TBDA01: Laboration 2

Måns Magnusson, Mattias Villani

30 augusti 2016

Instruktioner

- Laborationen ska göras **två och två**
- Labben ska vara en **PDF-rapport** med kod, analys och grafer. Ett tips är att använda R markdown. I rapporten ska följande ingå:
  - Båda studenternas namn och LiU-id.
  - Universitet och institution
  - Laborationsnummer
- Deadlinen för laborationen framgår av **kurshemsidan.**
- Laborationsrapporten skickas in via LISAM.

Innehåll

Kapitel 1

Introduktion till R

R är ett programmeringspråk för statistisk programmering som påminner mycket om Matlab. R bygger på öppen källkod och kan laddas ned **här**. R-Studio är en mycket populär IDE för R (som också påminner mycket om Matlab). Denna IDE finns att tillgå **här**. I R-Studio finns funktionalitet för literate programming med R markdown implementerat för att kombinera R kod med markdownsyntax. På detta sätt är det enkelt att generera rapporter med både text, grafik och kod. Det är R:s motsvarighet till Python Notebook.

För en ingång till R från andra språk kan onlineboken *Advanced R* rekommenderas som finns **här**. Kapitlen *Data structures*, *Subsetting* och *Functions* bör ge en snabb introduktion.

Även boken *The art of R programming* av Norman Matloff kan vara till hjälp som referenslitteratur. Boken finns **här**.

Videomaterial

- För en introduktion till syntaxen i R se Google developers R videomaterial **här**.
- Mer (detaljerat) videomaterial av Roger Peng finns att tillgå **här**.
- För att visualisera med basgrafiken finns **följande** introduktionsvideo.
- För mer komplicerad grafik rekommenderas `ggplot2`-paketet. En introduktionsvideo finns **här**.
- En introduktion till R markdown finns **här**.

Cheatsheets

- *R reference card v.2* av Matt Baggot med vanliga funktioner i R finns att tillgå **här**.
- *R markdown cheatsheet* av R-Studio med tips för R markdown finns att tillgå **här**.

Kapitel 2

Laboration

I denna laboration kommer vi gå djupare in på statistisk inferens. D.v.s. givet att vi observerat data, vilka slutsatser kan vi dra om parametrarna i våra sannolikhetsmodeller. När vi gör statistisk inferens kommer vi (oftast) att behöva göra **antaganden** om från vilken modell vårt datamaterial kommer. Det är därför nyttigt att ha med sig följande citat av George Box.

> ”All models are wrong, but some are useful.” - George E. P. Box

Obs!

> De flesta beräkningar i denna laboration är avrundade. Du kan mycket väl få något annorlunda resultat (men som avrundat bör ge samma resultat som i exemplen).

2.1 Likelihoodfunktionen

Vi ska nu visualisera likelihoodfunktionen. Nedan är 10 respektive 100 dragningar från en Gamma($\alpha = 4, \beta = 1$), vilket i detta exempel är vår ”sanna” fördelning. Observera att vi sätter slumpfröet till 4711. Det gör att vi kan replikera dessa dragningar exakt.

```
> set.seed(4711)
> x1 <- rgamma(n = 10, shape = 4, scale = 1)
> x1

 [1] 7.7323 6.5334 6.0959 2.7802 1.3565 5.1966 1.9279 5.4122 8.5859 6.4369

> x2 <- rgamma(n = 100, shape = 4, scale = 1)
```

Vi ska nu implementera log-likelihoodfunktioner (och anta att observationerna är obeorende av varandra). För att härleda en log-likelihoodfunktion utgår vi ofta från vår sannolikhetsfördelnings täthetsfunktion.

Nedan är ett exempel på log-likelihoodfunktionen för gammafördelade variabler och en vektor $\mathbf{x}$ med data.

$$l(\alpha, \beta|\mathbf{x}) = \ln\left(\prod_{i=1}^{n} \frac{\beta^\alpha}{\Gamma(\alpha)} x_i^{\alpha-1} e^{-\beta x_i}\right) \tag{2.1}$$

$$= n \cdot [\alpha \cdot \ln\beta - \ln\Gamma(\alpha)] + (\alpha - 1)\sum_{i=1}^{n} \ln(x_i) - \beta\sum_{i=1}^{n} x_i \tag{2.2}$$

Uppgift 1 Likelihoodfunktioner

a) Skapa en funktion i R du kallar `llgamma(x, alpha, beta)` som tar datapunkterna i en vektor `x` och värden för α och β och returnerar log-likelihoodvärdet för parametrarna givet det observerade materialet `x`. Inkludera funktionen i rapporten.
Tips! $\ln\Gamma(x)$ heter `lgamma()` och ln heter `log()` i R.
Nedan är ett exempel på värden funktionen ska returnera.

```
> llgamma(x = x1, alpha = 2, beta = 2)

Error in eval(expr, envir, enclos): could not find function "llgamma"
```

b) Beräkna och visualisera loglikelihood värden för $\mathbf{x}_1$ och $\mathbf{x}_2$ som simulerades ovan då $\alpha = 4$. Du behöver beräkna log-likelihooden för olika värden på β på intervallet $(0, 3]$, förslagsvis med en stegstorlek på 0.01. Vad är din visuella bedömning av funktionens maximum? **Tips!** `plot()`

c) Upprepa det du gjort i **b)** ovan men anta nu att $\beta = 1$ och visualisera på ett liknande sätt α på intervallet $(0, 10]$ för $\mathbf{x}_1$ och $\mathbf{x}_2$. Vad är din visuella bedömning av funktionens maximum?

d) Härled nu log-likelihoodfunktionen för en vektor $\mathbf{y}$ med normalfördelade datapunkter och inkludera härledningen i rapporten.
Implementera log-likelihoodfunktionen som en funktion i R som du kalla `llnorm(x, mu, sigma2)`. Funktionen ska kunna ta en vektor med datapunkter `x`, samt värden på σ och μ.
Inkludera funktionen i rapporten.
Nedan är ett exempel på hur funktionen ska fungera.

```
> llnormal(x = x1, mu = 2, sigma2 = 1)

Error in eval(expr, envir, enclos): could not find function "llnormal"
```

Tips! π är inkluderat som `pi` i R.

e) Visualisera på samma sätt som i **b)** och **c)** ovan log-likelihood funktionen för μ på intervallet $[0, 10]$ för de (gammafördelade) dragningarna $\mathbf{x}_1$ och $\mathbf{x}_2$ ovan. Vi antar att $\sigma^2 = 1$. Vad är dina slutsatser? Går det att se på log-likelihoodfunktionen om datat kommer från en gammafördelning?

2.2 Punktskattningar från populationsmodell

Vi har sett ovan att likelihoodfunktionen är en funktion som innehåller information om de parametrar vi är intresserade av. Vi vill då uppskatta dessa parametrar baserat på data vi har samlat in. En av de vanligaste metoderna för att göra detta är med vad som brukar kallas **maximum likelihood estimation** (MLE), som namnet antyder handlar det om att maximera likelihoodfunktionen vi studerade ovan. Är vi intresserade av att försöka uppskatta parametrarna i vår modell med ett enda värde per parameter är vi intresserad av en **punktskattning** av modellens parametrar. Det är också det vi ska studera i detta avsnitt.

För gammafördelningens parameter β går det att analytiskt härleda maximum för loglikelihoodfunktionen med avseende på parametern β. Detta kan göras på traditionellt sett genom att derivera log likelihoodfunktionen, sätta derivatan till 0 och lösa ut β. En punktskattning anges ofta med ett litet ”tak” över den parameter vi är intresserad av för att indikera att det är en skattning av parametern β.

$$\hat{\beta}_{MLE} = n\alpha \left(\sum_{i=1}^{n} x_i \right)^{-1} \tag{2.3}$$

Funktionen för att beräkna vår punktskattning är vad som kallas en **estimator**. Det värde funktionen returnerar kallas för **estimat**.

Uppgift 2 Punktskattning med MLE i en gammafördelning

Implementera estimatorn ovan som en funktion du kallar `gamma_beta_mle(x, alpha)` med parametrarna `x` (data) och `alpha`. Nedan är ett exempel på hur funktionen ska fungera:

```
> gamma_beta_mle(x = x1, alpha=2)

Error in eval(expr, envir, enclos): could not find function "gamma_beta_mle"
```

Skatta $\hat{\beta}_{MLE}$ med denna estimator för $\mathbf{x}_1$ och $\mathbf{x}_2$ ovan med $\alpha = 4$. Vad är dina slutsatser?

På ett liknande sätt är det möjligt att skatta parametrarna μ och σ^2 i en normalfördelning.

Uppgift 3 Punktskattning med MLE i en normalfördelning

a) MLE-estimatorn för μ och σ^2 i en normalfördelning finns **här**. Implementera dessa estimatorer som `norm_mu_mle(x)` och `norm_sigma2_mle(x)` med `x` (data) som argument. Se nedan för ett exempel på hur funktionerna ska fungera.

```
> test_x <- 1:10
> norm_mu_mle(x = test_x)

Error in eval(expr, envir, enclos): could not find function "norm_mu_mle"

> norm_sigma2_mle(x = test_x)

Error in eval(expr, envir, enclos): could not find function "norm_sigma2_mle"
```

b) Gör 10 och 100 dragningar från följande fördelning. Sätt slumpfröet till 42 innan du gör dina dragningar på samma sätt som $\mathbf{x}_1$ och $\mathbf{x}_2$ ovan.

$$\mathbf{y} \sim \mathcal{N}(\mu = 10, \sigma^2 = 4)$$

Använd sedan dina två estimatorer för att först skatta μoch σ^2 sedan. Vad är skillnaden mellan 10 och 100 dragningar? Vad beror detta på (**Tips!** Föregående laboration)?
Obs! Tänk på att `rnorm()` är parametriserad med σ, inte σ^2.

2.3 Samplingfördelning och väntevärdesriktighet

Vi ska nu studera egenskaperna hos dessa estimatorer genom simulering, detta är en vanlig metod för att utvärdera hur väl olika **estimatorer** fungerar. Vi ska nu studera **fördelningen** för de estimatorer vi har implementerat ovan.

Uppgift 4 Samplingfördelningen för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$

Som ett första steg ska vi studera fördelningen för de estimatorer vi har implementerat. Vi gör detta genom att upprepa skattningen för ett antal olika utfall från våra fördelningear och se hur våra estimatorer varierar när våra data varierar. Detta kallas estimatorns **samplingfördelning**.
Implementera pseudokoden nedan för att beräkna 2000 MLE-skattningar.

```
for i in 1 to 2000 do
   x1 = draw Gamma(n =10, alpha= 4, beta = 1)
   x2 = draw Gamma(n =100, alpha= 4, beta = 1)
   beta1_mle[i] = gamma_beta_mle(x = x1, alpha = 4)
   beta2_mle[i] = gamma_beta_mle(x = x2, alpha = 4)

   y1 = draw Normal(n = 10, mu = 10, sigma2 = 4)
   y2 = draw Normal(n = 100, mu = 10, sigma2 = 4)
   mu1[i] = norm_mu_mle(x = y1)
   mu2[i] = norm_mu_mle(x = y2)
   sigma2[i] = norm_sigma2_mle(x = y1, mu = 10)
   sigma2[i] = norm_sigma2_mle(x = y2, mu = 10)
```

a) Visualisera samplingfördelningarna för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$ då $n = 10$ och då $n = 100$ i ett histogram. Vad är dina slutsatser? **Tips!** `hist()`

b) Öka nu antalet dragningarna till $n = 1000$ för respektive fördelning. Vad är dina slutsatser? Hur förändras våra skattningar när vi har fler datapunkter som vi använder för att skatta våra parametrar? Vad beror detta på?

I föregående laboration så studerade vi fördelningars förväntade värde, $E(X)$, och varians, $Var(X)$. När vi arbetar med estimatorer kan vi också tala om en estimators förväntade värde på precis samma sätt som i föregående laboration.

Uppgift 5 Det förväntade värdet för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$

Beräkna det förväntade värdet för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$ baserat på de samplingfördelningar du beräknat ovan. Vad är din slutsats? Förändras det förväntade värdet om vi ökar urvalsstorleken?

På ett liknande sätt går det att beräkna en estimators varians.

2.4 Numerisk maximum likelihood estimation

De exempel vi arbetat med såhär långt har varit ”snälla” situationer där det finns analytiskt härledda estimatorer för våra estimatorer. Men som vi sett ovan (exempelvis för α i gammafördelningen) är det långt ifrån självklart att det finns analytiska resultat för att göra en MLE. I lite mer komplicerade fall vill vi därför istället använda oss av numeriska optimerare för att göra vår skattningar. I R finns funktionen `optim()` som är en bra numerisk optimerare som kan användas för MLE oavsett om vi känner likelihoodens gradient och hessian eller inte.

Nedan är ett exempel på hur `optim()` kan användas för att numeriskt finna MLE skattningar av μoch σ^2 i exemplet ovan. `optim()` kräver att parametrarna som ska optimeras finns i en vektor, likaså finner `optim()` ett minimum varför vi behöver multiplicera log likelihooden med -1. Det ger följande funktion:

```
> llnormal2(par = c(2, 1), x = x1)

Error in eval(expr, envir, enclos): could not find function "llnormal2"
```

Med denna funktion kan vi sedan använda `optim()` för att numeriskt finna våra maximum likelihoodskattningar. Jämför med resultaten vi fick ovan med de analytiska lösningarna.

```
> opt_res <- optim(c(0,1), llnormal2, x=test_x, method="L-BFGS-B", lower = c(-Inf,0))

Error in (function (par) : object 'llnormal2' not found

> opt_res$par

Error in eval(expr, envir, enclos): object 'opt_res' not found
```

Uppgift 6 Log-likelihoodfunktionen för betafördelningen

a) Härled (eller leta reda på) log-likelihoodfunktionen för betafördelningen och implementera den som en funktion i R som kan optimeras med `optim()`. Nedan är ett exempel på log-likelihoodfunktionen (multiplicerad med -1) som kan användas i `optim()`.

```
> llbeta(par = c(2, 2), x = c(0.01, 0.5, 0.99))

Error in eval(expr, envir, enclos): could not find function "llbeta"
```

b) Simulera 50 dragningar från en $\mathcal{B}(\alpha = 0.2, \beta = 2)$ och visualisera dragningarna med ett histogram. **Tips!** `hist()`

c) Använd `optim()` för att baserat på dessa dragningar och log-likelihoodfunktionen uppskatta parametrarna α och β i betafördelningen. Tänk på att α och β är definierade på $\mathbb{R}^+$ när du anger intervallen till `optim()`.

d) Upprepa **b)** och **c)** 2000 ggr (d.v.s. ta fram samplingfördelningen för $\hat{\alpha}_{MLE}$ och $\hat{\beta}_{MLE}$). Visualisera dessa samplingfördelningar i ett histogram. **Tips!** `hist()`