Maximum Likelihood Estimation and Model Fitting, CMEE MSc. Tin-Yu Hui

Practical 5 (26 Feb 2021)

Question 1 [Gamma MLE and moment estimators]

Let $X_1, X_2, \ldots, X_n$ be i.i.d gamma samples with parameters with parameters $\alpha > 0$ and $\beta > 0$. The pdf for the gamma distribution is $f_X(x) = \frac{\beta^\alpha}{\Gamma(\alpha)} x^{\alpha-1} e^{-\beta x}$, $0 < x < \infty$. Find the log-likelihood function for the two parameters.

i. Differentiate the log-likelihood function w.r.t. α and β, and show that the log-likelihood can only be maximised numerically without a closed-form solution.

ii. Another way to estimate the parameters is via the method of moments, or "moment matching". This method, as suggested by its name, estimates the parameter(s) by matching the (low-order) population moments with the sample moments. As explained on Day 1, the first two population moments are $E(X)$ and $E(X^2)$, which are functions of the parameters. Further, these population moments can be calculated from the mgf, which is $M_X(t) = \left(1 - \frac{t}{\beta}\right)^{-\alpha}$ for the case of gamma distribution. Find $E(X)$ and $E(X^2)$.

iii. What are sample moments? The first sample moment is $m_1 = \sum_{i=1}^{n} x_i$, the arithmetic average of our samples, and the second sample moment is $m_2 = \sum_{i=1}^{n} x_i^2$, the arithmetic average of our squared samples. As gamma is a two-parameter distribution, finding the first two moments will suffice. The final step is to "match" the population moments with the sample moments. That is:

$$\begin{cases} E(X) = m_1 \\ E(X^2) = m_2 \end{cases}$$

The remaining task is to find the pair of $(\tilde{\alpha}, \tilde{\beta})$ that satisfies this system of simultaneous equations. $(\tilde{\alpha}, \tilde{\beta})$ are the moment estimators.

Question 2 [Bivariate normal distribution and transformation, updated from Practical 2 Q3]

Given $\boldsymbol{X} = \begin{pmatrix} X_1 \\ X_2 \end{pmatrix}$, and $X \sim MVN(\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}, \boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix})$. The variance-covariance matrix $\boldsymbol{\Sigma}$ suggests that X_1 and X_2 both have variance of 1 and covariance ρ, $-1 < \rho < 1$. If $\rho > 0$ then the pair is positively correlated. If $\rho = 0$, then X_1 and X_2 are uncorrelated. Further, for multivariate normal r.v., $\rho = 0$ also implies independence.

i. Show that the eigenvalues of $\boldsymbol{\Sigma}$ are $\lambda_1 = 1 + \rho$ and $\lambda_2 = 1 - \rho$

ii. Please also show their associated unit eigenvectors are $\boldsymbol{v_1} = \begin{bmatrix} \frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$ and $\boldsymbol{v_2} = \begin{bmatrix} -\frac{1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} \end{bmatrix}$

iii. Let $\boldsymbol{P} = [\boldsymbol{v_1}\ \boldsymbol{v_2}] = \begin{bmatrix} \frac{1}{\sqrt{2}} & \frac{-1}{\sqrt{2}} \\ \frac{1}{\sqrt{2}} & \frac{1}{\sqrt{2}} \end{bmatrix}$, show that the inverse of $\boldsymbol{P}$ is $\boldsymbol{P^T}$.

iv. Show that $\boldsymbol{P^T \Sigma P}$ is a diagonal matrix. What else do you discover?

Decomposing $\boldsymbol{\Sigma}$ allows us to generate correlated multivariate normal r.v. from independent normal r.v.. Or, conversely, to "decorrelate" multivariate normal r.v. back into independent ones.

v. Let $\boldsymbol{Z} = \begin{pmatrix} Z_1 \\ Z_2 \end{pmatrix}$. $Z_1 \sim N(0, \lambda_1)$, $Z_2 \sim N(0, \lambda_2)$, and that Z_1 and Z_2 are independent. Then $\boldsymbol{X} = \boldsymbol{PZ}$ will follow multivariate normal distribution with $\boldsymbol{\mu} = \begin{pmatrix} 0 \\ 0 \end{pmatrix}$ and $\boldsymbol{\Sigma} = \begin{bmatrix} 1 & \rho \\ \rho & 1 \end{bmatrix}$.

```
# VALUE OF rho
rho<-0.6
# GENERATE z1 AND z2, INDEPENDENTLY
z1<-rnorm(20000, mean=0, sd=sqrt(1+rho))
z2<-rnorm(20000, mean=0, sd=sqrt(1-rho))

# TRANSFORM (z1, z2) INTO (x1, x2), VIA P-TRANSPOSE
x1<-z1/sqrt(2)-z2/sqrt(2)
x2<-z1/sqrt(2)+z2/sqrt(2)

# SCATTER PLOT OF (x1, x2)? LOOKS LIKE AN ELLIPSE?
plot(x1, x2)

# THE SAMPLE VARIANCE AND COVARIANCE OF x1 AND x2?
var(x1)
var(x2)
cov(x1, x2)
```

vi. Conversely, we can back transform the correlated $\boldsymbol{X}$ into independent normal r.v., via $\boldsymbol{Z} = \boldsymbol{P^T X}$

```
# LET'S REMOVE z1 AND z2 FROM OUR CURRENT R SESSION
rm(r1); rm(r2);

# WE BACK TRANSFORM (x1, x1) INTO (z1, z2), VIA t(P)
z1<-x1/sqrt(2)+x2/sqrt(2)
z2<-(-x1)/sqrt(2)+x2/sqrt(2)

# THE COVARIANCE BETWEEN z1 AND z2 SHOULD BE CLOSE TO ZERO
var(z1)
var(z2)
cov(z1, z2)
cor.test(z1, z2)
```

Eigen-decomposition has many applications, principle component analysis is one example.

Question 3 [ANOVA and mixed-effect model]

In this fictional example, a laboratory is developing a new treatment that can potentially make individuals "heathier". The response variable is $y_{i,j}$, which is a continuous score measuring one's health conditions. 30 individuals were recruited for this randomised trial, with half them being assigned to the treatment group, and half to the control group. The whole experiment was further repeated for 3 more times (4x30 individuals in total). Let us read the dataset into R and call it `dat`:

```
dat<-read.csv('drug.csv', header=T)

names(dat)
```

The explanatory variable is `treatment`, which indicates the treatment (0 for control, 1 for treatment) the individual was exposed to. The primary goal of this experiment is to study the effect of `treatment` (our fixed effect), thus the estimation of its coefficient as well as the associated C.I. is crucial. To test for the effect, one can choose to perform a one-way anova on all data points, or to perform four anovas separately on each `replicate`. I believe by now you are able to fit these linear models via log-likelihood or via the built-in `lm()` function. Unfortunately neither of the procedures is satisfactory, because the former neglects between-replicate variation, and the latter involves making multiple comparisons which can be difficult to deal with. One may further suggest to use a two-way anova by setting replicate as another fixed effect. It may help solve the issue of pseudo-replication, but the downside is that it creates three additional parameters. In other words, we spend three degrees of freedom explaining terms that we are not interested in. A more appropriate model is to treat `replicate` as a random effect.

Random effects have factor levels that drawn from a large population in which individuals differ in many ways, but we do not know exactly how or why they differ. These variations are usually beyond the control of the experimenter and beyond their immediate interest [Mick Crawley, GLM course]. Our replicate variable fits these descriptions very well, that we do not know what caused the

variation among our replicates, nor did we artificially make them different. All we would like is to incorporate this effect into our model. Our model, with both fixed and random effects, is a mixed-effect model:

$$y_{i,j} = a + b * treatment_{i,j} + \tau_j + \epsilon_{i,j}$$

$$where\ i = 1, 2, \dots, 30;\ j = 1,2,3,4$$

We have the same definition to the individual error term as before:

$$\epsilon_{i,j} \sim N(0, \sigma^2), iid$$

In the model you will find τ_j, the random effect. There are four of them here: $\tau_1, \tau_2, \tau_3, \tau_4$, one for each `replicate`. All individuals from the same replicate will share the same τ_j. Here we further assume that τ_j follows another normal distribution:

$$\tau_j \sim N\left(0, \sigma^2_{random}\right), iid$$

Now we do not wish to estimate the values of $\tau_1, \tau_2, \tau_3, \tau_4$ as they are unimportant. Instead, we only wish to estimate σ_{random} to summarise the among-replicate variation. The advantage of this model is that we are only adding 1 additional parameter to the model, as opposed to 3 (or number of factor levels-1) in the two-way fixed-effect model. Now let us construct the likelihood function for this four-parameter model, which is by definition the joint pdf of our observations:

$$L(a, b, \sigma, \sigma_{random}) = f\left(y_{1,1}, \dots, y_{30,1}, y_{1,2}, \dots, y_{30,2}, \dots, y_{1,3}, \dots, y_{30,3}, y_{1,4}, \dots, y_{30,4}\right)$$

Usually we will then express the joint pdf as the product of individuals pdfs. This step is only valid if the observations are independent. In this case, however, our observations are not entirely independent, as those within the same replicate are correlated (note: they share the same τ_j). Luckily, we are still permitted to break down the joint pdf into four chunks, as observations from different replicates are independent:

$$L(a, b, \sigma, \sigma_{random}) = f_{rep1}\left(y_{1,1}, \dots, y_{30,1}\right) f_{rep2}\left(y_{1,2}, \dots, y_{30,2}\right) f_{rep3}\left(y_{1,3}, \dots, y_{30,3}\right) f_{rep4}\left(y_{1,4}, \dots, y_{30,4}\right)$$

Without loss of generality, we discuss only $f_{rep1}\left(y_{1,1}, \dots, y_{30,1}\right)$, the joint pdf of the observations from the first replicate. We do not instantly know how this joint pdf or individual pdf $f\left(y_{i,1}\right)$ look like. But when the value of τ_1 is known (or given), then $y_{i,1}|\tau_1 \sim N(a + b * treatment_{i,1} + \tau_1, \sigma^2)$. Further, given τ_1, the observations within the first replicate are conditionally independent. We can rewrite the joint pdf for the first replicate as

$$f_{rep1}\left(y_{1,1}, \dots, y_{30,1}\right) = \int f_{rep1}\left(y_{1,1}, \dots, y_{30,1}|\tau_1\right) f(\tau_1) d\tau_1$$

$$= \int f\left(y_{1,1}|\,\tau_1\right) f\left(y_{2,1}|\tau_1\right) \dots f\left(y_{30,1}|\tau_1\right) f(\tau_1) d\tau_1$$

The first line holds because of the law of total probability. The nuisance τ_1 is integrated (marginalised) out. The second line holds because of conditional independence. Let us translate the model into R codes:

```
# WITHIN-REPLICATE LIKELIHOOD

within.rep.like<-function(parm, dat)

{
```

```
# DEFINE DATA
y<-dat$y
treatment<-dat$treatment==1
      f<-function(tau)
      {prod(dnorm(y, mean=parm[1]+parm[2]*treatment+tau,
sd=parm[3]))*dnorm(tau, mean=0, sd=parm[4])}
      f<-Vectorize(f, 'tau')
return(integrate(f, low=-Inf, upper=Inf, subdivisions=4000L)$value)
}
```

You should notice that this within-replicate likelihood involves an (one-dimensional) integral. The overall likelihood is the product of four within-replicate likelihoods, which means four integrals need to be evaluated for a set of parameters. We use the built-in `integrate()` function here.

```
# OVERALL LOG-LIKELIHOOD ACROSS FOUR REPLICATES. HARD-CODED.
overall.log.like<-function(parm, dat)
{
temp1<-within.rep.like(parm, dat[dat$replicate==1,])
temp2<-within.rep.like(parm, dat[dat$replicate==2,])
temp3<-within.rep.like(parm, dat[dat$replicate==3,])
temp4<-within.rep.like(parm, dat[dat$replicate==4,])
return(log(temp1)+log(temp2)+log(temp3)+log(temp4))
}
```

The next step is to maximise the overall log-likelihood to find the parameter estimates.

```
# MAXIMISE THE OVER LOG-LIKELIHOOD
initial.parm<-c(40, 2.3, 5, 6)
lower<-c(-50, -50, 0.001, 0.001)
upper<-c(100, 100, 100, 100)

# MAXIMISE WITH optim()
M<-optim(initial.parm, overall.log.like, dat=dat, method='L-BFGS-B',
      lower=lower, upper=upper,
      control=list(fnscale=-1), hessian=T)

# MAXIMISE WITH spg() FROM library(BB)
```

```
spg(initial.parm, overall.log.like, dat=dat,
      lower=lower, upper=upper,
      control=list(maximize=T))
```

I hope you get $\hat{b} = 2.33$. The treatment on average improves the health score by 2.33 units. I would also like to pay attention to the two sd estimates $\hat{\sigma}$ and $\hat{\sigma}_{random}$, as they explain the proportions of variation contributed by the individuals and the random effect.

Note:

We can of course continue the analysis by testing $H_0: b = 0$ or finding its C.I. via profiling. But I think the key message is that likelihood functions can be in forms of integrals (or even multidimensional integrals if there are more than one random effect). It can be computationally challenging to evaluate the log-likelihood value at a parameter value, let alone maximise it. Computational methods have been developed to work around the problem, or to approximate the integrals. In practice there are other ways to solve mixed-effect models, such as `lmer()` from `lme4` package.