Sapienza
Università Editrice

ANNALI DEL DIPARTIMENTO DI METODI E MODELLI PER L'ECONOMIA IL TERRITORIO E LA FINANZA

2018

Direttore Responsabile - Director
Alessandra De Rose

Direttore Scientifico - Editor in Chief
Roberta Gemmiti

Curatori del numero - Managing Editors
Alessandra De Rose, Ersilia Incelli

Copyright © 2018

Sapienza Università Editrice
Piazzale Aldo Moro 5 – 00185 Roma

www.editricesapienza.it
editrice.sapienza@uniroma1.it

Iscrizione Registro Operatori Comunicazione n. 11420

ISSN: 2385-0825

Pubblicato a novembre 2018

Antonio Parisi[*], Brunero Liseo[*]

STATISTICAL INFERENCE WITH SKEW T DISTRIBUTIONS: THE `MVST` `R` PACKAGE

Abstract. We consider a Bayesian analysis of a general regression model with skew-elliptically distributed errors. In particular, we describe the choice of the prior distributions and we propose a Monte Carlo algorithm which allows: i) fast and accurate estimates of the posterior distribution of the parameters ii) to perform model choice among nested families of skew-elliptical classes of densities. All the methods described in the paper are implemented in the new version of the `R` package `mvst`.
The basic regression model can be modified in several ways. As illustrative examples, we consider a multivariate response model and, with an additional module that interfaces with the package, a stochastic frontier analysis.

Keywords: regression models, *skew-normal*, *skew-t*, stochastic frontiers, `R`, model selection.

1. Introduction

In the last two decades there has been an explosion of interest around the possibility of constructing models which generalize the Gaussian distributions in terms of skewness and extra-kurtosis. Interest can be partially explained with the empirical observations of phenomena in different disciplines, which could not be easily represented via Gaussian distributions (see Genton (2004) and Azzalini (2014) for general accounts). In this perspective, different proposals of skew-Student t distributions have been proposed and they now play a significant role as empirical models for skew and/or heavy-tailed data.

In a seminal paper Azzalini (1985) introduced a class of densities, named *skew-normal*, which includes the normal density as a proper member. This class has been successively generalized to the multivariate case in Azzalini and Dalla Valle (1996) and Azzalini and Capitanio (1999). A p-dimensional *skew-normal* random vector $\boldsymbol{W}$ with scale matrix Σ and shape parameter $\boldsymbol{\alpha}$ has density function

$$f(\boldsymbol{w}) = 2\varphi(\boldsymbol{w}-\boldsymbol{\xi};\Sigma)\Phi\left(\boldsymbol{\alpha}^\top\boldsymbol{\omega}^{-1}(\boldsymbol{w}-\boldsymbol{\xi})\right), \quad \boldsymbol{w}\in\mathbb{R}^p,$$

where $\boldsymbol{\omega}$ is a diagonal $p\times p$ matrix with the i-th element equal to $\Sigma_{ii}^{1/2}$; $\boldsymbol{W}$ will be denoted as $\boldsymbol{W}\sim SN_p(\boldsymbol{\xi},\Sigma,\boldsymbol{\alpha})$. A formal definition of the *skew-t* distribution in terms of scale mixture of *skew-normal* random vectors was proposed by Azzalini and Capitanio (2003). In practice, if $\boldsymbol{W}\sim SN_p(\Sigma,\boldsymbol{\alpha})$ (with $\boldsymbol{\xi}=\boldsymbol{0}$), and V is a scalar r.v., independent of $\boldsymbol{U}$ and such that $\nu V\sim\chi^2_\nu$, then

$$\boldsymbol{Y} = \boldsymbol{\xi} + V^{-1/2}\boldsymbol{W}$$

is defined as a p-dimensional *skew-t* density. Azzalini and Capitanio (2003) derived a closed form expression of the density of $\boldsymbol{Y}$

$$f_{\boldsymbol{Y}}(\boldsymbol{y}) = 2t_p(\boldsymbol{y},\nu)\,T_1\left(\boldsymbol{\alpha}^\top\boldsymbol{\omega}^{-1}(\boldsymbol{y}-\boldsymbol{\xi})\sqrt{g(\boldsymbol{y};\nu)};\nu+p\right), \tag{1}$$

where $t_p(\boldsymbol{y};\nu)$ is the standard p-dimensional Student-t distribution with ν degrees of freedom,

$$g(\boldsymbol{y};\nu) = \frac{\nu+d}{Q(\boldsymbol{y};\nu)}, \quad Q(\boldsymbol{y};\nu) = (\boldsymbol{y}-\boldsymbol{\xi})^\top\Sigma^{-1}(\boldsymbol{y}-\boldsymbol{\xi}),$$

and $T_1(\boldsymbol{u},s)$ is the CDF of a scalar t distribution with s degrees of freedom, evaluated at $\boldsymbol{u}$.

[*] Università degli Studi di Roma “Tor Vergata”
[*] Sapienza Università di Roma

The *skew-normal* and *skew-t* models are characterized by remarkably elegant formal properties and flexibility. On the other hand, they also pose several inferential difficulties. In a frequentist setup, the estimate of the parameters of a *skew-normal* model is particularly challenging even in the univariate case. Many different methods have been proposed to overcome this problem, which are accounted and summarized in Azzalini (2014). Problems are mitigated in the *skew-t* family, where Azzalini and Arellano-Valle (2013) propose a maximum penalized likelihood approach for estimating the shape parameter of that model. However, to our knowledge, there is not a broadly satisfactory method for choosing among the two nested families of distributions.

From a Bayesian perspective, starting from a new parametrization of the bivariate *skew-normal* model (see for example, Azzalini, 2014), Liseo and Parisi (2013) propose a set of priors and a specifically designed sampler based on a sampling importance resampling philosophy (Celeux, Marin, and Robert, 2006). Parisi and Liseo (2017) extend the analysis to a more general p-variate *skew-t* model; see also Frühwirth-Schnatter and Pyne (2010) for an alternative approach, based on Gibbs sampler.

In this paper, and in the related package `mvst`, we extend the analysis to a (possibly multivariate) regression model

$$\boldsymbol{Y}_i = \boldsymbol{B}^\top \boldsymbol{X}_i + \boldsymbol{\eta}_i, \qquad i = 1, \dots, n, \tag{2}$$

where $\boldsymbol{\eta}_i$ is the error vector term, for which we assume a multivariate skew-elliptical distribution. In particular we adopt the *skew-t* family of densities or some special case of it; namely, the Student-t class of densities, the skew-normal one and, of course, the Gaussian model. See also Rubio and Genton (2016).
The paper is mainly devoted to the description of the new version of the `R` (`R` Core Team, , 2015) package `mvst`, which contains the implementation of the methods proposed in Liseo and Parisi (2013) and Parisi and Liseo (2017). Furthermore, the package also provides functions which allow us to

- generate pseudo-random draws from the multivariate skew-elliptical models;
- compute maximum *augmented* likelihood estimates of the parameters[1];
- evaluate the probability density function for skew-elliptical distributions.

There already exist other software packages which deal with the skew-elliptical distributions. Among the `R` packages, `sn` (Azzalini, 2018) is the most general one, at least for manipulating *skew-normal* and *skew-t* distributions and for making frequentist inference. There is, however, a number of other packages:

- `skewt` provides the probability density function (pdf), the cumulative distribution function (cdf), the quantile function and a device for generating pseudo-random values for the *skew-t* distribution of Fernández and Steel (1998),
- `EMMIXuskew` (Lee and McLachlan, 2013) fits, in a likelihood framework, the unrestricted multivariate *skew-t* mixture model,
- `EMMIXcskew` (Lee and McLachlan, 2018) fits, in a likelihood framework, the finite mixture of multivariate canonical fundamental *skew-t* distributions.

There are also resources for other programs, such as a `stata` suite of commands for fitting the *skew-normal* and *skew-t* models (Marchenko and Genton, 2010)[2].
To our knowledge, none of these packages provides functions which estimate the parameters of the model (2) from a Bayesian perspective.

The `mvst` makes use of the GNU Scientific Library (see Galassi et al., 2018) to speed up the heaviest part of the code. It also requires three `R` packages: `mvtnorm` (see Genz and Bretz, 2009 and Genz et al., 2018), `MCMCpack` (Martin, Quinn, and Park, 2011) and `mnormt` (Azzalini and Genz, 2016). It also makes use of three scripts available in the `RcppGSL` package (Eddelbuettel and Romain, 2017).

The paper is organized as follows: in the next Section 2 we detail the statistical model, propose an alternative parametrization and describe the prior distribution. Section 3 is devoted to a detailed description of the use of the package `mvst`. Sections 4 and 5 are dedicated to some specific illustrations in applications, namely a multivariate regression model and a stochastic frontier one (Kumbhakar and Lovell, 2000).

[1]Here the word augmented is related to a likelihood function augmented by a latent structure which allows a computationally simpler representation of *skew-normal* and *skew-t* density functions.

[2]A complete list of other resources is available at `http://azzalini.stat.unipd.it/SN/`.

2. The model

Let $\boldsymbol{\theta}^\star$ and $\boldsymbol{\theta}$ respectively denote the sets of the parameters $\{\boldsymbol{\xi}, \boldsymbol{\delta}, \Sigma, \nu\}$ and $\{\boldsymbol{\xi}, \boldsymbol{\psi}, G, \nu\}$, where

$$\begin{aligned}
\boldsymbol{\delta} &= \frac{1}{(1+\boldsymbol{\alpha}^\top \Omega \boldsymbol{\alpha})^{1/2}} \Omega \boldsymbol{\alpha}, \\
\boldsymbol{\psi} &= \boldsymbol{\omega}\boldsymbol{\delta}, \\
G &= \boldsymbol{\omega}(\Omega - \boldsymbol{\delta}\boldsymbol{\delta}^\top)\boldsymbol{\omega} = \Sigma - \boldsymbol{\psi}\boldsymbol{\psi}^\top.
\end{aligned}$$

We introduce both parametrizations since, from a Bayesian perspective, it is more natural to elicit the prior distribution using the $\boldsymbol{\theta}^\star$ parametrization, while the $\boldsymbol{\theta}$ parametrization is more efficient to implement our sampling strategy. Using the `mvst` package, it is possible to define a custom function to compute subjective prior density (see §5). As a default choice, the package uses the following prior structure

$$\pi(\boldsymbol{\theta}^\star) = \pi(\boldsymbol{\xi})\pi(\boldsymbol{\delta}|\Sigma)\pi(\Sigma)\pi(\nu),$$

with

$$\pi(\boldsymbol{\xi}) \propto 1, \quad \Sigma \sim IW(m, \Lambda) \text{ with } m = 0 \text{ and } \Lambda = \mathbf{0}.$$

For the number of degrees of freedom we adopt the objective prior proposed in Villa and Walker (2014) and Villa and Rubio (2018), where the parameter space for ν is restricted to integers between 1 and 30 since, after that value, a Student-t is no longer distinguishable, at least in a Kullback-Leibler sense, from a Gaussian distribution.

Conditional on the value of Σ, and hence on the value of Ω, the prior on $\boldsymbol{\delta}$ is uniform on the region of admissible values, that is

$$\pi(\boldsymbol{\delta}|\Sigma) = \left(\frac{\pi^{p/2}}{\Gamma(p/2+1)} \sqrt{|\Omega|} \right)^{-1}. \tag{3}$$

The Jacobian of the transformation $\boldsymbol{\theta}^\star \to \boldsymbol{\theta}$ is

$$|J| = \prod_{j=1}^{p} (G_{jj} + \psi_j^2)^{-1/2}.$$

Priors for the nested Student-t, *skew-normal* and normal models are coherent restrictions of the above prior, obtained by setting at specific values some of the above quantities.

2.1 An alternative representation of the skew-t model

There is a useful stochastic representation of the random vector $\boldsymbol{Y}$. Let $I_A(\cdot)$ be the indicator function of the set A and let us define

$$\begin{pmatrix} Z \\ \boldsymbol{X} \end{pmatrix} \sim N_{p+1} \left[\begin{pmatrix} 0 \\ \mathbf{0} \end{pmatrix}, \begin{pmatrix} 1 & \boldsymbol{\delta}^T \\ \boldsymbol{\delta} & \Omega \end{pmatrix} \right],$$

and

$$\boldsymbol{U} = (-1)^{I_{(-\infty,0)}(Z)} \boldsymbol{X}, \qquad V \sim \Gamma(\nu/2, \nu/2),$$

with V independent of U. Then, the random vector

$$\boldsymbol{Y} = \boldsymbol{\xi} + \boldsymbol{\omega} \boldsymbol{U} V^{-1/2} \sim ST_p(\boldsymbol{\xi}, \boldsymbol{\alpha}, \Sigma, \nu) \tag{4}$$

and the joint density of $(\boldsymbol{Y}, Z, V)$ is given by

$$\begin{aligned}
f_{p+2}(\boldsymbol{y}, z, v) &= f_p(\boldsymbol{y} \mid z, v) f(z) f(v) = N_p \left(\boldsymbol{\xi} + \boldsymbol{\omega}\boldsymbol{\delta} \frac{|z|}{\sqrt{v}}, \frac{1}{v} \boldsymbol{\omega}(\Omega - \boldsymbol{\delta}\boldsymbol{\delta}^\top)\boldsymbol{\omega} \right) \\
&\times \; N_1(z, 0, 1) \times \Gamma(v, \nu/2, \nu/2).
\end{aligned} \tag{5}$$

The above representation (4) will be used for performing Bayesian inference and for obtaining pseudo-random draws from the skew-elliptical models.

The density in eq. (5) is useful for finding the complete maximum likelihood (CML) estimators, that is, the estimators for the model parameters in the case in which the latent variables $\boldsymbol{z}$ and $\boldsymbol{v}$ were given. In that case, the likelihood has a simpler form and the CML estimators are obtained as

$$\begin{aligned}
\hat{\boldsymbol{\psi}}_{CML} &= \frac{1}{(\sum_{i=1}^n z_i^2)(\sum_{i=1}^n v_i) - (\sum_{i=1}^n |z_i|\sqrt{v_i})^2} \\
&\times \left[\left(\sum_{i=1}^n v_i\right)\left(\sum_{i=1}^n |z_i|\sqrt{v_i}\boldsymbol{y}_i\right) - \left(\sum_{i=1}^n |z_i|\sqrt{v_i}\right)\left(\sum_{i=1}^n v_i\boldsymbol{y}_i\right)\right], \\
\hat{\boldsymbol{\xi}}_{CML} &= \frac{1}{\sum_{i=1}^n v_i}\left[\left(\sum_{i=1}^n v_i\boldsymbol{y}_i\right) - \hat{\boldsymbol{\psi}}_{CML}\left(\sum_{i=1}^n |z_i|\sqrt{v_i}\right)\right], \\
\hat{G}_{CML} &= \frac{1}{n}\sum_{i=1}^n v_i\,\hat{\varepsilon}_i\,\hat{\varepsilon}_i^\top,
\end{aligned}$$

where

$$\hat{\varepsilon}_i = \boldsymbol{y}_i - \hat{\boldsymbol{\xi}}_{CML} - \hat{\boldsymbol{\psi}}_{CML}\frac{|z_i|}{\sqrt{v_i}}. \tag{6}$$

The estimator for ν does not have a closed form expression: it is the solution of the following equation

$$n\log(\hat{\nu}_{CML}/2) - n\,\psi(\hat{\nu}_{CML}/2) = \sum_{i=1}^n v_i - \sum_{i=1}^n \log(v_i) - n,$$

where $\psi(\cdot)$ denotes the digamma function.
The CML estimators are used in the initialization of the posterior sampler. They can also be useful in simulation studies, where the performances of a procedure can be measured with respect to the CML estimates instead of the real value of the parameters.

2.2 The sampling strategy

The outline of the algorithm employed to estimate the model has been already described in Parisi and Liseo (2017). Additional details regarding regression models with skewed errors will be given in §4.

The algorithm consists in an iterated version of the importance sampling: at each generic t-eth iteration, a population of N particles is generated from the proposal distribution of each model parameter. An importance weight $\tilde{\zeta}_j^{(t)}$ is then attached to each particle, and these weights will be used to resample the particles before the successive iteration. The particles, their weights and the list of the indices of the resampled particles are sufficient to compute any quantity of interest. In particular, the marginal likelihood results as a byproduct of the sampler (see Liseo and Parisi, 2013). Another example is given in Appendix A, which illustrates the computation of the Effective Sample Size, which is often employed to assess the performances of importance samplers.

3. Using the package

In this section we will illustrate the functions exported by the package using examples taken from the literature.

3.1 Model estimates

The main function of the package is `mcSE`, which performs a Monte Carlo sampling for a p-variate skew-elliptical model. As an illustration of the proposed approach, we consider the wine data used in Azzalini (2014), §5.2.1. The dataset contains chemical measurements from a set of 178 specimens of Italian wines, belonging to three cultivars (Barbera, Barolo and Grignolino). Data are available in the `sn` package.

```
R> library(sn)
R> data(wines)
```

Following the book's example, we considered only three variables (tartaric acid, malic acid and fixed acidity) measuread on the specimens from the Grignolino cultivar (71 observations).
We estimated a bivariate *skew-normal* model using the variables measuring the tartaric and malic acid in the specimens. We used the fixed acidity in §4, as a covariate in a regression model.

```
R> indices = which(wines[,1] == 'Grignolino')
R> y = cbind(wines[indices, 'tartaric'], wines[indices, 'malic'])
```

The `mcSE` function requires (at least) a dataset, the details for the sampler (namely, the number `N` of particles and the number `Ti` of iterations), and the `modelType`. Currently, the package implements the Normal (`modelType='N'`), Student-*t* (`'T'`), the *skew-normal* (`'SN'`) and the *skew-t* (`'ST'`) options.
The function also has the `warmUp` argument, which is a logical flag. If set as `TRUE`, it improves the initialization procedure by running some additional iterations with a small number of particles (by default, 3 iterations with $N/10$, $N/5$ and $N/2$ particles). Appendix B illustrates this procedure.
The optional argument `X` is useful in a regression setting and it is explained in §4.
Finally, the function also accepts a `control` list with tuning parameters. Among the others

- `Nwu` allows one to modify the number of warm-up iterations and the number of particles generated by each of them,
- `saveParticles` is a logical flag indicating whether the main objects of the simulation should be saved,
- `verbose` is a logical flag; if `TRUE`, details about the progress of the algorithm are printed.

In particular, if `saveParticles` is `TRUE`, the populations of proposed particles, their weights and the list of the indices of the resampled particles will be saved on file. These objects are sufficient to compute any quantity of interest.

The following lines estimate a bivariate *skew-normal* model for data in `y`, using $T = 6$ iterations with $N = 20000$ particles each

```
R> library(mvst)
R> set.seed(159)
R> fit = mcSE(y=y, N=20000, Ti=6, modelType='SN', warmUp=TRUE)
```

The `mcSE` object `fit` contains details of the estimated model. This output is represented in the form of a list containing, for each iteration, an estimate of the model parameters and marginal likelihood. Moreover, it also includes, for each iteration, the number of resampled particles and a perplexity measure (see Robert and Casella, 2010). Finally, it contains a table with some information about the parameters of the models.

The `fit` list is intended to be useful for other functions. In general, the number of resampled particles is the only element which could be of direct interest. For example, one could graphically assess the occurrence of the so-called "degeneracy phenomenon" plotting these values over the iterations. Figure 1 has been obtained using the command

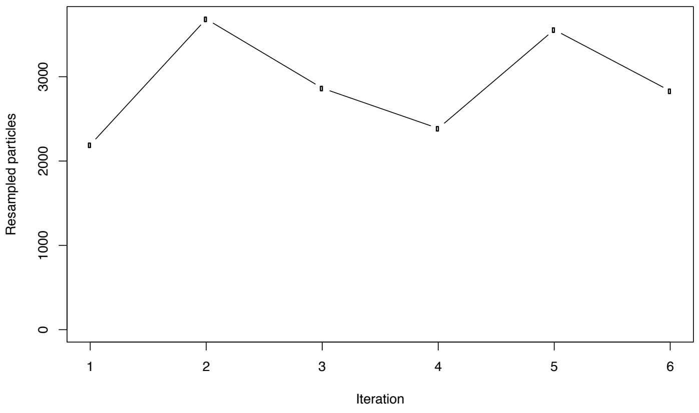
```
R> plot(fit$nResampled, xlab='Iteration', ylab='Resampled particles',
+   ylim=c(0,max(fit$nResampled)), pch=20, type='b')
```

In this particular case, there is no graphical evidence of degeneracy in the simulation. In fact, more than 2000 distinct particles (that is, more than 10% of the total) are resampled after each iteration, which is quite satisfactory for this kind of model.

A readable summary of the `fit` list can be obtained using the `summary` method for `mcSE` objects:

```
R> SNstats = summary(fit)

      Estimate  Std. Error      Q5%        Me      Q95%
xi1    1.90690     0.07639   1.7969   1.90887   2.03178
xi2    0.95151     0.08971   0.8152   0.95038   1.08555
```

Figure 1. Number of (distinct) particles resampled for each iteration.



```
G1      0.18879      0.03215   0.1462   0.18358  0.24264
G2     -0.02795      0.04028  -0.1022  -0.02308  0.02858
G3     -0.02795      0.04028  -0.1022  -0.02308  0.02858
G4      0.18852      0.07226   0.1086   0.16768  0.32609
psi1   -0.02513      0.08144  -0.1541  -0.02904  0.10745
psi2    1.33105      0.12874   1.1297   1.33203  1.52634

 log(p(y)) = -140.5493
```

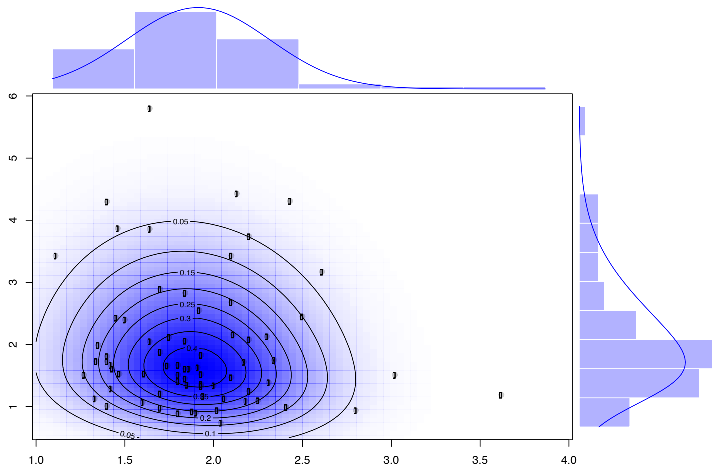
The table contains, for each model parameter, the estimates of the posterior mean, the standard error, the quantiles of order 5%, 50% and 95%. It also contains an evaluation of the marginal likelihood.

The package includes a `coef` method for the summary of the estimated models. It is possible to extract a list containing the point estimate $\hat{\boldsymbol{\theta}}$ using:

```
R> thetaHat = coef(SNstats)
R> thetaHat

$xi
[1] 1.9069013 0.9515116

$G
            [,1]        [,2]
[1,]  0.18878860 -0.02795028
[2,] -0.02795028  0.18851620

$psi
[1] -0.02513073  1.33105290
```

The package provides the function `dmvSE` which computes the probability density function of the considered models. For example, given $\hat{\boldsymbol{\theta}}$, the log-density at the point $\boldsymbol{y} = (2, 3)^\top$ is:

Figure 2. Contour plot of the density function of the estimated skew-normal model. Points represent the original data.



```
R> dmvSE(y=c(2,3), modelType='SN', theta=thetaHat, LOG=T)

[1] -1.780584
```

The function `bivPlot` draws the scatterplot, the histograms and the (joint and marginal) densities in Figure 2

```
R> bivPlot(y, modelType='SN', theta=thetaHat)
```

Notice that, in our parametrization, it is particularly simple to obtain the marginal distributions. In fact, if $\boldsymbol{Y} \sim ST_p(\boldsymbol{\xi}, \boldsymbol{\psi}, \Sigma, \nu)$ and j is a subset of $\{1, 2, \ldots, p\}$, then $\boldsymbol{Y}_j$ has distribution

$$\boldsymbol{Y}_j \sim ST_p(\boldsymbol{\xi}_j, \boldsymbol{\psi}_j, \Sigma_{jj}, \nu).$$

A proof is provided in Appendix C.

3.2 Model choice

The marginal log-likelihood returned by the `summary` function allows one to perform a model selection procedure using the Bayes factor.
We can define the vector `models`, collecting all competing models and initialize the vector `margLogLikes`, which will store the value of the marginal log-likelihood for each model

```
R> models = c('N', 'T', 'SN', 'ST')
R> nModels = length(models)
R> margLogLikes = numeric(nModels)
R> names(margLogLikes) = models
```

Iterating the estimation on all the candidate models, we obtain all the marginal log-likelihoods

```
R> for(iModel in 1:nModels){
R>  set.seed(159)
R>  fit = mcSE(y=y, N=20000, Ti=6, modelType=models[iModel], warmUp=T)
R>  margLogLikes[iModel] = summary(fit)$log.margLike
R> }
R> margLogLikes

        N         T        SN        ST
-147.1583 -140.6366 -140.5493 -136.1394
```

The marginal log-likelihood for the (multivariate) normal model is also available in closed form. Hence, it is possible to compute its exact value instead of the numerical approximation. In our case, the `MNmargLike` function returns a value which is, as expected, fairly close to its numerical approximation

```
R> MNmargLike(y, LOG=T)

[1] -147.1575
```

Using a uniform prior on the space of competing models, the *skew-t* model obtains the highest posterior probability (97.7%), indicating the presence of both skewness and extra-kurtosis in the data. The Normal model obtains a negligible value, while both the Student-t and *skew-normal* ones obtain approximately a 1% posterior probability.

3.3 Random number generator

The `mvst` package makes use of the representation given in eq. (4) to obtain pseudo-random draws from the *skew-t* model (and nested ones). As an example, let us suppose we want to generate draws from a *skew-normal* random variable with parameters

$$\boldsymbol{\theta}_0 = \{\xi = (1,2,3)^\top, G = \mathbb{I}_3, \psi = (0.3,0.4,0.2)^\top\}.$$

The parameter values are defined as a list

```
R> theta0 = list(xi=c(1,2,3), G=diag(3), psi=c(0.3,0.4,0.2))
```

In the *skew-normal* case, `theta0` contains the location parameter `xi`, the scale parameter `G` and the shape parameter `psi`. Any other element eventually present in the list would be ignored.

The function `rmvSE` requires

- `n`: the number of draws,
- `p`: the dimension of the r.v.,
- `X`: an optional argument which is useful in a regression setting,
- `modelType`: the model specification,
- `theta`: a *named list* with elements having the names of the model parameters.

```
R> set.seed(159)
R> skewData = rmvSE(n=2, p=3, X=NULL, modelType='SN', theta=theta0)
R> skewData

$y
          [,1]     [,2]     [,3]
[1,] 0.9548251 4.469901 4.356817
[2,] 0.3380907 2.381808 3.768723

$z
```

```
[1] -1.601373 -0.370687

$v
NULL

$X
NULL
```

The output list contains 4 elements: the draws `y`, the value of the latent variables `z` and `v` and `X`. In the example, the latent variables V are not defined for the *skew-normal* model, so the element `v` contains the `NULL` object.

3.4 Complete maximum likelihood estimates

Given a set of observations and the vector(s) of latent variables, the function `cmlSE` returns the CML estimates. To illustrate the function, we will generate 200 observations from the same r.v. as in the previous section

```
R> set.seed(159)
R> skewData = rmvSE(n=200, p=3, X=NULL, modelType='SN', theta=theta0)
```

We need the matrix of observations (`skewData$y`) and the value of the latent variables (`skewData$z`) to obtain the complete maximum likelihood estimates for the *skew-normal* model

```
R> fitCML = cmlSE('SN', y=skewData$y, z=skewData$z)
R> fitCML

$xi
[1] 1.118885 2.188577 2.879295

$G
            [,1]        [,2]       [,3]
[1,]  0.99016304 -0.01268072 0.14078652
[2,] -0.01268072  1.08991065 0.03684599
[3,]  0.14078652  0.03684599 0.97351426

$psi
[1] 0.2893048 0.7014148 0.2841432
```

The output contains a list with elements representing the CML estimate for $\boldsymbol{\xi}$, $\boldsymbol{\psi}$ and G.

4. A multivariate *skew-t* regression model

Consider the regression model in eq. (2), where $\boldsymbol{B}$ is a $(k \times p)$ matrix of regression parameters and $\boldsymbol{X}_i$ is a vector with the value of the k regressors for the i-eth observation. This model can be expressed as

$$\boldsymbol{y}_i \sim ST_p(\boldsymbol{B}^\top \boldsymbol{X}_i, \boldsymbol{\psi}, G, \nu).$$

It is necessary to find a proposal distribution for $\boldsymbol{B}$, while, for all the other parameters, it is sufficient to replace $\boldsymbol{\xi}$ with $\boldsymbol{B}^\top \boldsymbol{X}_i$.

Let $\boldsymbol{V}$ denote an $n \times n$ diagonal matrix whose i-eth value is v_i and let $\boldsymbol{H}$ denote a $n \times p$ matrix whose i-eth row contains the vector

$$\left(\boldsymbol{y}_i - \boldsymbol{\psi}\frac{|z_i|}{\sqrt{v_i}}\right)^\top.$$

Following Roy and Hobert (2010), under a flat prior for $\boldsymbol{B}$, its full conditional distribution is given by

$$(\boldsymbol{B}|\cdots) \sim MN(\boldsymbol{S}^{-1}\boldsymbol{C}_{\boldsymbol{\psi}}, \boldsymbol{S}^{-1}, \boldsymbol{G}),$$

where

$$\boldsymbol{S} = \boldsymbol{X}^\top \boldsymbol{V} \boldsymbol{X}, \qquad \boldsymbol{C}_{\boldsymbol{\psi}} = \boldsymbol{X}^\top \boldsymbol{V} \boldsymbol{H},$$

and the symbol $MN(\boldsymbol{M}, \boldsymbol{R}, \boldsymbol{\Delta})$ refers to a matrix normal random variable (see, for example Dawid, 1981 or Gupta and Nagar, 1999), with location $\boldsymbol{M}$ and scale parameters $\boldsymbol{R}$ and $\boldsymbol{\Delta}$.
Simulating draws from this distribution is simple, as it is linked to the multivariate normal distribution by a simple relation

$$\boldsymbol{B} \sim MN_{p\times k}(\boldsymbol{M}, \boldsymbol{R}, \boldsymbol{\Delta})$$

if and only if

$$\text{vec}(\boldsymbol{B}) \sim N_{pk}(\text{vec}(\boldsymbol{M}), \boldsymbol{R} \otimes \boldsymbol{\Delta}),$$

where $\otimes$ denotes the Kronecker product.

4.1 The Grignolino cultivar

The cmlSE, dmvSE, mcSE, rmvSE and MNmargLike functions support the presence of covariates. Where available, they should be collected into a design matrix (including the optional constant column) and provided with the X argument.
As an illustration of the proposed approach, we will again consider the wine data. We will estimate a multivariate response linear regression model in which both tartaric and malic acid (which have been already stored in the object y) are regressed over the fixed acidity. The following command creates the design matrix of the regression model

```
R> X = cbind(1, wines[indices, 'acidity'])
```

The model with skew-normally distributed errors can be estimated using the command

```
R> set.seed(159)
R> fit = mcSE(y=y, X=X, N=20000, Ti=6, modelType='SN', warmUp=T)
R> summary(fit)

        Estimate  Std. Error          Q5%         Me        Q95%
B1      1.291242    0.236532   9.448e-01   1.281072   1.7027275
B2      0.004004    0.002657  -3.206e-05   0.003803   0.0084142
B3     -0.984568    0.459512  -1.750e+00  -0.975135  -0.2206463
B4      0.037872    0.005238   2.964e-02   0.037660   0.0468244
G1      0.120600    0.045563   5.676e-02   0.117443   0.1990400
G2     -0.069191    0.048203  -1.537e-01  -0.063954  -0.0009862
G3     -0.069191    0.048203  -1.537e-01  -0.063954  -0.0009862
G4      0.586956    0.123401   4.270e-01   0.573384   0.8047241
psi1    0.350907    0.196738  -7.010e-02   0.404512   0.5774212
psi2   -0.189242    0.260757  -5.510e-01  -0.221620   0.2421843

 log(p(y)) = -133.3839
```

Repeating the procedure in §3.2, we find that the covariates can explain part of the skewness observed in the data. In fact, the Normal model obtains the highest posterior probability (43%), while the Student-t has a 41%.

5. The stochastic frontier model

The stochastic frontier model (see, for example, Kumbhakar and Lovell, 2000) represents a particular case of a regression model with skewed errors. We use this model to illustrate the possibility of modifying the prior distributions used as a default by the package.

A model for a production function can be written as in eq. (2), where the error term has a negative skewness, due to the inefficiency. On the contrary, for cost frontiers, the error term has a positive skewness. Hence, it is necessary to constrain the prior distribution of the shape parameter in order to allow only for positive (or negative) inefficiencies.

As an example, we consider the U.S. airline data from table F6.1 in Greene (2012) [3] and the cost frontier model given in §5 of Horrace and Parmeter (2018). The dataset contains 90 observations about costs (C), output (Q), price of fuel (PF), and load factor (LF) for 6 airlines over 15 years. Ignoring the panel structure, the cost model is given by

$$\log C_i = \beta_0 + \beta_1 \log Q_i + \beta_2 \log PF_i + \beta_3 LF_i + \eta_i \qquad i = 1, 2, \ldots, n.$$

In the following, we will assume a *skew-normal* distribution for the error terms η_i, with a positivity constraint on the shape parameter ψ. The model is univariate, hence here ψ is a scalar parameter.

The following code imports the dataset and defines the relevant objects

```
R> dataset = read.csv('TableF6-1.csv')
R> y = log(dataset[,'C'])
R> X = cbind(1, log(dataset[,'Q']), log(dataset[,'PF']), dataset[,'LF'])
```

To estimate the model, we need to provide a new function for the modified prior distribution. The function `logPriorDensUSF.R`, available at the url
`http://people.uniroma2.it/antonio.parisi/logPriorDensUSF.R`
computes the whole prior density, considering a prior on δ (see eq. 3) truncated in zero. This function should be passed to the `mcSE` function through the `control` argument

```
R> source('logPriorDensUSF.R')
R> set.seed(159)
R> fit = mcSE(y=y, X=X, N=20000, Ti=6, modelType='SN', warmUp=TRUE,
+   control=list(logPriorFunc='logPriorDensUSF'))
R> summary(fit)

      Estimate  Std. Error        Q5%         Me       Q95%
B1     9.48294    0.231064   9.092868    9.48113    9.87037
B2     0.88272    0.013119   0.860776    0.88231    0.90440
B3     0.45394    0.020153   0.420452    0.45381    0.48759
B4    -1.62984    0.342429  -2.182360   -1.64108   -1.07360
G      0.01492    0.002509   0.011349    0.01469    0.01931
psi    0.04529    0.028764   0.004455    0.04189    0.09612

 log(p(y)) = 49.89004
```

Notice that, for these kind of models, efficiencies often represent an object of interest. For example, they can be used to compute technical efficiency measures.
It isn't possible to estimate the inefficiencies using the standard output of the `mcSE` function. It is however possible to use the `saveParticles` option in order to save all quantities needed to estimate any quantity of interest. This point is discussed in Appendix D.

6. Future developments

In future releases of the package, an effort will be dedicated to the improvement of the speed of the `mcSE` function. This can be accomplished by exploiting the structure of the sampler, which allows a high degree of parallelization. Moreover, the code of the most demanding tasks can be translated in a lower level language.
Another possible goal is to provide a general interface which could allow an easier customization of the estimation function. In fact, it is already possible to generalize the field of applications in many directions, for example allowing for missing data in the response variable or estimating models for time series / panel data. However, this currently requires the knowledge of the internal structure of the package.

[3]available at `http://pages.stern.nyu.edu/~wgreene/Text/econometricanalysis.htm`.

References

AZZALINI A. (1985), A class of distributions which includes the normal ones, *Scandinavian Journal of Statistics*, 12, 171–178.

AZZALINI A. (2014), *The skew-normal and related families*, 3, *Monographs*. Cambridge University Press, Cambridge, with the collaboration of Antonella Capitanio.

AZZALINI A. (2018), *The R package sn: the skew-normal and related distributions such as the skew-t (version 1.5-2).* Università di Padova, Italia.

AZZALINI A., ARELLANO-VALLE R. B. (2013), Maximum penalized likelihood estimation for skew-normal and skew-t distributions, *Journal of Statistical Planning and Inference*, 143(2) 419–433.

AZZALINI A., CAPITANIO A. (1999), Statistical applications of the multivariate skew-normal distributions, *Journal of the Royal Statistical Society*, B, 61, 579–602.

AZZALINI A., CAPITANIO A. (2003), Distributions generated by perturbation of symmetry with emphasis on a multivariate skew t distribution, B, *Journal of the Royal Statistical Society*, B, 65, 367–389.

AZZALINI A., DALLA VALLE A. (1996), The multivariate skew-normal distribution, *Biometrika*, 83, 715–726.

AZZALINI A., GENZ A. (2016), *The R package mnormt: the Multivariate Normal and t Distributions (Version 1.5-5).*

CAPITANIO A., AZZALINI A., STANGHELLINI E. (2003), Graphical models for skew-normal variates, *Scandinavian Journal of Statistics*, 30(1) 129–144.

CELEUX G., MARIN J.-M., ROBERT C. P. (2006), Iterated importance sampling in missing data problems, *Computational Statistics and Data Analysis*, 50(12) 3386–3404.

DAWID A. P. (1981), Some matrix-variate distribution theory: notational considerations and a Bayesian application, *Biometrika*, 68(1) 265–274.

EDDELBUETTEL D., ROMAIN F. (2017), RcppGSL: easier GSL use from R via Rcpp.

FERNÁNDEZ C., STEEL M. F. J. (1998), On Bayesian modeling of fat tails and skewness, *Journal of the American Statistical Association*, 93(441) 359–371.

FRÜHWIRTH-SCHNATTER S., PYNE S. (2010), Bayesian inference for finite mixtures of univariate and multivariate skew-normal and skew-t distributions, *Biostatistics*, 11(2) 317–336.

GALASSI M. (2018), GNU scientific library reference manual.

GENTON M. G. (2004), *Skew-elliptical distributions and their applications: a journey beyond normality*. CRC/Chapman & Hall, London.

GENZ A., BRETZ F. (2009), *Computation of multivariate normal and t probabilities*, Lecture Notes in Statistics. Springer-Verlag, Heidelberg.

GENZ A., BRETZ F., MIWA T., MI X., LEISCH F., SCHEIPL F., BORNKAMP B., MAECHLER M., HOTHORNET T. (2018), *mvtnorm: multivariate normal and t distributions*, R package version 1.0-8.

GREENE W. (2012), *Econometric analysis*, Pearson International Edition. Pearson Education, Limited.

GUPTA A., NAGAR D. (1999), *Matrix variate distributions*, Monographs and Surveys in Pure and Applied Mathematics. Chapman and Hall/CRC.

HORRACE W. C., PARMETER C. F. (2018), A Laplace stochastic frontier model, *Econometric Reviews*, 37(3) 260–280.

KUMBHAKAR S. C., LOVELL C. A. K. (2000), *Stochastic frontier analysis*. Cambridge University Press, Cambridge.

LEE S. X., MCLACHLAN G. J. (2013), EMMIXuskew: An R package for fitting mixtures of multivariate skew-t distributions via the EM Algorithm, *Journal of Statistical Software*, 55(12) 1–22.

LEE S. X., MCLACHLAN G. J. (2018), EMMIXcskew: an R package for the fitting of a mixture of canonical fundamental skew t distributions, *Journal of Statistical Software, Articles*, 83(3) 1–32.

LISEO B., PARISI A. (2013), Bayesian inference for the multivariate skew-normal model: a population Monte Carlo approach, *Computational Statistics and Data Analysis*, 63, 125–138.

MARCHENKO Y. V., GENTON M. G. (2010), A suite of commands for fitting the skew-normal and skew-t models, *Stata Journal*, 10, 507–539.

MARTIN A. D., QUINN K. M., PARK J. H. (2011), MCMCpack: Markov Chain Monte Carlo in R, *Journal of Statistical Software*, 42(9) 22.

PARISI A., LISEO B. (2017), Objective Bayesian analysis for the multivariate skew-t model, *Statistical Methods & Applications*, 27, 277–295.

ROBERT C., CASELLA G. (2010), *Introducing Monte Carlo methods with R*. Springer - New York.

ROY V., HOBERT J. P. (2010), On Monte Carlo methods for Bayesian multivariate regression models with heavy-tailed errors, *Journal of Multivariate Analysis*, 101(5) 1190–1202.

RUBIO F. J., GENTON M. G. (2016), Bayesian linear regression with skew-symmetric error distributions with applications to survival analysis, *Statistics in Medicine*, 35(14) 2441–2454.

R CORE TEAM (2015), *R: a language and environment for statistical computing*, R Foundation for Statistical Computing, Vienna, Austria.

VILLA C., RUBIO F. (2018), Objective priors for the number of degrees of freedom of a multivariate t distribution and the t-copula, *Computational Statistics and Data Analysis*, 124, 197–219.

VILLA C., WALKER S. (2014), Objective prior for the number of degrees of freedom of a student t distribution, *Bayesian Analysis*, 9, 197–220.

A. Computing the Effective Sample Size

The Effective Sample Size (see §4.4 of Robert and Casella, 2010) represents a useful quantity which is often required to assess the performance of importance samplers. At a generic t-th iteration, if $\zeta_j^{(t)}$, $j = 1, 2, \ldots, N$, denote the normalized importance weights, the ESS is given by

$$ESS_t = \frac{1}{\sum_{j=1}^{N} \left(\zeta_j^{(t)}\right)^2}$$

We calculate this quantity in the example given in §3.1 in order to illustrate the possibility to compute any quantity of interest using the saved output, when the `saveParticles` option is `TRUE`. For example, modifying the command given above as

```
R> set.seed(159)
R> y = cbind(wines[indices, 'tartaric'], wines[indices, 'malic'])
R> fit = mcSE(y=y, N=20000, Ti=6, modelType='SN', warmUp=TRUE,
+    control=list(saveParticles=TRUE))
```

an `Output` folder will be created in the working directory. The name and the path of this directory can be set through the `outFolder` element of the `control` list. In this folder, for each iteration, three `rds` files are created, containing the proposed particles, the normalized (log-)importance weights and the indices of the particles which have been re-sampled at the end of the iteration.

In order to compute the ESS, we can rewrite it as

$$ESS_t = \frac{1}{\sum_{j=1}^{N} \exp\left\{2 \log\left(\zeta_j^{(t)}\right)\right\}}$$

where the normalized log-weights can be found in the `log.zeta1.rds` through `log.zeta6.rds` files (one file per iteration). The following code computes the ESS value for each iteration, using the files saved by the previous `mcSE` function, and produces a related plot.

```
R> ESS = numeric(6)
R> for(ti in 1:6){
R>  fileName = paste('Output/log.zeta', ti, '.rds', sep='')
R>  log.zeta.t = readRDS(fileName)
R>  ESS[ti] = 1 / sum(exp(2*log.zeta.t))
R> }
R> plot(ESS, ylim=c(0,max(ESS)), type='b', pch=20)
```

B. How many iterations / particles are necessary?

In general, Sampling Importance Resampling algorithms are run for a low number of iterations having a fairly high number of particles. It is important to stress that the proposed sampler, which is based on a Monte Carlo method, doesn't rely on convergence argument. However, when the posterior density is not unimodal, the sampler could stay stuck in a local mode for several iterations, until the other modes are explored. In this case, besides the waste of computational resources, results from the first iterations induce a bias in the final estimates.

As an example, it is well-known that the exploration of the posterior density of a *skew-normal* model poses several problems to samplers. To illustrate the point, it is possible to repeat the commands in §3.1, without the warm-up step.

```
R> set.seed(159)
R> fit = mcSE(y=y, N=20000, Ti=6, modelType='SN')
```

Figure 3. Effective Sample Size for each iteration.

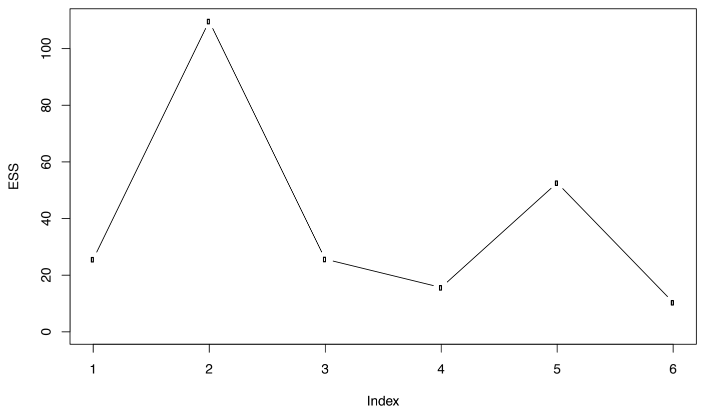


Figure 4. Contour plot of the density function of the skew-normal model estimated without the warm-up option.

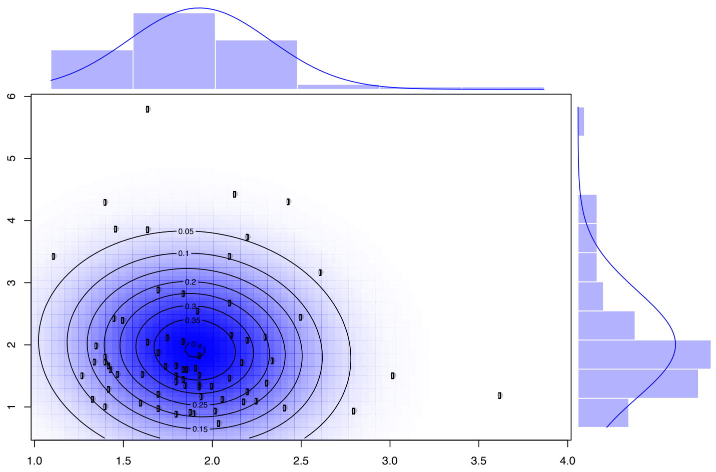
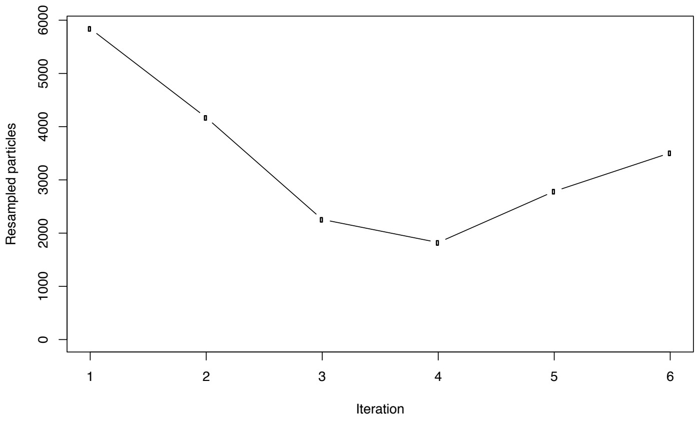


Figure 5. Number of (distinct) particles resampled for each iteration.



The contour plot in Figure 4 highlights the differences in the obtained inferences. The depicted density looks quite symmetric, even if a certain amount of skewness is evident from the data. The point estimate for the shape parameter is, in absolute value, much lower than the previous one:

```
R> SNstats = summary(fit)
R> coef(SNstats)$psi

[1] 0.0357801 0.5485207
```

The number of resampled particles, in Figure 5, is very high in the first two iterations (more than 25% of the particles has been resampled). It decreases during the third and fourth iteration, then it stabilizes at the same level as in the previous example in the last iterations.
A deeper look at the `fit` object reveals the problem. It is possible to extract the estimates of the shape parameter obtained in each iteration

```
R> fit$estlist$postMean$psi

              [,1]      [,2]
[1,]  0.050496910 0.1299646
[2,]  0.061213422 0.2631725
[3,]  0.274393719 0.4264509
[4,]  0.001787154 1.2526072
[5,] -0.002345185 1.3711511
[6,] -0.020744236 1.3942027
```

In the first two iterations, particles concentrate in a local mode of the posterior density, near the origin. In the third iteration, the sampler discovers the true mode of the posterior surface. This causes a dramatic drop in the number of resampled particles. Finally, in the last iterations, most of the particles are concentrated in a region which is closer to the estimates obtained in §3.1. The final estimate is however influenced by the results of the first two iterations, and is rather far from the “correct” value.

The `warmUp` argument of the `mcSE` function has been implemented to ease the exploration of the posterior distribution by providing a better initialization for the particles, at a low (or negative) cost: in the illustrated case, the algorithm is slightly faster when using the warm-up.
At the beginning of the algorithm, the particles are initialized using the stochastic representation of the model in eq. (4). Most of these particles will be generated in regions of low posterior density. Using the `warmUp` step, some iterations with a small number of particles rapidly explore the posterior density to find high density regions. These iterations will not be considered for any inferential purpose.

C. Marginal distributions

Let

$$\boldsymbol{Y} \sim ST_p(\boldsymbol{\xi}, \boldsymbol{\alpha}, \Sigma, \nu)$$

and let us denote by $\boldsymbol{Y}_i$, $i = 1, 2$ a partition of the components of the r.v. $\boldsymbol{Y}$, having respectively p_1 and p_2 components. It is known (see Azzalini and Capitanio, 2003 and Capitanio, Azzalini, and Stanghellini, 2003) that

$$\boldsymbol{Y}_1 \sim ST_{p_1}(\boldsymbol{\xi}_1, \boldsymbol{\alpha}_{1(2)}, \Sigma_{11}, \nu)$$

with

$$\begin{aligned}\boldsymbol{\alpha}_{1(2)} &= \frac{\boldsymbol{\alpha}_1 + (\Omega_{11})^{-1}\Omega_{12}\boldsymbol{\alpha}_2}{(1+\boldsymbol{\alpha}_2^\top \Omega_{22\cdot 1}\boldsymbol{\alpha}_2)^{1/2}} \\ \Omega_{22\cdot 1} &= \Omega_{22} - \Omega_{21}(\Omega_{11})^{-1}\Omega_{12}.\end{aligned} \tag{7}$$

Proposition. In the alternative parametrization, if

$$\boldsymbol{Y} \sim ST_p(\boldsymbol{\xi}, \boldsymbol{\delta}, \Sigma, \nu)$$

then the shape parameter $\boldsymbol{\delta}_1$ of the marginal distribution is given by the values of $\boldsymbol{\delta}$ of the components of $\boldsymbol{Y}_1$

$$\boldsymbol{Y}_1 \sim ST_{p_1}(\boldsymbol{\xi}_1, \boldsymbol{\delta}_1, \Sigma_{11}, \nu)$$

The same result is also true using the $\{\boldsymbol{\xi}, \boldsymbol{\psi}, G, \nu\}$ parametrization.

Proof: to prove this property, we denote by $\boldsymbol{\delta}_{1(2)}$ the shape parameter of $\boldsymbol{Y}_1$, and we will demonstrate that $\boldsymbol{\delta}_{1(2)} = \boldsymbol{\delta}_1$. To this aim, it is possible to express $\boldsymbol{\delta}$, as a function of $\boldsymbol{\alpha}$, as

$$\boldsymbol{\delta}(\boldsymbol{\alpha}) = (1 + \boldsymbol{\alpha}^\top\Omega\boldsymbol{\alpha})^{-1/2}\Omega\boldsymbol{\alpha}.$$

Hence, it is necessary to verify that

$$\boldsymbol{\delta}(\boldsymbol{\alpha}_{1(2)}) = \boldsymbol{\delta}_1(\boldsymbol{\alpha}). \tag{8}$$

For the RHS of eq. (8), define the $(p_1 \times p)$ matrix H

$$H = (\mathbb{I}_{p_1} : \mathbf{0}_{p_2}).$$

We have

$$\boldsymbol{\delta}_1(\boldsymbol{\alpha}) = H\boldsymbol{\delta}(\boldsymbol{\alpha}) = (1 + \boldsymbol{\alpha}^\top\Omega\boldsymbol{\alpha})^{-1/2}H\Omega\boldsymbol{\alpha}$$

For the LHS of (8), using eq. (7), in the alternative parametrization we have that

$$\begin{aligned}\boldsymbol{\delta}(\boldsymbol{\alpha}_{1(2)}) &= (1 + \boldsymbol{\alpha}_{1(2)}^\top\Omega_{11}\boldsymbol{\alpha}_{1(2)})^{-1/2}\Omega_{11}\boldsymbol{\alpha}_{1(2)} \\ &= (1+\boldsymbol{\alpha}_2^\top\Omega_{22\cdot 1}\alpha_2)^{1/2}(1+\boldsymbol{\alpha}^\top\Omega\boldsymbol{\alpha})^{-1/2}(1+\boldsymbol{\alpha}_2^\top\Omega_{22\cdot 1}\alpha_2)^{-1/2}H\Omega\boldsymbol{\alpha} = \\ &= (1+\boldsymbol{\alpha}^\top\Omega\boldsymbol{\alpha})^{-1/2}H\Omega\boldsymbol{\alpha}.\end{aligned}$$

Hence, eq. (8) holds.

D. Inefficiencies

Following the example in §5, here we will estimate the vector of the inefficiencies of the single airlines. From eq. (6), the inefficiencies are defined as $\boldsymbol{\psi}\frac{|z_i|}{\sqrt{v_i}}$ in the *skew-t* case, and $\boldsymbol{\psi}|z_i|$ in the *skew-normal* case. In the symmetric cases, the model doesn't admit inefficiencies.

At a generic t-th iteration, the inefficiency $u_i^{(t)}$ for the i-eth unit can be estimated as a mean of the inefficiencies given by the N particles, weighted by the normalized importance weights

$$\hat{u}_i^{(t)} = \sum_{j=1}^{N} \psi_j^{(t)} |z_{ij}^{(t)}| \zeta_j^{(t)}.$$

Using the normalized perplexity to weight these partial estimates, one obtains

$$\hat{u}_i = \sum_{ti=1}^{T} \hat{u}_i^{(t)} P^{(t)}.$$

Also in this case, we will use the `saveParticles` option to export the needed objects

```
R> set.seed(159)
R> y = log(dataset[,'C'])
R> X = cbind(1, log(dataset[,'Q']), log(dataset[,'PF']), dataset[,'LF'])
R> N = 20000
R> Ti = 6
R> fit = mcSE(y=y, X=X, N=N, Ti=Ti, modelType='SN', warmUp=TRUE,
+   control=list(logPriorFunc='logPriorDensUSF', saveParticles=TRUE))
```

The (unnormalized) perplexity can be found in the `fit` object

```
R> Pbar = fit$perplexity
R> P.t = Pbar / sum(Pbar)
```

The proposed particles, including the values of $z_{ij}^{(t)}$ and $\psi_j^{(t)}$ can be found in the `ProposedX.rds` files, where `X` denotes the iteration. For example, it is possible to extract the proposed values of $z_{ij}^{(1)}$ and $\psi_j^{(1)}$ using the commands

```
R> proposed = readRDS('Output/Proposed1.rds')
R> z.t = proposed$z
R> psi.t = proposed$psi
```

Finally, to extract the normalized importance weights $\zeta_j^{(1)}$ for the first iteration

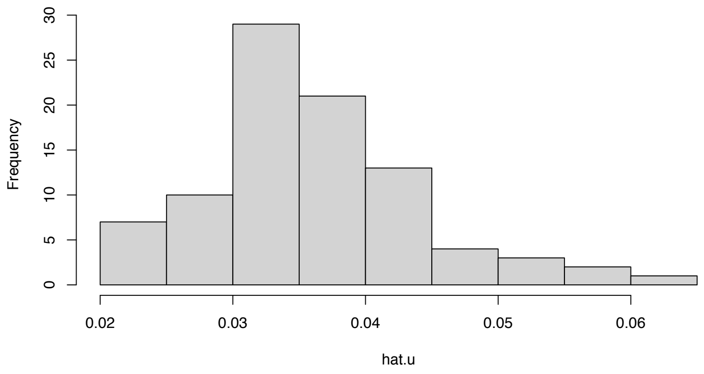
```
R> log.zeta.t = readRDS('Output/log.zeta1.rds')
```

Hence, we can estimate the whole vector $\hat{\boldsymbol{u}}$ of inefficiencies using the following code.

```
R> n = length(y)
R> hat.u.t = matrix(0, Ti, n)
R> for(ti in 1:Ti){
R>  proposed = readRDS(paste('Output/Proposed', ti,'.rds',sep=''))
R>  z.t = proposed$z
R>  psi.t = proposed$psi
R>  log.zeta.t = readRDS(paste('Output/log.zeta', ti,'.rds', sep=''))
R>  psi.mat = matrix(psi.t, N, 1) %*% rep(1, n)
R>  zeta.mat = matrix(exp(log.zeta.t), N, 1) %*% rep(1, n)
R>  hat.u.t[ti,] = apply(psi.mat * abs(z.t) * zeta.mat, 2, sum)
R> }
R> P.mat = matrix(P.t, Ti, 1) %*% rep(1, n)
R> hat.u = apply(hat.u.t * P.mat, 2, sum)
R> hist(hat.u, col='lightgray')
```

The histogram is given in Figure 6.

Figure 6. Histogram of the inefficiencies for the stochastic frontier model.


Frequency
hat.u