Matematisk Statistik

SF1901 Sannolikhetsteori och statistik, HT 2017
Laboration 2 för CFATE3 m.fl.

1 Introduktion

Denna demonstration är inte poänggivande, men syftar till att ge en djupare förståelse för några viktiga begrepp i kursen genom att illustrera dem med hjälp av MATLAB. Laborationen kommer att gås igenom vid en av föreläsningarna (se föreläsningsplanen för exakt datum). Vid detta föreläsningstillfälle har ni också har möjlighet att diskutera det som ni har kommit fram till under arbetet med laborationen.

Börja med att ladda ner följande filer från kurshemsidan.

- `plot_mvnpdf.m`
- `hist_density.m`

Se till att filerna ligger i den mapp du kommer att arbeta i. För att kontrollera att du har lagt filerna rätt, skriv `ls` och se om filerna ovan listas. Du kan skriva dina kommandon direkt i MATLAB-prompten men det är absolut att föredra att arbeta i editorn. Om den inte är öppen så kan du öppna den och skapa ett nytt dokument genom att skriva `edit lab2.m`. Koden som ges nedan är skriven i celler. En ny cell påbörjas genom att skriva två procenttecken. `Ctrl+Enter` exekverar innehållet i en cell.

2 Fördelningsfunktion och täthetsfunktion

Täthetsfunktionen f_X för en kontinuerlig stokastisk variabel X definieras av

$$P(X \in [a, b]) = \int_a^b f_X(x)dx,$$

och fördelningsfunktionen F_X ges av

$$F_X(x) = P(X \leq x) = \int_{-\infty}^{x} f_X(y)dy.$$

MATLAB har kommandon för de vanligaste sannolikhetsfördelningarna. För normalfördelningen $N(\mu, \sigma)$ ges exempelvis täthetsfunktionens värde i x,

$$f_X(x) = \frac{1}{\sqrt{2\pi}\sigma} e^{-\frac{(x-\mu)^2}{2\sigma^2}},$$

av kommandot `normpdf(x,mu,sigma)`. Följande kod genererar grafen av täthetsfunktionen för den standardiserade normalfördelningen $N(0, 1)$.

```
%% Tathetsfunktion for normalfordelning
    dx = 0.01;
    x = -10:dx:10;      % Skapar en vektor med dx som inkrement
    y = normpdf(x, 0, 1);
    plot(x,y)
```

Prova även att plotta täthetsfunktionen till normalfördelningen för några andra värden på parametrarna μ och σ, exempelvis $\mu = -1$, $\sigma = 0.1$ respektive $\mu = 2$, $\sigma = 2$.

För gammafördelningen med parametrar a och b är täthetsfunktionen

$$f_X(x) = \frac{1}{b^a \Gamma(a)} x^{a-1} e^{-x/b},$$

(observera att [2] använder en annan definition av parametrarna i gammafördelningen). Följande kod kan användas för att plotta denna täthetsfunktion.

```
%% Tathetsfunktion for gammafordelning
    dx = 0.01;
    x = -0:dx:10;       % Skapar en vektor med dx som inkrement
    y = gampdf(x,1,2);
    plot(x,y), hold on
    z = gampdf(x,5,1);
    plot(x,z,'r')
```

Även för fördelningfunktionerna finns kommandon för de vanligaste sannolikhetsfördelningarna. För gammafördelningen gäller exempelvis

```
%% Fordelningsfunktion for gammafordelning
    dx = 0.01;
    x = -0:dx:10;       % Skapar en vektor med dx som inkrement
    y = gamcdf(x,1,2);
    plot(x,y), hold on
    z = gamcdf(x,5,1);
    plot(x,z,'r')
```

Betrakta nu en stokastisk variabel X med täthetsfunktion

$$f_X(x) = \lambda e^{-\frac{x}{\lambda}} + \frac{\lambda}{x}, \quad x \in [1,\, 10]$$

för ett specifikt λ. Genom att lösa ekvationen $\int_1^{10} f_X(x)dx = 1$ numeriskt så kan vi härleda approximationen $\lambda = 0.4267$. Bestäm fördelningsfunktionen för X.

Svar: .....

För att få en bättre bild av täthetsfunktionen för X, så kan vi plotta den och exempelvis jämföra med täthetsfunktionen för en exponentialfördelad stokastisk variabel med väntevärde ett.

```
%% Jamforelse av tathetsfunktioner
    dx = 0.1;
    x = 0:dx:15;        % Skapar en vektor med dx som inkrement
    mu = 1;
    y = exppdf(x, mu);  % exponential-fordelningen
    plot(x,y), hold on
    lambda = 0.4267;
    f=(lambda*exp(-x/lambda)+lambda./x).*(x > = 1 & x < = 10);
    plot(x,f)
```

Diskutera skillnaden mellan fördelningarna.

3 Multivariat normalfördelning

Täthetsfunktionen för den multivariata normalfördelningen ritas upp av funktionen `plot_mvnpdf`. Vi undersöker hur funktionen fungerar och testar med några olika parametervärden. Parametrarna `mux` och `muy` kan anta alla reella värden, parametrarna `sigmax` och `sigmay` kan anta alla positiva värden och parametern `rho` kan anta alla värden på intervallet $[-1, 1]$. Observera att plotfönstret i funktionen `plot_mvnpdf` är fixt, så för parametervärden som är av storleksordningen tio eller större, så kommer merparten av täthetsfunktionen att hamna utanför plotfönstret.

```
%% Multivariat normal
    mux = 0; muy = -2; sigmax = 1; sigmay = 4; rho = 0.7;
    plot_mvnpdf(mux, muy, sigmax, sigmay, rho)
```

Hur påverkar olika parametervärden utseendet på plotten? Vad motsvarar de olika parametrarna?

4 Simulering av slumptal

Vi ska undersöka hur MATLAB kan användas för att generera slumptal. Följande kod genererar N stycken Exp(1/10)-fördelade slumptal, ritar upp ett histogram av slumptalen samt plottar den sanna täthetsfunktionen för Exp(1/10) ovanpå histogrammet som jämförelse.

```
%% Simulering av slumptal
    mu = 10;
    N = 1e4;
    y = exprnd(mu, N, 1);   % Genererar N exp-slumptal
    hist_density(y);    % Skapar ett normaliserat histogram
    t = linspace(0, 100, N/10); % Vektor med N/10 punkter
    hold on
    plot(t, exppdf(t, mu), 'r') % 'r' betyder rod linje
    hold off
```

Upprepa simuleringarna. Hur förhåller sig histogrammet till den röda linjen och hur förklaras variationen kring denna linje?

5 Stora talens lag

Stora talens lag säger att för oberoende, likafördelade stokastiska variabler $X_1, X_2, ...$, så gäller det att

$$S_n := \frac{1}{n}\sum_{i=1}^{n} X_i \to E[X_1], \quad \text{då } n \to \infty,$$

dvs. det aritmetiska medelvärdet konvergerar mot väntevärdet när antalet termer i medelvärdet går mot oändligheten. Vi ska nu undersöka denna konvergens genom att simulera de stokastiska variablerna X_i (som vi här låter vara exponentialfördelade) och studera beteendet hos medelvärdet S_n.

```
%% Stora talens lag
    mu = 0.5;
    M = 500;
    X = exprnd(mu, M, 1);
    plot(ones(M, 1)*mu, 'r-.')
    hold on
    for k = 1:M
        plot(k, mean(X(1:k)), '.')
        if k == 1
            legend('Sant \mu', 'Skattning av \mu')
        end
        xlabel(num2str(k)), pause(0.001)
    end
    hold off
```

Punkten med x-värde `k` är en skattning av medelvärdet av `k` exponentialfördelade stokastiska variabler. Ser det ut som förväntat?

6 Monte Carlo-simulering av väntevärden

Antag att vi vill bestämma väntevärdet av antalet ögon som kommer upp vid kast med en sexsidig tärning. Detta är inte svårt att beräkna för hand, men det går också att kasta tärningen många gånger och sedan räkna ut medelvärdet av dessa kast. Om $X_1, X_2, \ldots, X_n$ är likafördelade med väntevärde μ så gäller enligt Stora talens lag i [2] att

$$P\left(\left|\frac{1}{n}\sum_{i=1}^{n} X_i - \mu\right| < \varepsilon\right) \to 1$$

för varje $\varepsilon > 0$ när $n \to \infty$. Sannolikheten att skillnaden mellan medelvärdet och det sanna väntevärdet är mindre än ε går alltså mot ett när antalet observationer går mot oändligheten. Att använda detta samband och en slumptalsgenerator för att beräkna väntevärden kallas för Monte Carlo-metoder.

Idén bakom Monte Carlo-metoder är gammal och har funnits inom matematiken åtminstone sedan 1700-talet, men synen kom att förändras under andra halvan av 1900-talet då det på allvar blev möjligt att utföra stora beräkningar. Under 1940-talet utvecklade Stanislaw Ulam och John von Neumann metoder för att göra dessa "tärningskast" med hjälp av dator enligt [1]. Arbetet var kopplat till Manhattanprojektet vars syfte var att ta fram den första atombomben. Metoden namngavs efter casinot Monte Carlo i Monaco.

Antag att U är en stokastisk variabel som är likformigt fördelad över intervallet $[0, 2\pi]$. Om vi vill beräkna $E[\sin^2(U)]$, så kan vi göra detta analytiskt med hjälp av definitionen som

$$\begin{aligned} E[\sin^2(U)] &= \int_0^{2\pi} \sin^2(x)\frac{1}{2\pi}dx = \int_0^{2\pi} \frac{1-\cos(2x)}{4\pi}dx \\ &= \left[\frac{x}{4\pi} - \frac{\sin(2x)}{8\pi}\right]_0^{2\pi} = \frac{1}{2}, \end{aligned}$$

men väntevärdet kan också beräknas med Monte Carlo-metoder med följande kod.

```
%% Monte Carlo, del 1
    N = 1e5;
    U = rand(N, 1)*2*pi;
    mean(sin(U).^2);
```

Resultatet blir nära 1/2, men varierar något från gång till gång. Fördelen med Monte Carlo-metoder är att de kan användas även för väntevärden som är svåra att beräkna exakt. Låt exempelvis X och Y vara oberoende stokastiska variabler där $X \in \text{Exp}(4)$ och $Y \in N(0,1)$. Väntevärdet $E[(e^{X\cos(Y)}]$ ges då av

$$\begin{aligned} E[(e^{X\cos(Y)}] &= \int_0^\infty \int_{-\infty}^\infty e^{x\cos(y)} f_{X,Y}(x,y)dydx \\ &= \int_0^\infty \int_{-\infty}^\infty e^{x\cos(y)} \frac{1}{4} e^{-x/4} \frac{1}{\sqrt{2\pi}} e^{-y^2/2} dydx, \end{aligned}$$

vilket är en rätt knepig integral. Med Monte Carlo-metoder beräknas väntevärdet med följande kod.

```
%% Monte Carlo, del 2
    N = 1e5;
    X = exprnd(1/4,N,1);
    Y = randn(N,1);
    mean(exp(X.*cos(Y)));
```

Upprepa simuleringen av väntevärdet och se hur resultatet varierar. Prova också att variera N.

7 Monte Carlo-simulering av talet π

Vi ska nu använde Monte Carlo-metoder för att bestämma ett approximativt värde på talet π. Låt U och V vara två oberoende stokastiska variabler som är likformigt fördelade på $[-1,1]$. Paret (U,V) antar värden i $[-1,1]\times[-1,1]$ och kan ses som punkter i en kvadrat i planet. Sannolikheten att punkten (U,V) hamnar i enhetscirkeln är

$$P(\sqrt{U^2+V^2} \leq 1) = \frac{\text{arean av enhetscirkeln}}{\text{arean av kvadraten } [-1,1]\times[-1,1]} = \frac{\pi}{4}.$$

Vi kan skatta π på följande sätt. Vi simulerar först ett stort antal punkter $(U_1,V_1),(U_2,V_2),...,(U_N,V_N)$. För varje punkt (U_i,V_i) kontrollerar vi om $\sqrt{U_i^2+V_i^2} \leq 1$ och beräknar andelen punkter som hamnat i enhetscirkeln. Eftersom

$$\frac{\text{Antal punkter som hamnat i enhetscirkeln}}{N} \to P(\sqrt{(U^2+V^2} \leq 1)) = \frac{\pi}{4},$$

då $N \to \infty$, så gäller det för stora värden på N att

$$\pi \approx \frac{4 \cdot \text{Antal punkter som hamnar i enhetscirkeln}}{N}.$$

Följande kod genererar N punkter (U_i, V_i), plottar dem i planet samt beräknar motsvarande skattning av värdet på π. Kör koden flera gånger och variera N.

```
%% Monte Carlo, del 3.14
    N = 1e2;
    U = 2*rand(1,N,1)-1;    % Genererar U(-1,1)-ford. slumptal
    V = 2*rand(1,N,1)-1;
    plot(U,V,'o'), hold on  % Plottar de genererade punkterna
    X = -1:0.01:1;
    plot(X,sqrt(1-X.^2),'r')    % Plottar enhetscirkeln
    plot(X,-sqrt(1-X.^2),'r')
    Z = (sqrt(U.^2+V.^2)≤1);    % Beraknar narmevarde pa pi
    pi = 4*mean(Z);
```

8 Centrala Gränsvärdessatsen

Koden nedan simulerar exponentialfördelade slumptal och summerar sedan dessa. Studera koden och fundera ut vad `N` representerar.

```
%% Centrala gransvardessatsen
    M = 1e3;
    N = 4;
    mu = 5;
    X = exprnd(mu, M, N);
    S = cumsum(X, 2);
    for k = 1:N
        hist(S(:, k), 30)
        xlabel(num2str(k))
        pause(0.1)
    end
```

Justera N, vad händer när du ökar respektive minskar värdet? Varför? Vid vilket N ser det ut som att det inte gör någon skillnad att öka N? Vilken fördelning verkar summorna ha? Varför har de denna fördelning?

9 Simulering av konfidensintervall

Ett konfidensintervall med konfidensgrad $1 - \alpha$ för en (okänd) parameter μ innehåller det sanna μ med sannolikhet $1 - \alpha$. Vi ska försöka förstå innebörden av detta begrepp med hjälp av simuleringar. Följande kod använder $n = 25$ oberoende observationer från $N(2, 1)$- fördelningen för att skatta ett konfidensintervall för väntevärdet med konfidensgrad 95% (vi bortser från att vi vet vad det sanna värdet är). Detta upprepas 100 gånger så vi har 100

konfidensintervall. Hur många av dessa förväntar vi oss ska täcka den sanna parametern?

```
%% Simulering av konfidensintervall
    % Parametrar:
    n = 25; %Antal matningar
    mu = 2; %Vantevardet
    sigma = 1; %Standardavvikelsen
    alpha = 0.05;

    %Simulerar n * 100 observationer. (n observationer for ...
        varje intervall och 100 intervall)
    x = normrnd(mu, sigma,n,100); %n x 100 matris med varden

    %Skattar mu med medelvardet
    xbar = mean(x); %vektor med 100 medelvarden.

    %Beraknar de undre och ovre granserna
    undre = xbar - norminv(1-alpha/2)*sigma/sqrt(n);
    ovre = xbar + norminv(1-alpha/2)*sigma/sqrt(n);

    %Ritar upp alla intervall och markerar de som inte ...
        tacker det sanna vardet roda
    figure(1)
    hold on
    for k=1:100
        if ovre(k) < mu
            plot([undre(k) ovre(k)],[k k],'r')
            elseif undre(k) > mu
            plot([undre(k) ovre(k)],[k k],'r')
        else
            plot([undre(k) ovre(k)],[k k],'b')
        end
    end
    %b1 och b2 ar bara till for att figuren ska se snygg ut.
    b1 = min(xbar - norminv(1 - alpha/2)*sigma/sqrt(n));
    b2 = max(xbar + norminv(1 - alpha/2)*sigma/sqrt(n));
    axis([b1 b2 0 101]) %Minimerar mangden outnyttjat ...
        utrymme i figuren
    %Ritar ut det sanna vardet
    plot([mu mu],[0 101],'g')
    hold off
```

Vad visar de horisontella strecken och det vertikala strecket? Hur många av de 100 intervallen innehåller det sanna värdet på μ? Stämmer resultatet med dina förväntningar? Kör simuleringarna flera gånger.

Variera nu μ, σ, n och α (en i taget) och ser hur de olika parametrarna påverkar resultatet.

Referenser

[1] Eckhardt, Roger (1987) Stan Ulam, John von Neumann and the Monte Carlo, Method *Los Alamos Sci.*, Vol **15**, p. 131-43.

[2] Blom, G., Enger, J., Englund, G., Grandell, J., och Holst, L., (2005). Sannolikhetsteori och statistikteori med tillämpningar.