



R news and tutorials contributed by (750) R bloggers

- [Home](#)
- [About](#)
- [RSS](#)
- [add your blog!](#)
- [Learn R](#)
- [R jobs](#) ♦ ♦ ♦
- [Contact us](#)

Welcome!

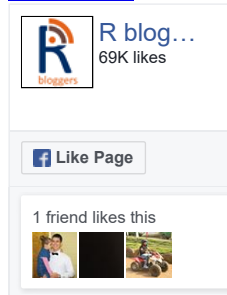
Follow @rbloggers

Here you will find daily **news and tutorials about R**, contributed by over 750 bloggers. There are many ways to **follow us** -

[By e-mail:](#)

44714 readers
BY FEEDBURNER

[On Facebook:](#)



If you are an **R blogger yourself** you are invited to [add your own R content feed to this site](#) (Non-English R bloggers should add themselves- [here](#))

[Jobs for R-users](#)

- [Data Scientist @ Garching bei München, Bayern, Germany](#)
- [Software Developer](#)
- [Senior Quantitative Analyst, Data Scientist](#)
- [R data wrangler](#)
- [Senior Data Scientist](#)

Search & Hit Enter

Popular Searches

- [googlevis](#)
- [heatmap](#)
- [twitter](#)



- [sql](#)
- [web Scraping](#)
- [eof](#)
- [hadoop](#)
- [Jeff Hemsley](#)
- [random forest](#)
- [3 d clusters](#)
- [anova](#)
- [blotter](#)
- [boxplot](#)
- [coplot](#)
- [decision tree](#)
- [discriminant](#)
- [financial](#)
- [ggplot](#)
- [background grid](#)
- [colour](#)
- [how to import image file to r](#)
- [maps](#)
- [purrr](#)
- [rattle](#)
- [Trading](#)
- [bar chart](#)
- [barplot](#)
- [Binary](#)
- [climate](#)
- [contingency table](#)
- [data frame](#)

Recent Posts

- [R Interface to Spark](#)
- [Data Science for Business – Time Series Forecasting Part 2: Forecasting with timekit](#)
- [Run massive parallel R jobs cheaply with updated doAzureParallel package](#)
- [Introduction to Set Theory and Sets with R](#)
- [Campaign Response Testing no longer published on Udemy](#)
- [Neural networks Exercises \(Part-1\)](#)
- [Introducing the MonteCarlo Package](#)
- [Words growing or shrinking in Hacker News titles: a tidy analysis](#)
- [Test-driving Microsoft Cognitive Toolkit in R using reticulate](#)
- [Deep Learning with R](#)
- [Add P-values and Significance Levels to ggplots](#)
- [UK R Courses](#)
- [Unconf projects 4: cityquant, notary, packagemetrics, pegax](#)
- [Tic Tac Toe Part 3: The Minimax Algorithm](#)

- [What is the tidyverse?](#)

Other sites

- [Jobs for R-users](#)
- [SAS blogs](#)

Fitting a Model by Maximum Likelihood

August 18, 2013

By [andrew](#)

Like 6 Share in Share 52

(This article was first published on [Exegetic Analytics » R](#), and kindly contributed to [R-bloggers](#))

220
SHARES

f Share

Tweet

[Maximum-Likelihood Estimation](#) (MLE) is a statistical technique for estimating model parameters. It basically sets out to answer the question: what model parameters are most likely to characterise a given set of data? First you need to select a model for the data. And the model must have one or more (unknown) parameters. As the name implies, MLE proceeds to maximise a likelihood function, which in turn maximises the agreement between the model and the data.

Most illustrative examples of MLE aim to derive the parameters for a [probability density function](#) (PDF) of a particular distribution. In this case the likelihood function is obtained by considering the PDF not as a function of the sample variable, but as a function of distribution's parameters. For each data point one then has a function of the distribution's parameters. The joint likelihood of the full data set is the product of these functions. This product is generally very small indeed, so the likelihood function is normally replaced by a log-likelihood function. Maximising either the likelihood or log-likelihood function yields the same results, but the latter is just a little more tractable!

Fitting a Normal Distribution

Let's illustrate with a simple example: fitting a normal distribution. First we generate some data.

```
> set.seed(1001)
>
> N <- 100
>
> x <- rnorm(N, mean = 3, sd = 2)
>
> mean(x)
[1] 2.998305
> sd(x)
[1] 2.288979
```

Then we formulate the log-likelihood function.

```
> LL <- function(mu, sigma) {
+   R = dnorm(x, mu, sigma)
+   #
+   -sum(log(R))
+ }
```

And apply MLE to estimate the two parameters (mean and standard deviation) for which the normal distribution best describes the data.

```
> library(stats4)
>
> mle(LL, start = list(mu = 1, sigma=1))
```

Call:
`mle(minuslogl = LL, start = list(mu = 1, sigma = 1))`

```
Coefficients:
      mu      sigma
2.998305 2.277506
Warning messages:
1: In dnorm(x, mu, sigma) : NaNs produced
2: In dnorm(x, mu, sigma) : NaNs produced
```

```
3: In dnorm(x, mu, sigma) : NaNs produced
```

Those warnings are a little disconcerting! They are produced when negative values are attempted for the standard deviation.

```
> dnorm(x, 1, -1)
[1] NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN
[30] NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN
[59] NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN
[88] NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN NaN
```

There are two ways to sort this out. The first is to apply constraints on the parameters. The mean does not require a constraint but we insist that the standard deviation is positive.

```
> mle(LL, start = list(mu = 1, sigma=1), method = "L",
      upper = c(Inf, Inf))
```

Call:

```
mle(minuslogl = LL, start = list(mu = 1, sigma = 1),
     lower = c(-Inf, 0), upper = c(Inf, Inf))
```

Coefficients:

```
      mu      sigma
2.998304 2.277506
```

This works because `mle()` calls `optim()`, which has a number of optimisation methods. The default method is BFGS. An alternative, the L-BFGS-B method, allows box constraints.

The other solution is to simply ignore the warnings. It's neater and produces the same results.

```
> LL <- function(mu, sigma) {
+   R = suppressWarnings(dnorm(x, mu, sigma))
+   #
+   -sum(log(R))
+ }
> mle(LL, start = list(mu = 1, sigma=1))
```

Call:

```
mle(minuslogl = LL, start = list(mu = 1, sigma = 1))
```

Coefficients:

```
      mu      sigma
2.998305 2.277506
```

The maximum-likelihood values for the mean and standard deviation are damn close to the corresponding sample statistics for the data. Of course, they do not agree perfectly with the values used when we generated the data: the results can only be as good as the data. If there were more samples then the results would be closer to these ideal values.

A note of caution: if your initial guess for the parameters is too far off then things can go seriously wrong!

```
> mle(LL, start = list(mu = 0, sigma=1))
```

Call:

```
mle(minuslogl = LL, start = list(mu = 0, sigma = 1))
```

Coefficients:

```
      mu      sigma
51.4840 226.8299
```

Fitting a Linear Model

Now let's try something a little more sophisticated: fitting a linear model. As before, we generate some data.

```
> x <- runif(N)
> y <- 5 * x + 3 + rnorm(N)
```

We can immediately fit this model using least squares regression.

```
> fit <- lm(y ~ x)
>
> summary(fit)
```

```
Call:
lm(formula = y ~ x)

Residuals:
    Min       1Q   Median       3Q      Max
-1.96206 -0.59016 -0.00166  0.51813  2.43778

Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept)   3.1080     0.1695   18.34 <2e-16 ***
x             4.9516     0.2962   16.72 <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
```

Residual standard error: 0.8871 on 98 degrees of freedom
Multiple R-squared: 0.7404, Adjusted R-squared: 0.7311
F-statistic: 279.5 on 1 and 98 DF, p-value: < 2.2e-16

The values for the slope and intercept are very satisfactory. No arguments there. We can superimpose the fitted line onto a scatter plot.

```
> plot(x, y)
> abline(fit, col = "red")
```

ml-estimation-scatter-fit

Pushing on to the MLE for the linear model parameters. First we need a likelihood function. The model is not a PDF, so we can't proceed in precisely the same way that we did with the normal distribution. However, if you fit a linear model then you want the residuals to be normally distributed. So the likelihood function fits a normal distribution to the residuals.

```
LL <- function(beta0, beta1, mu, sigma) {
  # Find residuals
  #
  R = y - x * beta1 - beta0
  #
  # Calculate the likelihood for the residuals (wi)
  #
  R = suppressWarnings(dnorm(R, mu, sigma))
  #
  # Sum the log likelihoods for all of the data points
  #
  -sum(log(R))
}
```

One small refinement that one might make is to move the logarithm into the call to dnorm().

```
LL <- function(beta0, beta1, mu, sigma) {
  R = y - x * beta1 - beta0
  #
  R = suppressWarnings(dnorm(R, mu, sigma, log = TRUE))
  #
  -sum(R)
}
```

It turns out that the initial guess is again rather important and a poor choice can result in errors. We will return to this issue a little later.

```
> fit <- mle(LL, start = list(beta0 = 3, beta1 = 1,
Error in solve.default(oout$hessian) :
  system is computationally singular: reciprocal condition
> fit <- mle(LL, start = list(beta0 = 5, beta1 = 3,
Error in solve.default(oout$hessian) :
  Lapack routine dgesv: system is exactly singular:
```

But if we choose values that are reasonably close then we get a decent outcome.

```
> fit <- mle(LL, start = list(beta0 = 4, beta1 = 2,
> fit
```

```
Call:
mle(minuslogl = LL, start = list(beta0 = 4, beta1 = 2,
  sigma = 1))
```

```
Coefficients:
    beta0    beta1      mu    sigma
3.5540217 4.9516133 -0.4459783 0.8782272
```

The maximum-likelihood estimates for the slope (beta1) and intercept (beta0) are not too bad. But there is a troubling warning about NaNs being produced in the summary output below.

```
> summary(fit)
Maximum likelihood estimation

Call:
mle(minuslogl = LL, start = list(beta0 = 4, beta1 =
  sigma = 1))

Coefficients:
      Estimate Std. Error
beta0  3.5540217      NaN
beta1  4.9516133  0.2931924
mu    -0.4459783      NaN
sigma  0.8782272  0.0620997

-2 log L: 257.8177
Warning message:
In sqrt(diag(object@vcov)) : NaNs produced
```

It stands to reason that we actually want to have the zero mean for the residuals. We can apply this constraint by specifying mu as a fixed parameter. Another option would be to simply replace mu with 0 in the call to dnorm(), but the alternative is just a little more flexible.

```
> fit <- mle(LL, start = list(beta0 = 2, beta1 = 1.5
  nobs = length(y))
> summary(fit)
Maximum likelihood estimation

Call:
mle(minuslogl = LL, start = list(beta0 = 2, beta1 =
  fixed = list(mu = 0), nobs = length(y))
```

```
Coefficients:
      Estimate Std. Error
beta0  3.1080423  0.16779428
beta1  4.9516164  0.29319233
sigma  0.8782272  0.06209969
```

```
-2 log L: 257.8177
```

The resulting estimates for the slope and intercept are rather good. And we have standard errors for these parameters as well.

How about assessing the overall quality of the model? We can look at the [Akaike Information Criterion](#) (AIC) and [Bayesian Information Criterion](#) (BIC). These can be used to compare the performance of different models for a given set of data.

```
> AIC(fit)
[1] 263.8177
> BIC(fit)
[1] 271.6332
> logLik(fit)
'log Lik.' -128.9088 (df=3)
```

Returning now to the errors mentioned above. Both of the cases where the call to mle() failed resulted from problems with inverting the [Hessian Matrix](#). With the implementation of mle() in the stats4 package there is really no way to get around this problem apart from having a good initial guess. In some situations though, this is just not feasible. There are, however, alternative implementations of MLE which circumvent this problem. The bbmle package has mle2() which offers essentially the same functionality but includes the option of not inverting the Hessian Matrix.

```
> library(bbmle)
>
> fit <- mle2(LL, start = list(beta0 = 3, beta1 = 1,
>
> summary(fit)
Maximum likelihood estimation

Call:
mle2(minuslogl = LL, start = list(beta0 = 3, beta1 =
  sigma = 1))
```

```
Coefficients:
      Estimate Std. Error z value Pr(z)
beta0  3.054021  0.083897  36.4019 <2e-16 ***
beta1  4.951617  0.293193  16.8886 <2e-16 ***
mu    0.054021  0.083897  0.6439 0.5196
```

```
sigma 0.878228    0.062100 14.1421 <2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.'
-2 log L: 257.8177
```

Here `mle2()` is called with the same initial guess that broke `mle()`, but it works fine. The summary information for the optimal set of parameters is also more extensive.

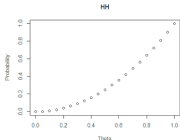
Fitting a linear model is just a toy example. However, Maximum-Likelihood Estimation can be applied to models of arbitrary complexity. If the model residuals are expected to be normally distributed then a log-likelihood function based on the one above can be used. If the residuals conform to a different distribution then the appropriate density function should be used instead of `dnorm()`.

Related



$$y_i = \mu \varepsilon_i$$

Maximum likelihood
October 13, 2011
In "R bloggers"



Who knew likelihood functions could be so pretty?
October 22, 2017
In "R bloggers"

Maximum Likelihood Estimation and the Origin of Life
September 8, 2013
In "R bloggers"

220
SHARES

Share

Tweet

To leave a comment for the author, please follow the link and comment on their blog:
[Exegetic Analytics » R](#)

R-bloggers.com offers [daily e-mail updates](#) about R news and [tutorials](#) on topics such as: [Data science](#), [Big Data](#), [R jobs](#), visualization ([ggplot2](#), [Boxplots](#), [maps](#), [animation](#)), programming ([RStudio](#), [Sweave](#), [LaTeX](#), [SQL](#), [Eclipse](#), [git](#), [hadoop](#), [Web Scraping](#)) statistics ([regression](#), [PCA](#), [time series](#), [trading](#)) and more...

If you got this far, why not **subscribe for updates** from the site?
Choose your flavor: [e-mail](#), [twitter](#), [RSS](#), or [facebook](#)...

Like 6

Share

Share

52

Comments are closed.

Search & Hit Enter

Recent popular posts

- [Deep Learning with R](#)
- [Add P-values and Significance Levels to ggplots](#)
- [Introducing the MonteCarlo Package](#)
- [How to create dot-density maps in R](#)

Most visited articles of the week

- [How to write the first for loop in R](#)
- [Installing R packages](#)
- [Using apply, sapply, lapply in R](#)
- [How to Make a Histogram with Basic R](#)
- [Tutorials for learning R](#)
- [How to perform a Logistic Regression in R](#)
- [Freedman's paradox](#)
- [In-depth introduction to machine learning in 15 hours of expert videos](#)
- [Shiny app to explore ggplot2](#)

Sponsors

EARL Conference
Tickets on sale now:
earlconf.com

EARL
ENTERPRISE APPLICATIONS OF THE R LANGUAGE

Learn R By Doing
www.DataCamp.com

Start Free Course



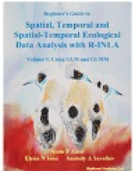
**ADVANCED VISUALIZATION
 FOR R SHINY AND DASH
 WITH PLOTLY**

TAKE THE MASTER CLASS →

RStudio Connect
 Push-Button Publishing | Scheduled Updates
 For all the work you create in R

R Studio **Free Evaluation**

Highland Statistics Ltd.
Course:
**Introduction to Regression
 Models with Spatial &
 Temporal Correlation**
Southampton, October 2017
Banff, December 2017



**Register for free updates
 in all areas of statistics**

**STATISTICS
 VIEWS**
 Bringing Statistics Together

WILEY

QUANTIDE
 knowledge from data

Quantide: statistical consulting and training

ODSC WEST
 San Francisco · November 2-4 **Register**

Learn everything from Predictive Analytics to Autonomous Vehicles

Accelerate AI / NOV 12 - 14
 SAN FRANCISCO

Learn how Data Science and AI impact your industry.

ai.odsc.com

R Courses for Professionals
Download R templates for machine learning

The fastest way to learn data science!

[VIEW SNEAK PEAK](#)

DATA SOCIETY

[R] Kenntnis-Tage 2017

Wissen & Vernetzen by  eoda

Data Science goes professional

Kassel
8.11. – 9.11.

FANTASY FOOTBALL ANALYTICS

USING STATISTICS & R

TO IMPROVE YOUR FANTASY FOOTBALL TEAM

Download a FREE eBook on R Programming!

Save 20% on All R Books

Promo Code **AZR08**

 **CRC Press**
Taylor & Francis Group
www.crcpress.com

 **NYC DATA SCIENCE ACADEMY**

Become a Data Scientist
Develop expertise in R, Python, Hadoop & Spark in just 12 Weeks

[Apply for Data Science Bootcamp](#)


jumping rivers

TRAINING: R, SCALA, STAN

STATWORX

[Data Science Service](#)

Data Science | Consulting | Development | Training


Try the FASTEST ML for R




Click for a Free Trial

 **YOTTAMINE ANALYTICS**

 Unique competence in statistics

 Training  Consulting

STATCON

Statistik: Consulting • Training • Software

<http://www.statcon.de>



[Contact us](#) if you wish to help support R-bloggers, and place **your banner here**.

[Jobs for R users](#)

- [Data Scientist @ Garching bei München, Bayern, Germany](#)
- [Software Developer](#)
- [Senior Quantitative Analyst, Data Scientist](#)
- [R data wrangler](#)
- [Senior Data Scientist](#)
- [Manager, Statistical Consulting & Data Science](#)
- [Financial Controller](#)

[Full list of contributing R-bloggers](#)

[R-bloggers](#) was founded by [Tal Galili](#), with gratitude to the [R](#) community.

Is powered by [WordPress](#) using a [bavotasan.com](#) design.

Copyright © 2017 **R-bloggers**. All Rights Reserved. [Terms and Conditions](#) for this website