Maximum Likelihood in R

Charles J. Geyer

September 30, 2003

1 Theory of Maximum Likelihood Estimation

1.1 Likelihood

A *likelihood* for a statistical model is defined by the same formula as the density, but the roles of the data x and the parameter θ are interchanged

$$L_x(\theta) = f_\theta(x). \tag{1}$$

Thus the likelihood is considered a function of θ for fixed data x, whereas the density is considered a function of x for fixed θ (but both are the same function of x and θ jointly).

Likelihood is actually a slightly more general concept, we also call

$$L_x(\theta) = h(x) f_\theta(x) \tag{2}$$

a likelihood for the model when $h(x)$ is any strictly positive valued function of x that does not contain the parameter θ. The reason for this extension of the notion is that all of the uses we make of the likelihood function will not be affected in any way by the presence or absence of $h(x)$. The way we make use of the extended definition is to simply drop multiplicative terms in the density that do not contain the parameter.

1.2 Maximum Likelihood Estimation

The so-called method of maximum likelihood uses as an estimator of the unknown true parameter value, the point $\hat{\theta}_x$ that maximizes the likelihood L_x. This estimator is called the maximum likelihood estimator (MLE). We say "so-called method" because it is not really a method, being rather vague in what is considered a maximizer. The likelihood function L_x need not have a maximizer, and even if it does, the maximizer need not be unique. It is often left unclear whether a global or local maximizer is intended. People often talk as if the global maximizer is intended, but theory says that a "good" local maximizer can have better properties than the global maximizer. So not only are global maximizers very hard to find, they aren't even desirable here.

So what is generally done is to start a good local optimization algorithm at a "good" starting point and take the solution produced by the algorithm to be the MLE (if the algorithm converges to a solution). Technically, what is required of the starting point to be "good" is that it obeys the square root law: its estimation error goes to zero like a constant divided by the square root of the sample size. Generally, one just uses the best estimator one can calculate as the starting point.

1.3 Expected Fisher Information

Because the log function is monotone, maximizing the likelihood is the same as maximizing the log likelihood

$$l_x(\theta) = \log L_x(\theta). \tag{3}$$

For many reasons it is more convenient to use log likelihood rather than likelihood.

The derivatives of the log likelihood function (3) are very important in likelihood theory. The moments of log likelihood derivatives satisfy some important identities. Note that if the likelihood is given by (2), then the log likelihood is given by

$$l_x(\theta) = \log h(x) + \log f_\theta(x) \tag{4}$$

and the derivatives (these are derivatives with respect to θ) do not involve the $\log h(x)$ term because it does not contain θ. So these derivatives are well defined and the same regardless of what $h(x)$ we use. Also note that the maximizer of (4) no matter how defined (local or global maximizer) does not depend on $h(x)$. Thus the MLE is the same (if defined) regardless of what $h(x)$ we use.

First, the first derivative has expectation zero

$$E_\theta\big\{\nabla l_x(\theta)\big\} = 0, \tag{5}$$

where ∇f denotes the vector of partial derivatives of a scalar function f of a vector variable, often called the *gradient* of f, and $\nabla f(x)$ denotes the value of the gradient at the point x.

Second, the variance of the first derivative is minus the expectation of the second

$$\operatorname{var}_\theta\big\{\nabla l_x(\theta)\big\} = -E_\theta\big\{\nabla^2 l_x(\theta)\big\}, \tag{6}$$

where $\nabla^2 f$ denotes the matrix of second partial derivatives of a scalar function f of a vector variable, often called the *hessian* of f, and $\nabla^2 f(x)$ denotes the value of the hessian at the point x.

Either side of (6) is called the *expected Fisher information* (or just "Fisher information" with no "expected" when it is clear what is meant) and is denoted $I(\theta)$.

1.4 Asymptotic Distribution of the MLE

The "large sample" or "asymptotic" approximation of the sampling distribution of the MLE $\hat{\theta}_x$ is multivariate normal with mean θ (the unknown true parameter value) and variance $I(\theta)^{-1}$. Note that in the multiparameter case $I(\theta)$ is a matrix so "inverse Fisher information" involves a matrix inverse.

Readers with previous exposure to likelihood theory may have a few questions here, mostly about n. So far we haven't made any assumptions about the data x. Specifically, we haven't assumed that x is a vector $(x_1, \ldots, x_n)$ of IID data. Likelihood theory works even when nothing is independent in the data (time series, spatial statistics, statistical genetics, and so forth). And we don't want to limit ourselves to IID. (Readers who wonder what "large sample" means when there is no n are invited to take my special topics course on advanced asymptotics next semester).

But for readers who have a hard time shaking off previous exposure to bad teaching that insisted on IID, we comment that in the IID case, if we write the Fisher information for sample size n as $I_n(\theta)$, then it satisfies the identity $I_n(\theta) = nI_1(\theta)$. This happens because the variance of a sum is the sum of the variances when the terms are independent. Hence it obviously does *not* hold for dependent data! What we are denoting $I(\theta)$ here is the Fisher information for the actual data being analyzed, that is, $I_n(\theta)$ in the case of an IID sample of size n.

Even when not analyzing dependent data (when you have to look at things our way), our convention that there is only one Fisher information of interest—and that is $I(\theta)$ the actual Fisher information for the actual data—is simpler that the conventional way which invites confusion between $I_n(\theta)$ and $I_1(\theta)$ and actually does confuse a lot of users.

1.5 Plug In and Observed Fisher Information

In practice, it is useless that the MLE has asymptotic variance $I(\theta)^{-1}$ because we don't know θ. If we knew θ, then we wouldn't be estimating it!

Hence we approximate the asymptotic variance by "plugging in" the estimated value of the parameter, that is, we use $I(\hat{\theta}_x)^{-1}$ as the approximate variance of the MLE.

Asymptotic theory guarantees that approximation of $I(\theta)$ by $I(\hat{\theta}_x)$ produces an error that is negligible compared to the main approximation of the actual sampling distribution of the MLE by $\text{Normal}\big(\theta, I(\theta)^{-1}\big)$. This is the *plug-in principle*, that plugging in $\hat{\theta}_x$ for θ in calculating asymptotic variance makes a negligible contribution to the error.

There is a second form of plug-in. The second derivative form of Fisher information $-E_\theta\big\{\nabla^2 l_x(\theta)\big\}$, is by the law of large numbers well approximated by the random variable itself. Thus we call

$$J_x(\theta) = -\nabla^2 l_x(\theta)$$

observed Fisher information and use it as another approximation to Fisher information. Of course, we still don't know θ, so we need a second "plug-in" using $J_x(\hat{\theta}_x)$ instead of $J_x(\theta)$. The plug-in principle applies here too. The error made by approximating $I(\theta)$ by $J_x(\hat{\theta}_x)$ is negligible compared to the error approximating the actual sampling distribution of the MLE by $\text{Normal}\big(\theta, I(\theta)^{-1}\big)$.

1.6 Summary of Theory

The asymptotic approximation to the sampling distribution of the MLE $\hat{\theta}_x$ is multivariate normal with mean θ and variance approximated by either $I(\hat{\theta}_x)^{-1}$ or $J_x(\hat{\theta}_x)^{-1}$.

2 Maximum Likelihood Estimation in R

2.1 The Cauchy Location-Scale Family

The (standard) *Cauchy Distribution* is the continuous univariate distribution having density

$$f(x) = \frac{1}{\pi} \cdot \frac{1}{1 + x^2}, \qquad -\infty < x < \infty. \tag{7}$$

The standard Cauchy distribution has no parameters, but it induces a two-parameter location-scale family having densities

$$f_{\mu,\sigma}(x) = \frac{1}{\sigma} \cdot f\left(\frac{x - \mu}{\sigma}\right) \tag{8}$$

If f is any distribution having mean zero and variance 1, then $f_{\mu,\sigma}$ has mean μ and variance σ^2. But the Cauchy distribution has neither mean nor variance. Thus we call μ the *location parameter* and σ the *scale parameter*.

Since the standard Cauchy distribution is clearly symmetric about zero, the Cauchy(μ, σ) distribution is symmetric about μ. Hence μ is the population median and a "good" estimate is the sample median. A robust scale estimator analogous to the sample median is the interquartile range (IQR). The IQR of the standard Cauchy distribution is

```
> qcauchy(3/4) - qcauchy(1/4)

[1] 2
```

Thus the population IQR of the Cauchy(μ, σ) distribution is 2σ, and hence a "good" estimate of σ is the sample IQR divided by 2.

2.2 Maximum Likelihood

2.2.1 One Parameter

The R function `nlm` minimizes arbitrary functions written in R. So to maximize the likelihood, we hand `nlm` the negative of the log likelihood (for any function f, minimizing $-f$ maximizes f).

For the Cauchy location model (μ is unknown, but $\sigma = 1$ is known) minus the log likelihood can be written either as

```
> mlogl <- function(mu, x) {
+     sum(-dcauchy(x, location = mu, log = TRUE))
+ }
```

using `dcauchy` to avoid having to know the formula (7) and (8) for the densities or as

```
> mlogl2 <- function(mu, x) {
+     sum(log(1 + (x - mu)^2))
+ }
```

using our knowledge of the Cauchy densities.

Either produces the same results on the simulated data

```
> n <- 30
> set.seed(42)
> x <- rcauchy(n)
```

Here the true "unknown" μ is zero, but we pretend we don't know that and see how good the MLE is as an estimator.

The following does the estimation

```
> mu.start <- median(x)
> mu.start
```

```
[1] -0.1955062
```

```
> out <- nlm(mlogl, mu.start, x = x)
> mu.hat <- out$estimate
> mu.hat
```

```
[1] -0.1816501
```

And the following does the estimation using the other "minus log likelihood" function

```
> out2 <- nlm(mlogl2, mu.start, x = x)
> mu.hat <- out$estimate
> mu.hat
```

```
[1] -0.1816501
```

2.2.2 A Simulation Study

We see for these data, the MLE is slightly better than the sample median. But this is just one data set. For random data sometimes the MLE will be better and sometimes the sample median will be better. As statisticians, what we are interested is in the sampling distributions of the two estimators, which we can easily study by simulation

```
> nsim <- 100
> mu <- 0
> mu.hat <- double(nsim)
> mu.twiddle <- double(nsim)
> for (i in 1:nsim) {
+     xsim <- rcauchy(n, location = mu)
+     mu.start <- median(xsim)
+     out <- nlm(mlogl, mu.start, x = xsim)
+     mu.hat[i] <- out$estimate
+     mu.twiddle[i] <- mu.start
+ }
> mean((mu.hat - mu)^2)

[1] 0.06203118

> mean((mu.twiddle - mu)^2)

[1] 0.08242236
```

The two numbers reported from the simulation are the mean square errors (MSE) of the two estimators. Their ratio

```
> mean((mu.hat - mu)^2)/mean((mu.twiddle - mu)^2)

[1] 0.7526013
```

is the *asymptotic relative efficiency* (ARE) of the estimators. Now we see the MLE is more accurate, as theory says it must be.

2.2.3 Two Parameters

Minus the log likelihood for the two-parameter Cauchy can be written

```
> mlogl3 <- function(theta, x) {
+     sum(-dcauchy(x, location = theta[1], scale = theta[2], log = TRUE))
+ }
```

and the MLE calculated by

```
> theta.start <- c(median(x), IQR(x)/2)
> theta.start

[1] -0.1955062  0.7125899

> out <- nlm(mlogl3, theta.start, x = x)
> theta.hat <- out$estimate
> theta.hat

[1] -0.1809299  0.7605561
```

2.3 Fisher Information

2.3.1 Observed Fisher Information

If asked nicely, `nlm` will calculate observed Fisher information evaluated at the MLE.

```
> out <- nlm(mlogl3, theta.start, x = x, hessian = TRUE)
> fish <- out$hessian
> fish

            [,1]        [,2]
[1,] 32.50733727  0.01480913
[2,]  0.01480913 19.34819096

> solve(fish)

              [,1]          [,2]
[1,]  3.076230e-02 -2.354550e-05
[2,] -2.354550e-05  5.168444e-02
```

2.3.2 Confidence Intervals

Inverse Fisher information gives the asymptotic variance matrix of the MLE. From it, we can construct asymptotic confidence intervals.

```
> conf.level <- 0.95
> crit <- qnorm((1 + conf.level)/2)
> inv.fish <- solve(fish)
> theta.hat[1] + c(-1, 1) * crit * sqrt(inv.fish[1, 1])

[1] -0.5246916  0.1628318

> theta.hat[2] + c(-1, 1) * crit * sqrt(inv.fish[2, 2])

[1] 0.3149737 1.2061385
```

These are, of course, *not* simultaneous confidence intervals. To get simultaneous coverage we would have to replace the critical value calculation by a Bonferroni correction

```
> crit

[1] 1.959964

> crit <- qnorm(1 - (1 - conf.level)/2/length(theta.hat))
> crit

[1] 2.241403
```

2.3.3 Expected Fisher Information

R has a function `deriv` that does derivatives of R expressions. But it isn't very sophisticated. It won't calculate likelihood derivatives here. So let's do the derivatives by pencil and paper.

First, the log likelihood itself

$$l_x(\mu, \sigma) = n \log(\sigma) - \sum_{i=1}^{n} \log\bigl(\sigma^2 + (x_i - \mu)^2\bigr)$$

The first derivatives are

$$\frac{\partial l_x(\mu, \sigma)}{\partial \mu} = \sum_{i=1}^{n} \frac{2(x_i - \mu)}{\sigma^2 + (x_i - \mu)^2}$$

$$\frac{\partial l_x(\mu, \sigma)}{\partial \sigma} = \frac{n}{\sigma} - \sum_{i=1}^{n} \frac{2\sigma}{\sigma^2 + (x_i - \mu)^2}$$

R doesn't do analytic integrals at all. But it does do numerical integrals, which is all we need to do Fisher information.

```
> theta.hat

[1] -0.1809299  0.7605561

> mu <- theta.hat[1]
> sigma <- theta.hat[2]
> grad1 <- function(x) 2 * (x - mu)/(sigma^2 + (x - mu)^2)
> grad2 <- function(x) (1/sigma - 2 * sigma/(sigma^2 + (x - mu)^2))
> fish.exact <- matrix(NA, 2, 2)
> fish.exact[1, 1] <- integrate(function(x) grad1(x)^2 * dcauchy(x,
+     mu, sigma), -Inf, Inf)$value
> fish.exact[2, 2] <- integrate(function(x) grad2(x)^2 * dcauchy(x,
+     mu, sigma), -Inf, Inf)$value
> fish.exact[1, 2] <- integrate(function(x) grad1(x) * grad2(x) *
+     dcauchy(x, mu, sigma), -Inf, Inf)$value
> fish.exact[2, 1] <- fish.exact[1, 2]
> round(n * fish.exact, 10)

         [,1]     [,2]
[1,] 25.93157  0.00000
[2,]  0.00000 25.93157

> fish

            [,1]        [,2]
[1,] 32.50733727  0.01480913
[2,]  0.01480913 19.34819096
```

In hindsight these matrices are misnamed. The matrix `fish` is observed Fisher information evaluated at the MLE, that is, $J_x(\hat{\theta}_x)$. The matrix `fish.exact` is expected Fisher information evaluated at the MLE, that is, $I_x(\hat{\theta}_x)$. Neither is "exact" both are approximations to the asymptotic variance of the MLE.

2.4 Using Derivative Information in Optimization

In its default mode of operation `nlm` uses derivatives calculated by finite differences. It will work better and faster if we supply the derivatives. We don't need more speed or accuracy in this small example, but for large numbers of parameters and complicated likelihoods, we do. So let's see how it works.

```
> mlogl4 <- function(theta, x) {
+     if (length(theta) != 2)
+         stop("length(theta) must be 2")
+     mu <- theta[1]
+     sigma <- theta[2]
+     value <- sum(-dcauchy(x, location = mu, scale = sigma, log = TRUE))
+     denom <- sigma^2 + (x - mu)^2
+     grad1 <- sum(-2 * (x - mu)/denom)
+     grad2 <- sum(-(1/sigma - (2 * sigma/denom)))
+     attr(value, "gradient") <- c(grad1, grad2)
+     return(value)
+ }
```

defines minus the log like and adds an attribute `"gradient"`, which is the gradient. So let's see how it works.

```
> theta.start

[1] -0.1955062  0.7125899

> out <- nlm(mlogl4, theta.start, x = x)
> out$estimate

[1] -0.1809294  0.7605566
```

Same answer as before to about six significant figures.

And what does our `mlogl4` do?

```
> mlogl4(out$estimate, x = x)

[1] 77.0496
attr(,"gradient")
[1] 4.367115e-05 5.032129e-05
```