjärta att omvandla till olika kedjekoder.

2. Om du gjort rätt i uppgift 1 kommer du att upptäcka att koden med åtta riktningar blir betydligt kortare än den med fyra riktningar. Varför? Kan man säga något allmänt om hur mycket kortare kod man får med åtta riktningar?

3. Rita formen som beskrivs av följande riktningskod, åtta riktningar. Tänk på att du ritar från pixelcentrum till pixelcentrum. 044443223265665444407775656111177772323111. Svaret är belöningen.

4. Rita formen som beskrivs av följande formnummer, fyra riktningar. Vrid svaret tills du ser ett husdjur. 0001000131002003030013031023003201303 1.

5. Riktningskoden kan användas för att manipulera formen på olika sätt. Hur kan du dubblera storleken på objektet genom att ändra koden? Hur kan du halvera (ungefär) storleken?