Mathematical Statistics II

Maximum Likelihood Estimation

Jesse Wheeler

Contents

1 Introduction **1**

2 Likelihood: an introduciton **1**

3 Joint Probabilities **5**

4 Examples **6**

5 Numeric Optimization **9**

6 Constraint Optimization **17**

1 Introduction

Overview

- The next approach we will discuss is Maximum Likelihood Estimation (MLE).
- As we will see, the MLE has several desirable properties, and as a result is often favored over approaches like the method of moments.
- The material for this section largely comes from Chapter 8.5 of Rice (2007), and various sections in Pawitan (2001).

2 Likelihood: an introduciton

What is likelihood?

- The term "likelihood" is often used colloquially to mean something analogous to probability. E.g., "What is the likelihood that it rains tomorrow?"
- When we use this term in statistics / mathematics, we mean something specific that isn't the same thing as probability.
- The use of the term "likelihood" was first made by R. A. Fisher, who was the architect and primary proponent of "likelihood-based-inference".
- We will start with the treatment of likelihood in the text "In all Likelihood" (Pawitan, 2001), which is a fantastic resource on the subject. (This will lead to some review...)

Coin Flips
We will revisit this example, as it is a great starting point to connect with existing understanding. Consider flipping a coin $N = 10$ times.

- In a probability class, we might assume the probability of heads $\theta = p$ is some number, say $p = 0.4$.
- If X is the number of heads, then $P_\theta(X = 8) = 0.011$ is something we can calculate exactly. In any interpretation of probability, this means that if we repeat the experiment 10,000 time, we expect around 110 times, the experiment will have 8 heads.
- Now what if, in a single experiment, we observe $X = 8$? **Based on this information alone**, what can we say about the parameter p? Information we have about p is *incomplete*(we've already done a pure frequentist interpretation of this, as well as a Bayesian interpretation. The MoM estimate is also easy to compute).
- However, we can conclude that p **cannot be zero or one**. This fact is available *deductively*, since if p were zero or one, then $X = 0$ or $X = 10$.
- Similarly, we are fairly certain that p is not close to zero, since observing $X = 10$ would be extremely unlikely in this case. For instance, the probability of observing $X = 8$ if $p = 0.10$ is only 3.6×10^{-7}, meaning we would only expect about 1 out of 3 million replicates to give $X = 8$ (or similar for observing $X \in \{8, 9, 10\}$).
- In contrast, $\theta = 0.6$ or $\theta = 0.7$ are *likely*, since $P_\theta(X = 8)$ would be 0.12, 0.23, respectively.
- Thus, we have found a *deductive* way of comparing different values of p, based on the observed data: Compare the probability of observed data under different values of p.
- Concretely, we take the probability model $P_\theta(X = 8)$, and treat it as a function of θ.
- Note this is opposite of what we did last semester, where we treat the pmf for a fixed θ as a function of x (the possible output).
- This is how we define the likelihood function (for this model, and more generally, for any pmf):

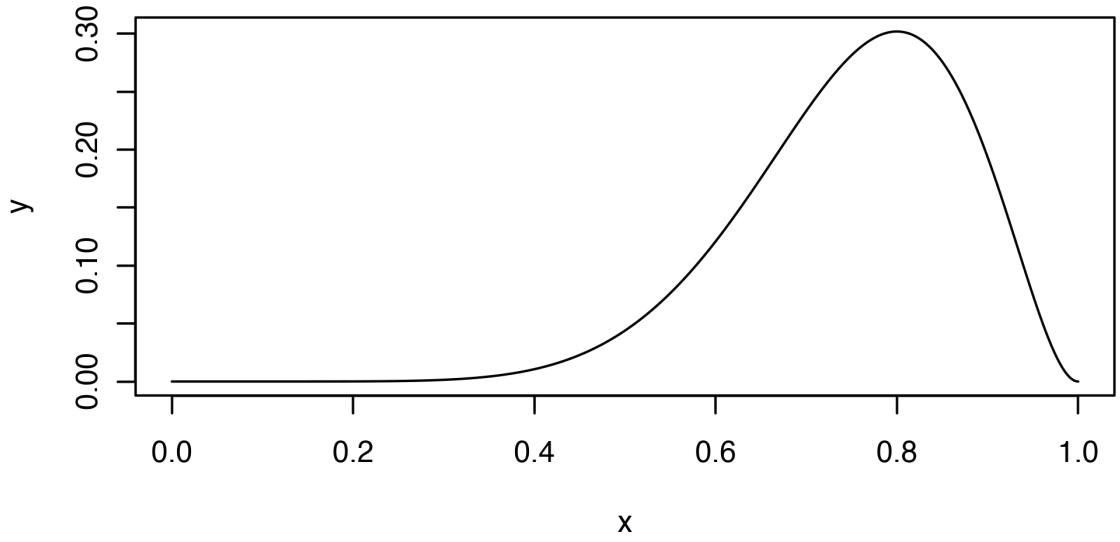
$$L(\theta) = P_\theta(X = 8) = f(x^*; \theta).$$

- For our specific coin-flipping example with $N = 10$, $X = 8$, the likelihood function is

$$L(\theta) = P_\theta(X = 8).$$

- This is plotted in the following way:

```
x <- seq(1e-8, 1-1e-8, length.out = 1000)
y <- dbinom(8, 10, x)
plot(x = x, y = y, type = 'l')
```



- From the figure, we see that p is unlikely to be less than 0.5,or greater than 0.95.
- Given the data alone (no prior), we should prefer a value somewhere in the middle of these values.
- We still have some uncertainty about the value of p, but the likelihood gives us a numerical way to compare values of θ. Stochastic uncertainty as a result of sampling is captured in the likelihood function $L(\theta)$.
- *The likelihood is not a probability.* Though it came from a probability, the likelihood function (a function of θ) does not satisfy the requirements to be a probability. In our previous example, we have:
$$\int_0^1 L(\theta)\, d\theta = 1/11 \neq 1.$$
- For discrete probability, the likelihood was *continuous*. Discrete likelihoods are possible, arising when we want to select from a list $\{\theta_1, \theta_2, \ldots\}$.
- The idea behind maximum likelihood estimation (MLE) is simple: our estimate is the value of θ that maximizes the likelihood function $L(\theta)$.
- The MLE is considered a *frequentist* approach. Why? It quantifies a maximum belief about a parameter, which is more Bayesian in nature than Frequentist.
- As we'll see later, the MLE has nice theoretical Frequentist *properties*, and as a result can be justified via the frequentist paradigm.
- Still, it has close connection to Bayesian estimation and interpretation. In fact, we'll discuss connections between the MLE and Bayesian statistics later.
- Often, maximizing the likelihood directly is challenging, so we maximize the log-likelihood instead.
- Other times, the likelihood has to be maximized numerically.

MLE of coin toss problem
Suppose we have $N = 10$ total tosses, and n total heads. Find the MLE of p, the probability of heads.

- First, describing what the likelihood function is. We already did this for this model, but a quick review, reinforcing common notation.
- We'll use a binomial(N, p) model; the parameter of interest is $\theta = p$.
- Let X be a random variable, denoting the number of heads. Thus, pmf of the model is:

$$P(X = x) = f(x; \theta) = \binom{N}{x} p^x (1-p)^{N-x}.$$

- We assume we observe a specific dataset x^*. In this case, we observe $x^* = n$. Fixing the model at the observed data, we can describe the probability of the observed data as a function of $\theta = p$:

$$L(\theta) = f(x^*; \theta) = \binom{N}{n} p^n (1-p)^{N-n},$$

  and the MLE is:

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, L(\theta),$$

  meaning that our estimate $\hat{\theta}$ is the *argument* that *maximizes* the likelihood function $L(\theta)$.
- As often is the case, this will be easiest if we take the natural logarithm, giving the log-likelihood. Note this is a monotonic transformation, so the argument that maximizes the log-likelihood is the same as the argument that maximizes the likelihood.
- Typically, we denote the log-likelihood using $\ell(\theta)$. In latex, this symbol is given by `\ell`.

$$\hat{\theta} = \underset{\theta}{\operatorname{argmax}}\, L(\theta) = \underset{\theta}{\operatorname{argmax}} \log\big(L(\theta)\big) = \underset{\theta}{\operatorname{argmax}}\, \ell(\theta).$$

- For our specific model, the log-likelihood is:

$$\ell(\theta) = \log \binom{N}{n} p^n (1-p)^{N-n} = \log\left(\binom{N}{n}\right) + n\log(p) + (N-n)\log(1-p).$$

- Now we see another recurring theme when finding the MLE: we end up with terms that don't impact the maximization. That is, the binomal term out front does not change the maximum value of p, so it can be ignored.
- To maximize, we take the derivative, and set equal to zero:

$$\frac{\partial \ell}{\partial \theta} = \frac{n}{p} - \frac{N-n}{1-p} = 0,$$

  or

$$\frac{n}{p} = \frac{N-n}{1-p} \implies n - np = Np - np.$$

- Solving for p, we get the MLE:

$$p = \frac{n}{N} \implies \hat{p} = \frac{n}{N} = \underset{\theta}{\operatorname{argmax}}\, L(\theta).$$

- A few notes:

– The MLE matches the frequentist-derived estimate. In most models, a first-principle estimate using frequentist interpretation of probability is not available, but the MLE will still match what the estimate *would be* if such an estimate were available.

– Setting a derivative equal to zero and solving only gives a critical point. Why did I assume it was a maximum? First, we plotted like likelihood function, and it is clear any critical point will be a maximum. More importantly, many of the classic probability models are part of what is known as the *exponential family*. In this family, all critical points of $L(\theta)$ will correspond to maximums. Next, theory we will establish later shows that as the number of observations increases, the likelihood function will be concave down around the MLE, suggesting that for even complex models, critical points in the region of high-likelihood will always be maximums.

Continuous models

- The interpretation of the likelihood function as the "the probability of the observed data x^*, considered as a function of θ" makes perfect sense in the discrete model case.
- For continuous models, the technical issue arises that the probability of any point value x is zero.
- We resolve the problem similar to what was done in Math 4450 and the John Rice text: approximate the probability by discretizing into small, discrete intervals:

  $$x^* \in (x^* - \epsilon/2, x^* + \epsilon/2),$$

  thus, the probability of observing something ϵ-close to the data is:

  $$\begin{aligned} L(\theta) &= P_\theta\big(X \in (x^* - \epsilon/2, x^* + \epsilon/2)\big) \\ &= \int_{x^*-\epsilon/2}^{x^*+\epsilon/2} f(x;\theta)\, d\theta \approx \epsilon f(x^*;\theta). \end{aligned}$$

- Then, since the likelihood is only meaningful up to a constant (we will discuss likelihood ratios later), then this has the same behavior as $L(\theta) = f(x^*;\theta)$.
- There are more advanced approaches to this problem, but this simple argument justifies the use of the pdf of a continuous random variable as the likelihood $L(\theta)$.
- **Going forward:** we once again will generalize a model $f(x;\theta)$ to mean either the pmf or pdf of a random variable. I will often say "density" as a blanket term, even if this corresponds to a pmf, not a density.
- Further, when we "integrate" a density, this means either:

  $$\int f(x;\theta)\, dx, \quad \text{If continuous}$$

  or

  $$\sum_x f(x;\theta), \quad \text{If discrete.}$$

3 Joint Probabilities

Likelihood with multiple observations

- Often the data we observe is multi-dimensional, rather than summarized as a single observation.

- In this case, the likelihood θ is still determined via the joint model:

$$L(\theta) = f(x^*; \theta) = f_{X_{1:N}}(x_1^*, x_2^*, \ldots, x_N^*; \theta).$$

- We are mostly focused in this class in the case were the observations are independent, meaning the likelihood factors:

$$L(\theta) = \prod_{i=1}^{N} f_{X_i}(x_i^*; \theta).$$

- We often further simplify this by assuming the data are identically distributed:

$$L(\theta) = \prod_{i=1}^{N} f_{X_1}(x_i^*; \theta).$$

- As we've seen, it's generally easier to maximize the log-likelihood. In the IID case:

$$\ell(\theta) = \log \prod_{i=1}^{N} f_{X_1}(x_i^*; \theta) = \sum_{i=1}^{n} \log f_{X_1}(x_i^*; \theta).$$

4 Examples

Examples of finding the MLE

Traffic data: Poisson Model
Returning to a motivating example, suppose we model traffic accidents in a given week as $X_1, X_2, \ldots, X_N$, where the data are iid Poisson(λ). Obtain the MLE for λ.

- If X_i are iid poisson, then the pmf of a single observation is:

$$f(x;\, \theta) = P_\theta(X_i = x) = \frac{\lambda^x e^{-\lambda}}{x!}.$$

- In this case, the parameter set of interest, θ is a single value: $\theta = \lambda$.
- Due to the IID assumption, the likelihood of each observation is given by:

$$L(\theta) = f_{X_{1:N}}(x_{1:N}^*;\, \theta) = \prod_{i=1}^{N} \frac{\lambda^{x_i^*} e^{-\lambda}}{x_i^*!}.$$

- We want to maximize this function with respect to λ; this is easier done after taking the log:

$$\begin{aligned}
\ell(\theta) &= \log f_{X_{1:N}}(x_{1:N}^*;\, \theta) \\
&= \sum_{i=1}^{N} \log f_{X_1}(x_i^*;\, \theta) \\
&= \sum_{i=1}^{N} \left(x_i^* \log \lambda - \lambda - \log(x_i^*!) \right) \\
&= \log \lambda \sum_{i=1}^{N} x_i^* - N\lambda - \sum_{i=1}^{N} \log(x_i^*!) \\
&= \log \lambda \sum_{i=1}^{N} x_i^* - N\lambda - c(x^*).
\end{aligned}$$

- In the last step, we replaced the sum of log-factorials with some constant function $c(x^*)$. As a function of θ, the maximum of $\ell(\theta)$ does not change with this constant, so it can be ignored for the purposes of maximization.
- Now to find the maximum, we can take the derivative and set equal to zero:

$$\ell'(\theta) = \frac{1}{\lambda}\sum_{i=1}^{N} x_i^* - N = 0,$$

  Thus,

$$\hat{\lambda} = \frac{1}{N}\sum_{i=1}^{N} x_i^* = \bar{x}_N^*.$$

- We formally need to check if this is indeed a maximum and not just a critical point. The second derivative with respect to λ is always negative (since x_i^* is non-negative), implying that it is indeed a maximum.
- We'll see later that as $N \to \infty$, then the likelihood function is *always* concave-down around the maximum, under fairly week conditions.
- We could also plot the likelihood / log-likelihood, which is informative on it's own right.
- Finally, note that this is the same estimator that was obtained via the MoM approach. This sometimes happens.

Two parameter model: Gaussian model
Suppose we model observations $X_1, \ldots, X_N$ as IID $N(\mu, \sigma^2)$ random variables. Find the MLE of $\theta = (\mu, \sigma^2)$.

- Though we have two parameters, the approach for finding the MLE is similar: find the likelihood function of the entire dataset, take the logarithm, and the compute (partial) derivatives and set equal to zero.
- In the final step, we might end up with a system of equations rather than just a single equation.
- There's also an interesting feature of this example that is applicable elsewhere: should we find the MLE of σ, or the MLE of σ^2? Fortunately, you get the same result either way (we'll show this later).
- In this case, either approach is straightforward. As practice, you may want to consider finding the MLE of σ^2. Here, we will treat σ as the parameter of interest, though sometimes it's easier to differentiate with respect to something like: $\theta_2 = \sigma^2$.
- Because of the IID assumption joint-likelihood function is:

$$L(\mu, \sigma) = f(x^*;\, \mu, \sigma) = \prod_{i=1}^{N} \frac{1}{\sigma\sqrt{2\pi}} \exp\Big(-\frac{1}{2}\big((x_i^* - \mu)/\sigma\big)^2\Big),$$

- The log-likelihood is therefore:

$$\begin{aligned}\ell(\mu, \sigma) &= \sum_{i=1}^{N} \Big(\log(1) - \log(\sigma\sqrt{2\pi}) - \frac{1}{2}\big((x_i^* - \mu)/\sigma\big)^2\Big) \\ &= -n\log(\sigma) - \frac{n}{2}\log 2\pi - \frac{1}{2\sigma^2}\sum_{i=1}^{N}(x_i^* - \mu)^2.\end{aligned}$$

- The partial derivatives are given by:

$$\frac{\partial \ell}{\partial \mu} = \frac{1}{\sigma^2}\sum_{i=1}^{N}(x_i^* - \mu) = 0$$
$$\frac{\partial \ell}{\partial \sigma} = -\frac{n}{\sigma} + \sigma^{-3}\sum_{i=1}^{N}(x_i^* - \mu)^2.$$

- Because $\sigma \geq 0$ (and zero if and only if the data are the same value), the first equation will only be zero if $\sum_{i=1}^{N}(x_i^* - \mu) = 0$, or if $\mu = \bar{x}_N^*$, which is independent of σ. Thus, the MLE is:

$$\hat{\mu} = \bar{x}_N^*.$$

- For the second equation, the partial derivative is zero if and only if

$$\frac{n}{\sigma} = \sigma^{-3}\sum_{i=1}^{N}(x_i^* - \mu)^2,$$

  or if

$$\sigma^2 = \frac{1}{n}\sum_{i=1}^{N}(x_i^* - \mu)^2.$$

  Since we are looking for the MLE of $\theta = (\theta, \sigma)$, we can now replace μ with it's maximizer $\hat{\mu} = \bar{x}_N^*$, giving us:

$$\hat{\sigma} = \sqrt{\frac{1}{n}\sum_{i=1}^{N}(x_i^* - \hat{\mu})^2} = \sqrt{\frac{1}{n}\sum_{i=1}^{N}(x_i^* - \bar{x}_N^*)^2}.$$

- Like the Poisson example, the MLE for the iid normal model has the same estimator as the MoM procedure.
- The sampling distribution for θ is therefore:

$$\hat{\mu} \sim N(\mu, \sigma^2/N), \quad N\hat{\sigma}^2/\sigma^2 \sim \chi^2_{N-1},$$

  and the two distributions are independent.

Plotting Normal Likelihood

- The likelihood function (not just to point estimate) will be used to measure uncertainty.
- For models with a single parameter, we often plot the likelihood curve.
- With more than one parameter, however, we have a *likelihood surface*.
- For the iid Normal(μ, σ^2) model, code for plotting this surface is available with course source-code.

Calculating the likelihood in R

- For standard distributions, the likelihood is easy to calculate in R. Recall the likelihood is:

$$L(\theta) = f(x_i^*; \theta).$$

- For $X_1, \ldots, X_n$ iid normal, the likelihood can be calculated using `dnorm`:

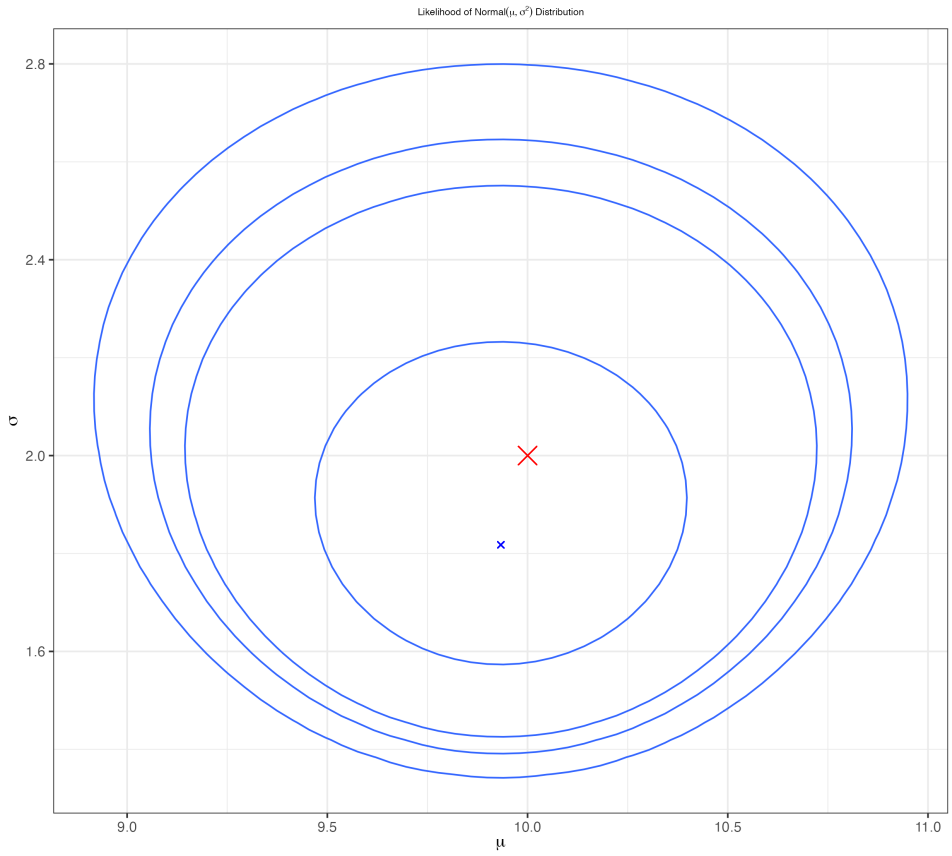


Figure 1: Likelihood surface of data generated from normal distribution.

```r
# Synthetic data
x_star <- rnorm(n = 100, mean = 4, sd = 3)
theta <- c(0, 1)  # value of theta
Likelihood <- prod(dnorm(x, mean = theta[1], sd = theta[2]))
loglik <- sum(
  dnorm(x, mean = theta[1], sd = theta[2], log = TRUE)
)
```

5 Numeric Optimization

Numeric Optimization

- In the previous examples, the MLE was available *analytically*.
- In many cases, however, there is no closed-form solution for the MLE, and it must be computed numerically.
- The next example demonstrates this, and then we will discuss optimization strategies.

Example: Gamma likelihood

Suppose we want to model data $X_1, X_2, \ldots, X_n$ as iid Gamma(α, λ), which has the density function:

$$f(x;\, \alpha, \lambda) = \frac{1}{\Gamma(\alpha)} \lambda^\alpha x^{\alpha-1} e^{-\lambda x}, \quad 0 \le x < \infty.$$

Find the MLE of $\theta = (\alpha, \lambda)$.

- The joint likelihood under the IID assumption is:

$$L(\theta) = \prod_{i=1}^{n} \frac{1}{\Gamma(\alpha)} \lambda^\alpha (x_i^*)^{\alpha-1} e^{-\lambda x_i^*}, \quad 0 \le x_i^* < \infty.$$

- We'll drop the indicator function, since the observations are all bigger than zero (otherwise a Gamma model is a bad choice), so the value is one. Taking logarithms, we get:

$$\begin{aligned} \ell(\alpha, \lambda) &= \sum_{i=1}^{n} \big(\alpha \log \lambda + (\alpha - 1) \log x_i^* - \lambda x_i^* - \log \Gamma(\alpha) \big) \\ &= n\alpha \log \lambda + (\alpha - 1) \sum_{i=1}^{n} \log x_i^* - \lambda \sum_{i=1}^{n} x_i^* - n \log \Gamma(\alpha). \end{aligned}$$

- The partial derivatives are:

$$\begin{aligned} \frac{\partial \ell}{\partial \alpha} &= n \log \lambda + \sum_{i=1}^{n} \log x_i^* - n \frac{\Gamma'(\alpha)}{\Gamma(\alpha)}, \\ \frac{\partial \ell}{\partial \lambda} &= \frac{n\alpha}{\lambda} - \sum_{i=1}^{n} x_i^*. \end{aligned}$$

- The second equation is the easiest to solve. Setting equal to zero, we have:

$$\frac{n\alpha}{\lambda} = \sum_{i=1}^{n} x_i^* = n \bar{x}_n^*.$$

  Thus, solving for λ, we get:

$$\lambda = \frac{\alpha}{\bar{x}_n^*}.$$

- The estimate of λ depends on the value of α. Since we are looking for a maximum of both, it needs to be a critical point, so we can write the estimate of λ as:

$$\hat{\lambda} = \frac{\hat{\alpha}}{\bar{x}_n^*},$$

  implying that we need to first maximize the likelihood for α.

- Plugging in $\hat{\lambda}$ into the second partial derivative, setting equal to zero, gives:

$$n \log \alpha - n \log \bar{x}_n^* + \sum_{i=1}^{n} \log x_i^* - n \frac{\Gamma'(\alpha)}{\Gamma(\alpha)} = 0.$$

- This equation *cannot be solved* in closed-form for α.

- The previous example leads us to consider numeric techniques for optimization and root finding.
- Note that this class is *not* an optimization course, so we'll only cover some of the most basic ideas.
- More modern and efficient numeric optimization techniques are readily available in R (or any other statistical software).
- For this class, we'll introduce some basic ideas like the Newton-Raphsom approach for root finding and optimization, as well as some other basic methods.

Newton-Raphsom root-finding algorithm

- Idea: start at a point θ_0, and approximate find the tangent line of $f(\theta)$ at the point θ_0:

$$y - f(\theta_0) = f'(\theta_0)\big(\theta - \theta_0\big)$$

- Then, find the root of the tangent line by setting $y = 0$, and solving for θ:

$$\theta = \theta_0 - \frac{f(\theta_0)}{f'(\theta_0)}.$$

- This root of the tangent line will be closer than our original guess θ_0, so we set:

$$\theta_1 = \theta_0 - \frac{f(\theta_0)}{f'(\theta_0)},$$

  and repeat:

$$\theta_{n+1} = \theta_n - \frac{f(\theta_n)}{f'(\theta_n)}.$$

- We stop based on some convergence criteria, often something like $|\theta_{n+1} - \theta_n| < \epsilon$, for a small choice of ϵ.
- (In class, check out wikipedia or draw a picture).
- We need now a starting point θ_0.
- Really we can pick anything, but it's best if we are close to the MLE.
- For our current problem (Gamma distribution), we could use the MoM estimator:

$$\hat{\alpha}_{\text{MoM}} = \theta_0 = \frac{\big(\bar{x}_n^*\big)^2}{\frac{1}{n}\sum_{i=1}^n \big(x_i^* - \bar{x}_n^*\big)^2}.$$

```
NR_root <- function(theta0, fn, deriv, tol = 1e-8, maxiter = 1000) {
  iter <- 0
  theta_old <- theta0
  theta_new <- theta_old + 10 * tol

  while(abs(theta_old - theta_new) > tol && iter < maxiter) {
    iter <- iter + 1
    theta_old <- theta_new
    theta_new <- theta_old - fn(theta_old) / deriv(theta_new)
  }
  cat("iters: ", iter, "\n")
  theta_new
}
```

```
alpha_fn <- function(alpha, data) {
  n <- length(data)
  n * log(alpha) - n * log(mean(data)) + sum(log(data)) - n * digamma(alpha)
}

alpha_deriv <- function(alpha, data) {
```

```
  n <- length(data)
  (n/alpha) - n * psigamma(alpha, 1)
}

set.seed(123)
data <- rgamma(n = 23, shape = 1, rate = 2)
```

```
# Not the exact MoM estimate, but close enough:
alpha_mom <- (mean(data)^2) / sd(data)

NR_root(
  theta0 = alpha_mom,
  fn = function(x) alpha_fn(x, data = data),
  deriv = function(x) alpha_deriv(x, data = data),
  tol = 1e-10
)

iters:  7
[1] 0.9728019
```

- Once the estimate of α is found, we can then plug it in to get the estimate of λ:

$$\hat{\lambda} = \frac{\hat{\alpha}}{\bar{x}_n^*} = 1.981.$$

Root-find considerations

- The function that we built for solving $f(\theta) = 0$ requires the derivative $f'(\theta)$.
- In some cases, the derivative is not readily available. Instead, we can approximate using the definition:

$$f'(\theta) = \lim_{h \to 0} \frac{f(\theta + h) - f(\theta)}{h} \approx \frac{f(\theta + \Delta) - f(\theta)}{\Delta}.$$

- We just pick Δ to be small, and we get a very good approximation of the derivative.
- Thus, we don't really need to derivative for uni-variate root-finding.
- The same mathematical approach can readily be extended into higher dimensional θ, replacing the derivative with a *gradient*.
- An important consideration, however, is that the results may depend on your starting parameter θ_0. In practice, you may want to try multiple values of θ_0.
- In most programming languages, there will be pre-built methods for root-fining and optimization.
- Our function we built works, but it doesn't do careful error checking, and it isn't optimized for speed.
- For univariate-root finding $f(\theta) = 0$ in R, we can use the `uniroot` function, which doesn't require a derivative.
- This function is very efficient for solving roots if $\theta \in \mathbb{R}$. For higher dimensions, we need to import a different package.

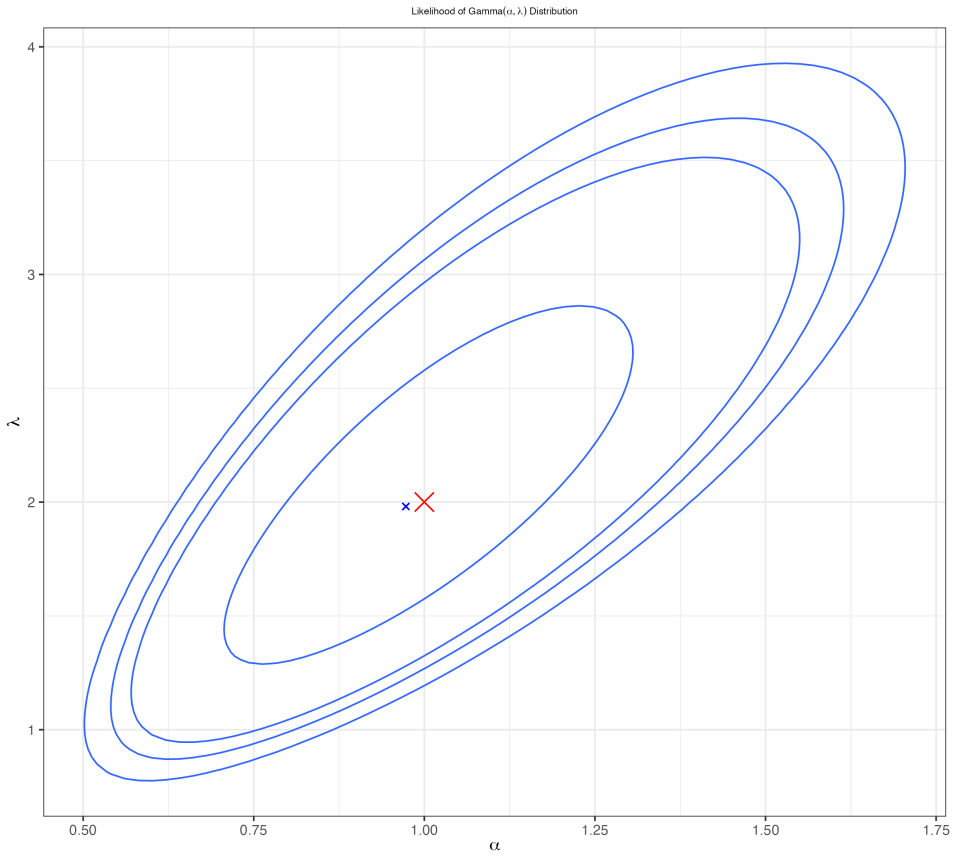


Figure 2: Likelihood surface of data generated from Gamma distribution.

```
uniroot(
  function(x) alpha_fn(x, data = data),
  interval = c(0.5, 2)
)

$root
[1] 0.9728068

$f.root
[1] -7.754612e-05

$iter
[1] 6

$init.it
[1] NA

$estim.prec
[1] 6.103516e-05
```

Direct Numeric Optimization

- Based on our last example, we solved for one parameter λ, and numerically found the other, α, via root-finding.
- In many cases, this is not possible. Instead, we might want to directly optimize both parameters.
- We will first introduce this by extending the Newton-Raphson to perform optimization rather than root-finding.
- The idea is simple: local maximum are just the zeros (roots) of the derivative function.
- Thus, we still perform Newton-Raphson, but on the derivative rather than the original function.
- Starting with initial estimate θ_0, we update following the equations until convergence:

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

- If θ is multivariate, then again we use the *gradient*: $\nabla f(\theta)$, which is a vector, and Hessian: $\nabla^2 f(\theta)$, which is a matrix:

$$\theta_{n+1} = \theta_n - \big(\nabla^2 f(\theta_n)\big)^{-1} \nabla\big(f(\theta_n)\big).$$

- This is the basic approach of many traditional machine learning algorithms:
  - Pick a model that depends on θ, and a loss-function $f(\theta)$ that depends on the data and model (here, our loss is the log-likelihood function).
  - If possible, calculate the derivative of the loss function with respect to θ. If not, approximate numerically.
  - If possible, calculate the Hessian of the loss function with respect to θ. If not, approximate numerically.
  - Start with initial guess for θ, take steps the size of the (approximate) hessian of $f(\theta)$, in the direction of the gradient $f(\theta)$.
- Many optimization strategies are variants of this basic approach: In practice, Hessians / Gradients may be expensive to compute.
- We will primarily focus our attention on pre-existing solutions.
- In R, the function we will use is called `optim`.
- There are several optimization methods available within this function
  - Nelder-Mead: a heuristic, direct search method. Does not require differentiability. Slow, but robust.
  - BFGS / L-BFGS-B: A Quasi-Newton approach, approximates either (or both) the Hessian and Gradient numerically. Extremely effective for (twice) differentiable functions, but slow as θ grows in dimension.
  - SANN: A stochastic approach. Hard to tune, but effective on "rough" surfaces
  - CG: Conjugate Gradients. More "fragile" than BFGS, but require less memory storage so can be useful for large θ.
  - Brent: Only univariate θ. In these cases, approximates the gradient and hessian, similar to what we proposed. Very effective, but requires univariate θ.
- How it works: Get function f, method you want to use, and starting point θ_0. Example:

```r
gamma_negloglik <- function(theta) {
  -dgamma(
    x = data,
    shape = theta[1], rate = theta[2],
    log = TRUE
  ) |> sum()
}
```

```r
optim(
  c(2, 5),
  fn = gamma_negloglik,
  method = 'L-BFGS-B',  # Constrained theta>0
  lower = c(1e-8, 1e-8)  # Constrained theta>0
)

$par
[1] 0.9728026 1.9806150

$value
[1] 6.641716

$counts
function gradient
      16       16

$convergence
[1] 0

$message
[1] "CONVERGENCE: REL_REDUCTION_OF_F <= FACTR*EPSMCH"
```

Gradient Descent and Variations

- In Newton-Raphson, the Hessian just tells us the *size* of the step.
- Often the gradient is available, but not the Hessian (or it is expensive to compute the inverse, since it is a matrix).
- In these cases, we often use a different approach called *gradient descent*.
- Idea: still step in the direction of the (negative) gradient, but the step size will just be small δ_n, and let δ_n shrink over-time:
$$\theta_{n+1} = \theta_n - \delta_n \nabla f(\theta_n).$$
- The idea is simple, but a lot of modern optimization techniques and theory are based on this idea, finding various strategies for specifying δ_n.
- Ex: BFGS approximates the Hessian, but scales poorly with dimension. In deep learning, θ has dimension in millions / billions, so BFGS won't work (nor does `optim`).
- The success of deep-learning and AI is largely due to the advent of *automatic differentiation*: software that calculates the exact gradient of $f(\theta_n)$, after simple calculations.
- Auto-diff libraries: PyTorch, TensorFlow, JAX, etc. Mostly available in Python, though some are available in R.

- Final approach we will discuss is *stochastic* gradient descent, which is effectively randomly sampling the data to compute an approximate gradient.
- The idea is that, even with auto-diff, gradients (with entire data) are expensive to compute.
- Thus, randomly sample data to get a stochastic approximation of the gradient; this is faster to compute, so we get more update-steps.
- This is the primary technique used in most deep-learning frameworks.

Maximum Likelihood: continued examples

Muon Decay
Returning to the Muon-Decay example we used MoM to solve. The density of iid observations $X_1, \ldots, X_n$ is:

$$f(x;\alpha) = \frac{1+\alpha x}{2}, \quad -1 \le x \le 1 \quad -1 \le \alpha \le 1.$$

Find the MLE of α.

- The joint-likelihood function, under IID assumptions and ignoring indicators, is:

$$L(\alpha) = \prod_{i=1}^{n} \frac{1+\alpha x_i^*}{2}.$$

- The log-likelihood is given by:

$$\ell(\alpha) = \sum_{i=1}^{n} \log(1+\alpha x_i^*) - n\log 2.$$

- Taking the derivative, we have:

$$\begin{aligned}\ell'(\alpha) &= \frac{\partial}{\partial\alpha}\sum_{i=1}^{n}\log(1+\alpha x_i^*) - \frac{\partial}{\partial\alpha} n\log 2 \\ &= \sum_{i=1}^{n}\frac{\partial}{\partial\alpha}\log(1+\alpha x_i^*) \\ &= \sum_{i=1}^{n}\frac{1}{1+\alpha x_i^*}.\end{aligned}$$

- Setting this equal to zero gives the following equation:

$$\sum_{i=1}^{n}\frac{x_i^*}{1-\alpha x_i^*} = 0.$$

- As before, there is no closed-form solution to solve this equation for α, and we need to take a computational approach. There are a couple of choices:
  - Solve for roots using a root finding algorithm, or directly optimize $\ell(\alpha)$?
  - What optimization procedure to use? Autodiff? Calculate second derivative and use Newton-Raphson? Quasi-newton, like BFGS?
- In this case, since a second derivative can readily be calculated, it's probably best to do Newton-Raphson, or a root-finding algorithm. We can use our own approach for root-finding of $\ell(\alpha)$, apply `uniroot` to $\ell'(\alpha)$, or `optim` to $\ell(\alpha)$.

- In class, we will practice at least of of these approaches, but it is recommended you try some of these approaches on your own.
- Regardless, the MoM might be a good choice for initializing the value α_0.

6 Constraint Optimization

Constrained optimization

- In all of our previous examples, we want to maximize the log-likelihood function $f(x_i^*; \theta)$, and there is some natural constraints of θ.
- Examples: In Gamma(α, λ), $\theta = (\alpha, \lambda)$, and we are constrained by $\alpha, \lambda > 0$.
- In the Gaussian model, $\theta = (\mu, \sigma)$. μ is unconstrained, but $\sigma \geq 0$.
- Similarly, in the Muon-decay example: $\theta = \alpha \in (0, 1)$.
- This leads to the issue of *constrained* optimization, and is typically a requirement for maximum likelihood estimation.
- There are a few common strategies:
- Limit search region to be within constraints: This is the basic approach used by the `uniroot` function. It's also used sometimes in `optim`, e.g., the L-BFGS-B method.
- Variable transformations. Modify the function so that the search algorithm can try all possible values, but apply a transformation that keeps it in the desired range.
  - Example: if $\theta > 0$ is a constraint, optimize over α with $\theta = e^\alpha$. Thus, $\alpha < 0$ is no problem, since $\theta > 0$. (*Example on next slide*)
  - Another common example is a logit transformation, which ensures $0 \leq \theta \leq 1$, via $\log(\theta/(1-\theta)) = \alpha$.
- In some cases, the constraint leads to an equation $g(\theta) = 0$, in which case we can apply something like Lagrange-multipliers. We will look at one of these cases as well.

Example: Gamma likelihood

MLE of Gamma likelihood using transformations
Revisit the numeric optimization of the Gamma likelihood, using variable transformations to deal with constraints.

```
gamma_negloglik2 <- function(alpha) {
  theta <- exp(alpha)
  -dgamma(
    x = data,
    shape = theta[1], rate = theta[2],
    log = TRUE
  ) |> sum()
}
```

```r
results <- optim(
  c(log(2), log(5)),
  fn = gamma_negloglik2,
  method = 'BFGS' # unconstrained
)
exp(results$par)  # Theta

[1] 0.9728029 1.9806131
```

Multinomial Cell probabilities

- In this example, we are bound by a curve, not a region. Lagrange Multipliers are a good analytic approach.
- Consider fitting a multinomial distribution to a frequency table.
- We let $X_1, \ldots, X_m$ be the counts in cells $1, \ldots, X_m$, and we assume that $(X_1, \ldots, X_m)$ follows a multinomial distribution.
- The distribution has has parameters n: total counts, and p_i probability of cell i.
- The total count $n = \sum_{i=1}^{b} X_i$, and p_i being the
- In this case, the X_i are *not* independent, so we don't just take product of IID observations.
- However, the likelihood can be calculated via the pmf of the multinomial distribution:

$$L(\theta) = f(x_i^*; \theta) = \frac{n!}{\prod_{i=1}^{m} x_i!} \prod_{i=1}^{m} p_i^{x_i^*}.$$

- Given $X_1, \ldots, X_m$, the number of trials n is known. The parameter vector of interest is then:

$$\theta = (p_1, \ldots, p_m).$$

- The log-likelihood is given by:

$$\ell(\theta) = \log n! - \sum_{i=1}^{m} \log x_i! + \sum_{i=1}^{m} x_i \log p_i.$$

- Note that we have the constraint $\sum_i p_i = 1$. Thus, we will solve this using the Lagrange-multiplier technique.
- Reminder: Suppose we are trying to maximize (or minimize) a function $f(\theta)$, with the constraint $g(\theta) = 0$. That is,

$$\max_{\theta} f(\theta)$$
$$\text{subject to } g(\theta) = 0$$

- This optimization problem can be solved by introducing the Lagrange function:

$$\mathcal{L}(\theta, \lambda) = f(\theta) + \lambda g(\theta),$$

- And solving the system of equations that arises from:

$$\nabla_{\theta,\lambda} \mathcal{L}(\theta, \lambda) = 0.$$

- Recall the partial derivative $\frac{\partial}{\partial\lambda}\mathcal{L}(\theta,\lambda) = g(\theta)$, which must be zero, giving the necessary constraint.
- Then, the condition $\nabla_\theta\mathcal{L}(\theta,\lambda) = 0$ ensures that the gradients of f and g are parallel. That is, geometrically, the constraint $g(\theta) = 0$ defines a curve or surface that we are constrained to; on this surface, the max/min value of f occurs when a level-set of f is tangent to the constraint curve $g(\theta) = 0$. This occurs only where $\nabla f(\theta) = \lambda\nabla g(\theta)$, which gives rise to the set of equations defined by $\nabla_\theta\mathcal{L}(\theta,\lambda) = 0$.

MLE of multinomial cell probabilities
Use Lagrange multipliers to find the MLE of a multinomial distribution, with $X_1, \ldots, X_m$.

- The log-likelihood, which we want to maximize, is the function:

$$\ell(\theta) = \log n! - \sum_{i=1}^{m} \log x_i^*! + \sum_{i=1}^{m} x_i^* \log p_i.$$

- However, we have the constraint:

$$g(\theta) = \sum_{i=1}^{m} p_i - 1 = 0.$$

- We can find the solution using Lagrange-multipliers. The Lagrangian is:

$$\mathcal{L}(\theta,\lambda) = \log n! - \sum_{i=1}^{m} \log x_i^*! + \sum_{i=1}^{m} x_i^* \log p_i + \lambda(\sum_{i=1}^{m} p_i - 1).$$

- Taking partial derivatives with respect to p_i gives:

$$\nabla_{\theta_i}\mathcal{L}(\theta,\lambda) = \frac{x_i^*}{p_i} + \lambda,$$

  Setting equal to zero and solving for p_i gives:

$$p_i = -\frac{x_i^*}{\lambda}.$$

- Now we still have the constraint function, given by:

$$\frac{\partial}{\partial\lambda}\mathcal{L}(\theta,\lambda) = 0,$$

  which gives:

$$\sum_{i=1}^{m} p_i = 1.$$

  Now we can solve for all of θ and λ, by combining these equations:

$$\sum_{i=1}^{m} p_i = \sum_{i=1}^{m} -\frac{x_i^*}{\lambda} = -\frac{n}{\lambda} = 1,$$

  which implies:

$$\lambda = -n,$$

  and therefore:

$$p_i = \frac{x_i^*}{n}.$$

- Often, we can describe parameters of a model as functions of other parameters.
- Example: in the multinomial model, we have parameters p_i representing probabilities for each cell. We might want to make these a function of θ, such that $p_i(\theta)$; then, we will still want to estimate θ using the data.
- In the case above the log-likelihood is then:

$$\ell(\theta) = \log n! - \sum_{i=1}^{m} \log x_i^*! + \sum i = 1^m x_i \log p_i(\theta).$$

- TODO: Finished 8.5.1 Example A, RICE.

Acknowledgments

- Compiled on January 25, 2026 using R version 4.5.1.

- We acknowledge students and instructors for previous versions of this course / slides.

References

Pawitan Y (2001). *In all likelihood: statistical modelling and inference using likelihood*. Oxford University Press. 1, 2

Rice JA (2007). *Mathematical statistics and data analysis*, volume 371. 3 edition. Thomson/Brooks/Cole Belmont, CA. 1