

# S-Plus functions for M-estimators of the parameters of the Gamma distribution

A. Marazzi, A. Randriamiharisoa

August 28, 1995. Revised: April 1997

## Abstract

Marazzi and Ruffieux (1996) describe an implementation of M-estimators for the Gamma distribution. This report is an appendix describing the corresponding S-plus functions.

## S-Plus functions

This appendix of Marazzi and Ruffieux (1996) describes a set of S-plus functions that compute M-estimates of the parameters of a Gamma distribution. The shrinking component estimates of the standardized parameters are used and the estimates  $\hat{\alpha}$  and  $\hat{\sigma}$  of the shape parameter  $\alpha$  and the scale parameter  $\sigma$  are returned. The functions and their purpose are:

<code>M.E.gamma</code>	M-estimates $\hat{\alpha}$ and $\hat{\sigma}$ .
<code>M.L.gamma</code>	Maximum likelihood estimates $\hat{\alpha}$ and $\hat{\sigma}$ .
<code>AV.ME.gamma</code>	Asymptotic covariance matrix of the M-estimates.
<code>AV.ML.gamma</code>	Asymptotic covariance matrix of the maximum likelihood estimates.
<code>Tab.gamma</code>	Computation and storage of the matrix $A_b$ and the vector $c_b$ for several values of $\alpha$ .
<code>Crv.gamma</code>	Visual check. This function draws the graph of the function $S_{p2}(\bar{\sigma}(\alpha), \alpha)$ that should cross the horizontal axis for $\alpha = \hat{\alpha}$ . A neat intersection indicates a precise solution.
<code>Sp2.gamma</code>	Preliminary numerical computations for <code>Crv.gamma()</code> .

The algorithms are programmed in FORTRAN using parts of the subroutine library ROBETH (Marazzi (1993)). They must be compiled and loaded into S-plus. For the the Windows version, a library is made available (use the command `library(robgam,T)`).

---

**function M.E.gamma**

M-estimates of the shape and scale parameters of the Gamma distribution.

---

**Specification**

```
function M.E.gamma(y, Table, sigma=0, cov=F,  
                  maxit=100, tol=0.001, til=0.001)
```

**Purpose**

This function finds a solution  $\hat{\theta} = (\hat{\tau}, \hat{\alpha})$  of equation (8), Section 2. It returns  $\hat{\alpha}$ ,  $\hat{\sigma} = \exp(\hat{\tau})$ ,  $\hat{\mu} = \hat{\alpha}\hat{\sigma}$ , the estimate  $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$  of the asymptotic covariance matrix  $V(\hat{\vartheta}, F)$  of  $\hat{\vartheta} = (\hat{\sigma}, \hat{\alpha})$ , and the corresponding estimate of the asymptotic variance of the mean estimate  $\hat{\sigma}\hat{\alpha}$ .

The functions  $A_b(\alpha)$  and  $c_b(\alpha)$  are assumed to be given as input arguments. They must be computed and stored before calling `M.E.gamma()` by means of the function `Tab.gamma()`.

**Method**

See Section 4. The estimate  $\hat{\theta}$  is the solution of  $S_{pj}(\sigma, \alpha) = 0$ ,  $j = 1, 2$  and is computed according to the algorithm described in Section 4.3. The asymptotic covariance matrix  $V(\hat{\vartheta}, F)$  (with  $F = F_{\hat{\sigma}, \hat{\alpha}}$ ) is computed according to formula (36), Section 5.2, which depends on  $V(\hat{\theta}, F) = M^{-1}QM^{-T}$  (formula (13), Section 3) and  $J(\theta)$  (Section 5.3).

**Arguments**

<code>y</code>	Observation vector.
<code>Table</code>	<code>Table</code> is the object (list) returned by the function <code>Tab.gamma()</code> and must be created before calling <code>M.E.gamma()</code> . The main component of <code>Table</code> is a matrix containing values of $A_b(\alpha)$ and $c_b(\alpha)$ for several values of $\alpha$ .
<code>sigma</code>	If <code>sigma = 0</code> on input, both equations $S_{pj}(\sigma, \alpha) = 0$ , $j = 1, 2$ are solved for $\sigma$ and $\alpha$ . If <code>sigma &gt; 0</code> on input, $\sigma$ is set equal to <code>sigma</code> and kept fixed. In this case, equation $S_{p2}(\sigma, \alpha)$ is solved for $\alpha$ .
<code>cov</code>	If <code>cov=T</code> on input, the matrices $M$ , $Q$ , and $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ , as well as the asymptotic variance of $\hat{\xi} = \hat{\sigma}\hat{\alpha}$ , are computed and returned.
<code>maxit</code>	Maximum number of iterations for the regula-falsi algorithm used to invert $S_{p1}$ with respect to $\sigma$ .
<code>tol</code>	Required relative precision of $\hat{\sigma}$ and $\hat{\alpha}$ .
<code>til</code>	Desired relative accuracy of the numerical integration used to compute $M$ and $Q$ .

## Value

List with the following components

alpha	Estimate $\hat{\alpha}$ .
sigma	Estimate $\hat{\sigma}$ .
mu	Estimate $\hat{\mu} = \hat{\alpha}\hat{\sigma}$ .
ok	ok = 1 on output, indicates that a solution has been reached within the desired accuracy.
Cov	Asymptotic covariance matrix $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ .
V.mu	Asymptotic variance of $\hat{\sigma}\hat{\alpha}$ .
M	Matrix $M$ .
Q	Matrix $Q$ .
c	Value of $c_b(\alpha)$ for $\alpha = \hat{\alpha}$ .
A	Value of $A_b(\alpha)$ for $\alpha = \hat{\alpha}$ .
call	The calling sequence.

---

## function M.L.gamma

Maximum likelihood estimates of the shape and scale parameters of the Gamma distribution.

---

## Specification

```
function M.L.gamma(y, cov=F, maxit=100, tol=0.001)
```

## Purpose

This function finds the solution  $(\hat{\sigma}, \hat{\alpha})$  of the maximum likelihood equations

$$\begin{cases} \frac{1}{n} \sum_{i=1}^n y_i = \alpha\sigma, \\ \frac{1}{n} \sum_{i=1}^n \ln y_i = \ln \sigma + \dot{\Gamma}(\alpha). \end{cases}$$

It returns  $\hat{\alpha}$ ,  $\hat{\sigma}$ ,  $\hat{\mu} = \hat{\alpha}\hat{\sigma}$ , an estimate of the asymptotic covariance matrix  $V(\hat{\vartheta}, F)$  of  $\hat{\vartheta} = (\hat{\sigma}, \hat{\alpha})$ , and an estimate of the asymptotic variance of the mean estimate  $\hat{\sigma}\hat{\alpha}$ .

## Method

The Newton algorithm is used in order to solve for  $\alpha$  the second equation, where  $\sigma$  is replaced by  $(\sum y_i/n)/\alpha$ . For the asymptotic covariance matrix of the parameter estimates, see function AV.ML.gamma().

## Arguments

y	Observation vector.
cov	If cov=T on input, the matrix $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ as well as the asymptotic variance of $\hat{\xi} = \hat{\sigma}\hat{\alpha}$ , are computed and returned.
maxit	Maximum number of iterations for the Newton algorithm.
tol	Required relative precision of $\hat{\sigma}$ and $\hat{\alpha}$ .

## Value

List with the following components

<code>alpha</code>	Estimate $\hat{\alpha}$ .
<code>sigma</code>	Estimate $\hat{\sigma}$ .
<code>mu</code>	Estimate $\hat{\mu} = \hat{\alpha}\hat{\sigma}$ .
<code>nit</code>	Reached number of iterations.
<code>Cov</code>	Asymptotic covariance matrix $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ .
<code>V.mu</code>	Asymptotic variance of $\hat{\sigma}\hat{\alpha}$ .
<code>call</code>	The calling sequence.

---

## function `AV.ME.gamma`

Asymptotic covariance matrix of the parameter M-estimates.

---

## Specification

```
function AV.ME.gamma(alpha, sigma, Table, til=0.001)
```

## Purpose

This function returns the estimate  $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$  of the asymptotic covariance matrix  $V(\hat{\vartheta}, F)$  of  $\hat{\vartheta} = (\hat{\sigma}, \hat{\alpha})$ , and the corresponding estimate of the asymptotic variance of  $\hat{\mu} = \hat{\sigma}\hat{\alpha}$ .

The functions  $A_b(\alpha)$  and  $c_b(\alpha)$  are assumed to be given as input arguments. They must be tabulated before calling `AV.ME.gamma()` by means of the function `Tab.gamma()`.

## Method

See Section 5. The asymptotic covariance matrix  $V(\hat{\vartheta}, F)$  (with  $F = F_{\hat{\sigma}, \hat{\alpha}}$ ) is computed according to formula (36), Section 5.2, which depends on  $V(\hat{\theta}, F) = M^{-1}QM^{-T}$  (formula (13), Section 3) and  $J(\theta)$  (Section 5.3).

## Arguments

<code>alpha</code>	Estimate $\hat{\alpha}$ .
<code>sigma</code>	Estimate $\hat{\sigma}$ .
<code>Table</code>	<code>Table</code> is the object (list) returned by the function <code>Tab.gamma()</code> and must be created before calling <code>M.E.gamma()</code> . The main component of <code>Table</code> is a matrix containing values of $A_b(\alpha)$ and $c_b(\alpha)$ for several values of $\alpha$ .
<code>til</code>	Desired relative accuracy of the numerical integration used to compute $M$ and $Q$ .

## Value

List with the following components

alpha	The value of $\alpha$ on input.
sigma	The value of $\sigma$ on input.
Cov	Asymptotic covariance matrix $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ .
V.mu	Asymptotic variance of $\hat{\sigma}\hat{\alpha}$ .
M	Matrix $M$ .
Q	Matrix $Q$ .
call	The calling sequence.

---

## function AV.ML.gamma

Asymptotic covariance matrix of the maximum likelihood estimates.

---

## Specification

```
function AV.ML.gamma(alpha, sigma)
```

## Purpose

For given values of  $\hat{\alpha}$  and  $\hat{\sigma}$ , this function returns an estimate of the asymptotic covariance matrix  $V(\hat{\vartheta}, F)$  of  $\hat{\vartheta} = (\hat{\sigma}, \hat{\alpha})$  and an estimate of the asymptotic variance of the mean estimate  $\hat{\mu} = \hat{\sigma}\hat{\alpha}$ .

## Method

The asymptotic covariance matrix of  $(\hat{\sigma}, \hat{\alpha})$  is estimated by  $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ , where

$$V(\hat{\vartheta}, F_{\sigma, \alpha}) = \frac{1}{\omega} C^T \begin{bmatrix} \ddot{\Gamma}(\alpha) & \sigma \\ \sigma & \alpha\sigma^2 \end{bmatrix} C,$$
$$\omega = \sigma^2(\alpha\ddot{\Gamma}(\alpha) - 1), \quad C = \begin{pmatrix} -\sigma^2 & 0 \\ 0 & 1 \end{pmatrix},$$

and  $\ddot{\Gamma}(t) = d^2 \ln \Gamma(t) / dt^2$  is the Trigamma function.

## Arguments

alpha	Estimate $\hat{\alpha}$ .
sigma	Estimate $\hat{\sigma}$ .

## Value

List with the following components

Cov	Asymptotic covariance matrix $V(\hat{\vartheta}, F_{\hat{\sigma}, \hat{\alpha}})$ .
V.mu	Asymptotic variance of $\hat{\sigma}\hat{\alpha}$ .
call	The calling sequence.

---

## function `Tab.gamma`

Computation and storage of the matrix  $A_b$  and the vector  $c_b$  for several values of  $\alpha$ .

---

### Specification

```
function Tab.gamma(b1=1.5, b2=1.7, alpha1=0.5, alpha2=20.5, k=101,  
                  A=c(0,0,0), monit=1,  
                  maxta=1,maxtc=1,maxit=100, til=0.001,tol=0.001)
```

### Purpose

For a given value of the tuning constant  $b = (b_1, b_2)$ , this function computes values of the matrix  $A_b(\alpha)$  and values of the vector  $c_b(\alpha)$  – as solutions of equations (9) and (10), Section 2, for the shrinking component case – for  $k$  equally spaced values of  $\alpha$  in a given interval  $[\alpha_1, \alpha_2]$ .

### Method

Values of  $A_b(\alpha)$  and  $c_b(\alpha)$  are computed for  $\alpha = \alpha_1 + (i-1)(\alpha_2 - \alpha_1)/(k-1)$ ,  $i = 1, \dots, k$ , as solutions of the system of equations (9)–(10). For a given  $\alpha$ , the algorithm for solving (9) and (10) alternates between improving a trial value of  $A_b(\alpha)$ , for a given  $c_b(\alpha)$ , and improving a trial value of  $c_b(\alpha)$ , for a given  $A_b(\alpha)$ . The improved value of  $A_b(\alpha)$  is computed by means of a finite number (`maxta`) of steps of the algorithm based on (23). An improved value of  $c_b(\alpha)$  is computed by means of a finite number (`maxtc`) of steps of an iterative algorithm for M-estimates of location (see, e.g., Hampel et al. (1986), p. 263) which has been adapted to equation (10).  $A_b$  is stored as a vector  $(a_{11}, a_{21}, a_{22})$ .

### Arguments

<code>b1</code>	Tuning constant $b_1$ .
<code>b2</code>	Tuning constant $b_2$ .
<code>alpha1</code>	Minimum value $\alpha_1$ of $\alpha$ .
<code>alpha2</code>	Maximum value $\alpha_2$ of $\alpha$ .
<code>k</code>	Number of values of $\alpha$ , for which $A_b(\alpha)$ and $c_b(\alpha)$ must be computed.
<code>A</code>	Initial value of $A_b(\alpha)$ . If <code>A=c(0,0,0)</code> , $A_b(\alpha)$ is set to the solution of (9) for $b_1 = b_2 = \infty$ (in this case the solution can be explicitly computed). Otherwise, the initial value is specified by the user.
<code>monit</code>	If <code>monit</code> > 0 and the iteration counter is a multiple of <code>monit</code> , the current values of $A_b(\alpha)$ and $c_b(\alpha)$ are displayed. If no iteration monitoring is required, set <code>monit</code> equal to 0 or to a negative integer value.
<code>maxta</code>	Maximum number of steps for improving $A_b(\alpha)$ .
<code>maxtc</code>	Maximum number of steps for improving $c_b(\alpha)$ .
<code>maxit</code>	Maximum number of cycles of the main algorithm.
<code>til</code>	Desired relative accuracy of numerical integrations.
<code>tol</code>	Desired relative precision of each element of $A_b(\alpha)$ and $c_b(\alpha)$ .

## Value

List with the following components

Table	Matrix with $k$ rows. The $i$ th row contains $(c_{b1}(\alpha), c_{b2}(\alpha), a_{b11}(\alpha), a_{b21}(\alpha), a_{b22}(\alpha))$ for $\alpha = \alpha_1 + (i - 1)(\alpha_2 - \alpha_1)/(k - 1)$ ( $i = 1, \dots, k$ ).
call	The calling sequence.

---

## function Sp2.gamma

Numerical computations for drawing a graph of the function  $S_{p2}(\bar{\sigma}(\alpha), \alpha)$  or a graph of the function  $S_{p2}(\sigma_0, \alpha)$ , where  $\sigma_0$  is a given value of  $\sigma$ .

---

## Specification

```
function Sp2.gamma(grid, sigma=0, y, Table, tol=0.001, maxit=1000)
```

## Purpose

By default (argument `sigma = 0`), `Sp2.gamma()` computes the function  $S_{p2}(\bar{\sigma}(\alpha), \alpha)$  defined in Section 4.3 for all values of  $\alpha$  contained in the vector argument `grid`. If  $\sigma_0$  is a given positive number and `sigma =  $\sigma_0$`  on input, `Sp2.gamma()` computes the function  $S_{p2}(\sigma_0, \alpha)$  defined in Section 4.3 for all values of  $\alpha$  contained in the vector argument `grid`. The graph of  $S_{p2}(\bar{\sigma}(\alpha), \alpha)$  or the graph of  $S_{p2}(\sigma_0, \alpha)$  (that can be drawn by means of the function `Crv.gamma()`) should cross the horizontal axis for  $\alpha = \hat{\alpha}$ . A neat intersection indicates a precise determination of  $\hat{\alpha}$ .

## Method

If `sigma = 0`, `Sp2.gamma()` uses a regula-falsi method to invert  $S_{p1}$  with respect to  $\sigma$ , for all values of  $\alpha$  given in `grid`, and obtain an accurate solution  $\bar{\sigma}(\alpha)$  of  $S_{p1}(\sigma, \alpha) = 0$ . It returns  $S_{p2}(\bar{\sigma}(\alpha), \alpha)$  for  $\alpha \in \text{grid}$ . If `sigma =  $\sigma_0$` , `Sp2.gamma()` computes and returns  $S_{p2}(\sigma_0, \alpha)$  for  $\alpha \in \text{grid}$ . The values of  $c_b(\alpha)$  and  $A_b(\alpha)$  contained in `Table`, and linear interpolation, are used to compute  $S_{p1}$  and  $S_{p2}$ .

## Arguments

<code>grid</code>	Vector containing the values of $\alpha$ for which computations are desired. For example, <code>grid=seq(alpha1,alpha2,length=100)</code> , where <code>alpha1</code> and <code>alpha2</code> are the limits of the interval for the graph.
<code>sigma</code>	See Purpose.
<code>y</code>	Observation vector.
<code>Table</code>	Matrix containing values of $A_b(\alpha)$ and $c_b(\alpha)$ for several values of $\alpha$ . This table must be created by means of the function <code>Tab.gamma()</code> before calling <code>Sp2.gamma()</code> .
<code>tol</code>	Required relative precision of $\bar{\sigma}(\alpha)$ in the regula-falsi algorithm.
<code>maxit</code>	Maximum number of iterations in the regula-falsi algorithm.

## Value

**Sp2** Matrix with `length(grid)` rows. If `sigma = 0`, the  $i$ th row contains  $(\alpha, S_{p2}(\bar{\sigma}(\alpha), \alpha), Iterm)$ , where  $\alpha$  is the  $i$ th element of `grid`. *Iterm* is equal to 1, if the regula-falsi algorithm converged and 0 otherwise. If `sigma =  $\sigma_0$` , the  $i$ th row contains  $(\alpha, S_{p2}(\sigma_0, \alpha))$ , where  $\alpha$  is the  $i$ th element of `grid`.

---

## function `Crv.gamma`

Draws the graph of the function computed by `Sp2.gamma()`

---

## Specification

```
function Crv.gamma(Sp2, x.lim=c(0,0), y.lim=c(0,0),
                  Title="", Xlab="Alpha", Ylab="")
```

## Purpose

`Crv.gamma()` draws the graph of the function  $(S_{p2}(\bar{\sigma}(\alpha), \alpha)$  or  $S_{p2}(\sigma_0, \alpha))$  prepared by means of `Sp2.gamma()`.

## Method

`Crv.gamma()` calls `plot()` in order to draw the function defined by the first two columns of the matrix object produced by `Sp2.gamma()`.

## Arguments

<b>Sp2</b>	Object returned by <code>Sp2.gamma()</code> .
<b>x.lim</b>	Argument of the S-plus function <code>par</code> : limits for the horizontal axis. By default ( <code>x.lim=c(0,0)</code> ) the extreme values of $\alpha$ found in <code>Sp2</code> are used.
<b>y.lim</b>	Argument of the S-plus function <code>par()</code> : limits for the vertical axis. By default ( <code>y.lim=c(0,0)</code> ) the extreme values found in the second column of <code>Sp2</code> are used.
<b>Title</b>	Plot title.
<b>Xlab</b>	Horizontal axis label.
<b>Ylab</b>	Vertical axis label.

## Value

None.

## Example

```
> alpha <- 5; sigma <- 1
> n <- 50; eps <- 0.06
> n1 <- n*(1-eps); n2 <- n-n1
> y <- c(rgamma(n1,shape=alpha)*sigma, runif(n2,min=0,max=50))
> Table <- Tab.gamma(b1=1.3,b2=1.3,k=201,alpha1=0.5,alpha2=10.5)
> ey <- M.E.gamma(y,Table,cov=T)
> Sp2 <- Sp2.gamma(seq(0.5,10.5,length=50),y,Table)
> win.graph()
> Crv.gamma(Sp2, Title="ey")
> ey$alpha; ey$sigma; ey$V.mu
```

## References

Marazzi A. (1993). Algorithms, Routines, and S functions for Robust Statistics. Chapman and Hall, New York.

Marazzi A., Ruffieux C. (1996). Implementing M-estimators of the Gamma distribution. In: Rieder H. (Ed.), *Robust Statistics, Data Analysis, and Computer intensive Methods*, In Honor of Peter Huber's 60th Birthday, Springer Verlag, Heidelberg.

Marazzi A. (1997). Estimating and testing the means of asymmetric distributions using M-estimators. Technical Report. Institut universitaire de medecine sociale et preventive. Bugnon 17, CH-1005 Lausanne.