QAD-Package

Konrad Medicus, Fabian Oberreiter,Tobias Hilgart

July 2, 2019

Contents

1 Introduction

The *"Quantification of Asymmetric Dependence"* package introduces a copula-based dependency measure capable of detecting and depicting asymmetry.

Dependency measures such as the various correlations are not applicable to measure asymmetric dependence. Using copulae, which describe the joint distribution of a random vector (X, Y) one can 'measure' the distance of a given copula to independence (dependency measure ζ_1, Trutschnig 2011, note) and fullfill all requirements for an asymmetric dependency measure, such as invariant under scale changes, 1 iff Y is a function of X, 0 iff independent etc.

However, when one works with samples – observations $(x_1, y_1), \ldots, (x_n, y_n)$ – the true distribution they are drawn from is generally unknown, and so is the copula then.

Thus there is a need to find a 'good estimate' for said copula in the sense, that it gives a good estimate of the dependency measure.

However, in said measure, the 'known' approximation called *empirical copula* (corresponds to empirical distribution) does not provide a good estimate.

This package introduces a 'smooth aggregation' of the empirical copula, called *empirical checkerboard copula*, that does provide a good estimates and enables the user to estimate the (asymmetric) dependence from a sample by using it.

2 qad

The qad – function does exactly that: From a given data.frame containing a bivariate sample – on column per variable– it calculates the empirical checkerboard copula and computes the dependency measure, thus quantifying the asymmetric dependence (qad) of two random variables X and Y.

2.1 Example I

```
n = 200
x = runif(n,-2,4)
y = sin(x^2)
df = data.frame(x,y)
model = qad(df, print = FALSE, permutation = TRUE)
model$results
```

```
##                          coef p.values
## 1        q(x1,x2) 0.6610848        0
## 2        q(x2,x1) 0.3781406        0
## 3 mean.dependence 0.5196127        0
## 4 asymmetry       0.2829442        0
```

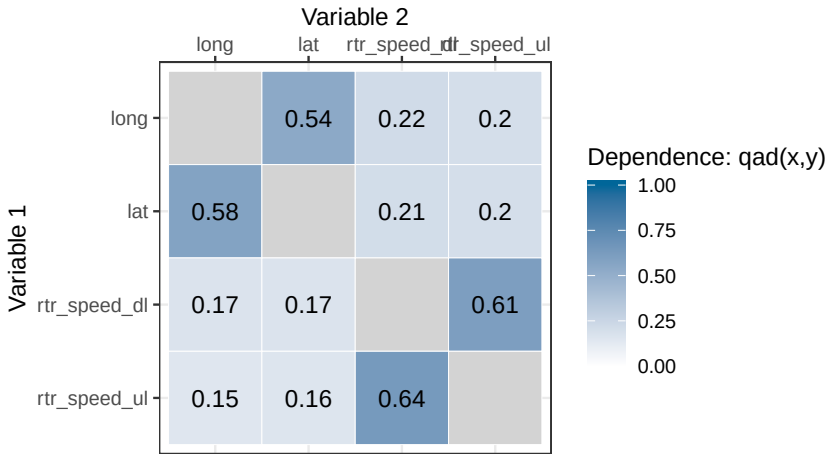
A permutated p-test can be performed to test for independence. The summary function can also be called on qad-objects. With pairwise.qad, one can calculate the qad for each pair of columns in a given data.frame, and heatmap.qad visualizes the results.

2.2 Example II

```
  load(url("http://www.trutschnig.net/RTR.RData"))
  df = RTR[sample(nrow(RTR),1000),
           c('long','lat','rtr_speed_dl','rtr_speed_ul')]
  model = pairwise.qad(df)
```

```
## [1] "Process..."
## [1] "Process: 1 / 4"
## [1] "Process: 2 / 4"
## [1] "Process: 3 / 4"
```

```
  heatmap.qad(model)
```



3 cci

Gives a confidence intervall, depending on the sample size, for the qad measure of two independent random variables. Thus, it can be used to test for independence.

```
c = cci(n, alternative = "one.sided")

x = runif(n,-2,4)
y = sin(x^2)
df = data.frame(x,y)
model = qad(df,print=FALSE)

if(coef(model, select = 'q(x1,x2)') %in% c){
 print('Accept H0')
}else{
 print('Reject H0')
}

## [1] "Reject H0"
```

4 the rest plot/predict/emp_ c_ copula/plot_ density

If one desires to calculate just the empirical copula (smoothing=FALSE) or the checkerboard aggregation (smoothing=TRUE), one can do so by plugging the data.frame(x,y) into the emp_ c_ copula() – function. The resulting mass/density matrix can be plotted via plot_ density.

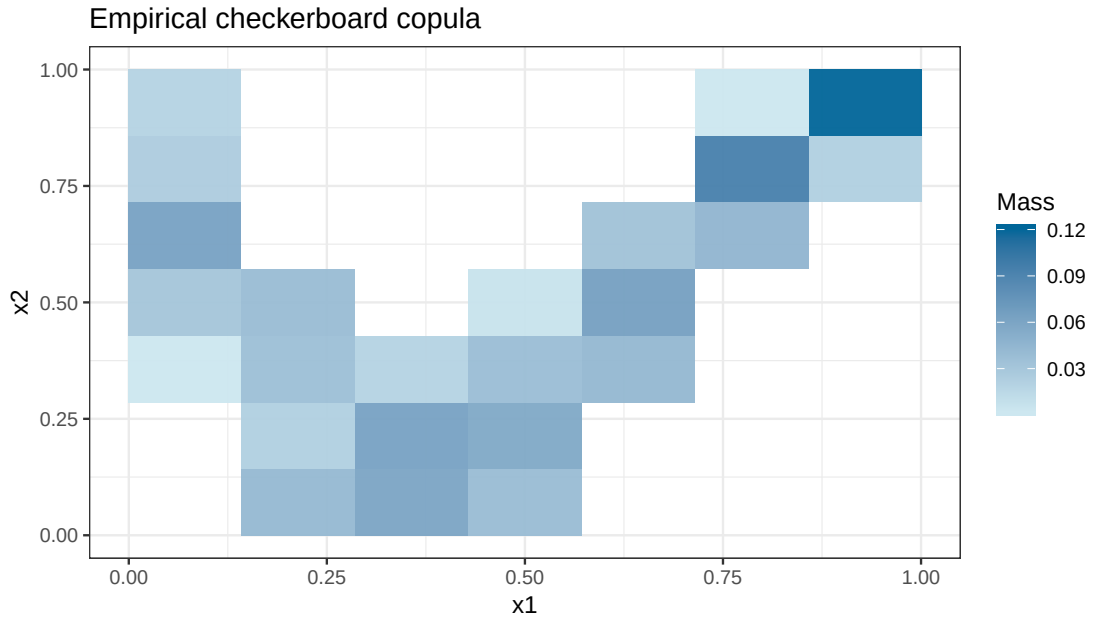
Alternatively, calling the plot – function on a qad-object directly plots it's mass matrix (from the checkerboard aggregation), one can also add the sample points and retransform from the copula into the original data setting.

Using the conditional probabilities, with the predict command on a qad-object, one gets the probabilities to land in certain intervals, given either certain x or y values (after specifying which are given). One can manually set the number of intervals, into which the range of the other RV is splitted.

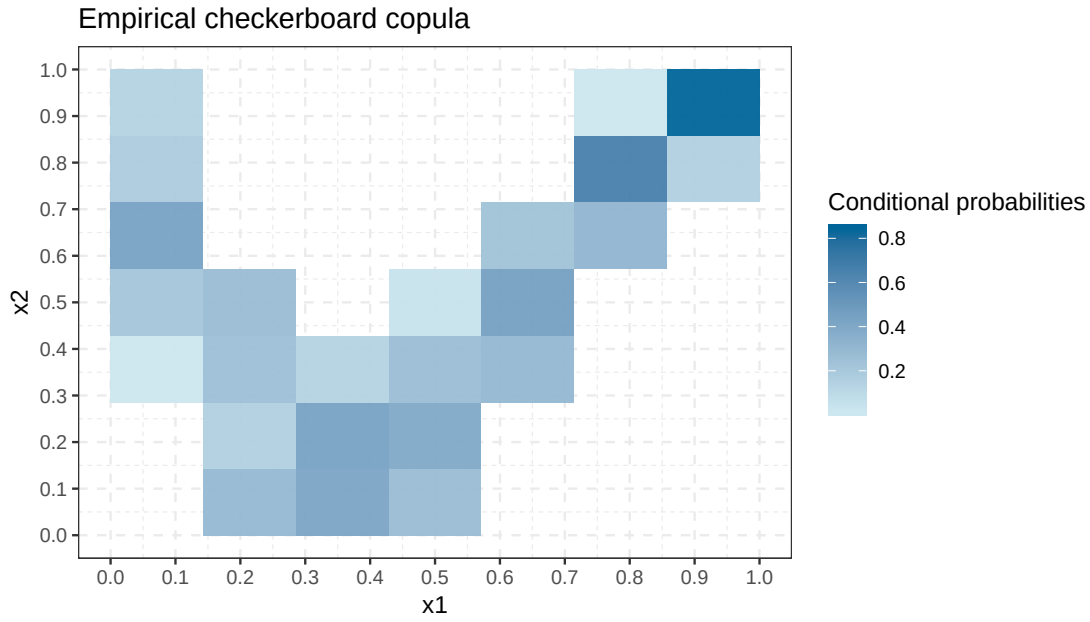
4.1 Example I

```
set.seed(2019)
x = rnorm(50,1,2)
y = x^2 + rnorm(50,0,1)
df = data.frame(x,y)

ECC = emp_c_copula(df, resolution = 7)
plot_density(ECC,density=FALSE)
```




```
model = qad(df,print=FALSE)
plot(model, copula = TRUE)
```



4.2 Example II

```
predict(model, values = c(-1,0,1,2,3), conditioned = "x1",
        nr_intervals = 5)

##    (-0.73,0.5] (0.5,1.87] (1.87,4.9] (4.9,7.27] (7.27,39.82]
## -1       0.364  0.3040000  0.3320000      0.000         0.000
## 0        0.596  0.3760000  0.0280000      0.000         0.000
## 1        0.440  0.4594286  0.1005714      0.000         0.000
## 2        0.000  0.2445714  0.5634286      0.192         0.000
## 3        0.000  0.0000000  0.0000000      0.096         0.904
```