MTH 511a - 2020: Lecture 15

Instructor: Dootika Vats

1 No closed-form MLEs

In the last lecture we introduced maximum likelihood estimation. Obtain MLE estimates for a problem requires maximizing the likelihood. However, it is possible that no analytical form of the maxima is possible!

This is a common challenge in many models and estimation problems, and requires sophisticated optimization tools. In the next few weeks, we will go over some of these optimization methods.

1.1 Examples

Example 1 (Gamma Distribution). Let $X_1, \dots, X_n \overset{iid}{\sim} \text{Gamma}(\alpha, 1)$. The likelihood function is

$$\begin{aligned} L(\alpha|x) &= \prod_{i=1}^{n} \frac{1}{\Gamma(\alpha)} x_i^{\alpha-1} e^{-x_i} \\ &= \frac{1}{\Gamma(\alpha)^n} \prod_{i=1}^{n} x_i^{\alpha-1} e^{-\sum x_i} \\ \Rightarrow l(\alpha) := \log L(\alpha|x) &= -n \log(\Gamma(\alpha)) + (\alpha - 1) \sum_{i=1}^{n} \log x_i - \sum_{i=1}^{n} x_i \end{aligned}$$

Taking first derivative

$$\frac{dl(\alpha)}{d\alpha} = -n \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} + \sum_{i=1}^{n} \log X_i \overset{set}{=} 0$$

Taking second derivative

$$\frac{d^2\, l(\alpha)}{d\alpha^2} = -n\frac{d^2}{d\alpha^2}\log(\Gamma(\alpha)) < 0 \quad \text{(polygamma function of order 1 is > 0)}$$

Here

$$\frac{d^2}{d\alpha^2}\log(\Gamma(\alpha))$$

is the polygamma function of order 1, which is always positive (look it up). So we know that the function is concave and a unique maximum exists, but not available in closed form. We cannot get an analytical form of the MLE for α. In such cases, we will use optimization methods.

2 Numerical optimization methods

We will cover three optimization methods:

- Newton-Raphson method
- Gradient ascent (descent)
- The MM algorithm

A general numerical optimization problem is framed in the following way. Let $f(\theta)$ be an *objective function* that is the main function of interest and needs to be either maximized or minimized. Then, we want to solve the following maximization

$$\theta_* = \arg\max_{\theta} f(\theta)$$

All the above three algorithms are such that they generate a sequence of $\{\theta_{(k)}\}$ such that the goal is for $\theta_{(k)} \to \theta_*$ in a deterministic manner (non-random convergence).

All methods that we will learn will find a local optima. Some will guarantee a local maxima, but not guarantee a global maxima, some will guarantee a local optima (so max or min), but not a global maxima. If the objective function is concave, then all methods will guarantee a global maxima!

Recall that a (univariate) function f is concave if $f'' < 0$ and a (multivariate) function f is concave if its Hessian is negative definite: for all $a \neq 0 \in \mathbb{R}^p$,

$$a^T \nabla^2 f a < 0\,.$$

2.1 Newton-Raphson's method

To solve this optimization problem, consider starting at a point $\theta_{(0)}$. Then subsequent elements of the sequence are determined in the following way. Suppose that the objective function f is such that a second derivative exists.

Since $f(\theta)$ is maximized at the unknown θ_*, $f'(\theta_*) = 0$. Applying first oder Taylor's expansion (and ignoring higher orders) to $f'(\theta_*)$ about the current iterate $\theta_{(k)}$

$$
\begin{aligned}
f'(\theta_*) &\approx f'(\theta_{(k)}) + (\theta_* - \theta_{(k)}) f''(\theta_{(k)}) = 0 \\
\Rightarrow \theta_* &\approx \theta_{(k)} - \frac{f'(\theta_{(k)})}{f''(\theta_{(k)})},
\end{aligned}
$$

where the approximation is best when $\theta_{(k)} = \theta^*$ and the approximation is weak when $\theta_{(k)}$ is far from θ_*. Thus, if we start from an arbitrary point using successive updates of the right hand side, we will get closer and closer to θ_*.

The Newton-Raphson method is

$$
\theta_{(k+1)} = \theta_{(k)} - \frac{f'(\theta_{(k)})}{f''(\theta_{(k)})}
$$

This works because when $f'(\theta_k) < 0$, the function is increasing at $\theta_{(k)}$, and thus Newton-Raphson increases $\theta_{(k)}$; and vice-versa. You stop iterating when $|\theta_{(k+1)} - \theta_{(k)}| < \epsilon$ for some chosen tolerance ϵ.

> *If the objective function is concave, the N-R method will converge to the global maxima. Otherwise it converges to a local optima or diverges!*

Example 2 (Gamma distribution continuted). Our objective function is the log-likelihood:

$$
f(\alpha) = -n \log(\Gamma(\alpha)) + (\alpha - 1) \sum_{i=1}^{n} \log x_i - \beta \sum x_i
$$

First derivative

$$
f'(\alpha) = -n \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} + \sum_{i=1}^{n} \log X_i
$$

Second derivative

$$
f''(\alpha) = -n \frac{d^2}{d\alpha^2} \log(\Gamma(\alpha)) < 0 .
$$

Thus the log-likelihood is concave, which implies there is a global maxima! The Newton-Raphson algorithm will converge to this global maxima.

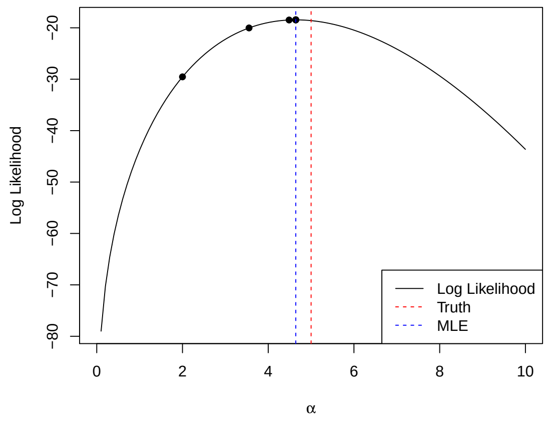
Start with a reasonable starting value: α_0. Then iterate with

$$
\alpha_{(k+1)} = \alpha_{(k)} - \frac{f'(\alpha_{(k)})}{f''(\alpha_{(k)})}
$$

Polygamma functions are in `psi` function in the `pracma` R package.

What is a good starting value α_0? Well, we know that the mean of a Gamma$(\alpha, 1)$ is α, so a good starting value is $\alpha_0 = n^{-1}\sum_{i=1}^{n} X_i$.

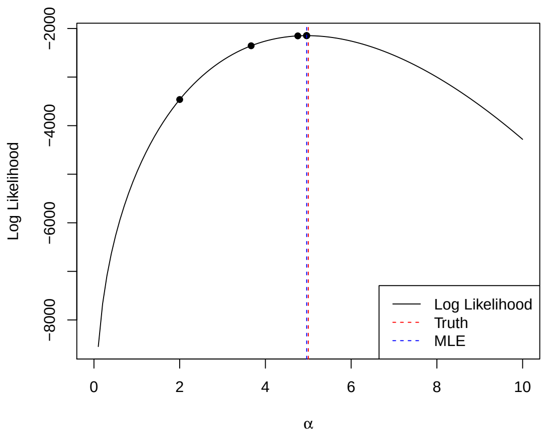
```
####################################################
### MLE for Gamma(alpha, 1)
####################################################
set.seed(100)
library(pracma) #for psi function

####################################################
# riginal data sample size is small first
# The NR methods estimates the MLE. Here the
# blue and red lines will not match because
# the data is not large enough for the consistency of
# the MLE to kick in.


alpha <- 5 #true value of alpha
n <- 10 # actual data size is small first
dat <- rgamma(n, shape = alpha, rate = 1)

alpha_newton <- numeric()
epsilon <- 1e-8 #some tolerance level preset
alpha_newton[1] <- 2 #alpha_0
count <- 1
tol <- 100 # large number
while(tol > epsilon)
{
  count <- count + 1

  #first derivative
  f.prime <- -n*psi(k = 0, alpha_newton[count - 1]) + sum(log(dat))

  #second derivative
  f.dprime <- -n*psi(k = 1, alpha_newton[count - 1])
  alpha_newton[count] <- alpha_newton[count - 1] - f.prime/f.dprime
  tol <- abs(alpha_newton[count] - alpha_newton[count-1])
}
alpha_newton
# [1] 2.000000 3.552357 4.487264 4.640581 4.643535 4.643536 4.643536

#Plot the log.likelihood for different values of alpha
alpha.grid <- seq(0, 10, length = 100)
log.like <- numeric(length = 100)
for(i in 1:100)
{
  log.like[i] <- sum(dgamma(dat, shape = alpha.grid[i], log = TRUE))
}
plot(alpha.grid, log.like, type = 'l', xlab = expression(alpha), ylab =
    "Log Likelihood")
abline(v = alpha, col = "red", lty = 2)
for(t in 1:count)
{
  points(alpha_newton[t], sum(dgamma(dat, shape = alpha_newton[t], log =
      TRUE)), pch = 16)
}
abline(v = tail(alpha_newton[count]), col = "blue", lty = 2)
legend("bottomright", legend = c("Likelihood", "Truth", "MLE"), lty =
    c(1,2,2), col = c("black", "red", "blue"))
```



Note the impact of the sample size of the original data. The Newton-Raphson algorithm converges to the MLE. If the data size is small, the MLE may not be close to the truth. This is why we see that the blue and red lines are far from each other. However, when increase the observed data to be 1000 observations, the consistency of the MLE should kick in and we expect to see the blue and red lines to be similar (below).

```
####################################################
# Increasing original data sample size.
# Now the MLE is closer to the "truth"
# and our NR method obtains the MLE.
# Blue and red lines should match a lot

# Randomly generate data
alpha <- 5 #true value of alpha
n <- 1000 # actual data size is small first
dat <- rgamma(n, shape = alpha, rate = 1)
alpha_newton <- numeric()
epsilon <- 1e-8 #some tolerance level preset
alpha_newton[1] <- 2 #alpha_0
count <- 1
```

```r
tol <- 100 # large number
while(tol > epsilon)
{
  count <- count + 1
  f.prime <- -n*psi(k = 0, alpha_newton[count - 1]) + sum(log(dat))
  f.dprime <- -n*psi(k = 1, alpha_newton[count - 1])
  alpha_newton[count] <- alpha_newton[count - 1] - f.prime/f.dprime
  tol <- abs(alpha_newton[count] - alpha_newton[count-1])
}
alpha_newton
# [1] 2.000000 3.666788 4.755252 4.957521 4.962359 4.962361 4.962361

#Plot the log.likelihood for different values of alpha
alpha.grid <- seq(0, 10, length = 100)
log.like <- numeric(length = 100)
for(i in 1:100)
{
  log.like[i] <- sum(dgamma(dat, shape = alpha.grid[i], log = TRUE))
}
plot(alpha.grid, log.like, type = 'l', xlab = expression(alpha), ylab =
    "Log Likelihood")
abline(v = alpha, col = "red", lty = 2)
for(t in 1:count)
{
  points(alpha_newton[t], sum(dgamma(dat, shape = alpha_newton[t], log =
      TRUE)), pch = 16)
}
abline(v = tail(alpha_newton[count]), col = "blue", lty = 2)
legend("bottomright", legend = c("Likelihood", "Truth", "MLE"), lty =
    c(1,2,2), col = c("black", "red", "blue"))
```



Example 3 (Location Cauchy distribution). Consider the location Cauchy distribution with mode at $\mu \in \mathbb{R}$. The goal is to find the MLE for μ.

$$f(x|\mu) = \frac{1}{\pi} \frac{1}{(1 + (x-\mu)^2)} \,.$$

First, we find the log-likelihood

$$L(\mu|X) = \prod_{i=1}^{n} f(X_i|\mu) = \pi^{-n} \prod_{i=1}^{n} \frac{1}{1 + (x_i - \mu)^2}$$

$$\Rightarrow l(\mu) := \log L(\mu|X) = -n \log \pi - \sum_{t=1}^{n} \log(1 + (X_i - \mu)^2) = f(\mu)\,.$$

It is evident that a closed form solution is difficult. So, we find the derivates.

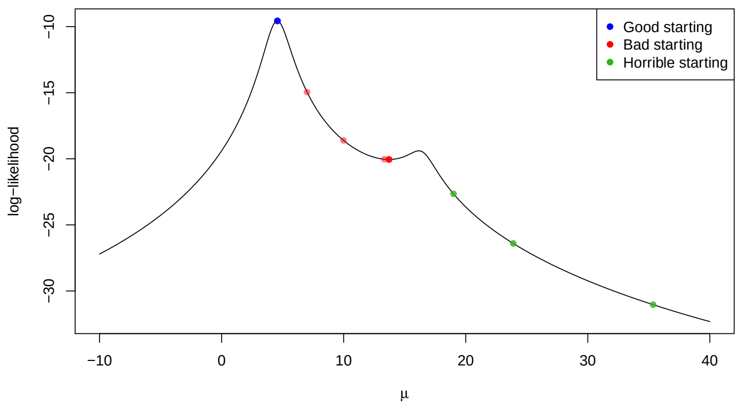
$$f'(\mu) = 2 \sum_{i=1}^{n} \frac{X_i - \mu}{1 + (X_i - \mu)^2}\,.$$

$$f''(\mu) = 2 \sum_{i=1}^{n} \left[2 \frac{(X_i - \mu)^2}{[1 + (X_i - \mu)^2]^2} - \frac{1}{1 + (X_i - \mu)^2} \right] ,$$

which may be positive or negative. So this is not a concave function, so we will be careful in choosing starting values.

1. Set $\mu_0 = \text{Median}(X_i)$ since the mean of Cauchy does not exist and the Cauchy centered at μ is symmetric around μ.
2. Determine:
$$\mu_{(k+1)} = \mu_{(k)} - \frac{f'(\mu_{(k)})}{f''(\mu_{(k)})}$$
3. Stop when $|\mu_{(k+1)} - \mu_{(k)}| < \epsilon$ for a chosen tolerance level ϵ.

The code for this is attached in the material, but below is a figure of what the log-likelihood looks like and the behavior of the Newton-Raphson algorithm for different starting values. It shows that choosing a good starting value is very important.



Newton-Raphson in Higher Dimensions

The NR method can be found in the same way using the multivariate Taylor's expansion. Let $\theta = (\theta_1, \theta_2, \ldots, \theta_p)$. Then first let ∇f denote the gradient of f and $\nabla^2 f$ denote the Hessian. So

$$\nabla f = \begin{bmatrix} \dfrac{\partial f}{\partial \theta_1} \\ \vdots \\ \dfrac{\partial f}{\partial \theta_p} \end{bmatrix} \quad \text{and} \quad \nabla^2 f = \begin{bmatrix} \dfrac{\partial^2 f}{\partial \theta_1^2} & \dfrac{\partial^2 f}{\partial \theta_1 \theta_2} & \cdots \\ \vdots & \vdots & \vdots \\ \dfrac{\partial^2 f}{\partial \theta_p \theta_1} & \cdots & \vdots \dfrac{\partial^2 f}{\partial \theta_p^2} \end{bmatrix}$$

Then, the function f is concave if $|\nabla^2 f| < 0$, where $|\cdot|$ is the determinant. *So always check that first to know if there is a unique maximum.*

Using a similar multivariate Taylor series expansion, the Newton-Raphson update solves the system of linear equations

$$\nabla f(\theta_{(k)}) + \nabla^2 f(\theta_k)(\theta_{k+1} - \theta_k) = 0\,.$$

If $\nabla^2 f(\theta_{(k)})$ is invertible, then you have that

$$\theta_{(k+1)} = \theta_{(k)} - \left[\nabla^2 f(\theta_{(k)})\right]^{-1} \nabla f(\theta_{(k)})\,.$$

Iterations are stopped with when $\|\theta_{(k+1)} - \theta_{(k)}\| < \epsilon$ for some chosen tolerance level, ϵ.

3 Questions

1. Can you implement the the Newton-Raphson procedure for linear regression and ridge regression?
2. What are some of the issues in implementing Newton-Raphson? Can we use it for any problem?
3. If the function is not concave and different starting values yield convergence to different points (or divergence), then what do we do?