

# Spearman's Rho for the AMH Copula: a Beautiful Formula

Martin Mächler

ETH Zurich

June 2014

---

## Abstract

We derive a beautiful series expansion for Spearman's rho,  $\rho(\theta)$  of the Ali-Mikhail-Haq (AMH) copula with parameter  $\theta$  which is also called  $\alpha$  or  $\theta$ . Further, via experiments we determine the cutoffs to be used for practically fast and accurate computation of  $\rho(\theta)$  for all  $\theta \in [-1, 1]$ .

*Keywords:* Archimedean copulas, Spearman's rho.

---

## 1. Introduction

A *copula* is a multivariate distribution function with standard uniform univariate margins. Standard references for an introduction are [Joe \(1997\)](#) or [Nelsen \(2007\)](#).

[Sklar \(1959\)](#) shows that for any multivariate distribution function  $H$  with margins  $F_j$ ,  $j \in \{1, \dots, d\}$ , there exists a copula  $C$  such that

$$H(x_1, \dots, x_d) = C(F_1(x_1), \dots, F_d(x_d)), \quad \mathbf{x} \in \mathbb{R}^d. \quad (1)$$

Conversely, given a copula  $C$  and arbitrary univariate distribution functions  $F_j$ ,  $j \in \{1, \dots, d\}$ ,  $H$  defined by (1) is a distribution function with marginals  $F_j$ ,  $j \in \{1, \dots, d\}$ .

## 2. Archimedean copulas

An *Archimedean generator*, or simply *generator*, is a continuous, decreasing function  $\psi : [0, \infty] \rightarrow [0, 1]$  which satisfies  $\psi(0) = 1$ ,  $\psi(\infty) := \lim_{t \rightarrow \infty} \psi(t) = 0$ , and which is strictly decreasing on  $[0, \inf\{t : \psi(t) = 0\}]$ . A  $d$ -dimensional copula is called *Archimedean* if it is of the form

$$C(\mathbf{u}; \psi) = \psi(\psi^{-1}(u_1) + \dots + \psi^{-1}(u_d)), \quad \mathbf{u} \in [0, 1]^d, \quad (2)$$

for some generator  $\psi$  with inverse  $\psi^{-1} : [0, 1] \rightarrow [0, \infty]$ , where  $\psi^{-1}(0) = \inf\{t : \psi(t) = 0\}$ . A necessary and sufficient condition for an Archimedean generator  $\psi$  to generate a proper copula in all dimensions  $d$  is that  $\psi$  is *completely monotone*, i.e.,  $(-1)^k \psi^{(k)}(t) \geq 0$  for all  $t \in (0, \infty)$  and  $k \in \mathbb{N}_0$ . See [Hofert and Maechler \(2011\)](#) and its references, for considerably more details.

### 2.1. The Ali-Mikhail-Haq (AMH) copulas

An Ali-Mikhail-Haq (AMH) copula with parameter  $\theta$ ,  $\theta \in [-1, 1)$  (where the right boundary,  $\theta = 1$  can sometimes be considered valid) has generator

$$\psi_{\text{AMH}}(t, \theta) = \frac{1 - \theta}{\exp(t) - \theta}. \quad (3)$$

For,  $\theta = 0$ , clearly  $\psi(t) = \exp(-t)$ , corresponds to independence. Both “rank based” association measures or correlations, Kendall's  $\tau$  and Spearman's  $\rho$ , are montone in  $\theta$ , and hence have the same sign as  $\theta$ .

Kendall's tau is equal to

$$\tau_\theta = 1 - \frac{2((1 - \theta)^2 \log(1 - \theta) + \theta)}{3\theta^2}, \quad (4)$$

for  $\theta \in [0, 1)$ ,  $\tau$  is in  $[0, \frac{1}{3})$ . The formula (4) needs care when  $\theta$  is close to zero, and we provide `tauAMH()` in the **copula** package, using a Taylor series for small  $|\theta|$ , see `help(tauAMH)`.

## 3. Spearman's Rho ( $\rho$ ) for AMH

### 3.1. The beautiful formula

Nelsen (2007, ex. 5.10, p. 172) provides the following formula for Spearman's  $\rho$  for the AMH copula,

$$\rho(\theta) = \frac{12(1 + \theta)}{\theta^2} \cdot \text{dilog}(1 - \theta) - \frac{24(1 - \theta)}{\theta^2} \cdot \log(1 - \theta) - \frac{3(\theta + 12)}{\theta}, \quad (5)$$

where his “dilogarithm”  $\text{dilog}(x) = \text{Li}_2(1 - x) = \text{polylog}(1 - x, 2)$ , and  $\text{Li}_2(x)$  is the usual definition of the dilogarithm (also called “Spence's function”),

$$\text{Li}_2(z) = - \int_0^z \frac{\ln(1 - u)}{u} du = \sum_{k=1}^{\infty} \frac{z^k}{k^2}, \quad z \in \mathbb{C} \setminus [1, \infty), \quad (6)$$

where the infinite sum is only applicable for  $|z| < 1$ .

With the boundaries for  $\theta \in \{-1, 1\}$ , this leads to a range of  $\rho$  in the interval  $[33 - 48 \log 2, 4\pi^2 - 39]$  or approximately  $[-0.2711, 0.4784]$ .

It is clear that formula (5) cannot be used for  $\theta = 0$  and further inspection reveals that it also heavily suffers from cancellation for  $|\theta| \ll 1$ .

In order to compute  $\rho$  accurately for all values of  $\theta$ , we look at the Taylor series of the respective terms in (5) and will find a beautiful infinite series formula for  $\rho(\theta)$ .

$$\begin{aligned} \rho(\theta) &= \frac{12(1 + \theta)}{\theta^2} \cdot \text{Li}_2(\theta) - \frac{24(1 - \theta)}{\theta^2} \cdot \log(1 - \theta) - \frac{3(\theta + 12)}{\theta} \\ &= 3/\theta \cdot (4(1 + \theta)/\theta \cdot \text{Li}_2(\theta) - 8(1 - \theta)/\theta \cdot \log(1 - \theta) - (\theta + 12)) \\ &= \frac{3}{\theta} \cdot r(\theta), \quad \text{where} \end{aligned} \quad (7)$$

$$r(\theta) := 4(1 + \frac{1}{\theta}) \cdot \text{Li}_2(\theta) - 8(\frac{1}{\theta} - 1) \cdot \log(1 - \theta) - (\theta + 12). \quad (8)$$

Now, we plug in the Taylor series of both  $\text{Li}_2(\theta) = \sum_{k=1}^{\infty} \frac{\theta^k}{k^2}$ , hence

$$\begin{aligned} r_1(\theta) &:= (1 + \frac{1}{\theta}) \cdot \text{Li}_2(\theta) = \text{Li}_2(\theta) + \frac{1}{\theta} \cdot \text{Li}_2(\theta) = \sum_{k=1}^{\infty} \frac{\theta^k}{k^2} + \sum_{k=1}^{\infty} \frac{\theta^{k-1}}{k^2} = \\ &= 1 + \sum_{k=1}^{\infty} \frac{k^2 + (k+1)^2}{k^2(k+1)^2} \theta^k, \end{aligned} \quad (9)$$

and  $\log(1 - \theta) = \theta + \frac{\theta^2}{2} + \frac{\theta^3}{3} + \dots = \sum_{k=1}^{\infty} \frac{\theta^k}{k}$ , hence

$$r_2(\theta) := (1 - \frac{1}{\theta}) \log(1 - \theta) = \sum_{k=1}^{\infty} \frac{\theta^k}{k} - \sum_{k=1}^{\infty} \frac{\theta^{k-1}}{k} = -1 + \sum_{k=1}^{\infty} \frac{\theta^k}{k(k+1)}. \quad (10)$$

Consequently, first from (8), then plugging in (9) and (10),

$$\begin{aligned} r(\theta) &= 4r_1(\theta) - 8r_2(\theta) - (12 + \theta) = \\ &= (4 \cdot 1 - 8(-1) - 12) + (4 \cdot \frac{5}{4} - 8 \cdot \frac{1}{2} - 1)\theta + \sum_{k=2}^{\infty} \left( \frac{4(k^2 + (k+1)^2)}{k^2(k+1)^2} - \frac{8}{k(k+1)} \right) \theta^k = \\ &= 0 + 0 \cdot \theta + \sum_{k=2}^{\infty} \frac{4(k^2 + (k+1)^2) - 8k(k+1)}{k^2(k+1)^2} \theta^k = \\ &= \sum_{k=2}^{\infty} \frac{4(2k^2 + 2k + 1)^2 - 8k(k+1)}{k^2(k+1)^2} \theta^k = \\ &= \sum_{k=2}^{\infty} \frac{4}{k^2(k+1)^2} \theta^k = \sum_{k=2}^{\infty} \frac{\theta^k}{\binom{k+1}{2}^2}, \end{aligned} \quad (11)$$

a beautiful formula with reciprocal binomial coefficients, and finally, as  $\rho(\theta) = \frac{3}{\theta} \cdot r(\theta)$  (7) from the above,

$$\rho(\theta) = \sum_{k=1}^{\infty} \frac{3}{\binom{k+2}{2}^2} \cdot \theta^k = \frac{\theta}{3} + \frac{\theta^2}{12} + \frac{3\theta^3}{100} + \frac{\theta^4}{75} + \dots \quad (12)$$

the “beautiful formula” for Spearman’s  $\rho$  of an AMH copula with parameter  $\theta$ . Compare this compact formula

$$\boxed{\rho(\theta) = \sum_{k=1}^{\infty} \frac{3\theta^k}{\binom{k+2}{2}^2}}$$

with the original three term formula (5) which involves  $\text{dilog}()$  and  $\log()$ , to understand why I call it *beautiful*. Note further that the “beautiful formula” clearly shows the approximate linearity of  $\rho(\theta)$  for small  $|\theta|$ . Note that the first few coefficients  $a_k$  in  $\rho(\theta) = \sum_{k=1}^{\infty} a_k \theta^k$  are

```
> require(sfsmisc) #--> mat2tex(), mult.fig(), eaxis()
> k <- 1:9; ak <- MASS::fractions(12/((k+1)*(k+2))^2)
> rbind(k = k, `\$a_k\$` = as.character(ak))
```

k	1	2	3	4	5	6	7	8	9
$a_k$	1/3	1/12	3/100	1/75	1/147	3/784	1/432	1/675	3/3025

### 3.2. Accurate and efficient R implementation of $\rho_{\text{AMH}}$

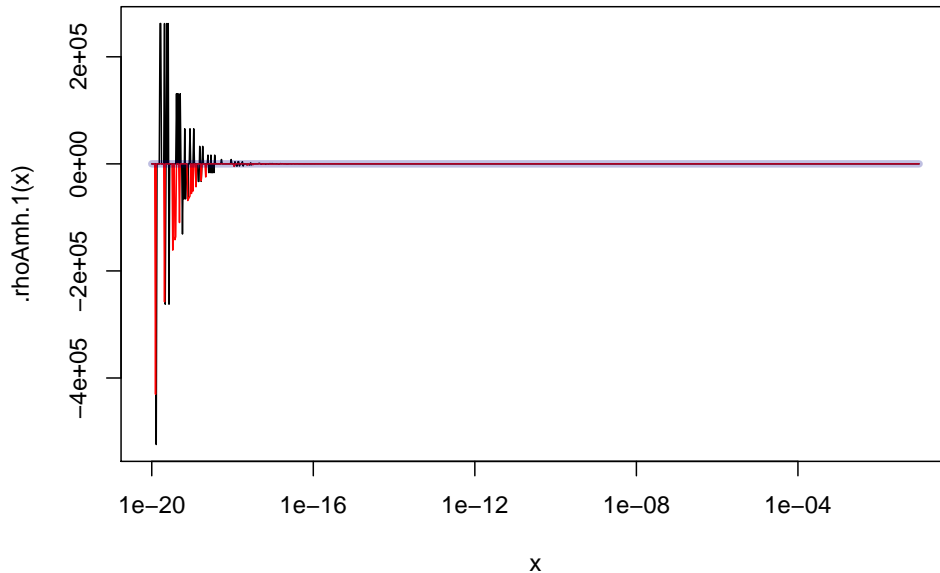
In the following R code, we use `a` as short form for the copula parameter  $\theta$  (which is also called  $\alpha$  in the literature):

```
> ##' Version 1: Direct formula from Nelsen:
> .rhoAmh.1 <- function(a) {
  Li2 <- gsl::dilog(a)
  12 * (1 + a) / a^2 * Li2 - 24 * (1 - a) / a^2 * log1p(- a) - 3 * (a + 12) / a
}
> .rhoAmh.1b <- function(a) {
  Li2 <- gsl::dilog(a)
  ## factored out 3/a from version 1:
  3/a * (4 * (1 + a) / a * Li2 - 8 * (1 - a) / a * log1p(- a) - (a + 12))
}
> ##' Version 2:
> .rhoAmh.2 <- function(a, e.sml = 1e-11) {
  stopifnot(length(a) <= 1)
  if(abs(a) < e.sml) { ## if |a| << 1, do better than the direct formula:
    a*(1/3 + a*(1/12 + a*(3/100 + a/75)))
  } else { ## regular a
    Li2 <- gsl::dilog(a)
    3/a * (4 * (1 + 1/a) * Li2 - 8 * (1/a - 1) * log1p(- a) - (a + 12))
  }
}
> ##' Series version with N terms:
> rhoAmh.T <- function(a, N) {
  stopifnot(length(N) == 1, N == as.integer(N), N >= 1)
  if(N <= 4)
    switch(N,
      a/3,
      a/3*(1 + a/4),
      a*(1/3 + a*(1/12 + a* 3/100)),
      a*(1/3 + a*(1/12 + a*(3/100 + a/75))))
  else { ## N >= 5
    n <- N:1 #--> sum smallest to largest
    if(is(a, "mpfr")) ## so all computations work in high precision
      n <- mpfr(n, precBits=max(.getPrec(a)))
    cf <- ## 3/choose(n+2, 2)^2
      3/((n+1)*(n+2)/2)^2
    a2n <- outer(n,a, function(x,y) y^x) ## a2n[i,j] := a[j] ^ n[i]
    colSums(cf * a2n)
  }
}
```

Now, the first graphical exploration, notably of the original Nelsen formula, `.rhoAmh.1()` and its variant very slight improvement `.rhoAmh.1b()`

```
> r1 <- curve( .rhoAmh.1 (x), 1e-20, .1, log="x", n=1025)
> r1b <- curve( .rhoAmh.1b(x), n=1025, add=TRUE, col=2)
> r2 <- curve( Vectorize(.rhoAmh.2)(x), n=1025, add=TRUE,
```

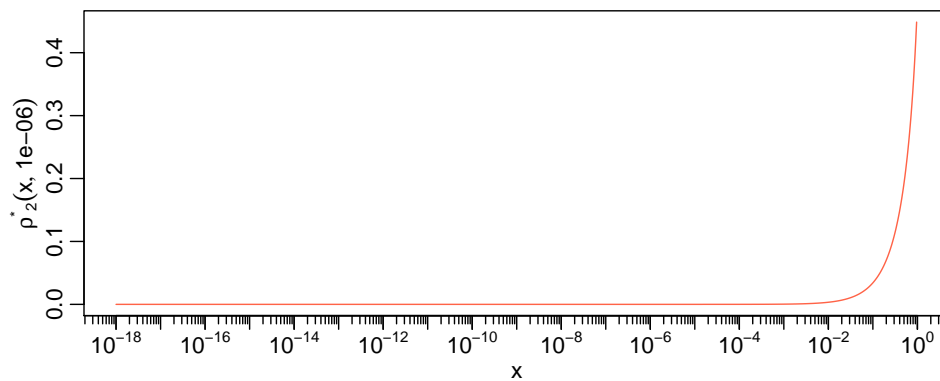
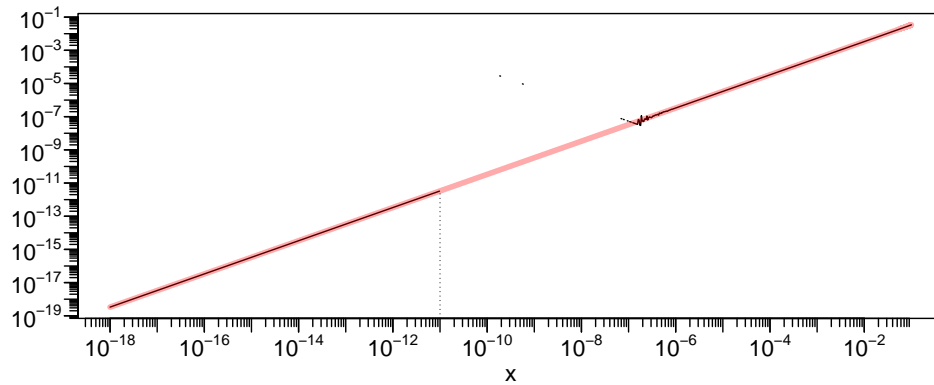
```
col=adjustcolor("blue4",1/4), lwd = 5)
> tab <- cbind(as.data.frame(r1), y.b = r1b$y, y2 = r2$y)
```



expose the big problems (y-values between -400'000 and 200'000 where  $|\rho| < 1$  is known!). Investigating `tab` shows that `1b` is very slightly better than `1`, but looking closer, e.g. also with `curve(.rhoAmh.1(x), 1e-20, .1, log="x", n=1025, ylim=c(-1,1)*.1)`, shows that Nelsen's direct formula is really unusable for  $|\theta| < 10^{-11}$  approximately.

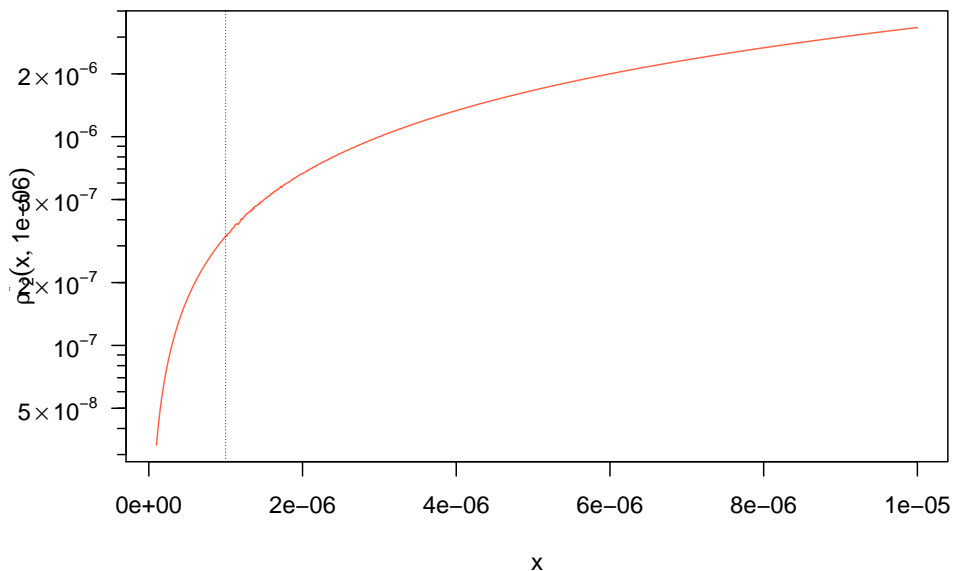
So, `.rhoAmh.2()` using a 4-terms series approximation for  $|\theta| < \mathbf{e.sml}$  is much better, but it is still not good enough, as is revealed by drawing it once with its default cutoff  $\mathbf{e.sml} = 10^{-11}$  and then in red with a higher cutoff  $10^{-6}$  (and in log-log and regular y-axis scale):

```
> if(require("sfsmisc")) {
  myAxes <- function(sides) for(s in sides) eaxis(s)
} else {
  myAxes <- function(sides) for(s in sides) axis(s)
}
> rhoAcurve <- function(k, ..., log = "",
  ylab = substitute({rho~"*"}[2](x, KK), list(KK=k)))
  curve(Vectorize(.rhoAmh.2)(x, k), n=1025, ylab=ylab, log=log,
    xaxt = if(grepl("x", log, fixed=TRUE)) "n" else "s",
    yaxt = if(grepl("y", log, fixed=TRUE)) "n" else "s", ...)
> e.s <- eval(formals(.rhoAmh.2)$e.sml); t0 <- e.s * .99999
> op <- sfsmisc::mult.fig(2, marP = -c(1.4,1,1,1))$old.par
> rhoAcurve(e.s, 1e-18, 1e-1, log = "xy", ylab=""); myAxes(1:2)
> lines(t0, .rhoAmh.2(t0), type="h", lty=3, lwd = 3/4)
> rhoAcurve(1e-6, add=TRUE, col=adjustcolor(2, 1/3), lwd=4)
> rhoAcurve(1e-6, 1e-18, 1, log="x", col="tomato"); myAxes(1)
> par(op)
```



So the default cutoff ( $10^{-11}$ ) is too small, as the explicit (Nelsen) formula breaks down between the cutoff and  $\approx 10^{-7}$ . Hence we are aiming for a cutoff  $> 10^{-7}$ , momentarily  $= 10^{-6}$ , and zoom into its neighborhood:

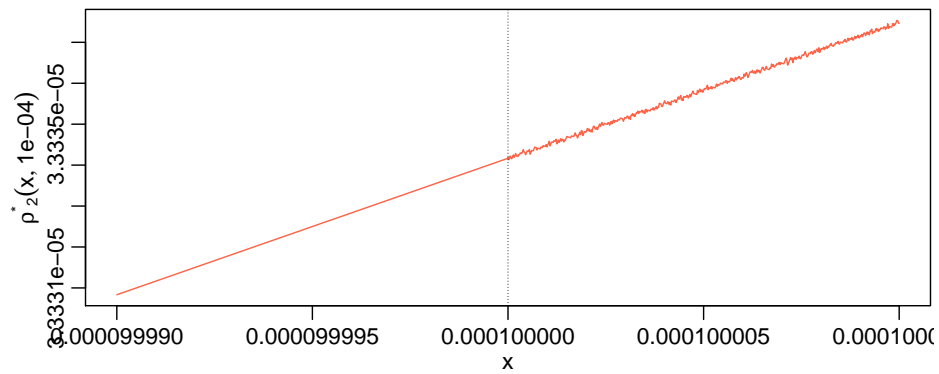
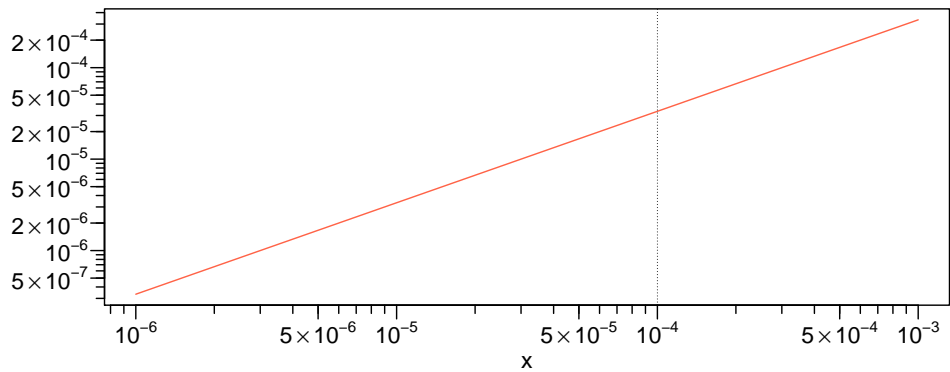
```
> rhoAcurve(1e-6, 1e-7, 1e-5, log = "y", col="tomato"); myAxes(2)
> abline(v=1e-6, lty=3, lwd=1/2)
```



Use still a larger cutoff:

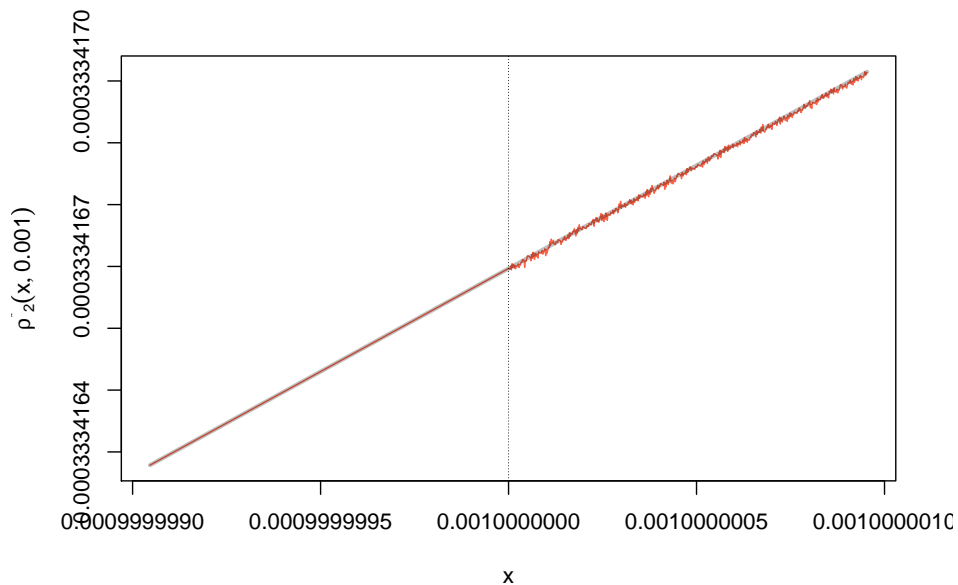
```
> cc <- 1e-4 ; op <- mult.fig(2, marP= -c(1,0,1,1))$old.par
> rhoAcurve(cc, 1e-6, 1e-3, log = "xy", col="tomato", ylab=""); myAxes(1:2)
```

```
> abline(v=cc, lty=3, lwd=1/2)
> ## zoom in extremely:
> rhoAcurve(cc, cc*(1-1e-4), cc*(1+1e-4), col="tomato")
> abline(v=cc, lty=3, lwd=1/2);      par(op)
```



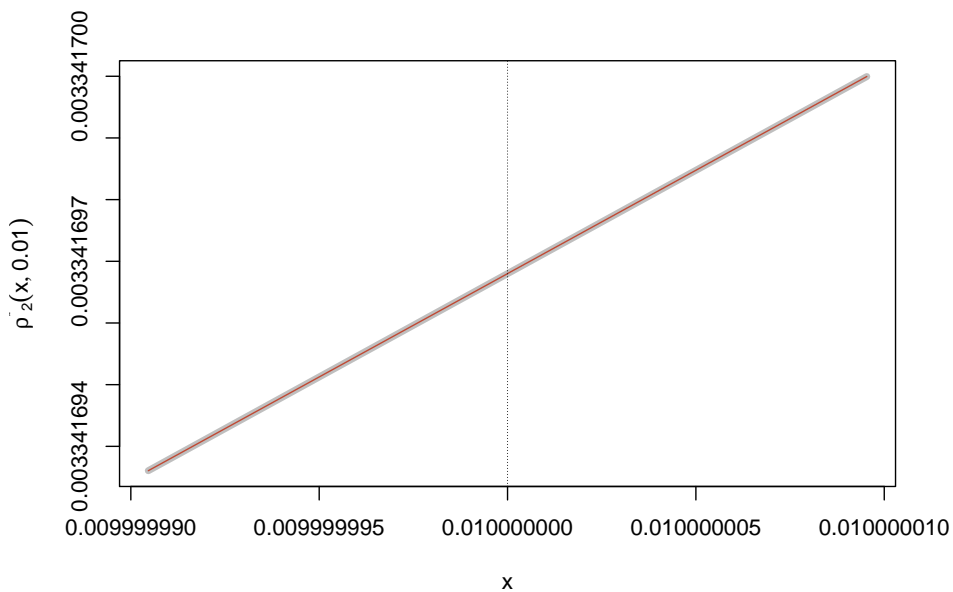
Still larger cutoff:

```
> cc <- 1e-3
> rhoAcurve(cc, cc*(1-2^-20), cc*(1+2^-20), log="y", yaxt="s", col="tomato")
> abline(v=cc, lty=3, lwd=1/2)
> rhoAcurve(cc*10, add=TRUE, col=adjustcolor(1,.25), lwd=3)
```



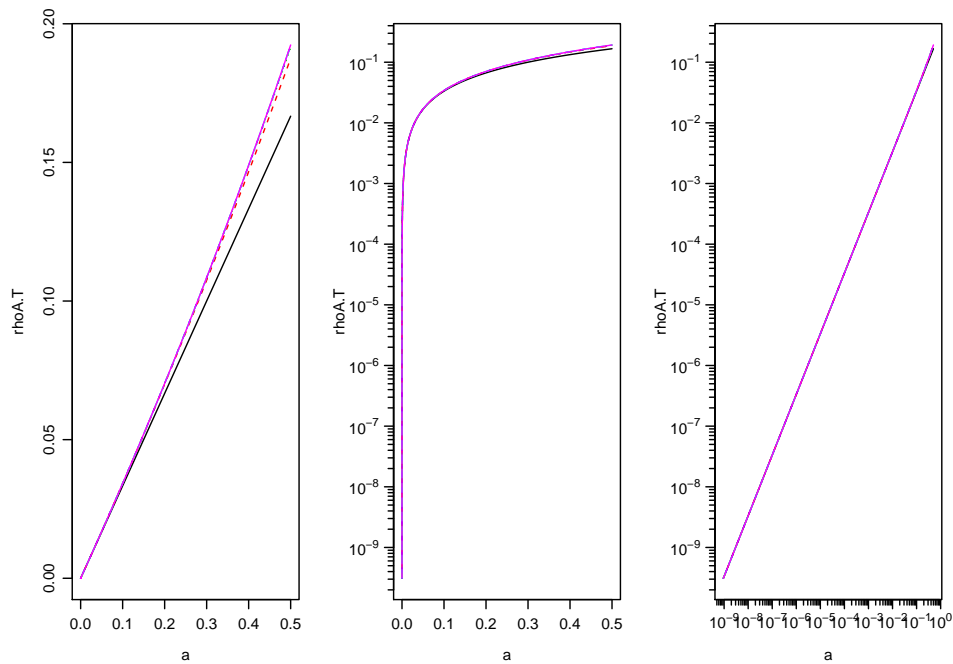
Still larger ...

```
> cc <- 0.01
> rhoAcurve(cc, cc*(1-2^-20), cc*(1+2^-20), log="y", yaxt="s", col="tomato")
> abline(v=cc, lty=3, lwd=1/2)
> rhoAcurve(cc*10, add=TRUE, col=adjustcolor(1,.25),lwd=5)
```



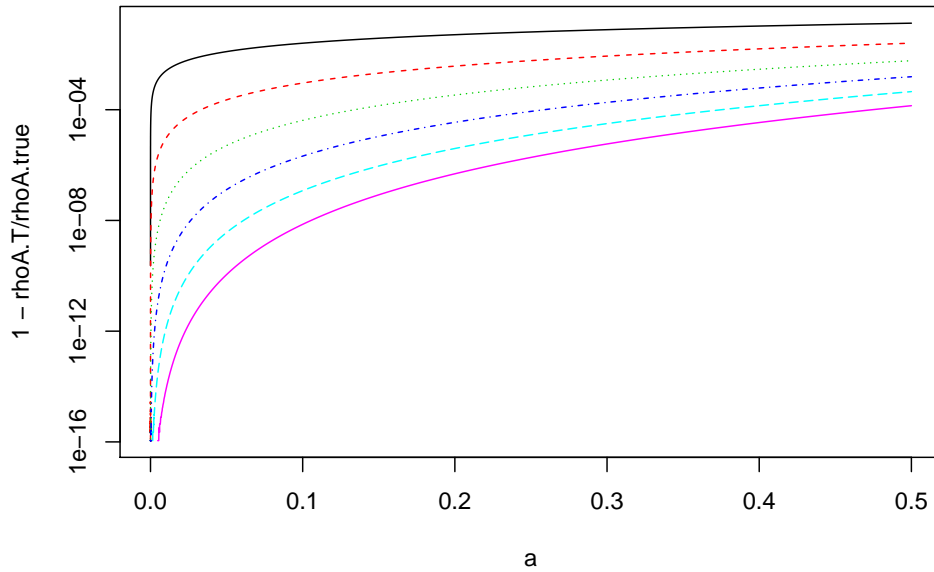
And “visibly”, it still seems perfect. This would suggest that a 4-terms approximation is to be preferred to the direct formula for  $|\theta| < 10^{-3}$ , possibly even  $|\theta| < 10^{-2}$ . We will determine the best  $k$ -terms series approximation for different cutoffs for  $k = 1, 2, 3, 4, 5$ , in the following. Looking at the series approximations (first order up to 6-th order) a first time,

```
> a <- 2^seq(-30,-1, by = 1/32)# 0 < a <= 0.5
> rhoA.T <- vapply(1:6, rhoAmh.T, a=a, numeric(length(a)))
> op <- mult.fig(mfcol=c(1,3), mgp=c(2.5,.8,0))$old.par
> matplot(a, rhoA.T, type="l")
> matplot(a, rhoA.T, type="l", log="y", yaxt="n") ; myAxes(2)
> matplot(a, rhoA.T, type="l", log="xy", axes=FALSE); myAxes(1:2);box()
> par(op)
```



Now, rather look at the *relative* approximation error of the different Taylor series approximations:

```
> rhoA.true <- rhoAmh.T(a,50)
> chk.w.mpfr <- FALSE ## Sys.info()[["user"]] == "maechler"
> if(chk.w.mpfr) {
  require(Rmpfr)## get the "really" "true" values:
  print(system.time(rhA.mp <- rhoAmh.T(mpfr(a, prec=256), 50))) ## 3.95 sec (lynne)
  print(system.time(rhA.mp1 <- rhoAmh.T(mpfr(a, prec=256), 60))) ## 4.54 sec
  stopifnot(all.equal(rhA.mp, rhoA.true, tol = 1e-15))
  print(all.equal(rhA.mp, rhoA.true, tol = 1e-20)) ## 6.99415....e-17 [64bit, lynne]
  ## see if the 50 terms have converged:
  print( all.equal(rhA.mp, rhA.mp1, tol = 1e-30) )
  ## "Mean relative difference: 2.4958....e-22"
  ## ==> 50 terms seem way enough for double prec
}
> matplot(a, 1 - rhoA.T / rhoA.true, type="l", log="y")
```



We rather provide a function for *visualizing* the relative approximation errors of the different Taylor series approximations in a flexible way:

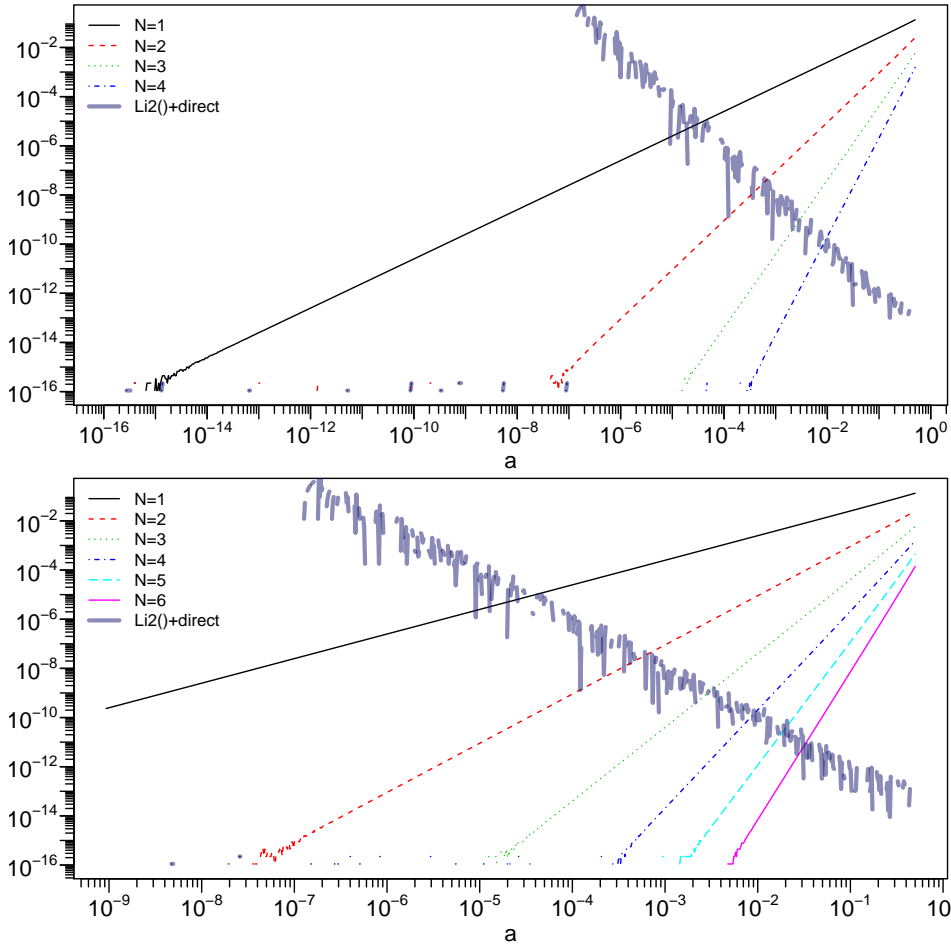
```
> pl.relE.rhoAMH <- function(N.max, N.inf = 50, N.min = 1, l2a.lim = c(-30, -1),
                             n.p.u = 2^round(log2(1000 / diff(l2a.lim))),
                             cut.rA2 = 1e-7,
                             colX = adjustcolor("midnightblue", 0.5), ...)
{
  stopifnot(length(l2a.lim) >= 2, l2a.lim < 0, n.p.u >= 1,
            N.max >= N.min, N.min >= 1, N.inf > N.max + 4,
            (N3 <- c(N.min, N.max, N.inf)) == as.integer(N3))
  a <- 2^seq(l2a.lim[1], l2a.lim[2], by = 1/n.p.u)
  N.s <- N.min:N.max
  rhoA.true <- rhoAmh.T(a, N.inf)
  rhoA.T <- vapply(N.s, rhoAmh.T, a=a, numeric(length(a))) # matrix
  rhoA.v2 <- Vectorize(.rhoAmh.2)(a, cut.rA2) # "Li2()+direct" below

  ## matplot() compatible colors and lty's
  cols <- palette()[1 + (N.s-1) %% 6]
  ltys <- (1:5) [1 + (N.s-1) %% 5]
  matplot(a, 1 - rhoA.T / rhoA.true, type="l", log="xy",
          col=cols, lty=ltys, axes=FALSE, frame=TRUE, ...)
  myAxes(1:2)
  lines(a, 1 - rhoA.v2 / rhoA.true, col= colX, lwd=3)
  legend("topleft", c(paste0("N=", N.s), "Li2()+direct"),
        col=c(cols, colX), lty=c(ltys, 1), lwd=c(rep(1, length(N.s)), 3),
        cex=.75, bty="n")
  invisible(list(a=a, rhoA.T=rhoA.T, rhoA.v2 = rhoA.v2))
}
```

Note that the “Li2()+direct” comparison is only for  $a = \theta > 10^{-7}$ , as that is used as cutoff per default,  $\text{cut.rA2} = 1e-7$ . And now look at the “very nice” pictures, using  $\text{l2a} = \log_2(a)$  to choose the range of  $a = \theta$ :

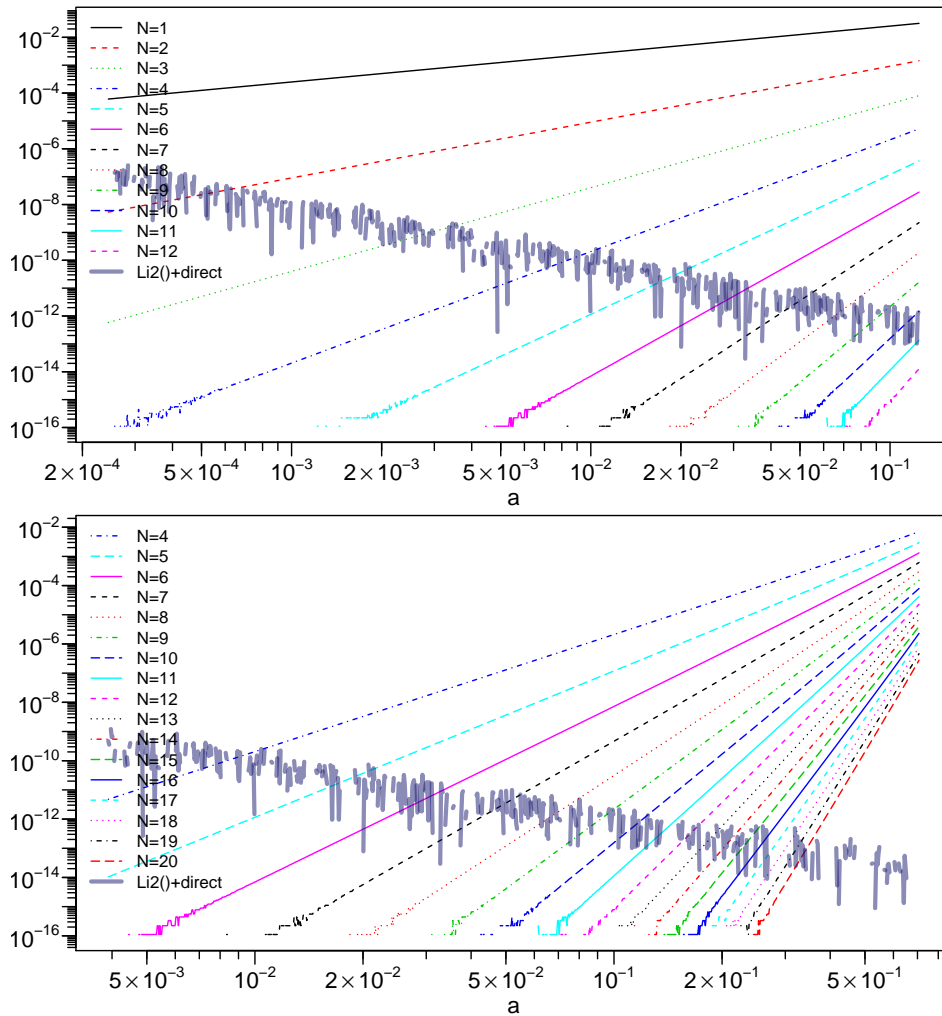
```
> op <- mult.fig(2, marP=-c(1.5, 1.5, 2, 1))$old.par
> pl.relE.rhoAMH(4, l2a=c(-53, -1), ylab="")
```

```
> pl.relE.rhoAMH(6, ylab="")
```



Successively zooming in “to the right”, to larger  $a$ , first, with range  $2^{-12} - 2^{-3}$ , and up to 12 terms, then zooming into range  $2^{-8} - 2^{-5}$ , and using 20,

```
> mult.fig(2, marP=-c(1.5,1.5,2,1))
> pl.relE.rhoAMH(12, l2a=c(-12, -3), ylab="")
> pl.relE.rhoAMH(20, l2a=c(-8, -.5), N.min = 4, ylab="")
```

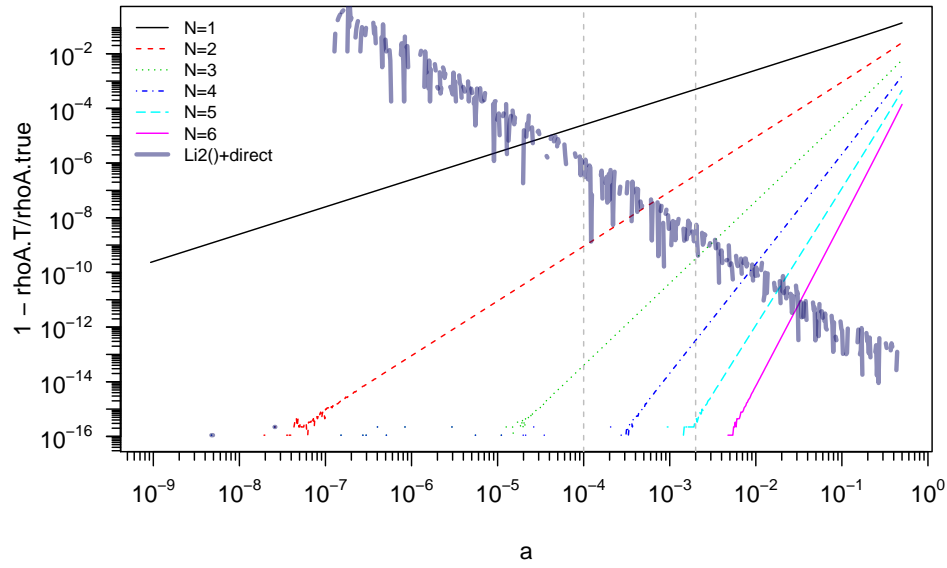


The next one is “just for fun”, to see if there is consistency when  $N \rightarrow N_\infty$ , i.e., our `N.inf = 50`, and not shown here:

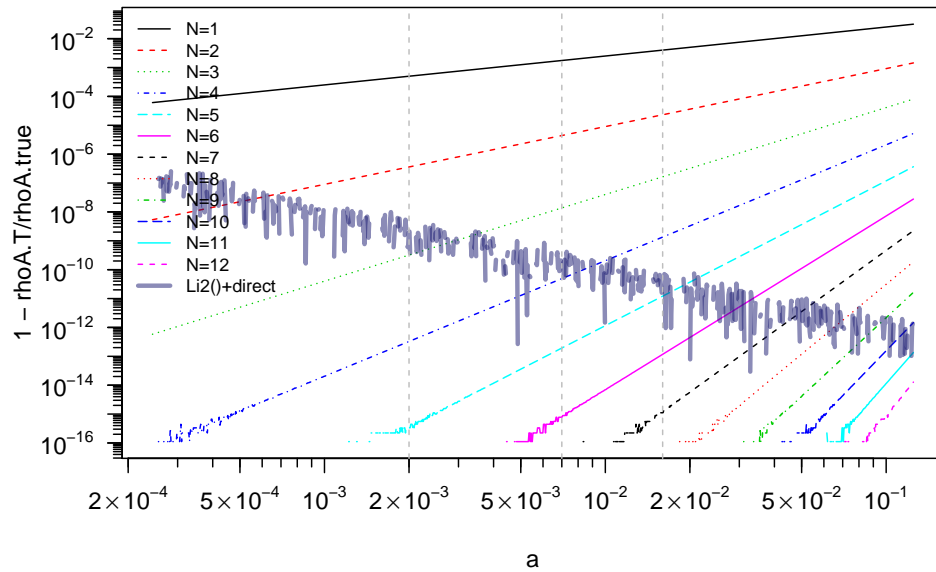
```
> par(op); pl.relE.rhoAMH(40, l2a=c(-5, -.5), N.min = 10)
```

The following plots are now used to read off the final cutoff used for the (hidden) `.rhoAmhCopula()` function in package **copula** which underlies `rho(amhCopula(.))`:

```
> pl.relE.rhoAMH(6)
> abline(v=1e-4, col="gray", lty=2)#-> N=2 cutoff
> abline(v=2e-3, col="gray", lty=2)#-> N=3 cutoff
```



```
> pl.relE.rhoAMH(12, l2a=c(-12, -3))
> abline(v= 2e-3, col="gray", lty=2)#-> N=3 cutoff
> abline(v= 7e-3, col="gray", lty=2)#-> N=4 cutoff
> abline(v=16e-3, col="gray", lty=2)#-> N=5 cutoff
```



Consequently, the implementation in **copula** is

```
> copula ::: .rhoAmhCopula
```

```
function (a)
{
  if (is.na(a))
    return(a)
  aa <- abs(a)
  if (aa < 7e-16)
    a/3
  else if (aa < 1e-04)
    a/3 * (1 + a/4)
  else if (aa < 0.002)
```

```

      a * (1/3 + a * (1/12 + a * 3/100))
else if (aa < 0.007)
  a * (1/3 + a * (1/12 + a * (3/100 + a/75)))
else if (aa < 0.016)
  a * (1/3 + a * (1/12 + a * (3/100 + a * (1/75 + a/147))))
else {
  3/a * (4 * (1 + 1/a) * dilog(a) - (if (a < 1)
    8 * (1/a - 1) * log1p(-a)
    else 0) - (a + 12))
}
}

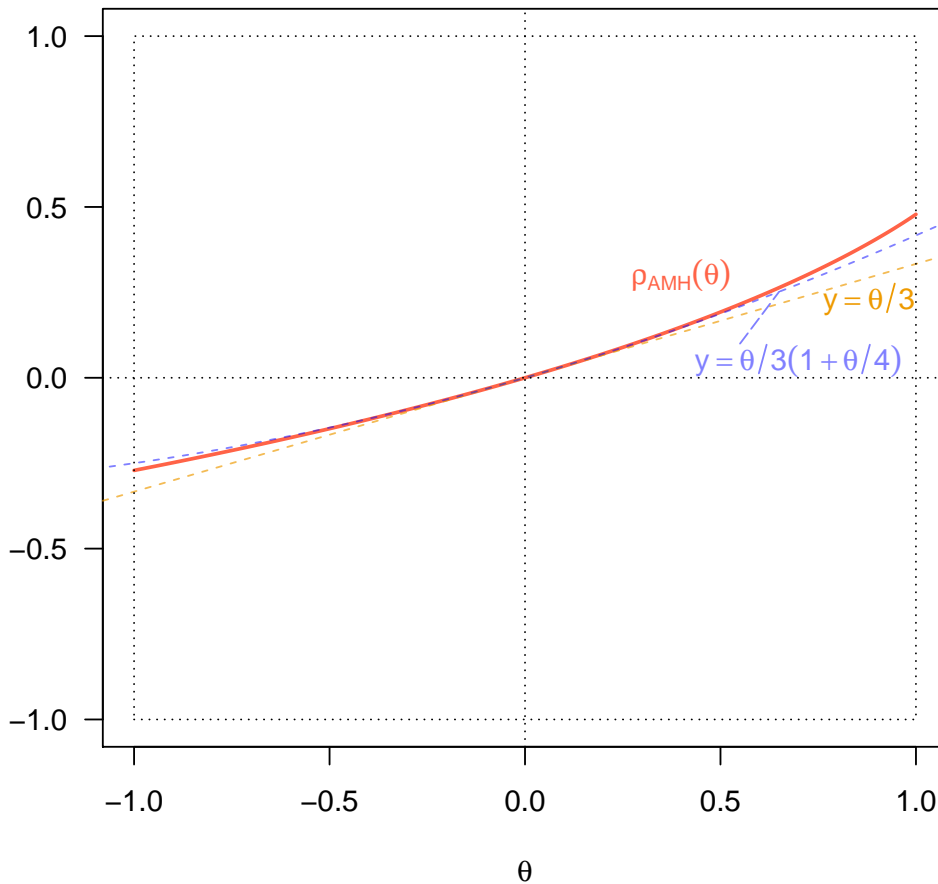
```

visualized on its full range  $[-1, 1]$ ,

```

> rhoAMH <- Vectorize(copula:::rhoAmhCopula)
> curve(rhoAMH, n=1025, -1, 1, ylim= c(-1,1), xlab = quote(theta),
  ylab="", col="tomato", lwd=2, las=1)
> abline(0, 1/3, lty=2, col=(adjustcolor(c2 <- "orange2", 2/3)))
> curve(x/3*(1+x/4), lty=2, col=(adjustcolor(c3 <- "blue", 1/2)),
  -1.1,1.1, add=TRUE); x. <- .65
> text(.4 , .3 , quote(rho[plain(AMH)](theta)),col="tomato")
> text(.88, .23, quote(y == theta/3), col=c2)
> text(.7, .05, quote(y == theta/3*(1+theta/4)), col=adjustcolor(c3, 1/2))
> segments(.55, .10, x., x./3*(1+x./4), lty="82", col=adjustcolor(c3, 1/2))
> abline(h=0,v=0, lty=3); rect(-1,-1,1,1, lty=3)

```



Finally, we may add some simple tests, that the **copula** package's `rho(<amhCopula>, *)` did not fulfill because of the notorious cancellations, previously. Note that in fact, we are only looking at very small (positive)  $\theta$ , and checking that already the *first* two order series approximations,

$$\rho_{\text{AMH}}(\theta) \approx \frac{\theta}{3} \left(1 + \frac{\theta}{4}\right) \approx \theta/3 \quad (13)$$

are all already good approximations or very accurate, depending on  $|\theta|$ :

```
> t0 <- seq(-1,1, by=2^-8)[1:512]
> t1 <- seq(-1/2, 1/2, by = 2^-8)
> th <- 10^-(6:99); i <- -(1:9)
> rth <- rhoAMH(th)
> stopifnot(all.equal(rhoAMH(1), 4*pi^2 - 39, tol = 8e-15), # <- gave NaN
  all.equal(rhoAMH(t0), t0/3 * (1 + t0/4), tol = 0.06),
  all.equal(rhoAMH(t1), t1/3 * (1 + t1/4), tol = 1/85),
  all.equal(rth, th / 3 * (1 + th/4), tol = 1e-15),
  all.equal(rth, th / 3, tol = 1e-6),
  all.equal(rth[i], th[i] / 3, tol = 6e-16))
> th <- 10^-(16:307)
> stopifnot(all.equal(th/3, rhoAMH(th), tol=4e-16),
  rho(amhCopula(0, use.indepC="FALSE")) == 0)
```

## Session Information

```
> toLatex(sessionInfo(), locale=FALSE)
```

- R version 3.5.2 Patched (2019-01-02 r75945), x86\_64-pc-linux-gnu
- Running under: Debian GNU/Linux buster/sid
- Matrix products: default
- BLAS: /srv/R/R-patched/build.19-01-03/lib/libRblas.so
- LAPACK: /srv/R/R-patched/build.19-01-03/lib/libRlapack.so
- Base packages: base, datasets, grDevices, graphics, grid, methods, parallel, splines, stats, stats4, tools, utils
- Other packages: Rmpfr 0.7-1, VGAM 1.0-6, abind 1.4-5, bbmle 1.0.20, copula 0.999-20, gmp 0.5-13.2, gridExtra 2.3, gsl 1.9-10.3, lattice 0.20-38, mev 1.11, qrng 0.0-5, randtoolbox 1.17.1, rngWELL 0.10-5, rugarch 1.4-0, sfsmisc 1.1-3
- Loaded via a namespace (and not attached): ADGofTest 0.3, DistributionUtils 0.6-0, GeneralizedHyperbolic 0.8-4, KernSmooth 2.23-15, MASS 7.3-51.1, Matrix 1.2-15, R6 2.3.0, Rcpp 1.0.0, Rsolnp 1.16, Runuran 0.24, SkewHyperbolic 0.4-0, assertthat 0.2.0, bayesplot 1.6.0, bindr 0.1.1, bindrcpp 0.2.2, boot 1.3-20, colorspace 1.3-2, compiler 3.5.2, crayon 1.3.4, digest 0.6.18, dplyr 0.7.8, evaluate 0.12, evd 2.3-3, expm 0.999-3, ggplot2 3.1.0, ggribes 0.5.1, glue 1.3.0, gmm 1.6-2, gtable 0.2.0, htmltools 0.3.6, ismev 1.42, knitr 1.21, ks 1.11.3, lazyeval 0.2.1, magrittr 1.5, mclust 5.4.2, mgcv 1.8-26,

munsell 0.5.0, mvtnorm 1.0-8, nleqslv 3.3.2, nlme 3.1-137, nloptr 1.2.1,  
 numDeriv 2016.8-1, partitions 1.9-19, pcaPP 1.9-73, pillar 1.3.1, pkgconfig 2.0.2,  
 plyr 1.8.4, polynom 1.3-9, pspline 1.0-18, purrr 0.2.5, revdbayes 1.3.2, rlang 0.3.0.1,  
 rmarkdown 1.11, rootSolve 1.7, sandwich 2.5-0, scales 1.0.0, spd 2.0-1, stabledist 0.7-1,  
 stringi 1.2.4, stringr 1.3.1, tibble 2.0.0, tidyselect 0.2.5, truncnorm 1.0-8, xfun 0.4,  
 xts 0.11-2, yaml 2.2.0, zoo 1.8-4

```
> my.strsplit( packageDescription("copula")$Date )
```

```
NA-- 2019-01-06
```

## References

- Hofert M, Maechler M (2011). “Nested Archimedean Copulas Meet R: The nacopula Package.” *Journal of Statistical Software*, **39**(9), 1–20. ISSN 1548-7660. URL <http://www.jstatsoft.org/v39/i09>.
- Joe H (1997). *Multivariate Models and Dependence Concepts*. Chapman & Hall/CRC.
- Nelsen RB (2007). *An Introduction to Copulas*. 2nd edition. Springer-Verlag, New York.
- Sklar A (1959). “Fonctions de Répartition à  $n$  Dimensions et Leurs Marges.” *Publications de L’Institut de Statistique de L’Université de Paris*, **8**, 229–231.

### Affiliation:

Martin Mächler  
 Seminar für Statistik, HG G 16  
 ETH Zurich  
 8092 Zurich, Switzerland  
 E-mail: [maechler@stat.math.ethz.ch](mailto:maechler@stat.math.ethz.ch)  
 URL: <http://stat.ethz.ch/people/maechler>