Polynomial Opt and mOpt Rho Functions in RobStatTM

April 4, 2023

R. Douglas Martin, Victor J. Yohai and Matias Salibian-Barrera

The Opt and mOpt Robust Regression Estimators

A regression M-estimator $\hat{\boldsymbol{\theta}}_M$ minimizes the objective function

$$\sum_{i=1}^{N} \rho\left(\frac{\hat{\epsilon}_i(\boldsymbol{\theta})}{\hat{s}}\right) \tag{1}$$

where $\hat{\epsilon}_i(\boldsymbol{\theta}) = \mathrm{r}_i - \mathbf{x}_i'\boldsymbol{\theta}$ are regression residuals, $\rho(x)$ is a symmetric loss function, and $\hat{s}$ is a robust scale estimate computed prior to the minimization. The value $\hat{\boldsymbol{\theta}}_M$ which minimmizes the objective (1) satisfies the equation

$$\sum_{i=1}^{N} \mathbf{x}_i \psi\left(\frac{\hat{\epsilon}_i(\hat{\boldsymbol{\theta}}_M)}{\hat{s}}\right) = \mathbf{0} \tag{2}$$

where $\psi(x) = \rho'(x)$. Using a high breakdown point but low efficiency intial estimate $\hat{\boldsymbol{\theta}}_M^0$, the above equation is solved using an iteratively reweighted least squares (IRWLS) algorithm, based on a weight function $w(x) = \psi(x)/x$. The functions $\rho(x)$ and $\psi(x)$ are referred to as *rho* and *psi* functions, respectively. Related details are available in Chapter 5 of Maronna et al. (2019).

The `RobStatTM` function `lmrobdetMM`, through its argument `control = lmrobdet.control`, allows the user to choose one of two optimal rho functions, an *optimal* rho function $\rho_{\mathrm{opt}}(x)$ and a *modified optimal* rho function $\rho_{\mathrm{mopt}}(x)$, each of which depend on a tuning conast c that controls the normal distribution efficiency of the regression estimator. The rho function $\rho_{\mathrm{opt}}(x)$ is optimal in the sense that in minimizes the maximum bias due to outliers, subject to a specified normal distribution efficiency.[1] The rho function $\rho_{\mathrm{mopt}}(x)$ is a very slight modification of $\rho_{\mathrm{opt}}(x)$, which is made to insure convergence of the IRWLS algorithm, but retains the robustness toward outliers of $\rho_{\mathrm{opt}}(x)$. The formula for the rho function $\rho_{\mathrm{opt}}(x)$ is provided in the `RobStatTM` companion Vignette "Optimal Bias Robust Regression Psi and Rho Revisited", where the formulas for the psi functions $\psi_{\mathrm{opt}}(x)$ and $\psi_{\mathrm{mopt}}(x)$ are provided. In that Vignette, it is explained that the rho function obtained by integration of the psi function is an *error function* (erf), for which a numerical approximation is needed. The latter was implemented with R code in the package `pracma` by Hans W. Borchers, who based his R code on a Fortran algorithm in Zhang and Jin (1996). For use in `lmrobdetMM`, a C implentation of the erf approximation was developed by Kjell Konis.

The Polynomial Versions of the Opt and mOpt Rho Functions

For the sake of improvements in speed and transparency, that C code erf approximation for obtaining the optimal rho functions $\rho_{\mathrm{opt}}(x)$ and $\rho_{\mathrm{mopt}}(x)$ has been replaced with accurate polynomial versions. However, we retained the original C code erf versions in the V0 versions $\rho_{\mathrm{optV0}}(x)$ and $\rho_{\mathrm{moptV0}}(x)$ for users who

[1] See Section 5.8.1 of Maronna et al. (2019) for details. The main idea is also sketched in the open source article Martin and Xia (2022) that may be downloaded fom the Publisher's website.

wish to check the performance of the new polynomial forms of $\rho_{\text{opt}}(x)$ and $\rho_{\text{mopt}}(x)$ against the original erf approximation versions.

It is quite common to use a robust estimator with a 0.95 (95%) normal distribution efficiency, which is often the default efficiency in software package implementations. Figure 1 displays the 0.95 (95%) normal distribution efficiency $\rho_{\text{opt}}(x)$ and $\psi_{\text{opt}}(x)$ original C code analytic functions as the solid curved lines, and the polynomial versions as overlaid dots. The polynomial version is virtually the same as the erf version, ,but is much faster.

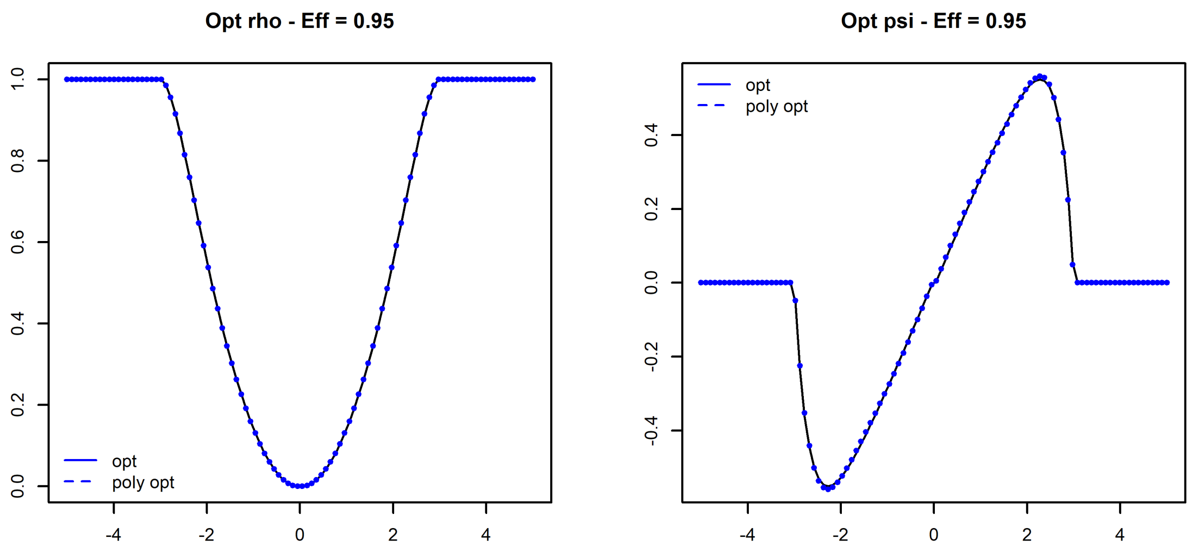

Figure 1: Optimal Rho and Psi for 95% Normal Distribution Efficiency

Figure 2 displays the 0.95 (95%) normal distribution $\rho_{\text{mopt}}(x)$ and $\psi_{\text{mopt}}(x)$ with C code analytic modified functions as the solid curved lines, and the polynomial versions as overlaid dots. Again, the two versions are virtually the same.

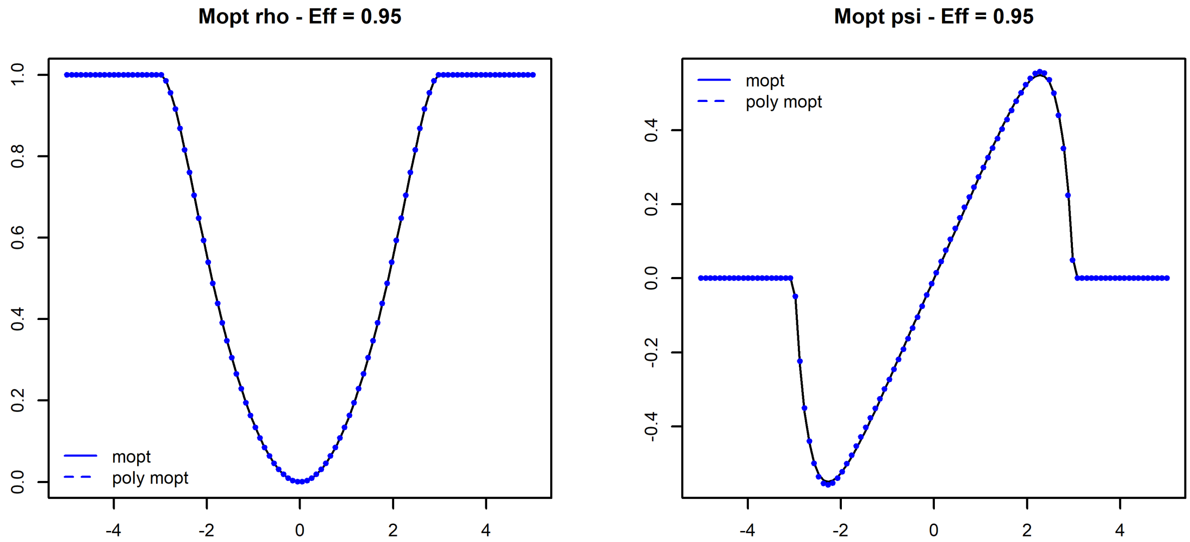

Figure 2: Modified Optimal Rho and Psi for 95% Normal Distribution Efficiency

The Polynomial Fits to the erf Functions

The polynomial representation of each of the erf bassed rho and psi functions is obtained by fitting an odd polynomial of degree 7 to the psi function, and then integrating to get an even polynomial of degree 8 representation of the rho function. The odd polynomial of degree 7 representation of each of the two psi functions is obtained by a least squares fit of the polynomial to the erf versions of $\psi_{\text{opt}}(x)$ and $\psi_{\text{mopt}}(x)$. Keep in mind that the rho functions and psi functions depend upon their normal distribution efficienies, which is controlled by a tuning parameter c, and so the coefficients of the polynomials will depend on the normal distribution efficiencies of the opt and mopt regression estimators. The function `polyapproxpar(eff, rhoname)`, where `eff` is the normal distribution efficiency and `rhoname` is either `opt` or `mopt`, computes these coefficients as illustrated in the following code for `eff = 0.95` and `rhoname = opt`.

```
library(RobStatTM)
source('polyapprox.R')
eff <- 0.95
poly <- polyapproxpar(eff, 'opt')
round(poly$coef,4)
```

[1] -0.0104 0.3158 -0.0275 0.0078 -0.0010

```
poly$a
```

[1] 0.03305454

```
poly$b
```

[1] 3.003281

```
poly$u1
```

[1] -0.0001724852

```
poly$u2
```

[1] 0.9996474

Note that the above coefficients are the constant and the coeffients of x^1, x^3,x^5, x^7 for the polynomial for $\psi_{\text{opt}}(x)$, and the integral of this polynomial is the 8th degree polynomial for $\rho_{\text{opt}}(x)$. The constant a defines the interval $[-a, +a]$ within which $\psi_{\text{opt}}(x) = 0$ and $\rho_{\text{opt}}(x) = 0$, which is apparent in Figure 1. The constant b defines the interval $[-b, +b]$ outside of which $\psi_{\text{opt}}(x) = 0$ and $\rho_{\text{opt}}(x)$ is constant, which is also apparent in Figure 1. The value u1 is the value of the $\rho_{\text{opt}}(x)$ polynomial at $x = \pm a$, and u2 is the value of that polynomial at $x = \pm b$.

However, users will sometimes want to use a lower normal distribution efficiency than (0.95) 95% in order to obtain more robustness toward outliers. We now show Figures 3 and 4 what the opt and mopt rho and psi functions look like for 0.65 (65%) normal distribution efficiency, and these are displayed in .

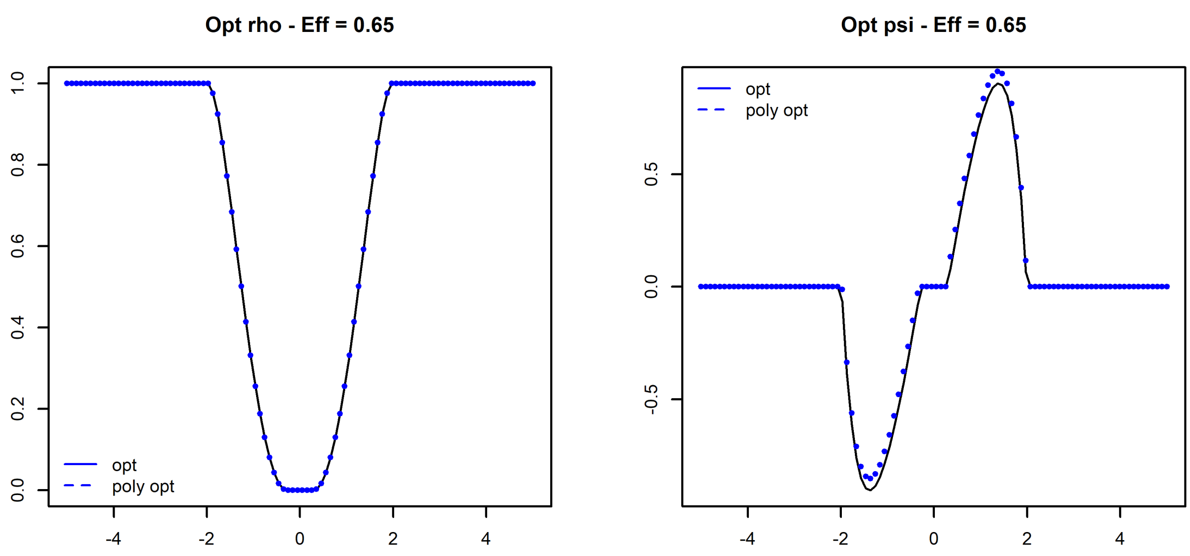

Figure 3: Optimal Rho and Psi for 65% Normal Distribution Efficiency

What is striking about Figure 3 is the curious flat spot where the psi function is equal to zero on the interval $[-a, +a]$ around the origin, and since the the rho function is the integral of the psi function, the rho function is also to zero on that interval.

It is interesting to take a look at the polynomial coefficients and related parameters for the 65% normal distribution function opt function, using the same `polyapproxpar` function as above. The polynomial values for the 65% efficient opt estimator are:

```
library(RobStatTM)
source('polyapprox.R')
eff <- 0.65
poly <- polyapproxpar(eff, 'opt')
round(poly$coef,4)

## [1] -0.3668  1.2891 -0.1843  0.0142 -0.0097

poly$a

## [1] 0.2879384

poly$b

## [1] 1.986329

poly$u1

## [1] -0.05249455

poly$u2

## [1] 0.9473982
```

Note that our earlier use of the above code for the 95% efficient opt psi revealed the very tiny interval $[-0.033, +0.033]$ around the origin where the psi and rho values are equal to zero, and in a second close look at Figure 1 one can see a hint at this fact in the psi function. The difference in the values of a in the intervals $[-a, +a]$ for the 95% and 65% versions of $\rho_{\text{opt}}(x)$ reflects the general behavior of the interval that its length increases as the normal distribution efficieny decreases. Note that for $\rho_{\text{mopt}}(x)$ and $\psi_{mopt}(x)$ in Figure 4 there is no such flat spot.

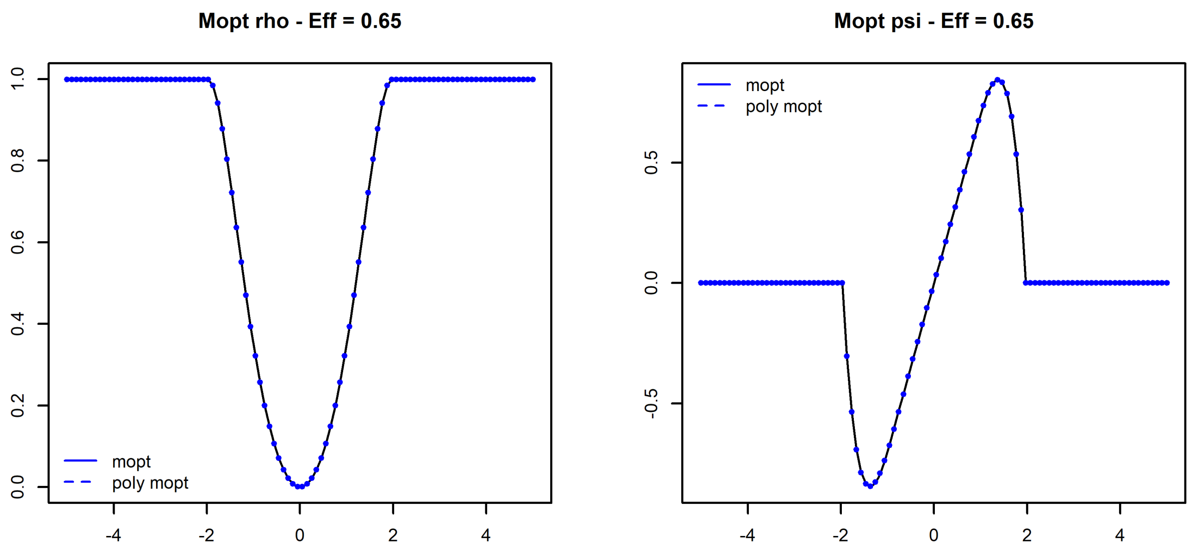

Figure 4: Modified Optimal Rho and Psi for 65% Normal Distribution Efficiency

Furthermore, using `polyapproxpar` for the 65% efficient mopt estimator confirms that there is no flat spot:

```
library(RobStatTM)
source('polyapprox.R')
eff <- 0.65
poly <- polyapproxpar(eff, 'mopt')
round(poly$coef,4)

## [1]  0.0000  0.6783  0.0781 -0.0532 -0.0033

poly$a

## [1] 0

poly$b

## [1] 1.963787

poly$u1

## [1] 0

poly$u2

## [1] 0.9991142
```

The Difficulty Caused by the opt Estimator Psi Flat Spot

It turns out that this flat spot results in a weight function $w(x) = \psi(x)/x$ that is equal to zero on $[-a, +a]$, then increases for a while as $|x|$ increases, and then decreases. This turns out to cause problems for the IRWLS algorithm, which requires a weight function that is non-inreasing as $|x|$ increases from zero. That is the reason for creating the modified psi and rho functions which do not have such a flat spot. For further details concerning the effect of the flat spot on the weight function, and the method of contructing the mopt psi and rho functions, see the companion Vignette "Optimal Bias Robust Regression Psi and Rho Revisited".

References

Maronna, R. A. et al. (2019). *Robust Statistics: Theory and Methods (with R)*. 2nd ed. John Wiley & Sons.

Martin, R. D. and Xia, D. Z. (Mar. 2022). "Efficient Bias robust Regression for Time series Factor Models". In: *Journal of Asset Management*, pp. 1–20. ISSN: 1470-8272. DOI: `10.1057/s41260-022-00258-0`. URL: `https://link.springer.com/content/pdf/10.1057/s41260-%20022-00258-0.pdf`.

Zhang, S. and Jin, J. (1996). *Computation of Special Functions*. New York: John Wiley & Sons, Inc. ISBN: 0-471-11963-6.