

CopReg Vignette: Use of CFI for variable selection/elimination

04 October 2026

CFI = comparative fit index

Reference for the use of CFI: Brechmann, E. C. and Joe, H. (2015). Truncation of vine copulas using fit indices. Journal of Multivariate Analysis, 138:19-33.

```
library(CopReg)
library(igraph)
```

1. Create correlation matrix

Input elements for `getVinearray` function: vanderWaarden/polychoric/polyserial correlation matrix from training set extracted from `data(donkey)`

```
n <- 400
rmat <- matrix(c(
  1.000, 0.193, -0.097, 0.408, 0.499, 0.256, 0.558,
  0.193, 1.000, -0.024, 0.509, 0.597, 0.381, 0.611,
  -0.097, -0.024, 1.000, -0.147, -0.144, 0.023, -0.095,
  0.408, 0.509, -0.147, 1.000, 0.622, 0.532, 0.763,
  0.499, 0.597, -0.144, 0.622, 1.000, 0.657, 0.875,
  0.256, 0.381, 0.023, 0.532, 0.657, 1.000, 0.745,
  0.558, 0.611, -0.095, 0.763, 0.875, 0.674, 1.000), 7,7)
```

2. Illustration of CFI=0.95

Use of `getVinearray` function and interpretation of output

```
#vineobj95 <- getVinearray(rmat, n, method = "MSTleaf", cfi_bd = 0.95)
#vineobj95 <- getVinearray(rmat, n, method = "MSTx", cfi_bd = 0.95)
#vineobj95 <- getVinearray(rmat, n, method = "forwardR", cfi_bd = 0.95)
vineobj95 <- getVinearray(rmat, n, method = "forwardD", cfi_bd = 0.95)
names(vineobj95)
#> [1] "vine_array" "trunc_level" "cfi" "cfi_vector" "treeweight"
#> [6] "fitval" "pcmat"
print(vineobj95$trunc_level)
#> [1] 4
print(vineobj95$fitval)
#> [1] 0.5668414 0.7625183 0.8902764 0.9621231
print(vineobj95$cfi_vector)
#> [1] 0.5704513 0.7655835 0.8925362 0.9634711
print(vineobj95$pcmat)
#>      [,1] [,2]      [,3]      [,4]      [,5]      [,6]      [,7]
#> [1,]  0 0.622 0.4080000 0.2560000 0.3810000 -0.02400000 0.8750000
#> [2,]  0 0.000 0.3430279 0.4844461 0.1068110 0.03477626 0.5770602
#> [3,]  0 0.000 0.0000000 0.5058531 0.3793640 -0.10333267 0.2552377
#> [4,]  0 0.000 0.0000000 0.0000000 0.4371845 -0.16609218 0.2362427
```

```
print(vineobj95$vine_array)
#>      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
#> [1,]    5    5    4    1    6    2    5
#> [2,]    0    4    5    4    1    6    4
#> [3,]    0    0    1    5    4    1    1
#> [4,]    0    0    0    6    5    4    6
#> [5,]    0    0    0    0    2    0    0
#> [6,]    0    0    0    0    0    3    0
#> [7,]    0    0    0    0    0    0    7
```

Interpretation: can maybe try vine regression with explanatory variables 5,4,1,6. Can delete variables 2,3.

For vine regression, use the following vine array.

```
# donkeysub95.tr <- donkey.tr[, -c(2,3)] # see donkey-v2.Rmd vignette
Asub95 <- vineobj95$vine_array
Asub95 <- Asub95[-c(5,6), -c(5,6)]
Asub95[Asub95>3] <- Asub95[Asub95>3]-2
Asub95[4,5] <- 1
print(Asub95)
#>      [,1] [,2] [,3] [,4] [,5]
#> [1,]    3    3    2    1    3
#> [2,]    0    2    3    2    2
#> [3,]    0    0    1    3    1
#> [4,]    0    0    0    4    1
#> [5,]    0    0    0    0    5
```

Response variable (last column is now indexed as variable 5 instead of 7. Variables 4,5,6 reindexed as 2,3,4 respectively.

Convert `donkeysub95.tr` to hypercube data `udata95` and input to `CopReg` with arguments `data=udata95`, `vine_array=Asub95`, `is_disc = c(T,F,F,F,F)`.

3. Illustration of CFI=0.98

higher value of CFI, more vine trees

Use of `getVinearray` function and interpretation of output

```
vineobj98 <- getVinearray(rmat, n, method = "forwardD", cfi_bd = 0.98)
print(vineobj98$trunc_level)
#> [1] 5
print(vineobj98$fitval)
#> [1] 0.5668414 0.7625183 0.8902764 0.9621231 0.9819459
print(vineobj98$cfi_vector)
#> [1] 0.5704513 0.7655835 0.8925362 0.9634711 0.9823255
print(vineobj98$pcmat)
#>      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
#> [1,]    0 0.622 0.4080000 0.2560000 0.3810000 -0.02400000 0.8750000
#> [2,]    0 0.000 0.3430279 0.4844461 0.1068110 0.03477626 0.5770602
#> [3,]    0 0.000 0.0000000 0.5058531 0.3793640 -0.10333267 0.2552377
#> [4,]    0 0.000 0.0000000 0.0000000 0.4371845 -0.16609218 0.2362427
#> [5,]    0 0.000 0.0000000 0.0000000 0.0000000 -0.17546421 0.2225887
print(vineobj98$vine_array)
#>      [,1] [,2] [,3] [,4] [,5] [,6] [,7]
#> [1,]    5    5    4    1    6    2    5
```

```
#> [2,] 0 4 5 4 1 6 4
#> [3,] 0 0 1 5 4 1 1
#> [4,] 0 0 0 6 5 4 6
#> [5,] 0 0 0 0 2 5 2
#> [6,] 0 0 0 0 0 3 0
#> [7,] 0 0 0 0 0 0 7
```

Interpretation: can maybe try vine regression with explanatory variables 5,4,1,6,2. Can delete variables 3.

For vine regression, use the following vine array.

```
#donkeysub98.tr <- donkey.tr[, -3] # see donkey.Rmd vignette
Asub98 <- vineobj98$vine_array
Asub98 <- Asub98[-6, -6]
Asub98[Asub98>3] <- Asub98[Asub98>3]-1
print(Asub98)
#>      [,1] [,2] [,3] [,4] [,5] [,6]
#> [1,] 4 4 3 1 5 4
#> [2,] 0 3 4 3 1 3
#> [3,] 0 0 1 4 3 1
#> [4,] 0 0 0 5 4 5
#> [5,] 0 0 0 0 2 2
#> [6,] 0 0 0 0 0 6
```

Response variable (last column is now indexed as variable 6 instead of 7. Variables 4,5,6 reindexed as 3,4,5 respectively.

Convert `donkeysub98.tr` to hypercube data `udata98` and input to `CopReg` with arguments `data=udata98`, `vine_array=Asub98`, `is_disc = c(T,T,F,F,F,F)`.