Sannolikhet och statistik med Matlab

Måns Eriksson

Inledning

Det här kompendiet är tänkt att användas för självstudier under kursen *Sannolikhet och statistik* vid Uppsala universitet. Målet är att använda Matlab för att illustrera olika begrepp och resultat inom sannolikhetsteori och statistik, med förhoppningen om att det ska ge en djupare förståelse för materialet. Dessutom diskuterar vi hur man kan använda Matlab för beskrivande statistik och rutinräkningar. Kompendiet är däremot inte tänkt som en guide till Matlab, och läsaren förutsätts ha använt programmet tidigare. Förutom Matlab-instruktioner finns på några ställen i texten också fördjupande avsnitt för den som vill veta mer eller är extra intresserad av någon del av kursen. Dessa är märkta med en stjärna (*). Avsnitten är inte nödvändigtvis svårare än de övriga, men dyker lite djupare ner i ämnet än kursen i övrigt.

I texten används genomgående samma notation som i boken av Alm/Britton.

På kompendiets hemsida `http://www.math.uu.se/~eriksson/sos/` finns alla kodstycken nedladdningsbara som `.m`-filer. Eventuella feltryck och korrigeringar kommer också att publiceras där.

Det här är första versionen av kompendiet och utvecklingen av det kommer att fortsätta. Kom därför gärna med kommentarer under kursens gång eller i kursutvärderingen!

Uppsala, 4 september 2009
Måns Eriksson
`eriksson@math.uu.se`

Innehåll

1 Beskrivande statistik

När man undersöker ett datamaterial är man ofta intresserad av sammanfatta informationen i materialet genom olika tal och figurer. Ofta är det behändigt att använda något datorprogram för att ta fram den informationen. I det här avsnittet tittar vi kort och översiktligt på hur Matlab kan användas för de vanligaste figurerna och måtten.

1.1 Läges- och spridningsmått

I tabellen nedan visas hur man får de vanligaste läges- och spridningsmåtten med hjälp av Matlab då det datamaterial som man vill undersöka finns sparat i vektorn `x`.

Mått	**Kommando**
Medelvärde	`mean(x)`
Stickprovsstandardavvikelse	`std(x)`
Stickprovsvarians	`var(x)`
Minsta värdet	`min(x)`
Undre kvartil	`quantile(x,0.25)`
Median	`median(x)`
Övre kvartil	`quantile(x,0.75)`
Största värdet	`max(x)`

Övning 1.1. *Undersök datamaterialet i Exempel 6.1 i Alm/Britton med hjälp av Matlab.*

1.2 Grafisk illustration

En lämplig första illustration av ett datamaterial är ofta att kort och gott plotta observationerna. Genom att rita in dem i ett diagram får man en första bild av hur spritt materialet är och om det exempelvis verkar vara centrerat kring något visst värde. I Matlab kan man plotta materialet med kommandot `scatter(1:length(x),x)`.

Datamaterialet i vektorn `x` kan annars åskådliggöras grafiskt med ett lådagram eller ett histogram. Lådagram fås i Matlab genom kommandot `boxplot(x)`.

Ett histogram med tio intervall fås med kommandot `hist(x)`. Vill man istället ha `n` stycken intervall skriver man `hist(x,n)`. Om `y` är en vektor med intervallgränser kan man få ett histogram med dessa gränser genom kommandot `hist(x,y)`. Det är slutligen också möjligt att få ett histogram med relativa frekvenser längs den vertikala axeln, på ett sådant sätt att staplarnas sammanlagda area blir 1. Detta är av intresse när vi senare i kursen studerar fördelningar för slumpvariabler och kan göras genom att man skriver

```
[i,r] = hist(x);
bar(r,i/(sum(i)*(r(2)-r(1))))
```

Övning 1.2. *Plotta längderna för flickorna i datamaterialet i Exempel 6.1 i Alm/Britton. Fundera utifrån bilden på om du ser några observationer som är extra intressanta eller om du kan säga något om materialet i allmänhet.*

Övning 1.3. *Rita och jämför lådagram och histogram för de olika könen i datamaterialet.*

1.3 Flerdimensionella material

Om man har ett tvådimensionellt datamaterial sparat i vektorerna `x` och `y` fås korrelationskoefficienten med kommandot

```
K=corrcoef(x,y); K(1,2)
```

Man får ett spridningsdiagram genom att skriva `scatter(x,y)`.

Övning 1.4. *Gör Övning 6.4.1 i Alm/Britton med hjälp av Matlab.*

2 Slumpvariabler

Även om de funktioner för beskrivande statistik som vi diskuterat ovan är nyttiga så kommer det inte att vara dem som tonvikten ligger på i resten av den här texten. När vi försöker förstå olika begrepp och resultat inom sannolikhetsteori och statistik är det nyttigt att undersöka hur olika slumpvariabler och satser fungerar i praktiken - och vi kan använda Matlab för att simulera utfall av slumpvariabler. Det innebär att vi kan använda programmet för att studera hur en slumpvariabel beter sig eller för att dra ett stickprov från en fördelning som vi är intresserade av.

För flera av de vanligaste fördelningarna kan Matlab dessutom användas för att beräkna fördelningsfunktionen $F(x) = P(X \leq x)$, täthetsfunktionen $f(x)$ eller sannolikhetsfunktionen $P(X = x)$ för olika x. Programmet kan därmed användas istället för de tabeller som återfinns längst bak i Alm/Britton och formelsamlingen.

2.1 Några vanliga fördelningar

Nedan finns de Matlabfunktioner som används för att beräkna $F(x)$, $f(x)$ och $P(X = x)$ samt generera m stycken observationer från olika viktiga slumpvariabler.

Diskreta fördelningar

Fördelning	$\mathbf{F(x) = P(X \leq x)}$	$\mathbf{P(X = x)}$	**Generera m observationer**
$Bin(n,p)$	`binocdf(x,n,p)`	`binopdf(x,n,p)`	`binornd(n,p,1,m)`.
$Po(\lambda)$	`poisscdf(x,lambda)`	`poisspdf(x,lambda)`	`poissrnd(lambda,1,m)`
Likformig på $1, 2, \ldots, N$	`unidcdf(x,N)`	`unidpdf(x,N)`	`unidrnd(N,1,m)`

Kontinuerliga fördelningar

Fördelning	$\mathbf{F(x) = P(X \leq x)}$	$\mathbf{f(x)}$	**Generera m observationer**
$Re(a,b)$	`unifcdf(x,a,b)`	`unifpdf(x,a,b)`	`unifrnd(a,b,1,m)`.
$N(\mu, \sigma^2)$	`normcdf(x,my,sigma)`	`normpdf(x,my,sigma)`	`normrnd(my,sigma,1,m)`
$Exp(\beta)$	`expcdf(x,1/beta)`	`exppdf(x,1/beta)`	`exprnd(1/beta,1,m)`

2.2 Väntevärden, standardavvikelser och kvantiler

De egenskaper som kanske är viktigast för att beskriva en slumpvariabel är dess väntevärde och standardavvikelse. Väntevärdet sägs ofta beskriva var observationer av slumpvariabeln hamnar "i genomsnitt", medan standardavvikelsen beskriver spridningen av observationerna. Dessutom är man ofta intresserad av fördelningens kvantiler. Vi ska studera hur dessa hänger ihop för normalfördelningen och exponentialfördelningen, för att förhoppningsvis få en lite bättre känsla för vad begreppen innebär.

En titt på normalfördelningen

Normalfördelningen har två parametrar - väntevärdet μ och variansen σ^2. Vi ska titta på hur dessa påverkar läge, spridning och kvantiler för fördelningen. Intervallet mellan α- och $(1-\alpha)$-kvantilen för normalfördelningen är viktigt inom statistiken, så vi märker ut dessa för $\alpha = 0.05$. Sannolikheten att en observation hamnar mellan de två kvantilerna är 0.90 (kontrollera det!). Kvantilerna fås i Matlab med kommandot `norminv(1-alfa,my,sigma)`.

Vi börjar med att titta på hur väntevärdet μ påverkar fördelningen.

Exempel 2.1. *Parametern μ i normalfördelningen*

```
% Samma väntevärde men olika varians
sigma=1; i=1;
for my=[0 1 -2]
  subplot(3,2,i); hold on; plot(-4:0.01:4,normpdf(-4:0.01:4,my,sigma),'r');
    plot(norminv(0.95,my,sigma),0,'o'); plot(norminv(0.05,my,sigma),0,'o');
    hold off; xlabel(sprintf('Täthetsfunktion för N(%g,%g)',my,sigma^2));
  subplot(3,2,i+1); plot(-4:0.01:4,normcdf(-4:0.01:4,my,sigma),'r');
    xlabel(sprintf('Fördelningsfunktion för N(%g,%g)',my,sigma^2));
  i=i+2;
end
```

Variansen förändras inte då väntevärdet ändras. Vad händer med kvantilerna när väntevärdet ändras? Med avståndet mellan dem?

Härnäst undersöker vi hur variansen σ^2 (och därmed också standardavvikelsen σ) påverkar fördelningen.

Exempel 2.2. *Parametern σ^2 i normalfördelningen*

```
% Samma varians men olika väntevärde
my=0; i=1;
for sigma=[0.2 1 sqrt(2) 3]
  subplot(4,2,i); hold on; plot(-6:0.01:6,normpdf(-6:0.01:6,my,sigma),'r');
    plot(norminv(0.95,my,sigma),0,'o'); plot(norminv(0.05,my,sigma),0,'o');
    hold off; xlabel(sprintf('Täthetsfunktion för N(%g,%g)',my,sigma^2));
  subplot(4,2,i+1); plot(-6:0.01:6,normcdf(-6:0.01:6,my,sigma),'r');
    xlabel(sprintf('Fördelningsfunktion för N(%g,%g)',my,sigma^2));
  i=i+2;
end
```

Väntevärdet förändras inte då variansen ändras. Men vad händer med kvantilerna när variansen får ett nytt värde? Med avståndet mellan dem? Hur hänger detta ihop med tolkningen av variansen och standardavvikelsen som ett mått på hur utspridda observationerna av slumpvariabeln är?

En titt på exponentialfördelningen

Exponentialfördelningen skiljer sig från normalfördelningen bland annat genom att den bara har en parameter, β. När parametern ändras så förändras både väntevärdet och standardavvikelsen; faktum är att de båda har värdet $1/\beta$. Exponentialfördelningens kvantiler fås med kommandot `expinv(alfa,1/beta)`.

Exempel 2.3. *Parametern β i exponentialfördelningen*

```
% Olika värdena på parametern beta
i=1;
for beta=[0.5 1 3 8]
  subplot(4,2,i); hold on; plot(0:0.01:6,exppdf(0:0.01:6,1/beta),'r');
    plot(expinv(0.95,1/beta),0,'o'); plot(expinv(0.05,1/beta),0,'o');
    plot(1/beta,0,'*'); hold off;
    xlabel(sprintf('Täthetsfunktion för Exp(%g)',beta));
  subplot(4,2,i+1); plot(0:0.01:6,expcdf(0:0.01:6,1/beta),'r');
    xlabel(sprintf('Fördelningsfunktion för  Exp(%g)',beta));
  i=i+2;
end
```

Här ser vi att spridningen minskar när väntevärdet minskar, eftersom väntevärdet och standardavvikelsen är kopplade till varandra.

Jämförelser av fördelningar

Som tidigare nämnts så är väntevärde och standardavvikelse viktiga när man beskriver en slumpvariabel. Men hur lika är olika slumpvariabler som har samma väntevärde och standardavvikelse? Vi undersöker $N(1,1)$- och $Exp(1)$-fördelningarna, båda med väntevärde och standardavvikelse lika med 1. Vi börjar med att rita upp täthets- och fördelningsfunktioner.

Exempel 2.4. *Normalfördelning och exponentialfördelning*

```
% Täthets- och fördelningsfunktioner
subplot(2,1,1); hold on; plot(-2:0.01:4,normpdf(-2:0.01:4,1,1),'r');
  plot(norminv(0.95,1,1),0,'o'); plot(norminv(0.05,1,1),0,'o');
  plot(0:0.01:4,exppdf(0:0.01:4,1),'b'); plot(expinv(0.95,1),0,'*');
  plot(expinv(0.05,1),0,'*'); hold off;
  xlabel('Täthet: N(1,1) i rött och Exp(1) i blått. Normalförd-kvantiler
                         som cirklar, Expförd-kvantiler som stjärnor.');
subplot(2,1,2); hold on; plot(-2:0.01:4,normcdf(-2:0.01:4,1,1),'r');
  plot(0:0.01:4,expcdf(0:0.01:4,1),'b'); hold off;
  xlabel('Fördelningsfunktioner: N(1,1) i rött och Exp(1) i blått');
```

Funktionerna är ganska olika varandra. Men kan man se några skillnader i praktiken? Vi jämför$N(1,1)$-fördelningen med $Exp(1)$-fördelningen genom att simulera 200 observationer från respektive fördelning med Matlab och plotta dem bredvid varandra.

Exempel 2.5. *Normalfördelning och exponentialfördelning - simulering*

```
% Simulering
n=200; % Antalet observationer som ska simuleras från varje fördelning
hold on;
  scatter(1:n, normrnd(1,1,1,n),'r'); % N(1,1) i rött.
  scatter((n+1):(2*n), exprnd(1,1,n),'b'); % Exp(1) i blått.
  % Lägger till en grön linje för väntevärdet 1:
  plot(1:(2*n),zeros(1,2*n)+1,'g');
  xlabel('Simulerade observationer: N(1,1) i rött och Exp(1) i blått');
hold off;
```

Troligen är de ganska lika varandra ovanför den gröna linjen, men väldigt olika under den (hur väl tycker du att det stämmer överens med täthetsfunktionen?). Trots att de bägge fördelningarna har samma väntevärde och varians så beter de sig på rätt olika vis! Även om väntevärde och varians är bra som en första beskrivning av en slumpvariabels beteende så säger de tydligen inte allt om fördelningen.

En märklig fördelning

Väntevärde och varians är ofta bra för att beskriva olika egenskaper hos fördelningar, men det finns fall där de inte går att använda. Ett exempel är *Cauchyfördelningen*. Det är en fördelning som trots att den är symmetrisk saknar väntevärde och har oändlig varians.

Cauchyfördelningen finns inte implementerad i Matlabs statistikpaket, men n stycken slumpvariabler från fördelningen kan genereras med hjälp av $Re(0,1)$-fördelningen genom kommandot[1]

```
tan(pi*(unifrnd(0,1,1,n)-1/2))
```

För att illustrera hur fördelningen beter sig provar vi att generera och plotta 100 Cauchyfördelade slumptal:

```
plot(1:100,tan(pi*(unifrnd(0,1,1,100)-1/2)),'p')
```

Vad kan du säga om fördelningens beteende? Prova att köra koden ovan flera gånger. Varierar plotten mycket från gång till gång? Får du utifrån det någon känsla för varför Cauchyfördelningen inte har något väntevärde men har oändlig varians?

Fördelningen sägs ha *tunga svansar*, vilket innebär att en förhållandevis stor del av dess värden kommer att hamna långt från fördelningens mitt. Därmed kan de värden den antar också variera väldigt kraftigt. Trots att Cauchyfördelningen tillsynes har så pass dåliga egenskaper så används den ganska flitigt i stokastisk modellering. Den dyker upp inom fysiken (för att beskriva resonanser och spektrallinjer i spektroskopi) och används inom försäkringsmatematik (där de flesta utbetalningarna är små men där det ibland kommer några som är väldigt stora).

[1]Metoden som används bygger på invertering av fördelningsfunktionen och beskrivs i kapitel 5 i Alm/Britton.

3 Stora talens lag

Så gott som all kvantitativ forskning bygger på att man får mer information genom att samla in mer data. Våra erfarenheter säger oss att ju fler mätningar vi gör, desto säkrare blir vår uppskattning av värdet på den uppmätta kvantiteten. Att så är fallet även i vår matematiska modell för sannolikheter och slumpvariabler är innebörden av en av sannolikhetsteorins viktigaste satser - Stora talens lag.

3.1 Konvergens mot väntevärdet

Stora talens lag säger, i viss mening, att då $X_1, \ldots, X_n$ är oberoende slumpvariabler med väntevärde μ och ändlig varians så $\bar{X}_n \to \mu$, då $n \to \infty$, där $\bar{X}_n = \frac{1}{n}\sum_{i=1}^{n} X_i$. Vi ska kontrollera det genom simulering i specialfallet då X_i är likafördelade $N(0,1)$.

Exempel 3.1. *Stora talens lag*

```
summa=0; i=1; n=0;
medel=zeros(1,5);
for m=[10 90 900 9000 90000]
  summa=summa+sum(normrnd(0,1,1,m));
  n=n+m;
  medel(i)=summa/n;
  text=sprintf('n=%g, medelvärde: %g \n',n,medel(i));
  disp(text)
  i=i+1;
end
```

Verkar medelvärdet konvergera mot väntevärdet 0?

Man kan också fråga sig vad som händer med medelvärdet om X_i inte har något väntevärde. Vi har tidigare anmärkt att Cauchyfördelningen saknar väntevärde och provar därför att göra samma simulering som ovan, men med Cauchyfördelade slumpvariabler.

Exempel 3.2. *Medelvärde för Cauchyfördelningen*

```
summa=0; i=1; n=0;
medel=zeros(1,5);
for m=[10 90 900 9000 90000]
  summa=summa+sum(tan(pi*(unifrnd(0,1,1,m)-1/2)));
  n=n+m;
  medel(i)=summa/n;
  text=sprintf('n=%g, medelvärde: %g \n',n,medel(i));
  disp(text)
  i=i+1;
end
```

Verkar medelvärdet konvergera då `n` växer? Prova gärna att köra koden ovan flera gånger för att se om det blir någon skillnad.

3.2 *STL och beräkningsvetenskap

Ett intressant tillämpningsområde för Stora talens lag är beräkningsvetenskap. Ett exempel är numerisk integration. Vi ska studera hur man kan approximera π med hjälp av Matlab och STL.

Vi vet från tidigare kurser i analys att $\int_0^1 \sqrt{1-x^2}dx = \pi/4$. Kurvan ligger i kvadraten $\{(x,y) : 0 \leq x \leq 1, 0 \leq y \leq 1\}$ i $\mathbb{R}^2$. Fundera på hur man utifrån detta kan använda simulering av slumpvariabler och Stora talens lag för att approximera π! Rita figur!

En lösning på problemet är följande. Låt (X_i, Y_i), $i = 1, \ldots, n$ vara oberoende slumpvariabler tillhörande den tvådimensionella likformiga fördelningen på den just nämnda kvadraten[2] (se avsnitt 3.9.1 i Alm/Britton) och låt $Z_i = 1$ om paret (X_i, Y_i) ligger under eller på kurvan $\sqrt{1-x^2}$ och $Z_i = 0$ om paret ligger över kurvan. Eftersom (X_i, Y_i) är likformigt fördelade på kvadraten så är sannolikheten att de ligger under kurvan lika med arean under kurvan delat med kvadratens area, så

$$P(Z_i = 1) = P(Y \leq \sqrt{1-X^2}) = \frac{\pi/4}{1} = \pi/4.$$

Eftersom sannolikheten är densamma för alla i så innebär det att $\sum_{i=1}^n Z_i \sim Bin(n, \pi/4)$, så att $E(\sum_{i=1}^n Z_i) = n \cdot \pi/4$. Då säger, lite informellt uttryckt, Stora talens lag att $\frac{1}{n}\sum_{i=1}^n Z_i \to \pi/4$ då $n \to \infty$. Vi får därmed att $\pi \approx 4 \cdot \frac{1}{n}\sum_{i=1}^n Z_i$ när n är stort.

En implementering i Matlab kan se ut så här:

Exempel 3.3. *Approximation av π*

```
n=100;
sumZ=0;
for m=1:n
  X=unifrnd(0,1); Y=unifrnd(0,1);
  if Y<sqrt(1-X^2)
    sumZ=sumZ+1;
  end
end
piapprox=4*sumZ/n
```

Den här typen av beräkningsmetoder kallas Monte Carlo-metoder. De konvergerar i allmänhet ganska långsamt; de blir som bäst $O(1/\sqrt{n})$ medan mer konventionella beräkningsmetoder för numerisk integration kan nå $O(1/n^4)$. Det kan kanske avskräcka från användandet av Monte Carlo-integration, men det ska poängteras att det fina med den här typen av metoder är att de är förhållandevis enkla att implementera även i högre dimensioner, där de vanliga metoderna blir ohanterliga.

Övning 3.1. *Prova att använda exempelvis $n = 100$, $n = 10000$ och $n = 1000000$ i Exempel 3.3 ovan. Verkar det som att konvergenshastigheten är $O(1/\sqrt{n})$?*

Monte Carlo-metoder används även för andra typer av beräkningar än integration. De förekommer exempelvis inom olika typer av ingenjörsarbete, inom finansvärlden, inom telekommunikation, i modellering av molekylära system och inom artificiell intelligens.

[2]...vilket är samma sak som att $X_i \sim Re(0,1)$ och $Y_i \sim Re(0,1)$ där X_i och Y_i är oberoende.

4 Centrala gränsvärdessatsen

Ett av sannolikhetsteorins viktigaste (och kanske mest mystiska) resultat är *Centrala gränsvärdessatsen*, förkortat *CGS*. Satsen handlar om hur normaliserade summor av slumpvariabler beter sig när antalet variabler i summan blir allt större.

Den kanske viktigaste praktiska tillämpningen av CGS är att satsen möjliggör approximationer. Något förenklat säger den att om $X_1, \ldots, X_n$ är oberoende likafördelade slumpvariabler med ändlig varians så är summan av dem approximativt normalfördelad om n är "stort". Det innebär att man approximativt kan räkna ut olika sannolikheter för summan. Många vanliga statistiska metoder bygger på just detta faktum och man använder CGS bland annat för att konstruera konfidensintervall; se kapitel 7 i Alm/Britton och avsnitt 6 i kompendiet.

Vi ska nu ta en närmare titt på CGS för att se vad satsen egentligen betyder och vad som menas med att n ska vara "stort". Som en förberedelse passar vi på att studera icke-normaliserade summor av slumpvariabler.

4.1 Summor av slumpvariabler

Det finns många tillfällen då man är intresserad av att räkna med funktioner av slumpvariabler. Det kanske vanligaste exemplet är summor. Vi vet att när man räknar med vanliga reella tal så är $\sum_{i=1}^{10} x = 10 \cdot x$. Men vad händer om man tittar på summan $X_1 + X_2 + ... + X_{10}$, där X_i är oberoende likafördelade slumpvariabler?

Till att börja med kan vi konstatera att det i allmänhet *inte* gäller att $X_1 + X_2 + ... + X_{10} = 10 \cdot X_1$. Vänsterledet är summan av tio oberoende slumpvariabler medan högerledet är tio gånger den första slumpvariabeln. De behöver (förstås?) inte vara lika med varandra. Däremot kan man misstänka att de kanske har samma fördelning, vilket i så fall skulle vara den stokastiska motsvarigheten till att $\sum_{i=1}^{10} x = 10 \cdot x$.

Låt oss för enkelhets skull anta att slumpvariablerna har väntevärde 0 och varians 1. Räknereglerna för väntevärden ger oss att både $10 \cdot X_1$ och $\sum_{i=1}^{10} X_i$ har väntevärdet 0 (kontrollera det!). Det ger en första indikation på att det kanske kan vara så att de har samma fördelning. För att undersöka det hela närmare antar vi att $X_i \sim N(0, 1)$ och tar Matlab till hjälp för att simulera 100 slumptal från fördelningen för $10 \cdot X_1$ och 100 slumptal från fördelningen för $\sum_{i=1}^{10} X_i$.

Exempel 4.1. *Jämförelse av* $10 \cdot X_1$ *och* $\sum_{i=1}^{10} X_i$

```
n=100; % Vill simulera 100 obs vardera från de två varianterna
hold on;
 scatter(1:n, 10*normrnd(0,1,1,n),'b'); % 10*X
 scatter((n+1):(2*n), sum(normrnd(0,1,10,n)),'r'); % X_1+...+X_10
 % Lägger till en grön linje för väntevärdet 0:
 plot(1:(2*n),zeros(1,2*n),'g');
hold off;
```

Verkar de blå punkterna, simulerade från fördelningen för $10 \cdot X_1$, och de röda punkterna, från $\sum_{i=1}^{10} X_i$, ha samma fördelning?

Övning 4.1. *Jämför fördelningarna för* $10 \cdot X_1$ *och* $\sum_{i=1}^{10} X_i$ *genom att simulera observationer och rita lådagram och/eller histogram.*

Övning 4.2. *Använd räknereglerna för väntevärden, varians och normalfördelningen för att bestämma fördelningarna för* $10 \cdot X_1$ *och* $\sum_{i=1}^{10} X_i$ *om* X_i *är oberoende och* $N(0,1)$*-fördelade. Stämmer resultatet med den slutsats du drog av exemplet ovan?*[3]

Det finns alltså skillnader mellan fördelningarna! Förklaringen ges av varianserna; att multiplicera den första slumpvariabeln med 10 gör att värdet som antas varierar mycket mer än om man summerar tio likafördelade slumpavariabler. Det gäller alltså att passa sig litegrann när man räknar med slumpvariabler, så att man inte av gammal vana råkar använda de räkneregler som gäller för reella tal.

Övning 4.3. *Utgående från resultatet i den förra övningen, kan du finna ett tal* a *så att* $a \cdot X_1$ *har samma fördelning som* $\sum_{i=1}^{n} X_i$*? Vilken fördelning har* $\frac{1}{a}\sum_{i=1}^{n} X_i$*?*[4]

4.2 Hur stort är "stort"?

I praktiken säger CGS att om $X_1, \ldots, X_n$ är oberoende likafördelade slumpvariabler med väntevärde μ och varians $\sigma^2 < \infty$ så är

$$\frac{\sum_{i=1}^{n} X_i - n\mu}{\sigma\sqrt{n}} \approx N(0,1) \tag{1}$$

då n är "stort". Vi ska studera $(\sum_{i=1}^{n} X_i - n\mu)/(\sigma\sqrt{n})$ för olika värden på n och två olika fördelningar för X_i.

Låt först $X_i \sim Re(0,1)$. Då är $\mu = 1/2$ och $\sigma^2 = 1/12$. Funktionen `unifrnd(0,1,1,n)` ger n stycken $Re(0,1)$-fördelade slumptal. Vi börjar med att använda den för att rita ett histogram över 1000 slumptal. Verkar de vara $Re(0,1)$-fördelade?

```
hist(unifrnd(0,1,1,1000))
```

Härnäst frågor vi oss vilken fördelning summan av två $Re(0,1)$-fördelade slumpvariabler får. Vi genererar två vektorer med 1000 slumptal vardera:

```
X=unifrnd(0,1,1,1000); Y=unifrnd(0,1,1,1000);
Z=X+Y; % Definiera Z=X+Y
hist(Z) % Rita ett histogram för Z
```

Verkar summan ha en fördelning som liknar normalfördelningen mer än vad $Re(0,1)$-fördelningen gör?

Vad händer i så fall när vi lägger ihop ännu fler tal? Vi provar att låta n gå från 1 till 9 och normaliserar genom att subtrahera summans väntevärde ($n\mu = \frac{n}{2}$) och dela med dess standardavvikelse ($\sigma\sqrt{n} = \sqrt{\frac{n}{12}}$), för att få $(\sum_{i=1}^{n} X_i - n\mu)/(\sigma\sqrt{n})$. Vi simulerar ett antal observationer och ritar deras histogram, tillsammans med täthetsfunktionen för $N(0,1)$-fördelningen. För vilka värden på n tycker du att approximationen (1) verkar vara godtagbar i det här fallet?

[3] *Svar:* $10 \cdot X_1 \sim N(0,100)$ och $\sum_{i=1}^{10} X_i \sim N(0,10)$. Inte samma!
[4] *Svar:* $a = \sqrt{n}$. $N(0,1)$.

Exempel 4.2. *CGS för rektangelfördelningen*

```
% Vi delar upp ritfönstret i 9 rutor för att undersöka.
X=zeros(1,10000); % Skapar en tom vektor.
for n=1:9
  subplot(3,3,n) % Aktiverar ruta n
  X=X+unifrnd(0,1,1,10000); % Summan av X_i-variablerna
  Y=(X-0.5*n)./sqrt(n/12); % Normaliserar summan
  hold on
    [i,Yut] = hist(Y,-4:0.348:4);
    bar(Yut,i/(sum(i)*0.348)); % Ger histogram med relativ frekvens
    % Lägger till N(0,1)-täthetsfunktionen
    plot(-4:0.01:4,normpdf(-4:0.01:4),'r')
    text=sprintf('Normaliserade summan av n=%g variabler', n);
    xlabel(text)
    ylabel('Rel. frekvens')
  hold off
end
```

Som en jämförelse tittar vi nu på den normaliserade summan $(\sum_{i=1}^{n} X_i - n\mu)/(\sigma\sqrt{n})$ då $X_i \sim Exp(1)$. Då är $\mu = \sigma^2 = 1$. Vi tittar på värden på n mellan 1 och 9. Verkar approximationen (1) vara godtagbar för något av dessa n i det här fallet? Blir approximationen bättre då n ökar?

Exempel 4.3. *CGS för exponentialfördelningen*

```
X=zeros(1,10000); % Skapar en tom vektor.
for n=1:9
  subplot(3,3,n) % Aktiverar ruta n
  X=X+exprnd(1,1,10000);
  Y=(X-n)./sqrt(n); % Normalisera
  hold on
    [i,Yut] = hist(Y,-4:0.348:4);
    bar(Yut,i/(sum(i)*0.348)); % Ger histogram med relativ frekvens
    % Lägger till N(0,1)-täthetsfunktionen
    plot(-4:0.01:4,normpdf(-4:0.01:4),'r')
    text=sprintf('Summan av n=%g variabler', n);
    xlabel(text)
    ylabel('Rel. frekvens')
  hold off
end
```

Uppenbarligen påverkar fördelningens utseende vad det innebär att n är "stort". Allmänt kan man säga att konvergensen i CGS ofta går snabbare om X_i har en fördelning som är symmetrisk kring dess väntevärde μ, det vill säga en fördelning vars täthetsfunktion ser likadan ut på båda sidorna av μ. $Re(0,1)$-fördelningen är symmetrisk kring väntevärde $1/2$ men $Exp(1)$-fördelningen är inte symmetrisk kring väntevärdet 1 (vilket vi såg i Exempel 2.3).

Övning 4.4. *Studera* $(\sum_{i=1}^{n} X_i - n\mu)/(\sigma\sqrt{n})$ *då* $X_i \sim Exp(1)$ *och* n *är 15, 20, 25, 30, 50 och 100. Verkar approximationen godtagbar för något av dessa* n*?*

4.3 Centrala gränsvärdessatsen och approximationer

Exempel 3.61 på sidan 173 i Alm/Britton visar hur man kan räkna ut sannolikheter för binomialfördelningen genom normalapproximation. Approximationen motiveras med hjälp av CGS och vi ska därför studera exemplet igen här. Som tidigare nämnts ger kommandot `binocdf(x,n,p)` fördelningsfunktionen $P(X \leq x)$ när $X \sim Bin(n, p)$. Vi kan använda det för att räkna ut den exakta sannolikheten och jämföra den med den approximativa sannolikheten.

Exempel 4.4. *Normalapproximation av binomialfördelning (Ex. 3.61 i Alm/Britton)*

```
% Approximativ sannolikhet
p1=normcdf(40.5,29.1,sqrt(20.37))-normcdf(19.5,29.1,sqrt(20.37))

% Exakt sannolikhet
p2=binocdf(40,97,0.3)-binocdf(19,97,0.3)

% Differens
p2-p1
```

Tycker du att differensen, det vill säget felet i approximationen, är acceptabel i det här fallet?

Övning 4.5. *Lös problem 336 d) i Alm/Britton exakt med hjälp av Matlab. Är felet i approximationen godtagbart?*

5 Punktskattningar

Vi har i avsnitt 1 i kompendiet sett hur man kan använda Matlab för att räkna ut exempelvis stickprovsmedelvärden och stickprovsvarianser. Eftersom dessa ofta används för att skatta parametrar i olika fördelningar så kan man alltså enkelt använda Matlab för att få fram de skattningar man vill ha.

5.1 *Estimatorer är slumpvariabler

Ett mycket viktigt första steg när man börjar studera statistik är att inse skillnaden mellan *estimatorn*, som är en funktion av slumpvariabler, och *estimatet*, det vill säga skattningen, som är en funktion av det observerade stickprovet. Estimatorn är en slumpvariabel och själva skattningen är en observation av denna.

Att estimatorn är en slumpvariabel innebär att vi kan studera den precis som andra slumpvariabler och beräkna exempelvis dess väntevärde och standardavvikelse. På så vis kan vi teoretiskt jämföra olika estimatorer för att avgöra vilken som är bäst. För det mesta vill vi att estimatorn ska vara väntevärdesriktig (så att den ”i genomsnitt” ger

det rätta värdet) och att dess standardavvikelse skall vara så liten som möjligt (så att skattningen förhoppningsvis inte avviker så mycket från det sanna parametervärdet).

För många estimatorer kan man relativt enkelt räkna ut väntevärde och standardavvikelse. Exempelvis gäller det att om $X_1, \ldots, X_n$ är oberoende $N(\mu, \sigma^2)$-fördelade slumpvariabler så är stickprovsmedelvärdet $\bar{X} \sim N(\mu, \sigma^2/n)$ - denna används som bekant för att skatta väntevärdet μ.

Stickprovsmedelvärdet är inte den enda tänkbara skattningen av μ. En normalfördelad slumpvariabel med väntevärde μ har även median μ, så en tänkbar estimator är stickprovsmedianen $\hat{X}$. Fördelningen för $\hat{X}$ är, liksom väntevärde och standardavvikelse, betydligt svårare att räkna ut än motsvarande egenskaper för stickprovsmedelvärdet. Här kommer Matlab till vår undsättning!

Antag att vi har 10 observationer från $N(\mu, 1)$-fördelningen. Vi vill veta om den bästa estimatorn är stickprovsmedelvärdet $\bar{X}$ eller stickprovsmedianen $\hat{X}$. Vi vet att $E(\bar{X}) = \mu$ och att $V(\bar{X}) = 1/10$. Genom att simulera ett antal observationer av $\hat{X}$ kan vi skatta $E(\bar{X})$ och $V(\bar{X})$.

Vi vet inte hur väntevärdet och variansen för $\hat{X}$ beror på μ, men vi kan prova att stoppa in olika värden på μ för att se om estimatorn är väntevärdesriktig och om variansen ändras då μ ändras. Vi provar att sätta μ lika med 0, 1 och 5 och att simulera 1000 stickprov för varje väntevärde. Vi räknar ut medianen i varje stickprov och gör därmed 1000 simuleringar vardera av $\hat{X}$ för de olika värdena på μ. Vi använder sedan dessa för att skatta $E(\hat{X})$ ocyh $V(\hat{X})$ i de olika fallen.

Exempel 5.1. *Medianen som skattning av μ*

```
med=zeros(3,1000); % Skapar en matris med enbart nollor
for i=1:1000
  med(1,i)=median(normrnd(0,1,1,10)); % Medianen av 10 N(0,1)-obs.
  med(2,i)=median(normrnd(1,1,1,10)); % Medianen av 10 N(1,1)-obs.
  med(3,i)=median(normrnd(5,1,1,10)); % Medianen av 10 N(5,1)-obs.
end
my0=sprintf('my=0. Väntevärde: %g, varians: %g \n',mean(med(1,:)),
                                                    var(med(1,:)));
my1=sprintf('my=1. Väntevärde: %g, varians: %g \n',mean(med(2,:)),
                                                    var(med(2,:)));
my5=sprintf('my=5. Väntevärde: %g, varians: %g',mean(med(3,:)),
                                                    var(med(3,:)));
disp([my0 my1 my5])
```

Verkar $\hat{X}$ vara väntevärdesriktig? Verkar estimatorns varians bero på μ? Är variansen större eller mindre än $V(\bar{X}) = 1/10$? Utifrån detta, vilken av estimatorerna tycker du är att föredra när man vill skatta μ för normalfördelningen?

5.2 *Outliers

Ibland händer det att stickprovet innehåller en (eller flera) observationer som verkar avvika från de andra på ett misstänkt sätt, framförallt genom att vara ovanligt stora eller ovanligt små. Sådana observationer kallas *outliers* och kan exempelvis bero på felaktigt inmatade data, ändrade förutsättningar för en viss mätning, föroreningar eller

ren slump. De kan också vara en indikation på att något är fel med den nuvarande modellen.

Vi såg i det förra avsnittet att stickprovsmedelvärdet i allmänhet är en bättre estimator för parametern μ i normalfördelningen än vad stickprovsmedianen är. Vi ska nu studera hur bra de båda estimatorerna är när stickprovet innehåller en outlier i form av en "förorening". Antag att man tror sig ha gjort 10 observationer av en $N(\mu, 1)$-fördelade slumpvariabel, men att en av observationerna istället har gjorts från en $N(\mu + 4, 1)$-fördelad slumpvariabel. Hur påverkar det väntevärdet och variansen för estimatorerna? I koden nedan antar vi för enkelhets skull att $\mu = 0$.

Exempel 5.2. *Estimatorer och outliers*

```
skattningar=zeros(2,1000); % Skapar en matris med enbart nollor
for i=1:1000
  stpr=normrnd(0,1,1,9); % Nio N(0,1)-obs
  stpr(1,10)=normrnd(4,1,1,1); % En N(4,1)-obs
  skattningar(1,i)=mean(stpr); % Medelvärdet
  skattningar(2,i)=median(stpr); % Medianen
end
medel=sprintf('Medelvärde. Väntevärde: %g, varians: %g \n',
            mean(skattningar(1,:)), var(skattningar(1,:)));
medi=sprintf('Median. Väntevärde: %g, varians: %g',
            mean(skattningar(2,:)), var(skattningar(2,:)));
disp([medel medi])
```

Är skattningarna väntevärdesriktiga? Om inte, är någon av dem bättre än den andra? Är medelvärdet att föredra framför medianen även i det här fallet?

6 Konfidensintervall

Konfidensintervall är ofta ett bra alternativ till rena punktskattningar, eftersom de säger mer om osäkerheten i skattningen. Konfidensgraden är ett mått på hur effektiv metoden som man konstruerat intervallet med är - om konfidensgraden är 95% så ger den i 95% av fallen ett konfidensintervall som täcker det sanna parametervärdet. Viktigt att komma ihåg är att det är intervallets gränser, och inte den parameter man undersöker, som är stokastiska. Därmed kan man inte efter att ha räknat ut konfidensintervallet i ett visst fall säga att sannolikheten att det täcker parametervärdet är 95% - det är samma sak som att slå en tärning, titta på resultatet och sedan påstå att sannolikheten är 1/6 att man just slog en sexa!

6.1 Intervall för parametrar

Ett alternativ till att leta upp fördelningars kvantiler i tabeller är att använda Matlab för att hitta dem. Då har man även möjlighet att använda sådana värden på

fördelningens parametrar (eller på α) som inte finns i de vanliga tabellerna. Kommandon för de vanligaste kvantilerna anges i tabellen nedan.

Kvantil	**Kommando**
λ_α	`norminv(1-alfa)`
$t_\alpha(f)$	`tinv(1-alfa,f)`
$\chi^2_\alpha(f)$	`chi2inv(1-alfa,f)`

Dessa kan användas för att beräkna olika konfidensintervall som vi är intresserade av. Givet ett stickprov, från en normalfördelning med känd standardavvikelse σ, sparat i vektorn `x`, fås ett $(1-\alpha)\%$ konfidensintervall för väntevärdet μ på följande vis.

```
[mean(x)-norminv(1-alfa/2)*sigma/sqrt(length(x));
 mean(x)+norminv(1-alfa/2)*sigma/sqrt(length(x))]
```

Övning 6.1. *Ta fram motsvarande kod för konfidensintervall för μ då σ är okänd respektive konfidensintervall för σ^2 då μ är okänd. Kontrollera din kod genom att jämföra dess resultat med resultat från lämpliga exempel ur Alm/Britton.*

Alternativt kan man använda Matlabs inbyggda funktioner för beräkning av konfidensintervall för väntevärdet:

Konfidensintervall	**Kommando**
För μ, normalfördelning, σ känt	`[h p ci]=ztest(x,0,sigma,alfa); ci`
För μ, normalfördelning, σ okänt	`[h p ci]=ttest(x,0,alfa); ci`

6.2 *Hur bra är konfidensintervall med normalapproximation?

I avsnitt 7.6.4 i Alm/Britton diskuteras användningen av normalapproximation för att konstruera konfidensintervall med approximativ konfidensgrad. Vi ska här genom simulering studera hur nära den sökta konfidensgraden den approximativa konfidensgraden faktiskt hamnar.

Vi simulerar 10000 stickprov av storlek `n` från $Exp(1)$-fördelningen och beräknar för vart och ett av stickproven ett approximativt 95% konfidensintervall för väntevärdet $1/\beta$ genom normalapproximation. Intervallet har formen $I_{1/\beta} = (\bar{x} \pm \lambda_{\alpha/2} - d)$ där medelfelet $d = \bar{x}/\sqrt{n}$. Slutligen kontrollerar vi hur stor andel av konfidensintervallen som innehöll det sanna parametervärdet $\beta = 1$. Om approximationen är bra bör andelen ligga nära den sökta konfidensgraden 0.95.

Exempel 6.1. *Konfidensgrad vid normalapproximation*

```
% Simulerar 10000 konfidensintervall och räknar antalet
% som innehåller det riktiga parametervärdet.
alfa=0.05; beta=1; n=100; antal=0;
for i=1:10000
  x=exprnd(1/beta,1,n); xmean=mean(x);
  int=[xmean-norminv(1-alfa/2)*xmean/sqrt(n)
       xmean+norminv(1-alfa/2)*xmean/sqrt(n)];
  % Kontrollera om 1 ligger i intervallet:
  if (int(1)<=1/beta && 1/beta<=int(2))
    antal=antal+1;
  end
end
% Skattad konfidensgrad:
konfidensgrad=antal/10000
```

Hur bra tycker du att approximationen är för `n=10`? Prova att öka stickprovsstorleken `n` och se hur konfidensgraden påverkas. Stämmer det överens med vad du förväntade dig? Prova också att ändra värdet på exponentialfördelningens parameter `beta`. Påverkar det hur bra approximationen är?

Övning 6.2. *Gör en motsvarande undersökning av konfidensgrad för konfidensintervall för p i $Bin(n,p)$-fördelningen för olika värden på n och p.*

7 Regression

Regression leder ofta till långa (och tröttsamma) beräkningar. När man har större datamaterial gör man klokt i att använda en dator för att utföra dem och Matlab har därför ett antal funktioner för att utföra olika typer av regression. Vi ska här nosa lite på funktionerna för enkel linjär regression samt titta på hur man kan använda Matlab för att illustrera kurvanpassningen.

7.1 Enkel linjär regression

Om vi har ett datamaterial sparat i vektorerna `x` och `y` så kan vi skatta parametrarna α och β i modellen $y = \alpha + \beta x$ med kommandot `polyfit(x,y,1)`. Detta returnerar en vektor med (i tur och ordning) β^* och α^*.

För att illustrera hur det kan gå till tittar vi på Exempel 9.4 ur Alm/Britton.

Exempel 7.1. *Medeltemperatur och latitud*

```
% Exempel 9.4 i Alm/Britton
lat=[66.6 63.5 63.1 60.4 59.2 59.3 57.4 57.6 57.8 56.7 55.7];
temp=[-0.6 4.0 4.2 5.8 7.0 7.6 6.0 7.6 7.7 7.5 8.5];
andpunkter=[min(lat) max(lat)];
koeff=polyfit(lat,temp,1)
hold on;
  scatter(lat,temp)
  plot(andpunkter,koeff(2)+koeff(1)*andpunkter,'r')
hold off;
```

Jämför med de skattningar och den bild som finns i exemplet i boken. Verkar de stämma överens?

Övning 7.1. *Använd resultatet i exemplet för att prediktera medeltemperaturen i Uppsala (latitud 59.9).*[5]

Övning 7.2. *Använd verktygen från Avsnitt 6 för att konstruera ett 95% konfidensintervall för β (utan att anta att σ är känd).*[6]

[5] `koeff(2)+koeff(1)*59.9` ger den predikterade medeltemperaturen 5.8 grader.
[6] $I_\beta = (-0.94, -0.51)$