*Journal of Statistical Software*

*February 2014, Volume 56, Issue 12.* *http://www.jstatsoft.org/*

# Introducing localgauss, an R Package for Estimating and Visualizing Local Gaussian Correlation

**Geir Drage Berentsen**
University of Bergen

**Tore Selland Kleppe**
University of Bergen

**Dag Tjøstheim**
University of Bergen

**Abstract**

Quantifying non-linear dependence structures between two random variables is a challenging task. There exist several bona-fide dependence measures able to capture the strength of the non-linear association, but they typically give little information about *how* the variables are associated. This problem has been recognized by several authors and has given rise to the concept of local measures of dependence. A local measure of dependence is able to capture the "local" dependence structure in a particular region. The idea is that the global dependence structure is better described by a portfolio of local measures of dependence computed in different regions than a one-number measure of dependence. This paper introduces the R package **localgauss** which estimates and visualizes a measure of local dependence called local Gaussian correlation. The package provides a function for estimation, a function for local independence testing and corresponding functions for visualization purposes, which are all demonstrated with examples.

*Keywords*: local Gaussian correlation, measure of dependence, R.

## 1. Introduction

The Pearson correlation coefficient is by far the most widely used measure of dependence between two stochastic variables. For variables with a linear relationship it measures both the strength and the direction of the (linear) dependence, and for jointly Gaussian variables it completely characterizes the dependence structure. However, it is well known that it is not adequate in many cases, and one can easily construct examples where there is complete dependence, but where the correlation coefficient is 0 (e.g., take $Y = X^2$ where $X$ is any symmetric variable with mean 0 and existing third moment). Other valid measures of dependence exist (see e.g., Rosenblatt and Wahlen 1992; Skaug and Tjøstheim 1993; Tjøstheim 1996; Székely, Rizzo, and Bakirov 2007; Székely and Rizzo 2009), but unlike the Pearson correlation, these measures typically do not distinguish between positive and negative dependencies. Indeed,

in general a signed one-number measure cannot properly capture non-linear dependence (see e.g., Embrechts, McNeil, and Straumann 2002). The issue of defining positive and negative dependence for variables in a non-linear relationship is a rather delicate one (see e.g., Lehmann 1966). For such variables there can be regions within their support where there is stronger dependence, positive or negative, and other regions where it is weaker. For example, it has been observed that the dependence between financial variables becomes stronger during an economic downturn, leading to a stronger association between large losses. It may be more suitable to describe such situations by a local measure of dependence that can characterize the dependence structure within different regions of the data, rather than using a one-number measure.

This paper accompanies the R package **localgauss**, publicly available for Linux, Mac and Windows at the Comprehensive R Archive Network (CRAN, http://CRAN.R-project.org/package=localgauss), which estimates and visualizes a novel local measure of dependence called local Gaussian correlation, introduced by Tjøstheim and Hufthammer (2013). See also Støve, Tjøstheim, and Hufthammer (2012), Berentsen, Støve, Tjøstheim, and ø (2013), Støve and Tjøstheim (2014) and Berentsen and Tjøstheim (2014). R is a free software environment for statistical computing and graphics (R Core Team 2013), and the R language is widely used among statisticians.

The rest of the article is organized as follows. A brief review of a competing local measures of dependence is presented below. An introduction to local Gaussian correlation and the package **localgauss** is then described in Section 2. Finally, in Section 3, we demonstrate the usage of the package on several examples.

### 1.1. Local measures of dependence

For two random variables $X_1$ and $X_2$, a local dependence measure is allowed to vary within the support of $(X_1, X_2)$. Besides this property, local measures of dependence proposed in the statistical literature have quite different properties and motivations (see e.g., Berentsen and Tjøstheim 2014, and references therein). The local measure with most similarities to our methodology is perhaps the "local dependence function" proposed by Holland and Wang (1987). This measure is derived by generalizing the concept of cross-product ratios in discrete two-way contingency tables to the case of a continuous bivariate density and is given by

$$\gamma(x_1, x_2) = \frac{\partial^2}{\partial x_1 \partial x_2} \log f(x_1, x_2),$$

where $f$ is the bivariate density of $(X_1, X_2)$. Further motivation for $\gamma$ is given in Jones (1996) along with an empirical counterpart based on kernel estimates of the density and its derivatives. Properties of $\gamma$ are given in Jones (1996) and a method for visualizing the dependence structure in bivariate data based on $\gamma$ can be found in Jones and Koch (2003). Elements of this methodology are adopted in our software to visually present the local dependence measure described in Section 2, and in many situations the two approaches give broadly similar results (see Berentsen and Tjøstheim 2014). Software for the competing methodology can be obtained from the second author of Jones and Koch (2003) upon request.

## 2. Local Gaussian correlation and package localgauss

The idea of local Gaussian correlation is very simple. Let $X = (X_1, X_2)$ be a random variable with a general bivariate density $f$. Then in a neighborhood of each point $x = (x_1, x_2)$ we fit a bivariate Gaussian density

$$\psi(v, \theta(x)) = \frac{1}{2\pi\sigma_1(x)\sigma_2(x)\sqrt{1-\rho^2(x)}} \exp\left[-\frac{1}{2}\frac{1}{1-\rho^2(x)}\left(\frac{(v_1-\mu_1(x))^2}{\sigma_1^2(x)} - 2\rho(x)\frac{(v_1-\mu_1(x))(v_2-\mu_2(x))}{\sigma_1(x)\sigma_2(x)} + \frac{(v_2-\mu_2(x))^2}{\sigma_2^2(x)}\right)\right]. \quad (1)$$

where $v = (v_1, v_2)^\top$ is the running variable in the Gaussian distribution and

$$\theta(x) = (\mu_1(x), \mu_2(x), \sigma_1(x), \sigma_2(x), \rho(x)),$$

with $\mu_i(x)$, $i = 1, 2$ the local means, $\sigma_i(x)$, $i = 1, 2$ the local standard deviations and $\rho(x)$ the local correlation at the point $x$. The population values of the local parameters $\theta_b(x) = \theta(x)$ are obtained by minimizing the local penalty function

$$q = \int K_b(v - x)\left[\psi\left(v, \theta(x)\right) - \log \psi(v, \theta(x)) f(v)\right] \mathrm{d}v, \quad (2)$$

where $K_b(v-x) = (b_1 b_2)^{-1} K(b_1^{-1}(v_1 - x_1)) K(b_2^{-1}(v_2 - x_2))$ is a product kernel with bandwidth $b = (b_1, b_2)$, and we define the "local Gaussian correlation" $\rho_b(x) = \rho(x)$ as the last element of the vector $\theta(x)$ that minimizes $q$. Moving to another point $x'$, $q$ can be used to obtain a new Gaussian approximation density $\psi(v, \theta(x'))$, which approximates $f$ in a neighborhood of $x'$. In this way $f$ may be represented by a family of Gaussian bivariate densities as $x$ varies, and in each specific neighborhood of $x$, the local dependence properties are described by $\rho(x)$. We may define the (local) dependence to be positive if $\rho(x) > 0$ and negative if $\rho(x) < 0$.

The penalty function $q$ was used in Hjort and Jones (1996) for density estimation purposes, and later by Tjøstheim and Hufthammer (2013) in the development of local Gaussian correlation. In general, $q$ can be used to fit any parametric distribution to $f$ locally, and in Hjort and Jones (1996) it is argued that $q$ can be interpreted as a locally weighted Kullback-Leibler criterion for measuring the distance between $f(\cdot)$ and the chosen parametric distribution (in our case $\psi(\cdot, \theta(x))$).

Given observations $X_i = (X_{i1}, X_{i2})$, $i = 1, \ldots, n$, from $f$ the corresponding estimates $\hat{\theta}(x)$ are obtained by maximizing the local log-likelihood function (see Hjort and Jones 1996)

$$L\big(X_1, \ldots, X_n, \theta(x)\big) = n^{-1} \sum_i K_b(X_i - x) \log \psi(X_i, \theta(x)) - \int K_b(v - x)\psi(v, \theta(x))\, \mathrm{d}v. \quad (3)$$

To see that the local likelihood function (3) is consistent with the penalty function $q$, first observe that when $\theta(x)$ is chosen to minimize $q$, it satisfies the set of equations

$$\frac{\partial q}{\partial \theta_j} = \int K_b(v - x) \frac{\partial}{\partial \theta_j}\{\log(\psi(v, \theta(x)))\}[\psi(v, \theta(x)) - f(v)]\, \mathrm{d}v = 0, \quad j = 1, \ldots, 5. \quad (4)$$

Using the notation

$$u_j(\cdot, \theta) = \frac{\partial}{\partial \theta_j}\{\log \psi(\cdot, \theta)\}, \quad (5)$$

and assuming $\mathsf{E}\{K_b(X_i - x)u_j(X_i, \theta(x))\} < \infty$, the law of large numbers gives almost surely

$$\begin{aligned}\frac{\partial L}{\partial \theta_j} &= n^{-1}\sum_i K_b(X_i - x)u_j(X_i, \theta(x)) - \int K_b(v - x)u_j(v, \theta(x))\psi(v, \theta(x))\,\mathrm{d}v \\ &\to \int K_b(v - x)u_j(v, \theta(x))[f(v) - \psi(v, \theta(x))]\,\mathrm{d}v = -\frac{\partial q}{\partial \theta_j}, \quad j = 1, \ldots, 5,\end{aligned} \tag{6}$$

as $n \to \infty$ and we see that (6) can be identified with (4). Note that as $b \to \infty$ (3) reduces to the ordinary log-likelihood for a Gaussian distribution $\psi$ plus a constant, and hence $\rho(x)$ reduces to the ordinary global Gaussian correlation. For more details about the local parameters and estimation using local likelihood, including limiting behavior as $b_i \to 0$, $i = 1, 2$, and estimation of standard errors, we refer to Tjøstheim and Hufthammer (2013).

### 2.1. Function `localgauss`

The equations $\frac{\partial L}{\partial \theta_j} = 0$, $j = 1, \ldots, 5$, with $\frac{\partial L}{\partial \theta_j}$ given by (6) do not in general have closed form solutions, hence (3) must be maximized directly. In the R package **localgauss** the function `localgauss()` maximizes (3) for different values of $x$ using a modified Newton's method with line-search and returns an S3 object of class '`localgauss`'. The function (3) is not everywhere concave, and eigenvalue modification is therefore employed to ensure that the scaling matrix is positive definite (Nocedal and Wright 1999, Chapter 6). The optimizer and objective function are written in Fortran 90 (Metcalf and Reid 1999), and the source code for the gradient and Hessian of (3) are generated using the automatic differentiation tool **TAPENADE** (Hascoët and Pascual 2004).

The number $M$ of points $x = (x_1, x_2)$ for which we want to estimate the local Gaussian correlation can be manually specified by the argument `xy.mat` which is an $M$ times 2 matrix. Alternatively, the selection of these points can be done using the methodology of Jones and Koch (2003). First a regular $N \times N$ grid is placed across the area of interest. The regular grid is then screened by selecting the grid points $x_1, \ldots, x_M$ satisfying $\hat{f}(x_j) \geq C$, for some constant $C$ and a density estimator $\hat{f}$. For simple and quick implementation this is done using the R package **MASS** (Venables and Ripley 2002). In `localgauss()` the values of $N$ and $C$ are set by the arguments `gsize` and `hthresh`, respectively. If `xy.mat` is not specified it will be selected internally by the method described above. Since (3) has to be optimized for every estimation point, the computational time increases proportionally with the size of `xy.mat`. Calling `localgauss()` with a 500 times 2 `xy.mat`, even for a rather large sample size ($n \approx 1000$), should not take longer than a few seconds with a personal computer of today's standard. An overview of the other arguments and output of `localgauss()` is given in Table 1. Note that we have used the notation `x` and `y` for the marginal data vectors rather than $x_1$ and $x_2$.

Choosing the bandwidth $b = (b_1, b_2)$ in `localgauss()` is left to the user. In this way, the local Gaussian correlation can be estimated with different bandwidths, reflecting the dependence structure on different scales of locality increasing to the global correlation as $b \to \infty$. The population value $\rho_b(x) = \rho(x)$ is defined as the minimizer of (2) with the bandwidth $b$ fixed. If one wants a more objective choice for estimating the limiting value $\rho(x)$ as $b \to 0$, note that Tjøstheim and Hufthammer (2013) propose a bandwidth algorithm that aims to balance the variance of the estimated local parameters $\hat{\theta}(x)$ versus the bias of the resulting density estimate $\psi(\cdot, \hat{\theta}(x))$. An alternative method is described in Berentsen *et al.* (2013) in the context of

| Arguments | Description | Default |
|---|---|---|
| `x`, `y` | The two data vectors. | |
| `b1`, `b2` | The bandwidth in the $x$-direction and $y$-direction, respectively. | |
| `gsize` | The gridsize (only used if `xy.mat` is not specified). | `15` |
| `hthresh` | Grid points where a non-parametric density estimate is lower than `hthresh` are omitted (only used if `xy.mat` is not specified). | `0.001` |
| `xy.mat` | A $M$ times 2 matrix of points where the local parameters are to be estimated. | `NULL` |
| **Results** | **Description** | |
| `par.est` | $M$ times 5 matrix of parameter estimates, with columns $\mu_1$, $\mu_2$, $\sigma_1$, $\sigma_2$ and $\rho$. | |
| `eflag` | $M$-vector of exit flags from the optimizer. Estimations with exit flags other than 0 should not be trusted. | |
| `hessian` | The negative Hessian of the objective function (3). | |

Table 1: Summary of arguments and results of `localgauss()`.

copula goodness-of-fit testing. A method of general applicability is likelihood cross-validation as explained in Berentsen and Tjøstheim (2014). It has worked well in cases tested by us, but may require screening of outlying observations (see Berentsen and Tjøstheim 2014).

### 2.2. Function `localgauss.indtest`

In an independence testing situation it may be of interest to pinpoint possible discrepancies between the data and the null-hypothesis of independence. Such discrepancies can be investigated more closely by doing several "local tests of independence" as described in Berentsen and Tjøstheim (2014). This idea is similar but not identical to the idea of local permutation testing described in Jones and Koch (2003).

The local test of independence is based on estimating the null-distribution of the estimated local Gaussian correlation by re-sampling from the product of the empirical marginal distributions. This is a standard way of estimating the null-distribution for any statistic suitable for independence testing. The difference in the present case is that we may have a portfolio of test statistics $\hat{\rho}(x_i)$ for several different points $x_i$, and that we need to produce the null distribution for each of them. In the R package **localgauss** this can be done by passing a '`localgauss`' object, produced by the function `localgauss()`, to the function `localgauss.indtest()`. The null-distribution is estimated for each point specified in `xy.mat` in the '`localgauss`' object. The function `localgauss.indtest()` then returns an S3 object of class '`localgauss.indtest`', and the result of the test can be found in the vector `test.results`: An estimated local correlation for the original data significantly larger than the null-distribution is indicated with +1; an estimated local correlation for the original data insignificant with respect to the null-distribution is indicated with 0; an estimated local correlation for the original data significantly smaller than the null-distribution is indicated with −1. Recall that when $b \to \infty$, the local log-likelihood (3) reduces to the ordinary log-likelihood for Gaussian variables. Note therefore that all the results in `test.results` will coincide with that of a bootstrap test of the global Gaussian correlation when $b \to \infty$.

In general, the choice of bootstrap replicas `R` is a compromise between reducing the effect of

| Arguments | Description | Default |
|---|---|---|
| `locobj` | S3 object of class '`localgauss`' produced by the `localgauss` function. | |
| `R` | Number of bootstrap replica. | `10` |
| `alpha` | Significance level (note: two sided test). | `0.10` |
| `seed` | Random seed used for bootstrap. | `1` |
| **Results** | **Description** | |
| `localgauss` | Simply returns argument `locobj`. | |
| `upper` | Vector containing the 1−`alpha`/2 quantiles of the null-distributions. | |
| `lower` | Vector containing the `alpha`/2 quantiles of the null-distributions. | |
| `test.results` | Vector containing the test results. | |

Table 2: Summary of arguments and results of `localgauss.indtest()`.

random sampling error and available computing power and time. Our experience suggests that using `R` = 500 bootstrap replicas is adequate, with larger values having little effect on the results. This means that the computational time can be significant even in situations where the original '`localgauss`' object only takes a few seconds to produce. The re-sampling loop is constructed using the R package **foreach** (Kane, Emerson, and Weston 2013; Revolution Analytics and Weston 2013b) which supports parallel execution for users with multiple processors/cores on their computer or access to multiple nodes of a cluster. The function `localgauss.ind()` will automatically run in parallel if a "parallel backend" compatible with the **foreach** package is registered. Registering a parallel backend in R can be done via e.g., the R package **doParallel** (Revolution Analytics and Weston 2013a) which supports both Unix-like systems and Windows. When no parallel backend is registered `localgauss.ind()` will issue a warning that it is running sequentially. Note that the computational time depends to a large degree on the size of the `xy.mat` in the '`localgauss`' object. An overview of the other arguments and output of `localgauss.indtest()` is given in Table 2.

## 2.3. Graphics

Graphics are important for proper interpretation and presentation of the results given by the functions `localgauss()` and `localgauss.indtest()` described above. The following plotting routines are based on the R package **ggplot2** (Wickham 2009).

The S3 method for graphically displaying a '`localgauss`' object is invoked by applying `plot()` to such an object. The function displays the estimated local Gaussian correlation $\hat{\rho}(x_i)$ in tiles for each point in `xy.mat`. The numerical value of $\hat{\rho}(x_i)$ is indicated by a color gradient between `highcol` (which represents large values) and `lowcol` (which represents low values). This color gradient may be taken to be continuous (`divergent.col.grad = FALSE`) or divergent with 0 as midpoint (`divergent.col.grad = TRUE`). We recommend the former choice when the range of the values of $\hat{\rho}(x_i)$ is small. For the latter choice, values close to 0 are indicated with the color white and we recommend this choice in the presence of both positive and negative values of $\hat{\rho}(x_i)$. If `plot.text = TRUE` the numerical value is added to each tile. An overview of the other arguments of the S3 `plot` method for '`localgauss`' objects are given in Table 3.

| Arguments | Description | Default |
|---|---|---|
| `x` | S3 object of class '`localgauss`' produced by `localgauss()`. | |
| `plot.text` | If `TRUE`, the numerical values of the estimated local correlation are added to each tile. | `TRUE` |
| `plot.points` | If `TRUE`, the original observations are overlain. | `FALSE` |
| `tsize` | The font size used if `plot.text` is `TRUE`. | 3 |
| `lowcol` | The color used to indicate small values of $\hat{\rho}(x_i)$. | `"cyan"` |
| `highcol` | The color used to indicate large values of $\hat{\rho}(x_i)$. | `"magenta"` |
| `point.col` | The color used for observation points if `plot.points` is `TRUE`. | `"black"` |
| `point.size` | The size of observation points if `plot.points` is `TRUE`. | 1 |
| `xlab`, `ylab` | The label of the $x$-axis and $y$-axis, respectively. | |
| `divergent.col.grad` | If `TRUE`, a divergent color gradient between `lowcol` and `highcol` with 0 as midpoint is used. If `FALSE` an ordinary color gradient between `lowcol` and `highcol` is used. | `TRUE` |

Table 3: Summary of arguments of S3 `plot` method for '`localgauss`' objects.

| Arguments | Description | Default |
|---|---|---|
| `x` | S3 object of class '`localgauss.indtest`' produced by the `localgauss.indtest` function. | |
| `plot.points` | If `TRUE`, the original observations are overlain. | `FALSE` |
| `poscol` | Color indicating $+1$ test result. | `"magenta"` |
| `negcol` | Color indicating $-1$ test result. | `"cyan"` |
| `point.col` | The color used for observations points if `plot.points` is `TRUE`. | `"black"` |
| `point.size` | The size of observations points if `plot.points` is `TRUE`. | 1 |
| `xlab`, `ylab` | The label of the $x$-axis and $y$-axis, respectively. | |

Table 4: Summary of arguments of S3 `plot` method for '`localgauss.indtest`' objects.

Similarly, an S3 `plot` method for graphically displaying a '`localgauss.indtest`' object is available. The `test.results` from the '`localgauss.indtest`' object are displayed in tiles for each point in `xy.mat`. The values of `test.results` are indicated with `poscol` for $+1$, `negcol` for $-1$ and white for 0. The resulting plot is an analogue to the "dependence map" proposed in Jones and Koch (2003) for the local dependence function. It can be seen as a compromise between the very fine details given by the estimated local Gaussian correlation and the crude characterization of dependence given by a single global measure. An overview of the other arguments of the S3 `plot` method for '`localgauss.indtest`' objects are given in Table 4.

## 3. Usage examples

In the following examples, all R code demonstrated assumes that the package has been loaded and we set the random seed to make all the examples reproducible:

```
R> library("localgauss")
R> set.seed(1)
```

### 3.1. Example 1: Bivariate Gaussian variables

We start by illustrating the basic usage of `localgauss()` in the case when $X_1$ and $X_2$ are standard Gaussian with (global) correlation $\rho = 0.6$. The following code estimates local parameters in the points $(-1, 1)$, $(0, 0)$ and $(1, 1)$:

```
R> x <- rnorm(1000)
R> y <- 0.6 * x + sqrt(1 - 0.6^2) * rnorm(1000)
R> xy.mat <- rbind(c(-1, 1), c(0, 0), c(1, 1))
R> lg.out <- localgauss(x = x, y = y, xy.mat = xy.mat, b1 = 1, b2 = 1)
```

As a precautionary measure one should check that the output `eflag` only contains a string of 0's indicating that the optimization succeeded:

```
R> lg.out$eflag
[1] 0 0 0
```

The estimated local parameters are then given by the output `par.est`:

```
R> lg.out$par.est
```

```
            mu_1          mu_2    sig_1     sig_2       rho
[1,]  0.02255388  0.0023722762 1.007630 0.9785351 0.5538728
[2,]  0.00913216  0.0006094236 1.015119 1.0021872 0.5681855
[3,] -0.02580466 -0.0319621089 1.023635 0.9948138 0.5830581
```

Note that the local Gaussians are not centered at the estimation points given by `xy.mat`. Indeed, in the case of Gaussian $f$ with parameters $\theta = (\mu_1, \mu_2, \sigma_1, \sigma_2, \rho)$, the solution to the penalty function (2) is given by $\theta(x) = \theta$, hence the local Gaussians will be located at the mean $\mu = (\mu_1, \mu_2)$. This also means that we have the appealing property that $\rho(x) = \rho$, and as seen above the corresponding estimate is quite close to $\rho = 0.6$.

The following code creates a '`localgauss`' object where `xy.mat` is generated internally using the arguments `gsize` and `hthresh`:

```
R> lg.out2 <- localgauss(x = x, y = y, b1 = 1, b2 = 1, gsize = 15,
+    hthresh = 0.01)
```

Typically, '`localgauss`' objects where the `xy.mat` is generated internally are well suited for plotting using their corresponding S3 `plot` method. To check that the value of `hthresh` is suitable we recommend overlaying the observations in the plot to visually check that estimation is done only in points with a neighborhood containing a "reasonable" amount of observations (otherwise the value of `hthresh` should be adjusted). For the previous '`localgauss`' object this is done by

```
R> plot(lg.out2, plot.text = FALSE, plot.points = TRUE)
```

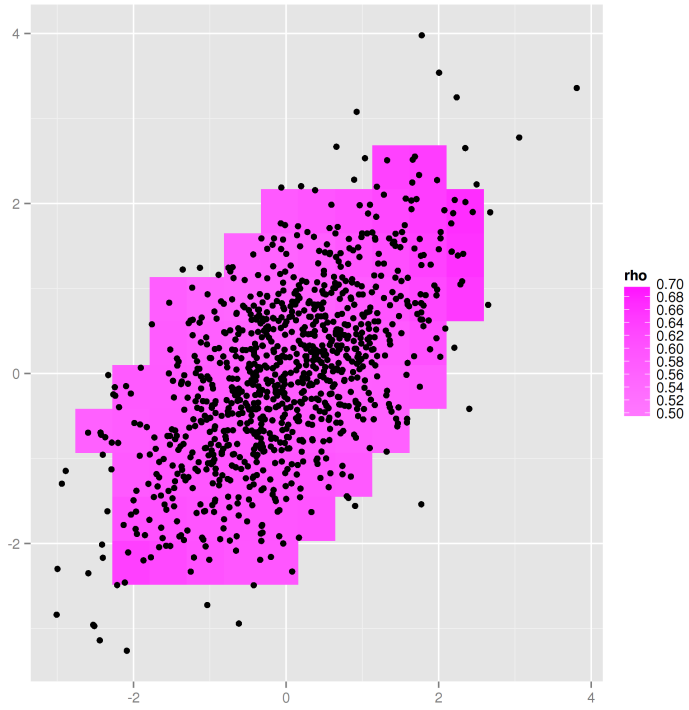


Figure 1: Estimated local Gaussian correlation with overlain observations and value indicated by color in the Gaussian case.

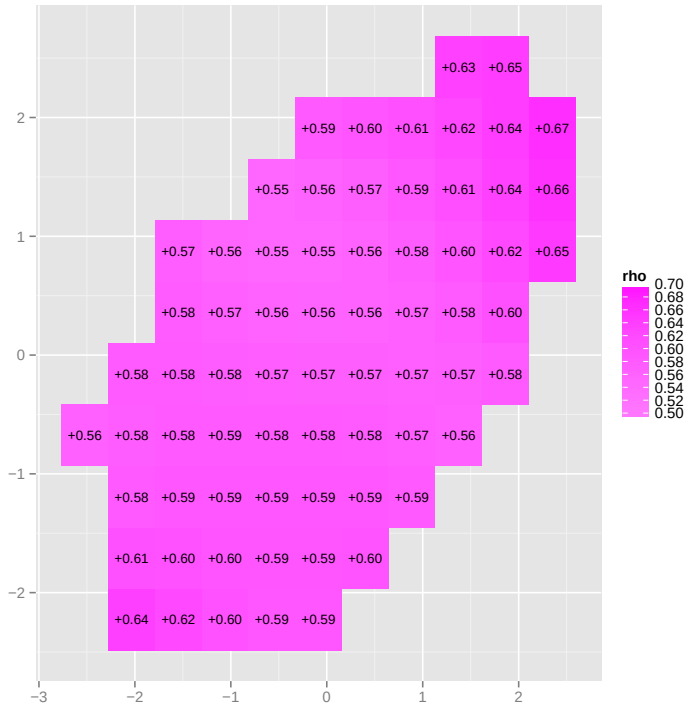


Figure 2: Numerical value of estimated local Gaussian correlation in the Gaussian case.

which produces the plot shown in Figure 1. Indeed, we see that the neighborhood of all points in the area of estimation (colored pink) contains observations. By default the S3 `plot` method will include the numerical value of the estimated local correlation. Thus we need only write

```
R> plot(lg.out2)
```

to obtain Figure 2. As expected, the local Gaussian correlation is constant equal to 0.6 everywhere modulo the sampling variation.

### 3.2. Example 2: Wikipedia model

Several simulated examples where the ordinary Pearson correlation fails to capture non-linear dependence structure can be found at Wikipedia (2012). One of these examples can be simulated in the following way

```
R> x <- runif(1000, -1, 1)
R> y <- (x^2 + runif(1000, 0, 1/2)) * sample(c(-1, 1), 1000, replace = TRUE)
```

A scatter-plot of these variables is displayed in Figure 3, where we clearly see a strong non-linear relationship between the two variables. However, the sample correlation is

```
R> cor(x, y)

[1] -0.02702666
```

which is due to the symmetry between positive and negative dependence in the data. The following code creates a '`localgauss`' object and plots it without including numerical values in each tile:

```
R> lg.out <- localgauss(x = x, y = y, b1 = 0.5, b2 = 0.5, gsize = 100,
+    hthresh = 0.15)
R> plot(lg.out, plot.text = FALSE)
```

This results in Figure 4. Note that the initial gridsize is `gsize` $\times$ `gsize` $= 100 \times 100$, which after the screening procedure resulted in a 4684 times 2 `xy.mat`. This means that (3) must be maximized for 4684 different points which on the first author's laptop (with 2.4 GHz Intel Core i5 CPU and 8 GB memory running Linux) took about one minute. Such an amount of estimation points is hardly necessary in practice, and the neighboring estimates in this example are extremely close in value. This also means that the corresponding plot produced by the S3 `plot` method will be very smooth.

### 3.3. Example 3: Uranium exploration data set

The Uranium exploration data set of Cook and Johnson (1986) consists of 655 chemical analyzes from water samples collected from the Montrose quad-rangle of Western Colorado. These data can be obtained via the R package **copula** (Yan 2007):

```
R> data("uranium", package = "copula")
R> attach(uranium)
```

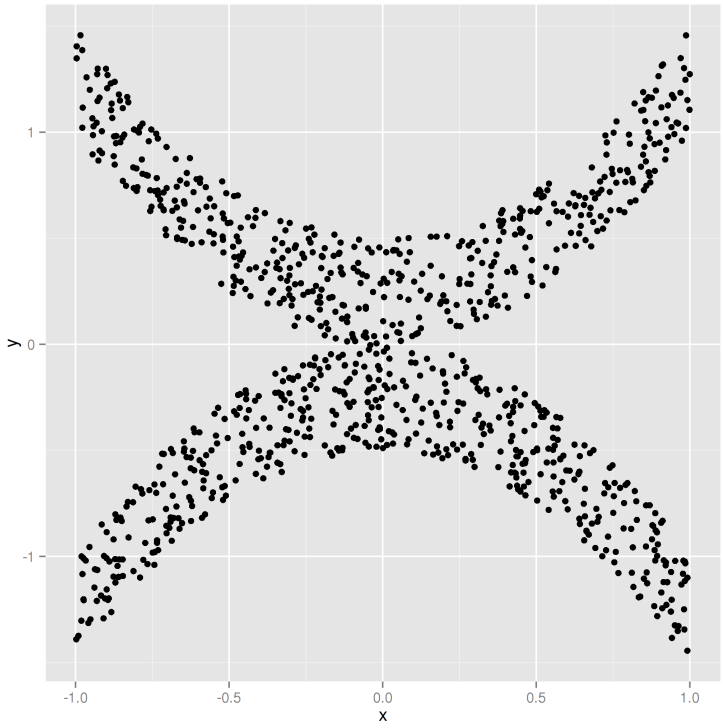


Figure 3: Scatter plot of $n = 1000$ simulated data from the Wikipedia example.

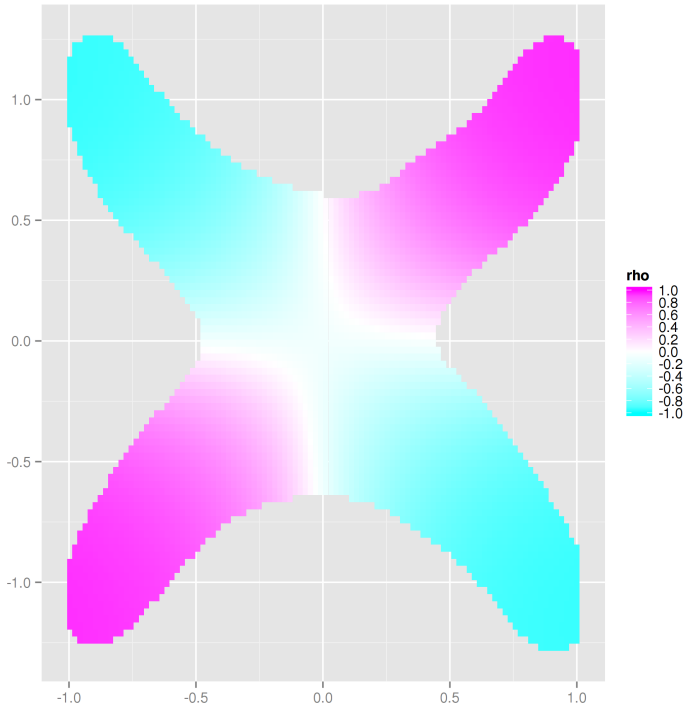


Figure 4: Estimated local Gaussian correlation for the Wikipedia example with value indicated by color.

Note that package **copula** requires that the GNU Scientific Library (**GSL**) is installed on the computer (see Galassi, Davies, Theiler, Jungman, Alken, Booth, and Rossi 2009). Amongst several other elements, concentrations of cesium (Cs) and scandium (Sc) were measured. We are interested in the relation between the concentration of these two elements. Point estimates of local Gaussian parameters at the points $(1.8, 0.7)$ (the lower tail) and $(2.3, 1.2)$ (the upper tail) are obtained by

```
R> xy.mat <- rbind(c(1.8, 0.7), c(2.3, 1.2))
R> lg.out <- localgauss(x = Cs, y = Sc, xy.mat = xy.mat, b1 = 0.6, b2 = 0.4)
R> lg.out$par.est
```

```
   mu_1     mu_2     sig_1     sig_2       rho
[1,] 2.041581 1.023168 0.2173486 0.1715325 0.4262585
[2,] 2.037460 1.020956 0.2432937 0.1679532 0.3224928
```

The following code creates a '`localgauss`' object suitable for plotting:

```
R> lg.out <- localgauss(x = Cs, y = Sc, b1 = 0.6, b2 = 0.4, gsize = 15,
+    hthresh = 0.1)
```

To check if we have chosen a suitable value of `hthresh` we include the observations in the first plot:

```
R> plot(lg.out, plot.text = FALSE, plot.points = TRUE,
+    xlab = "Cs", ylab = "Sc", divergent.col.grad = FALSE)
```

which results in Figure 5. We see that the neighborhood of all points in the area of estimation contains observations indicating a good choice of `hthresh`. Note that with the choice `divergent.col.grad = FALSE` a continuous color gradient is used, which is the better choice for discriminating between values when the range of the values is small. Including numerical values of the estimated local Gaussian correlation can be done by

```
R> plot(lg.out, xlab = "Cs", ylab = "Sc", divergent.col.grad = FALSE)
```

as displayed in Figure 6. Figure 6 indicates that small concentrations of cesium are typically associated with small values of scandium, while large values of cesium are associated with medium to large values of scandium.

### 3.4. Example 4: Non-linear regression

Finally we illustrate the usage of `plot()` for '`localgauss.ind`' objects and `localgauss.ind()` when $X_1$ and $X_2$ are given by the non-linear regression model $X_2 = X_1^2 + \epsilon$, where $X_1$ and $\epsilon$ are independent standard normal. This is yet another example where the ordinary Pearson's correlation is zero even though $X_1$ and $X_2$ are dependent. We start by creating a '`localgauss`' object with estimates of the local parameters in the points $(-1, 1)$, $(0, 0)$ and $(1, 1)$ for $n = 500$ observations simulated from the model:

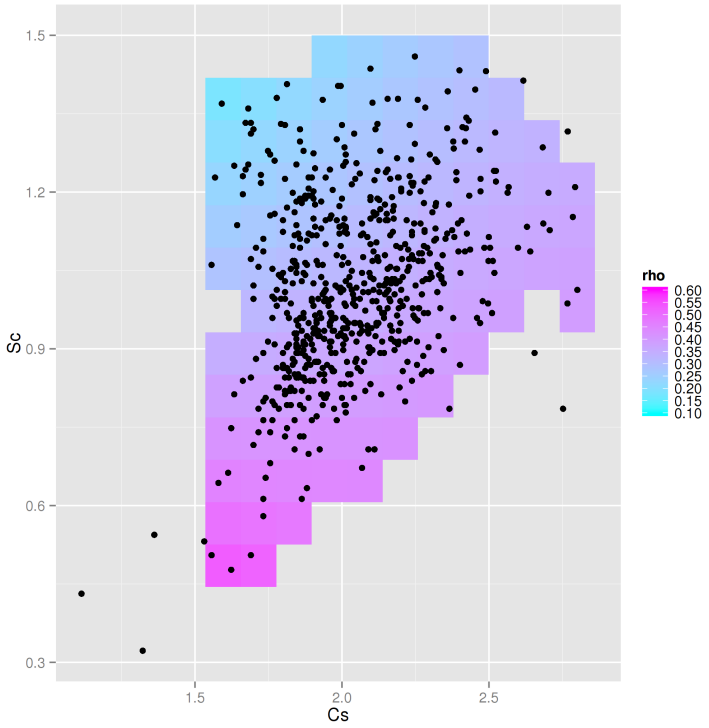


Figure 5: Estimated local Gaussian correlation with overlain observations and value indicated by color for the Uranium exploration data set.

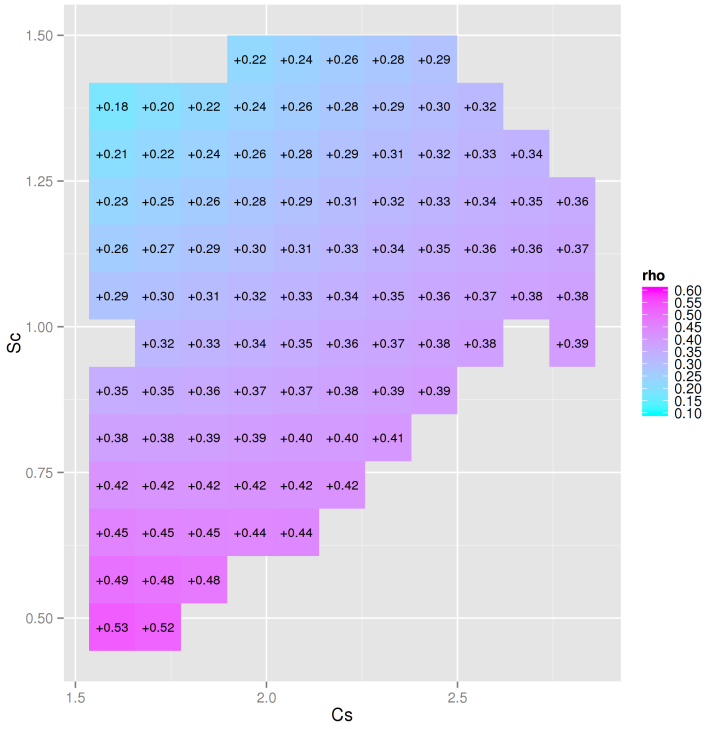


Figure 6: Numerical value of estimated local Gaussian correlation for the Uranium exploration data set.

```
R> x <- rnorm(500)
R> y <- x^2 + rnorm(500)
R> xy.mat <- rbind(c(-1, 1), c(0, 0), c(1, 1))
R> lg.out1 <- localgauss(x, y, b1 = .5, b2 = 1, xy.mat = xy.mat)
```

Performing the local test of independence in the same points can be done by passing the '`localgauss`' object to `localgauss.ind()`:

```
R> ind.out1 <- localgauss.indtest(lg.out1, R = 500)
R> ind.out1$test.results
[1] -1  0  1
```

We see that the local Gaussian correlation is significantly negative in $(-1, 1)$, significantly positive in $(1, 1)$ and neither in $(0, 0)$ at a (default) 10% significance level. In the previous computation no parallel backend is registered and the bootstrapping is done sequentially. Nevertheless, since the null-distribution of $\hat{\rho}(x_i)$ is constructed for only three points, the computational time is quite reasonable (about 10 seconds on the first author's laptop). The following code creates a '`localgauss`' object where the `xy.mat` is generated internally and plots it with overlain observations:

```
R> lg.out2 <- localgauss(x, y, b1 = .5, b2 = 1, hthresh = 0.015, gsize = 30)
R> plot(lg.out2, plot.text = FALSE, plot.points = TRUE)
```

This results in Figure 7. Applying `localgauss.indtest()` on this '`localgauss`' object can be quite time consuming since the new `xy.mat` contains 118 points. The computational time can be shortened by registering a parallel backend if the user has multiple cores/processors on his/hers computer. On a Unix-like system with 28 cores (say) a parallel backend can be registered using the **doParallel** package in the following way:

```
R> library("doParallel")
R> registerDoParallel(cores = 28)
```

The function `localgauss.indtest()` will then automatically run in parallel:

```
R> ind.out2 <- localgauss.indtest(lg.out2, R = 500)
```

On a computer with 4 AMD Opteron 6128 CPUs (a total of 32 cores) and 128 GB of memory this took about 25 seconds, while using just one core on the same computer (sequential run) took about 11 minutes. The result can then be passed to the `plot` method for '`localgauss.indtest`' objects for a graphical display of the test results:

```
R> plot(ind.out2)
```

This results in Figure 8 which displays the regions where $\hat{\rho}(x_i)$ is significantly positive and negative.

## 4. Conclusions

This paper introduces the R package **localgauss** which estimates and visualizes local Gaussian correlation. The package enables users to easily utilize the methodology of local Gaussian

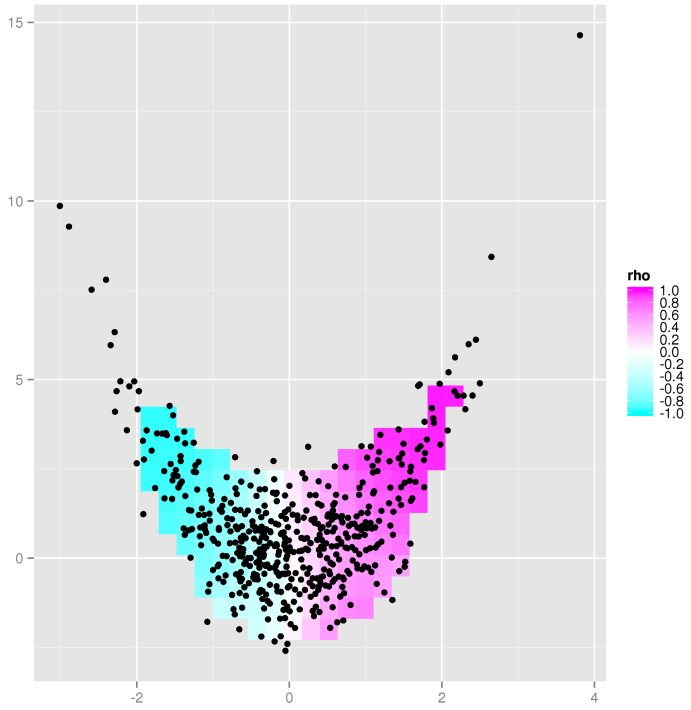


Figure 7: Estimated local Gaussian correlation with $n = 500$ overlain observations from the non-linear regression model.

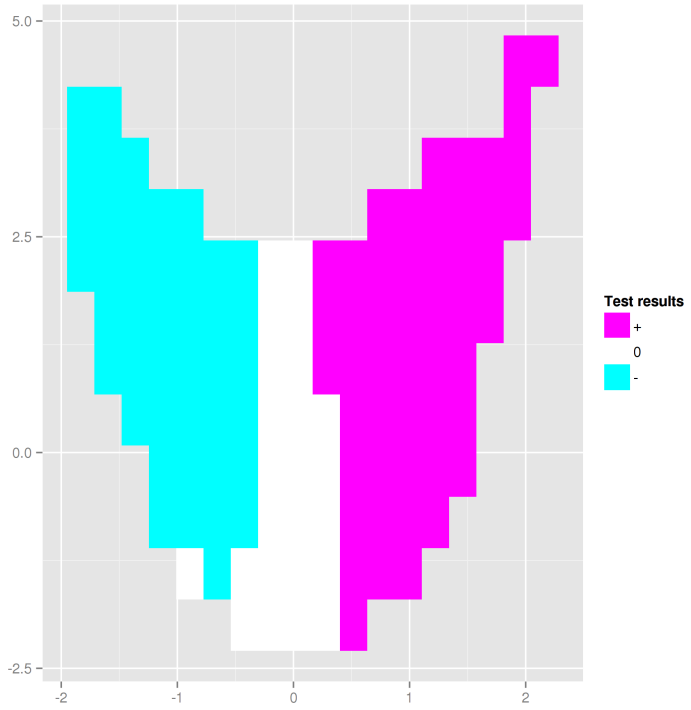


Figure 8: Result of local independence test.

correlation without constructing their own optimization routines. The package can be used for exploring the dependence structure in datasets and can be seen as a supplement to the raw information given by a scatter-plot. Other applications of the package can be found in Berentsen and Tjøstheim (2014) and Berentsen *et al.* (2013). The former article illustrates how the local Gaussian correlation can be used to construct both global and local tests of independence, while the latter article discusses how the local Gaussian correlation can be used to describe and recognize the dependence structure in various copula models. The package has also been used for parts of the work in Støve and Tjøstheim (2014).

## Acknowledgments

The authors are grateful to Karl Ove Hufthammer who laid the foundation for the present plotting routine used in the S3 `plot` method for '`localgauss`' objects. We would also like to thank three anonymous referees for useful comments and suggestions which helped improve both the **localgauss** package and its presentation in this paper.

## References

Berentsen GD, Støve B, Tjøstheim D, øTN (2013). "Recognizing and Visualizing Copulas: An Approach Using Local Gaussian Approximation." Working paper, URL http://people.uib.no/gbe062/local-gaussian-correlation/.

Berentsen GD, Tjøstheim D (2014). "Recognizing and Visualizing Departures from Independence in Bivariate Data Using Local Gaussian Correlation." *Statistics and Computing*. doi:10.1007/s11222-013-9402-8. Forthcoming.

Cook RD, Johnson ME (1986). "Generalized Burr-Pareto-Logistic Distributions with Applications to a Uranium Exploration Data Set." *Technometrics*, **28**(2), 123–131.

Embrechts P, McNeil A, Straumann D (2002). *Risk Management: Value at Risk and Beyond*, chapter Correlation and Dependence in Risk Management: Properties and Pitfalls, pp. 176–223. Cambridge University Press.

Galassi M, Davies J, Theiler BG, Jungman G, Alken P, Booth M, Rossi F (2009). *GNU Scientific Library Reference Manual*. 3rd edition. Network Theory Ltd.

Hascoët L, Pascual V (2004). "**TAPENADE** 2.1 User's Guide." *Technical Report 0300*, INRIA. URL http://www.inria.fr/rrrt/rt-0300.html.

Hjort NL, Jones MC (1996). "Locally Parametric Nonparametric Density Estimation." *The Annals of Statistics*, **24**(4), 1619–1647.

Holland PW, Wang YJ (1987). "Dependence Function for Continuous Bivariate Densities." *Communications in Statistics – Theory and Methods*, **16**(3), 863–876.

Jones MC (1996). "The Local Dependence Function." *Biometrika*, **83**(4), 899–904.

Jones MC, Koch I (2003). "Dependence Maps: Local Dependence in Practice." *Statistics and Computing*, **13**(3), 241–255.

Kane MJ, Emerson J, Weston S (2013). "Scalable Strategies for Computing with Massive Data." *Journal of Statistical Software*, **55**(14), 1–19. URL http://www.jstatsoft.org/v55/i14/.

Lehmann EL (1966). "Some Concepts of Dependence." *The Annals of Mathematical Statistics*, **37**(5), 1137–1153.

Metcalf M, Reid JK (1999). *Fortran 90/95 Explained*. 2nd edition. Oxford University Press, Inc., New York.

Nocedal J, Wright SJ (1999). *Numerical Optimization*. Springer-Verlag.

R Core Team (2013). *R: A Language and Environment for Statistical Computing*. R Foundation for Statistical Computing, Vienna, Austria. URL http://www.R-project.org/.

Revolution Analytics, Weston S (2013a). ***doParallel**: Foreach Parallel Adaptor for the **parallel** Package*. R package version 1.0.6, URL http://CRAN.R-project.org/package=doParallel.

Revolution Analytics, Weston S (2013b). ***foreach**: Foreach Looping Construct for R*. R package version 1.4.1, URL http://CRAN.R-project.org/package=foreach.

Rosenblatt M, Wahlen BE (1992). "A Nonparametric Measure of Independence under a Hypothesis of Independent Components." *Statistics and Probability Letters*, **15**(3), 245–252.

Skaug HJ, Tjøstheim D (1993). "A Nonparametric Test of Serial Independence Based on the Empirical Distribution Function." *Biometrika*, **80**(3), 591–602.

Støve B, Tjøstheim D (2014). "Measuring Asymmetries in Financial Returns: An Empirical Investigation Using Local Gaussian Correlation." In N Haldrup, M Meitz, P Saikkonnen (eds.), *Essays in Nonlinear Time Series Econometrics*. Oxford University Press. Forthcoming.

Støve B, Tjøstheim D, Hufthammer KO (2012). "Using Local Gaussian Correlation in a Nonlinear Re-Examination of Financial Contagion." Revised manuscript, URL http://folk.uib.no/gbe062/local-gaussian-correlation/.

Székely GJ, Rizzo ML (2009). "Brownian Distance Covariance." *The Annals of Applied Statistics*, **3**(4), 1236–1265.

Székely GJ, Rizzo ML, Bakirov NK (2007). "Measuring and Testing Dependence by Correlation of Distances." *The Annals of Statistics*, **35**(6), 2769–2794.

Tjøstheim D (1996). "Measures of Dependence and Tests of Independence." *Statistics: A Journal of Theoretical and Applied Statistics*, **28**(3), 249–284.

Tjøstheim D, Hufthammer KO (2013). "Local Gaussian Correlation: A New Measure of Dependence." *Journal of Econometrics*, **172**(1), 33–48.

Venables WN, Ripley BD (2002). *Modern Applied Statistics with S*. 4th edition. Springer-Verlag, New York. URL http://www.stats.ox.ac.uk/pub/MASS4.

Wickham H (2009). ***ggplot2**: Elegant Graphics for Data Analysis*. Springer-Verlag, New York. URL http://had.co.nz/ggplot2/book.

Wikipedia (2012). "Correlation and Dependence." Accessed 2012-09-10, URL http://en.wikipedia.org/wiki/Correlation_and_dependence.

Yan J (2007). "Enjoy the Joy of Copulas: With a Package **copula**." *Journal of Statistical Software*, **21**(4), 1–21. URL http://www.jstatsoft.org/v21/i04/.

## Affiliation:

Geir Drage Berentsen, Tore Selland Kleppe, Dag Bjarne Tjøstheim
Department of Mathematics
University of Bergen
Postboks 7800
5020 Bergen, Norway
E-mail: geir.berentsen@math.uib.no, Tore.Kleppe@math.uib.no,
Dag.Tjostheim@math.uib.no
URL: http://folk.uib.no/tkl083,
http://www.uib.no/persons/Dag.Tjostheim

*Journal of Statistical Software* http://www.jstatsoft.org/
published by the American Statistical Association http://www.amstat.org/
Volume 56, Issue 12 *Submitted:* 2012-10-19
February 2014 *Accepted:* 2013-09-11