TDAB01 SANNOLIKHETSLÄRA OCH STATISTIK

LABB 2: ESTIMATORER OCH DERAS SAMPLINGEGENSKAPER

JOSE M. PEÑA, MÅNS MAGNUSSON, MATTIAS VILLANI
IDA, LINKÖPINGS UNIVERSITET

1. INSTRUKTIONER

- Laborationen ska göras **två och två**
- Labben ska vara en **PDF-rapport** med kod, analys och grafer. I rapporten ska följande ingå:
  - Båda studenternas namn och LiU-id.
  - Laborationsnummer
  - Uppgifterna ni besvarar (t ex som rubriker).
- Ett tips är att använda R markdown. Här finns en R markdownmall att utgå ifrån. Antingen kan ni skapa PDF direkt från denna mall i R-Studio (med TeX) eller så skapar ni ett Word/HTML dokument som sedan skrivs ut till PDF.
- Laborationsrapporten skickas in via LISAM. Där hittar ni också **deadlinen**.

2. 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.

2.1. **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.

2.2. **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.

3. LABORATION

I denna laboration kommer vi att gå djupare in på statistisk inferens. Allså, 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.

”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).

3.1. **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. Kom ihåg att α heter `shape` och β heter `scale` för gammafördelningen i R. 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)
> x2

  [1] 4.79331 6.40788 1.25250 7.06992 2.67750 1.74628 5.69628 4.95335
  [9] 3.64830 5.56426 3.02685 5.63416 4.69548 3.87418 4.72661 4.91524
 [17] 8.29348 6.57035 7.10917 4.31000 6.82118 3.66030 1.03559 9.27426
 [25] 2.56627 3.50256 3.08015 4.86450 4.30884 6.73306 2.70627 4.57985
 [33] 2.67979 2.27968 2.00768 1.40964 4.99229 5.48001 4.51633 5.14000
 [41] 3.77294 3.45746 2.85032 2.46253 3.36590 3.49688 2.41453 5.07472
 [49] 3.81785 3.65560 5.87285 4.31166 9.40850 8.03080 3.48122 5.37745
 [57] 7.74533 8.56121 0.71804 4.34067 3.02849 5.75616 1.44207 3.84525
 [65] 5.10882 2.07482 3.58257 0.94767 2.44326 4.97860 3.52383 1.69256
 [73] 9.03576 3.61321 2.45790 2.94222 4.97442 1.56972 2.01078 7.01523
 [81] 6.40548 1.39056 2.86646 1.20835 1.59489 4.04473 7.87768 2.75054
 [89] 5.82472 3.32521 2.56622 2.55995 5.58725 4.63006 4.74396 3.84933
 [97] 1.64613 4.71465 2.28516 3.08871
```

Vi ska nu använda log-likelihoodfunktionen. 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. Vi antar här att observationerna är iid (independent and identically distributed). Då,

$$\begin{aligned}\ln L(\alpha, \beta|\mathbf{x}) &= \ln\left(\prod_{i=1}^{n} \frac{\beta^\alpha}{\Gamma(\alpha)} x_i^{\alpha-1} e^{-\beta x_i}\right) \\ &= n \cdot [\alpha \cdot \ln\beta - \ln\Gamma(\alpha)] + (\alpha - 1)\sum_{i=1}^{n} \ln(x_i) - \beta \sum_{i=1}^{n} x_i\end{aligned}$$

3.1.1. *Likelihoodfunktioner.*

(1) Skapa en funktion i R som du kallar `llgamma(x, alpha, beta)` som tar datapunkterna i en vektor $\mathbf{x}$, värden för α och β och returnerar log-likelihoodvärdet för parametrarna givet det observerade data $\mathbf{x}$. Inkludera funktionen i rapporten. Nedan är ett exempel på värden funktionen ska returnera.

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

[1] -75.19
```

Tips! $\ln\Gamma()$ heter `lgamma()` och $\ln()$ heter `log()` i R.

(2) 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. För vilket av de upprepade värdena för β får du det maximala värdet på log-likelihoodfunktionen ? Visualisera log-likelihoodfunktionen för de olika värdena på β.

(3) Upprepa det du gjort i (2) ovan men anta nu att $\beta = 1$ och visualisera på ett liknande sätt α på intervallet (0,10] för $\mathbf{x}_1$ och $\mathbf{x}_2$. Vilket värde på α ger det maximala värdet för log-likelihood-funktionen ?

(4) 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 kallar `llnorm(x, mu, sigma2)`. Funktionen ska kunna ta en vektor med datapunkter $\mathbf{x}$, samt värden på μ och σ^2. Inkludera funktionen i rapporten. Nedan är ett exempel på hur funktionen ska fungera.

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

[1] -87.257
```

Tips! π är inkluderat som `pi` i R.

(5) Visualisera på samma sätt som i (2) och (3) ovan log-likelihoodfunktionen för μ på intervallet [0,10] för de (gammafördelade) dragningarna $\mathbf{x}_1$ och $\mathbf{x}_2$ ovan. Vi antar att $\sigma^2 = 1$. Visualisera gammafördelnings täthetsfunktion (med α och β som maximerade log-likelihoodfunktionen i (2) och (3)) och normalfördelningens täthetsfunktion (med μ som maximerade log-likelihoodfunktionen i (4)). Visualisera också datamaterialet (som ett histogram med `hist()`). Vilken modell, normalfördelningen eller gammafördelningen, tycker du passar datamaterialet bäst ?

3.2. **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 den 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. Då,

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

Funktionen för att beräkna vår punktskattning för ett antal datapunkter är vad som kallas en estimator. Det värde funktionen returnerar kallas för estimat eller skattning.

3.2.1. *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)

[1] 0.38419
```

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

3.2.2. *Punktskattning med MLE i en normalfördelning.*

(1) 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)

[1] 5.5

> norm_sigma2_mle(x = test_x)

[1] 8.25
```

(2) Gör 10 och 10000 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. **Obs!** Tänk på att `rnorm()` är parametriserad med σ, inte σ^2.

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

Använd sedan dina två estimatorer för att först skatta μ och σ^2 sedan. Vad är skillnaden mellan 10 och 10000 dragningar ? Vad beror detta på ?

3.3. **Numerisk maximum likelihood estimation.** De exempel vi arbetat med såhär långt har varit snälla situationer där det finns analytiskt härledda 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.

Nedan är ett exempel på hur `optim()` kan användas för att numeriskt finna MLE skattningar av μ och 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<-function(par=c(2,1),x){
+   -llnormal(x,par[1],par[2])
+ }
```

Med denna funktion kan vi sedan använda `optim()` för att numeriskt finna våra maximum likelihoodskattningar. Argumentet `par` i `optim()` är de initiala värdena där optimeraren startar. Tänk på att ge initieringsvärden som är definierade i funktionen. För att undvika att `optim()` ska söka i exakt 0 kan man ange `.Machine$double.eps` i `lower` argumentet som lägsta positiva numeriska värdet.

```
> opt_res <- optim(par = c(0,1), fn = llnormal2, x=test_x, method="L-BFGS-B",
+                  lower = c(-Inf, .Machine$double.eps))
>
> opt_res$par
[1] 5.499999 8.249989
```

3.3.1. *Log-likelihoodfunktionen för betafördelningen.*

(1) Härled (eller leta reda på) log-likelihoodfunktionen för betafördelningen och implementera den som en funktion i R som kan optimeras med `optim()`.

(2) Simulera 100 dragningar från en $Beta(\alpha = 0.2, \beta = 2)$ och visualisera dragningarna med ett histogram.

(3) 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 och startvärdena till `optim()`.

3.4. **Estimatorers samplingfördelning.** Vi ska nu studera egenskaperna hos estimatorer genom simulering, detta är en vanlig metod för att utvärdera hur väl olika estimatorer fungerar.

3.4.1. *Samplingfördelningen för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$.*

(1) Vi ska studera fördelningen för de estimatorer vi har implementerat ovan. Vi gör detta genom att upprepa skattningen för ett antal olika utfall från våra fördelningar 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 = 10000, 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 = 10000, mu = 10, sigma2 = 4)
   mu1[i] = norm_mu_mle(x = y1)
   mu2[i] = norm_mu_mle(x = y2)
   sigma1[i] = norm_sigma2_mle(x = y1)
   sigma2[i] = norm_sigma2_mle(x = y2)
```

Visualisera samplingfördelningarna för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$ då n=10 och då n=10000 i ett histogram. Vad är dina slutsatser ?

(2) Uppgiften ovan är orealistik. I praktiken brukar vi ha bara ett stickprov, dvs vi brukar inte ha tillgång till populationen som vi hade ovan. Då, vi kan inte dra flera stickprov från populationen för att approximera samplingfördelningarna för våra estimatorer. Vi kan lösa detta problem genom att låtsas vårt enda stickprov är populationen och vi drar flera stickprov från denna falska population. Denna metod heter bootstrap. Man brukar dra **med återläggning** (dvs, `replace = TRUE` i R) och lika många punkter som i det ursprungliga stickprovet. Implementera pseudokoden nedan för att beräkna 2000 bootstrap MLE-skattningar.

```
x1 = draw Gamma(n = 10, alpha= 4, beta = 1)
x2 = draw Gamma(n = 10000, alpha= 4, beta = 1)
y1 = draw Normal(n = 10, mu = 10, sigma2 = 4)
y2 = draw Normal(n = 10000, mu = 10, sigma2 = 4)
for i in 1 to 2000 do
   beta1_mle[i] = gamma_beta_mle(x = sample(x1, 10, replace = TRUE), alpha = 4)
   beta2_mle[i] = gamma_beta_mle(x = sample(x2 , 10000, replace = TRUE), alpha = 4)
   mu1[i] = norm_mu_mle(x = sample(y1, 10, replace = TRUE))
   mu2[i] = norm_mu_mle(x = sample(y2, 10000, replace = TRUE))
   sigma1[i] = norm_sigma2_mle(x = sample(y1, 10, replace = TRUE))
   sigma2[i] = norm_sigma2_mle(x = sample(y2, 10000, replace = TRUE))
```

Visualisera boostrapfördelningarna för $\hat{\beta}_{MLE}$, $\hat{\mu}_{MLE}$ och $\hat{\sigma^2}_{MLE}$ då n=10 och då n=10000 i ett histogram. Jamför dem med samplingfördelningarna i (1). Vad är dina slutsatser ?