Avd. Matematisk statistik

KTH Matematik

Tobias Rydén

2011-09-19

SF1905 Sannolikhetsteori och statistik: Lab 1 ht 2011

Förberedelser. Innan du går till laborationen, läs igenom den här handledningen, repetera exponentialfördelningen, gå igenom uppgifterna 4.12 och 6.21 i läroboken samt repetera hur du kan lösa dem.

Temat för den här datorlaborationen är *simulering*. Sannolikhetsteoridelen av kursen handlar om hur man genom *beräkningar* kan ta fram olika storheter som sannolikheter, väntevärden osv, för en given stokastisk modell. För mer komplicerade system är det ibland inte alls möjligt att göra exakta beräkningar, eller så är de så tidskrävande att man avstår.

I sådana sammanhang kan *simulering* vara ett alternativ. Simulering innebär att man med hjälp av en dator simulerar ett antal replikeringar av det stokastiska systemet, och sedan använder t ex medelvärden eller empiriska kvantiler (mer om det nedan) för att uppskatta de storheter man söker. I den här laborationen skall vi göra detta för några enkla problem, men grundprinciperna går att använda på långt mer komplicerade problem som vi inte kan lösa med enkla beräkningar.

I det allra enklaste fallet kan det vara fråga om att uppskatta väntevärdet för en fördelning. Antag att vi har en fördelningsfunktion F, och låt X vara en stokastisk variabel med denna fördelningsfunktion. Antag också att vi vill uppskatta tillhörande väntevärde, μ säg. Om vi nu drar observationer $x_1, x_2, \ldots, x_n$ från F, kan vi uppskatta μ med hjälp av

$$\widehat{\mu} = \frac{1}{n}\sum_{k=1}^{n} x_k.$$

Att detta är en rimlig uppskattning följer av stora talens lag.

Om vi vill uppskatta $F(a) = P(X \leq a)$ för något tal a, så kan vi göra det genom att räkna ut hur stor andel av de simulerade observationerna som

är $\leq a$. Vi kan skriva detta som

$$\widehat{F}(a) = \frac{\text{antal } x_k \text{ som är} \leq a}{n} = \frac{1}{n}\sum_{k=1}^{n} \mathbf{I}(x_k \leq a);$$

här är $\mathbf{I}$ en s k *indikatorfunktion*, som tar värdet 1 om villkoret inom parentes är uppfyllt, annars 0. Alltså är $\mathbf{I}(x_k \leq a)$ lika med 1 precis för de k sådana att $x_k \leq a$, så att summan ovan räknar antalet index k som uppfyller villkoret.

1 Introduktion till Matlab

I samtliga laborationer i den här kursen kommer *Matlab*, som är ett interaktivt program för numeriska beräkningar, att användas. Det är också ett programmeringsspråk. Matlab finns på de flesta datorer på KTH, och till fördelarna med programmet hör att det ser i stort sett likadant ut oberoende av på vilket sorts dator man kör det. Om du vill ha mer att läsa om Matlab så finns det olika handledningar att ladda ned och köpa.

Börja med att logga in på ditt vanliga konto. Starta sedan Matlab genom att klicka på ikonen. Programmet avlutas med kommandot `exit`.

Till att börja med kan man tänka på Matlab som en avancerad räknedosa som beräknar uttryck. Man skriver in vad man vill ha gjort och Matlab svarar.

```
>> 3*11.5+2.3^2/4

ans =

   35.8225
```

Variabler tilldelas värden med tecknet `=` och finns sedan kvar i minnet. Pröva att tilldela några variabler värden.

```
>> a=1;
>> b=sqrt(36);
>> width=3.89;
>> who

Your variables are:

a         b         width
```

Kommandot `who` visar alltså vilka variabler som finns i minnet. En variabel, t ex `b`, kan raderas ur minnet med `clear b`. Läggs ett semikolon, `;`, till efter en kommandorad, skrivs resultatet inte ut på skärmen. Matlab kan också hantera vektorer och matriser, och de hanteras precis lika enkelt som ovan.

```
>> x=[1 3 7]

x =

     1     3     7

>> y=[2 1 8]'

y =

     2
     1
     8

>> z=[1 2 ; 3 4]

z =

     1     2
     3     4

>> w=rand(1,4)

w =

    0.2190    0.0470    0.6789    0.6793
```

Tecknet `'` betyder som synes transponat och semikolon används för att skilja rader åt i matriser. Funktionen `rand(m,n)` ger en $m \times n$-matris med slumptal som är rektangelfördelade mellan 0 och 1.

Notationen för algebraiska uttryck är den vanliga, men kom ihåg att multiplikationstecknet `*` tolkas som matrismultiplikation. Elementvis multiplikation mellan två matriser `A` och `B` av samma dimensioner skrivs `A.*B`.

I Matlab finns alla vanliga funktioner inbyggda, t ex

```
exp log sin asin cos acos tan atan.
```

Observera att `log` är den naturliga logaritmen. Pröva att plotta en funktion, t ex genom följande kommandon.

```
>> x=0.5:0.1:2
>> help log
>> y=log(x)
>> plot(x,y)
```

Den första raden tilldelar `x` en vektor som löper från 0.5 till 2 i steg om 0.1. Hjälpfunktionen `help` ger information om en funktion. Skriver du bara `help`, visas en lista med tillgängliga funktioner, sorterade efter funktionspaket (ett sådant paket kallas en *toolbox*).

I Matlab finns det en stor mängd funktioner som har att göra med sannolikhetsteori och statistik. Se `help stats`. Titta speciellt under rubriken *Random Number Generators*, som kommer i början på den långa listan av funktioner.

2 Väntevärde av exponentialfördelning

Matlabs funktion för att simulera exponentialfördelade stokastiska variabler heter `exprnd`. Använd gärna Matlabs hjälpkommando `help` för att ta reda på precis hur funktionen `exprnd` fungerar, dvs skriv `help exprnd`!

Funktionen kan också användas för att simulera vektorer (eller matriser) av oberoende exponentialfördelade variabler. T ex ger

```
>> n=1000;
>> x=exprnd(2.5,n,1);
```

en $n \times 1$-vektor av värden från exponentialfördelningen med väntevärde 2.5.

Antag att vi nu inte vet att denna fördelning har väntevärdet just 2.5, och att vi vill uppskatta detta från våra simulerade data. Det kan vi göra genom att beräkna medelvärdet:

```
>> mean(x)
```

Hur bra blir din uppskattning? Pröva att göra om simuleringen och medelvärdesberäkningen några gånger. Pröva också olika värden på n!

3 Svanssannolikheter för exponentialfördelning

Vi ska nu använda simulering för att uppskatta svanssannolikheten $P(X > x) = 1 - F_X(x)$ för en exponentialfördelning. Syntaxen `(x>5)` i Matlab ger

en vektor av samma storlek som x, men där ett element i vektorn är 1 eller 0 beroende på om motsvarande element i x uppfyller villkoret > 5 eller inte. Du kan allså simulera en vektor x av data som ovan, och sedan skriva

```
>> mean(x>5)
```

för att beräkna skattningen $1 - \widehat{F}(5)$. Vad får du för värde? Vad är den sanna svanssannolikheten? Prova med olika n, och olika svanssannolikheter (dvs byt 5 mot något annat)!

4 Simulering av uppgift 4.12 i läroboken

Uppgift 4.12 i läroboken handlar om att beräkna sannolikheten att den ena av två oberoende exponentialvariabler är mindre är den andra. Du kan uppskatta den sannolikheten genom att simulera två vektorer `x` och `y` av oberoende observationer från två exponentialfördelningar med olika väntevärden, och sedan se hur ofta en x-variabel är mindre än motsvarande y-variabel. Du kan skapa en vektor av 1:or och 0:or som ovan genom syntaxen `(x<y)` i Matlab. Använd detta för att uppskatta sannolikheten i uppgift 4.12 för värden på a och b som du väljer!

5 Simulering av uppgift 6.21 i läroboken

För att lösa uppgift 6.21 i läroboken kan man använda centrala gränsvärdessatsen (CGS). Eftersom CGS är en approximation, så blir det dock inte en exakt lösning. En lösning med hjälp av simulering är naturligtvis också en uppskattning, dvs en approximation, men genom att göra en tillräckligt stor simulering så kan vi göra felet hur litet vi vill. Det kan vi inte med CGS.

I Matlab kan vi använda funktionen `randsample` för att simulera en vektor av 1000 dragningar från den diskreta fördelningen i uppgift 6.21:

```
>> randsample(0:3,1000,true,[0.4 0.2 0.3 0.1])
```

Skriv `help randsample` för att ta reda på varför det är rätt syntax!

Nu vill vi summera just 1000 dragningar från denna fördelning för att få *en* simulering av det totala antalet barn i bostadsområdet.

```
>> sum(randsample(0:3,1000,true,[0.4 0.2 0.3 0.1]))
```

Naturligtvis måste vi göra många simuleringar av det totala antalet barn, precis som i de föregående problemen. Vi kan använda en slinga för det:

```
>> n=500; x=zeros(n,1);
>> for i=1:n, x(i)=sum(randsample(0:3,1000,true,[0.4 0.2 0.3 0.1])); end
```

För Matlab-puritanen som vill arbeta med vektoroperationer, erbjuder vi alternativet

```
>> n=500
>> x=sum(reshape(randsample(0:3,n*1000,true,[0.4 0.2 0.3 0.1]),1000,n));
```

som gör samma sak.

Vi har nu en vektor av längd n med lika många simuleringar av det totala antalet barn. Uppgiften handlar om att ta fram 10%-kvantilen för den fördelning som vi simulerar från, dvs om X är en stokastisk variabel som representerar det totala antalet barn så vill vi hitta en punkt a sådan att $P(X \leq a) = 0.9$.

Denna kvantil kan vi uppskatta med hjälp av den *empiriska 10%-kvantilen* för vårt stickprov, som (väsentligen) är den punkt a sådan att 90% av observationerna är $\leq a$. I Matlab kan vi skriva

```
>> quantile(x,0.9)
```

Observera att man anger sannolikhetsmassan som skall ligga till *vänster* om punkten man söker (här: 0.9), till skillnad från i läroboken. Läs gärna om funktionen genom `help quantile`!

Hur många simuleringar n måste du göra för att få en bra skattning?