Maximum Likelihood Estimation

Prepared by: Harsh Patel (hkp49), Josh Levine (jl2108), Jaini Patel (jp1891), Yifan Liao (yl1463) and Aayush Shah (avs93).

Suppose we have a random sample $X_1, X_2, \cdots, X_n$ whose assumed probability distribution depends on some unknown parameter θ. Our primary goal here will be to find a point estimator $u(X_1, X_2, \cdots, X_n)$, such that $u(x_1, x_2, \cdots, x_n)$ is a "good" point estimate of θ, where $x_1, x_2, \cdots, x_n$ are the observed values of the random sample. For example, if we plan to take a random sample $X_1, X_2, \cdots, X_n$ for which the X_i are assumed to be normally distributed with mean μ and variance σ^2, then our goal will be to find a good estimate of μ, say, using the data $x_1, x_2, \cdots, x_n$ that we obtained from our specific random sample.

It seems reasonable that a good estimate of the unknown parameter θ would be the value θ of that maximizes the probability, that is, the likelihood of getting the data we observed. So, that is the idea behind the method of maximum likelihood estimation.

Formal Definitions of some important terms:

Let $X_1, X_2, \cdots, X_n$ be a random sample from a distribution that depends on one or more unknown parameters $\theta_1, \theta_2, \cdots, \theta_m$ with probability density (or mass) function $f(x_i; \theta_1, \theta_2, \cdots, \theta_m)$. Suppose that $(\theta_1, \theta_2, \cdots, \theta_m)$ is restricted to a given parameter space Ω. Then:

1) When regarded as a function of $\theta_1, \theta_2, \cdots, \theta_m$, the joint probability density (or mass) function of $X_1, X_2, \cdots, X_n$:

$$L(\theta_1, \theta_2, \ldots, \theta_m) = \prod_{i=1}^{n} f(x_i; \theta_1, \theta_2, \ldots, \theta_m)$$

   $((\theta_1, \theta_2, \cdots, \theta_m)$ in $\Omega)$ is called the **likelihood function**.

2) If:

$$[u_1(x_1, x_2, \ldots, x_n), u_2(x_1, x_2, \ldots, x_n), \ldots, u_m(x_1, x_2, \ldots, x_n)]$$

   is the m-tuple that maximizes the likelihood function, then: $\hat{\theta}_i = u_i(X_1, X_2, \ldots, X_n)$ is the **maximum likelihood estimator** of θ_i, for $i = 1, 2, \cdots, m$.

3) The corresponding observed values of the statistics in (2), namely:

$$[u_1(x_1, x_2, \ldots, x_n), u_2(x_1, x_2, \ldots, x_n), \ldots, u_m(x_1, x_2, \ldots, x_n)]$$

   are called the **maximum likelihood estimates** of θ_i, for $i = 1, 2, \cdots, m$.

Implementation of Maximum Likelihood Estimation Method:

Suppose we have a random sample $X_1, X_2, \cdots, X_n$ for which the probability density (or mass) function of each X_i is $f(x_i; \theta)$. Then, the joint probability mass (or density) function of $X_1, X_2, \cdots, X_n$, which we'll (not so arbitrarily) call $L(\theta)$ is:

$$L(\theta) = P(X_1 = x_1, X_2 = x_2, \ldots, X_n = x_n) = f(x_1; \theta) \cdot f(x_2; \theta) \cdots f(x_n; \theta) = \prod_{i=1}^{n} f(x_i; \theta)$$

The first equality is of course just the definition of the joint probability mass function. The second equality comes from that fact that we have a random sample, which implies by definition that the X_i are independent. And, the last equality just uses the shorthand mathematical notation of a product of indexed terms. Now, in light of the basic idea of maximum likelihood estimation, one reasonable way to proceed is to treat the **likelihood function** $L(\theta)$ as a function of , and find the value of θ that maximizes it.

The simplest way to solve the likelihood estimation is by using the log likelihood function, in which we take logarithm of the likelihood function,

$$l(\theta) = \ln L(\theta)$$

Maximum Likelihood estimation for different distributions:

Below are codes for the maximum likelihood estimation functions for different distributions with the parameters to be estimated.

1) Bernoulli distribution: estimate (probability p)

```
bernoulli_mle <- function(input_data)
{
  bernoulli_mle <- mean(input_data)
  return(bernoulli_mle)
}
```

2) Binomial distribution: estimate(probability p) for a given n (assume that n is known to us)

```
binomial_mle <-function(input_data,n)
{
  m <- length(input_data)
  binomial_mle <- mean(input_data)/n
  return(binomial_mle)
}
```

3) Poisson distribution: estimate(lambda)

```
poisson_mle <- function(input_data)
{
  lambda_value_mle <- mean(input_data)
  return(lambda_value_mle)
}
```

4) Geometric distribution: estimate(probaility p)

```
geometric_mle <- function(input_data)
{
  geometric_mle <- (1.0/(mean(input_data)+1))
  return(geometric_mle)
}
```

5) Exponential distribution: estimate(beta)

```
exponential_mle <- function(input_data)
{
  beta_mle <- 1/mean(input_data)
  return(beta_mle)
}
```

6) Normal distribution: estimate(mean,variance)

```r
normal_mle <-function(input_data)
{
  n <- length(input_data)
  mean_mle <- mean(input_data)
  variance_mle <- sum((input_data-mean_mle)^2)/n
  return(c(mean_mle,variance_mle))
}
```

7) Uniform distribution: estimate(a,b)

```r
uniform_mle <-function(input_data)
{
  a_mle <- min(input_data)
  b_mle <- max(input_data)
  return(c(a_mle,b_mle))
}
```

8) Gamma distribution: estimate(alpha,beta)

```r
gamma_mle <-function(input_data)
{
  s <- log(mean(input_data)) - mean(log(input_data))
  alpha_mle <- (3-s)/(12*s) + sqrt((s-3)^2 + 24*s)/(12*s)
  beta_mle <- mean(input_data)/alpha_mle
  return(c(alpha_mle,beta_mle))
}
```

9) Beta distribution: estimate(a,b)

Since the MLE derivatives for a and b are not closed form equations, we use the Newton-Raphsen method to iteratively find the MLE estimations for a and b.

```r
beta_mle <- function(input_data)
{
  # Initializing alpha and beta parameters using Method of moment estimates
  input_data_mean <- mean(input_data)
  input_data_variance <- (sum(input_data * input_data))/length(input_data)
  a_mle <- ((input_data_mean ^ 2) - (input_data_mean * input_data_variance))/(input_data_variance - (inp
  b_mle <- (a_mle * (1 - input_data_mean))/(input_data_mean)
  final_val <- c(a_mle, b_mle)

   for(index in 1:100){

    g1 <- digamma(a_mle) - digamma(a_mle + b_mle) - (sum(log(input_data)))/length(input_data)

    g2 <- digamma(b_mle) - digamma(a_mle + b_mle) - (sum(log(1 - input_data))/length(input_data))

    g <- c(g1, g2)

    # Calculating G matrix of second derivatives:
    G1_val <- trigamma(a_mle) - trigamma(a_mle + b_mle)
```

```
    G2_val <- -trigamma(a_mle + b_mle)
    G3_val <- trigamma(b_mle) - trigamma(a_mle + b_mle)
    G <- matrix(c(G1_val, G2_val, G2_val, G3_val), nrow = 2, ncol = 2, byrow = TRUE)
    G_inverse <- solve(G)

    # Final values for the iteration: Theta_mle(i+1) = Theta_mle(i) - G_inverse*g

    final_val <- final_val - t(G_inverse %*% g)
    a_mle <- final_val[1]
    b_mle <- final_val[2]
  }

  return(c(a_mle,b_mle))

}
```

10) Chi-squared distribution: estimate(v)

```
chi_sqaure_mle <- function(input_data)
{
  ## Here we have used the property of chi-square being a special case of Gamma distribution.
  ## A gamma distribution (alpha,beta) with alpha = v/2 and beta = 1/2, is a chi-squared RV with v degr
  result_mle <- gamma_mle(input_data)
  v_mle <- 2* result_mle[1]
  return(v_mle)
}
```

11) Multinomial distribution: estimate(n)

```
# multinomial distribution: estimate(n)
library(MASS)
multinomial_mle <- function(input_data){
  a = nrow(input_data)
  p = c(0,0,0,0,0)
  for(i in 1:a)
    p[i]<-1-((var(input_data[i,]))/mean(input_data[i,]))
  n = sum(rowMeans(input_data))/sum(p[1:a])
  return (n)
}
```

12) Multivariate distribution: estimate(vari)

```
# multivariate distribution: estimate(vari)

multivariate_normal_mle <- function(input_data){
  mu_hat = colMeans(input_data)
  summation = var(input_data)
}
```

Below is the code to find Maximum Likelihood Estimators for different distributions on sample size of 10000.

```
mle_test_function <-function(distribution, num_samples = 10000)
{
  if(distribution=="bernoulli")
  {
    p = 0.7
    print("-------Bernoulli-------")
    print(paste("Population parameter: ", p))
    input_data <- rbinom(num_samples,1,p)
    p_mle <- bernoulli_mle(input_data)
    print(paste("Estimated parameter: ", p_mle))
    return(p_mle)
  }

  else if(distribution=="binomial")
  {
    n = 1000
    p = 0.6
    print("-------Binomial-------")
    print(paste("Population parameter: ", p))
    input_data <- rbinom(num_samples,n,p)
    p_mle <- binomial_mle(input_data,n)
    print(paste("Estimated parameter: ", p_mle))
    return(p_mle)
  }

  else if(distribution=="geometric")
  {
    p = 0.6
    print("-------Geometric-------")
    print(paste("Population parameter: ", p))
    input_data <- rgeom(num_samples,p)
    p_mle <- geometric_mle(input_data)
    print(paste("Estimated parameter: ", p_mle))
    return(p_mle)
  }

  else if(distribution=="poisson")
  {
    lambda = 0.03
    print("-------Poisson-------")
    print(paste("Population parameter: ", lambda))
    input_data <- rpois(num_samples,lambda)
    lambda_mle <- poisson_mle(input_data)
    print(paste("Estimated parameter: ", lambda_mle))
    return(lambda_mle)
  }

  else if(distribution=="uniform")
  {
    a = 0
    b = 5
    print("-------Uniform-------")
    print(paste("Population parameter: ", a, b))
```

```
  input_data <- runif(num_samples,a,b)
  theta_mle <- uniform_mle(input_data)
  print(paste("Estimated parameter: ", theta_mle))
  return(theta_mle)
}

else if(distribution=="normal")
{
  mean = 0
  variance = 1
  print("-------Normal-------")
  print(paste("Population parameter: ", "mean = ", mean,"variance = ",variance))
  input_data <- rnorm(num_samples,mean,variance) # mean = 0, variance = 1
  theta_mle <- normal_mle(input_data)
  print(paste("Estimated parameter: ", theta_mle))
  return(theta_mle)
}
else if(distribution=="exponential")
{
  beta = 0.2
  print("-------Exponential-------")
  print(paste("Population parameter: ", beta))
  input_data <- rexp(num_samples,beta)
  beta_mle <- exponential_mle(input_data)
  print(paste("Estimated parameter: ", beta_mle))
  return(beta_mle)
}

else if(distribution=="gamma")
{
  alpha = 10
  beta = 0.4
  print("-------Gamma-------")
  print(paste("Population parameter: ", alpha,beta))
  input_data <- rgamma(num_samples,alpha,scale = beta) # alpha = 2, beta = 0.4
  theta_mle <- gamma_mle(input_data)
  print(paste("Estimated parameter: ", theta_mle))
  return(theta_mle)
}

else if(distribution=="beta")
{
  alpha = 10
  beta = 5
  print("-------Beta-------")
  print(paste("Population parameter: ", alpha,beta))
  input_data <- rbeta(num_samples, alpha, beta) # alpha = 3, beta = 5
  theta_mle <- beta_mle(input_data)
  print(paste("Estimated parameter: ", theta_mle))
  return(theta_mle)
}

else if(distribution=="Chi-square")
```

```
  {
    v = 4
    print("-------Chi-square-------")
    print(paste("Population parameter: ", v))
    input_data <- rchisq(num_samples,v) # degree of freedom (v = 4)
    v_mle <- chi_sqaure_mle(input_data)
    print(paste("Estimated parameter: ", v_mle))
    return(v_mle)
  }

  else if(distribution == "multinomial")
  {
    p = c(0.15,0.05,0.4,0.1,0.3)
    print("-------Multinomial-------")
    input_data = rmultinom(num_samples,size=5,p)
    print(paste("Population parameter: ",length(p)))
    a = nrow(input_data)
    multinom_mle <- multinomial_mle(input_data)
    print(paste("Estimated parameter: ", multinom_mle))
    return (multinom_mle)
  }

  else if(distribution=="multivariate")
  {
    vari = c(10,3,3,2)
    print("-------Multivariate-------")
    sigma = matrix(vari,2,2)
    input_data = mvrnorm(n = num_samples, rep(0, 2), sigma)
    print(paste("Population parameter: ",vari))
    multivariate_mle <- multivariate_normal_mle(input_data)
    print(paste("Estimated parameter: ", multivariate_mle))
  }

  else
  {
    return("Wrong distribution")
  }
}
```

Below are the outputs of estimated parameters obtained by applying maximum likelihood estimation method on each probability distribution.

```
num_samples = 10000  #Number of samples generated for each distribution

mle_test_function("bernoulli")
```

```
## [1] "-------Bernoulli-------"
## [1] "Population parameter:  0.7"
## [1] "Estimated parameter:  0.6976"

## [1] 0.6976
```

```r
mle_test_function("binomial")
```

```
## [1] "-------Binomial-------"
## [1] "Population parameter:  0.6"
## [1] "Estimated parameter:  0.5999561"

## [1] 0.5999561
```

```r
mle_test_function("geometric")
```

```
## [1] "-------Geometric-------"
## [1] "Population parameter:  0.6"
## [1] "Estimated parameter:  0.595770032767352"

## [1] 0.59577
```

```r
mle_test_function("exponential")
```

```
## [1] "-------Exponential-------"
## [1] "Population parameter:  0.2"
## [1] "Estimated parameter:  0.199066812611437"

## [1] 0.1990668
```

```r
mle_test_function("poisson")
```

```
## [1] "-------Poisson-------"
## [1] "Population parameter:  0.03"
## [1] "Estimated parameter:  0.0293"

## [1] 0.0293
```

```r
mle_test_function("normal")
```

```
## [1] "-------Normal-------"
## [1] "Population parameter:  mean =  0 variance =  1"
## [1] "Estimated parameter:  -0.0134977837924129"
## [2] "Estimated parameter:  0.996505963361796"

## [1] -0.01349778  0.99650596
```

```r
mle_test_function("uniform")
```

```
## [1] "-------Uniform-------"
## [1] "Population parameter:  0 5"
## [1] "Estimated parameter:  0.000323440181091428"
## [2] "Estimated parameter:  4.99758006772026"

## [1] 0.0003234402 4.9975800677
```

```r
mle_test_function("gamma")
```

```
## [1] "-------Gamma-------"
## [1] "Population parameter:  10 0.4"
## [1] "Estimated parameter:  9.97725425242152"
## [2] "Estimated parameter:  0.401307063920786"

## [1] 9.9772543 0.4013071
```

```r
mle_test_function("beta")
```

```
## [1] "-------Beta-------"
## [1] "Population parameter:  10 5"
## [1] "Estimated parameter:  9.96943743951381"
## [2] "Estimated parameter:  4.9958738924558"

## [1] 9.969437 4.995874
```

```r
mle_test_function("Chi-square")
```

```
## [1] "-------Chi-square-------"
## [1] "Population parameter:  4"
## [1] "Estimated parameter:  3.83807594901865"

## [1] 3.838076
```

```r
mle_test_function("multinomial")
```

```
## [1] "-------Multinomial-------"
## [1] "Population parameter:  5"
## [1] "Estimated parameter:  5.03441799054938"

## [1] 5.034418
```

```r
mle_test_function("multivariate")
```

```
## [1] "-------Multivariate-------"
## [1] "Population parameter:  10" "Population parameter:  3"
## [3] "Population parameter:  3"  "Population parameter:  2"
## [1] "Estimated parameter:  10.0242540974728"
## [2] "Estimated parameter:  2.94472657777345"
## [3] "Estimated parameter:  2.94472657777345"
## [4] "Estimated parameter:  1.95921198885998"
```

Results:

Below is the table that shows the given parameters and the obtained estimated parameters.

Distribution	Actual Parameter 1	Actual Parameter 2	Estimated Parameter 1	Estimated Parameter 2
Bernoulli (p)	p = 0.7	N/A	$\hat{p}$ = 0.6976	N/A
Binomial(n, p)	n = 1000	p = 0.6	$\hat{n}$ = 1000	$\hat{p}$ = 0.5999561
Geometric(p)	p = 0.6	N/A	$\hat{p}$ = 0.595770032767352	N/A
Exponential(β)	β = 0.2	N/A	$\hat{\beta}$ = 0.199066812611437	N/A
Poisson(λ)	λ = 0.03	N/A	λ = 0.0293	N/A
Normal(μ, σ^2)	μ = 0	σ = 1	$\hat{\mu}$ = -0.0134977837924129	$\hat{\sigma}$ = 0.996505963361796
Uniform(a, b)	a = 0	b = 5	$\hat{a}$ = 0.000323440181091428	$\hat{b}$ = 4.99758006772026
Gamma(α, β)	α = 10	β = 0.4	$\hat{\alpha}$ = 9.97725425242152	$\hat{\beta}$ = 0.401307063920786
Beta(α, β)	α = 10	β = 5	$\hat{\alpha}$ = 9.96943743951381	$\hat{\beta}$ = 4.9958738924558
Chi-Square(p)	p = 4	N/A	$\hat{p}$ = 3.83807594901865	N/A
Multinomial(n, p)	n = 5	N/A	$\hat{n}$ = 5.03441799054938	N/A
Multivariate Normal($\mu, \sum$)	$vari$ = 10 $vari$ = 3 $vari$ = 3 $vari$ = 2	N/A	$\hat{vari}$ = 10.0242540974728 $\hat{vari}$ = 2.94472657777345 $\hat{vari}$ = 2.94472657777345 $\hat{vari}$ = 1.95921198885998	N/A

Conclusion:

As we can see from our output results, that the estimated values obtained using the method of maximum likelihood estimation is quite consistent and accurate for most of the probability distributions with at most two parameters (when the sample size is 1000). Our estimator accurately estimated the parameter values with an error of ~0.05, which is a small value.So, we can say, that this estimating method will provide accurate estimates, since the empirical distribution converges in some sense to the probability distribution, making the values of parameters almost equal.

Reference:

1) https://online.stat.psu.edu/stat415/lesson/1/1.2