How to use the tcG package

Introduction

This package contains all the functions needed to fit the models presented in Boutigny et al. (to be published)[1].

The main functions of the package are `tcG.fit`, `extGP.fit` that are used to fit the models and `res.plot` can be used to draw plots. You can check the documentation pages.

```
# install: change the character with the path to your zip file
# devtools::install("path/tcG-main")

# load
library(tcG)
```

```
?tcG.fit
?extGP.fit
?res.plot
```

Two types of models can be fitted with this package, all aiming at modeling precipitation which we will note Y.

Meta-Gaussian models

A meta-Gaussian model can be written as

$$Y = 0 * \mathbb{I}_{X<0} + \psi(X) * \mathbb{I}_{X \geq 0}, \text{ with } X \sim \mathcal{N}(\mu, 1)$$

where $\mathbb{I}$ is the indicator function equals to 1 if the condition is true and 0 else. μ is the parameter that controls the probability of dry measurement. The transformation ψ is called the anamorphosis, and 4 options are available in the package:

- `gp` (for Generalized Pareto): $\psi(x) = y_m + \sigma x^{\frac{1}{\alpha}} \exp \frac{\xi x^2}{2}$
- `power`: $\psi(x) = y_m + \sigma x^{\frac{1}{\alpha}}$
- `quadratic-power`: $\psi(x) = y_m + \sigma_1 x^{\frac{1}{\alpha}} + \sigma_2 x^{\frac{2}{\alpha}}$
- `power-exp`: $\psi(x) = \sigma_2(\exp(\sigma_1 x^{1/\alpha}) - 1)$

y_m is the minimal value that can be observed.

The choice of the anamorphosis will always be controlled by the argument `name`.

[1] Boutigny M, Ailliot P, Chaubet A, Naveau P, Saussol B (to be published). "Modelling rainfall from sub-hourly to daily scale with a heavy tailed meta-Gaussian model." *Water Resources Research*.

Extended Generalized Pareto model

This model is the one of Naveau et al. (2016)[2]. It models only the positive part of the distribution and can be written as

$$Y_+ = y_m + \sigma H_\xi^{-1}(U^{1/\alpha})$$

where $U \sim Unif(0,1)$, H_ξ is the cdf of a GPD and y_m is the minimal value that can be observed.

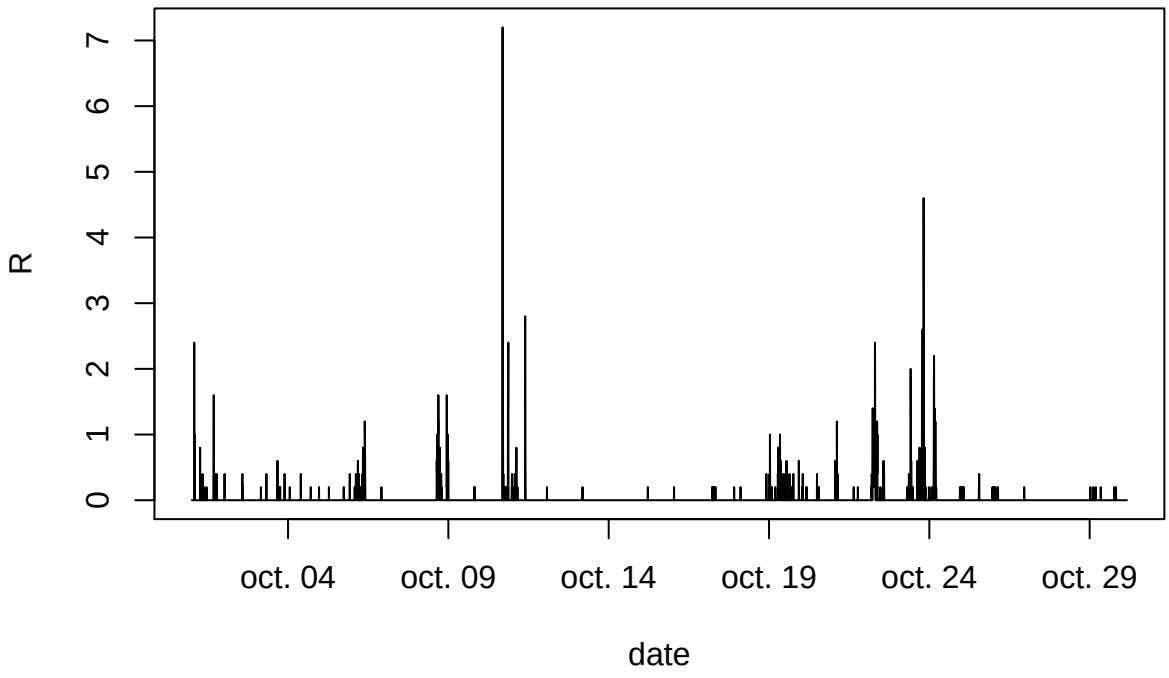
Data

Load the package and the provided data set.

```
data(guip)

# using only October to avoid long calculations
guip=guip[lubridate::month(guip$date)==10,]

plot(guip[1:7000,], type="l")
```



[2]Naveau P, Huser R, Ribereau P, Hannart A (2016). "Modeling jointly low, moderate, and heavy rainfall intensities without a threshold selection." *Water Resources Research*, **52**(4), 2753–2769.

Fitting meta-Gaussian models

First fit

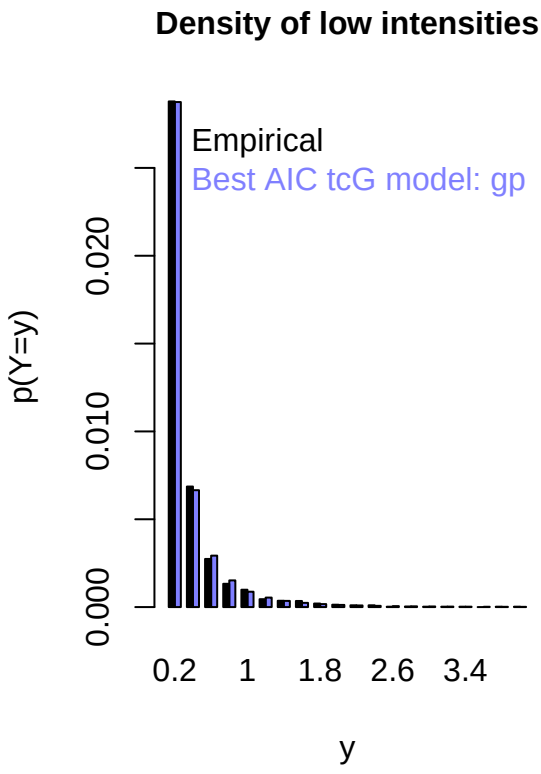
As a first example the `gp` transformation will be fitted. The argument `name` is a vector containing the names of the transformations that I want to fit. `init` is a list, named according to those transformations, which contains initial values for the parameters:

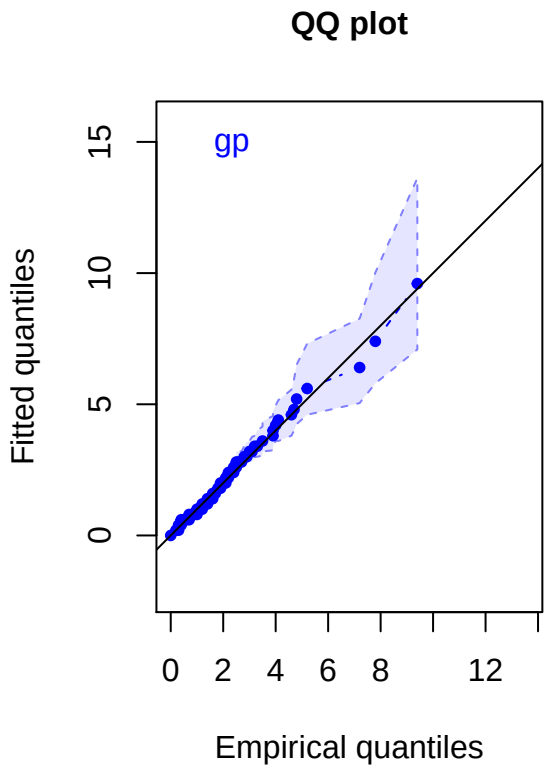
```
init=list("power"=c(mu, sigma, alpha),
          "gp"=c(mu, sigma, alpha, xi),
          "power-exp"=c(mu, sigma1, sigma2, alpha),
          "quadratic-power"=c(mu, sigma1, sigma2, alpha))
```

y_m is the minimal value that can be observed and *step* is the precision of the data. Finally `R` and `bootstrap` control the bootstrap replicates.

```
sort(unique(guip$R))
#>  [1] 0.0 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9
#> [20] 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1 3.2 3.3 3.5 3.9 4.0 4.1 4.6
#> [39] 4.7 4.8 5.2 7.2 7.8 9.4

res=tcG.fit(guip$R, name="gp", init=list("gp"=c(-1,0.5,0.5,0.5)), ym=0.2, step=0.2, R=50)
#> [1] "gp"
```





```
# fitted parameters
res$par
#> $gp
#> [1] -1.7207684  0.5829441  0.6563083  0.4389557
```

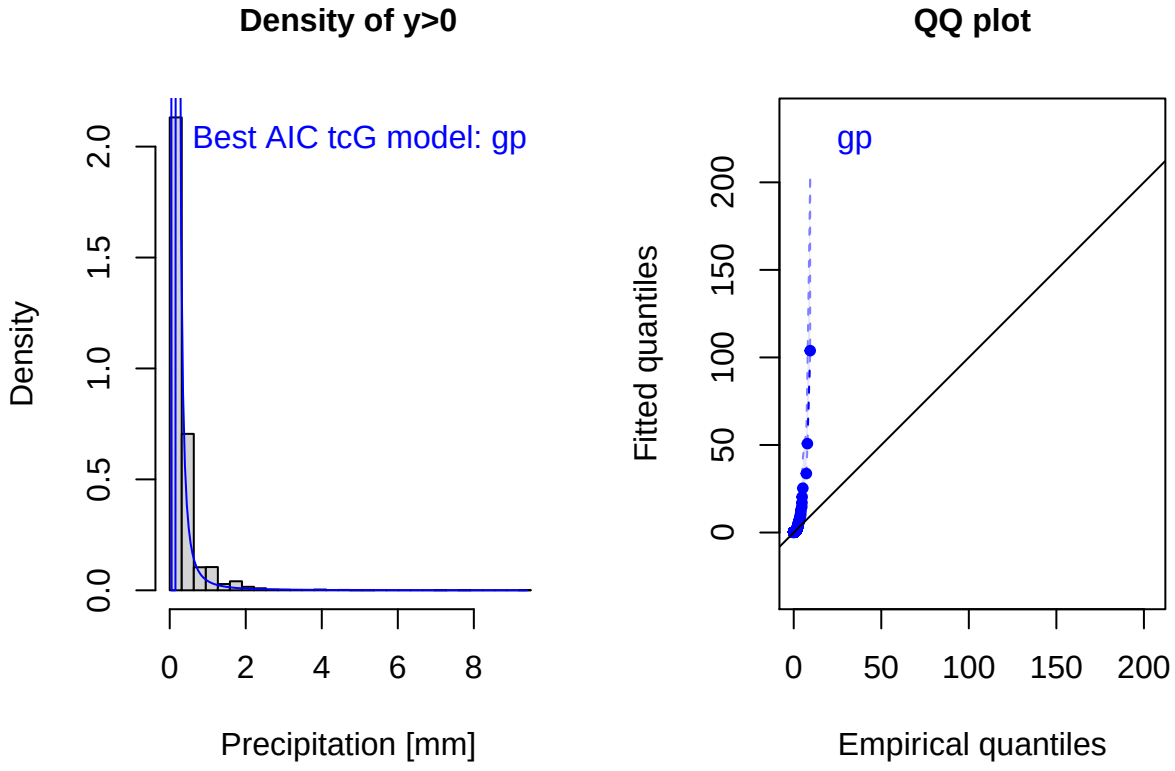
The default results include a plot for low intensities with a barplot zoomed on the 20 first steps, and a quantile-quantile plot that gives a better view of the global fit. The light area on the qqplot give the 95% interval computed with the bootstrap replicates.

The impact of discretization

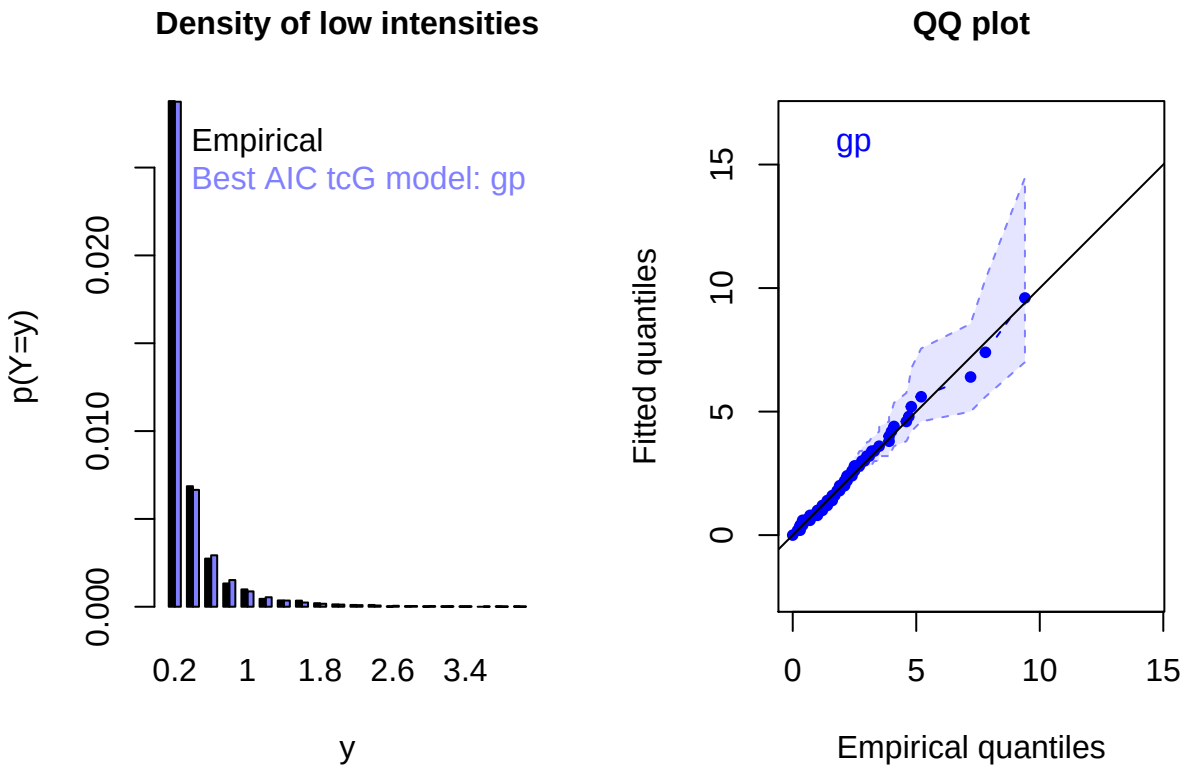
Taking into account the discretization of precipitation is especially important at fine time scales. In the package there are two options to fit the models: the continuous likelihood and the discrete one. Choosing one is made by inquiring the precision of the data. If `step==0` (default) the continuous likelihood is used, else it is the discrete one.

```
sort(unique(guip$R))
#>  [1] 0.0 0.2 0.3 0.4 0.5 0.6 0.7 0.8 0.9 1.0 1.1 1.2 1.3 1.4 1.5 1.6 1.7 1.8 1.9
#> [20] 2.0 2.1 2.2 2.3 2.4 2.5 2.6 2.7 2.8 2.9 3.0 3.1 3.2 3.3 3.5 3.9 4.0 4.1 4.6
#> [39] 4.7 4.8 5.2 7.2 7.8 9.4

res_cont=tcG.fit(guip$R, name="gp", init=list("gp"=c(-1,0.5,0.5,0.5)), ym=0, step=0, R=50)
#> [1] "gp"
```

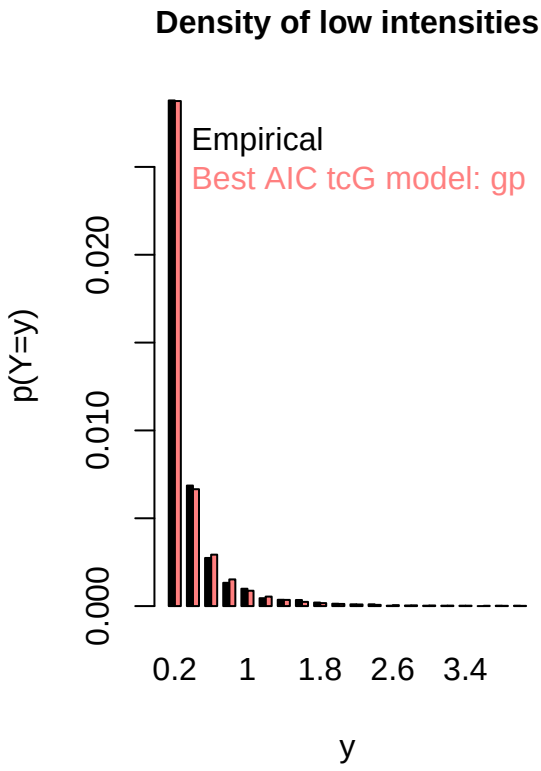


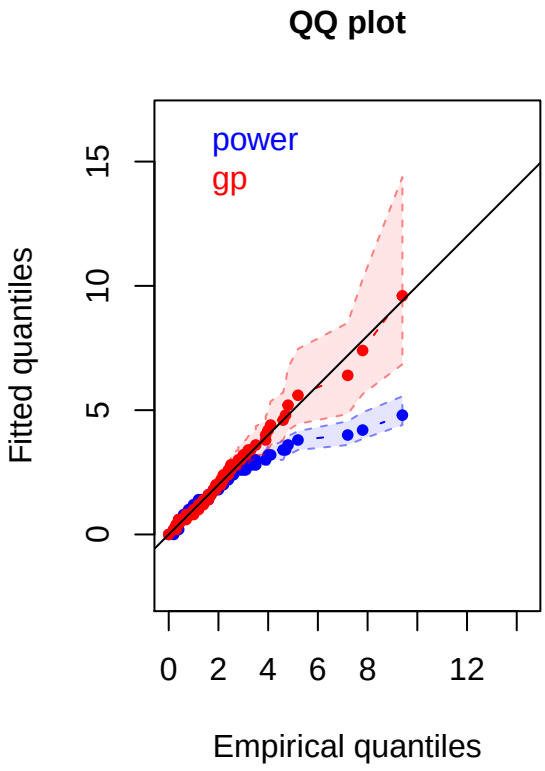
```
res_dis=tcG.fit(guip$R, name="gp", init=list("gp"=c(-1,0.5,0.5,0.5)), ym=0.2, step=0.2, R=50)
#> [1] "gp"
```



Comparing several meta-Gaussian models

```
res=tcG.fit(guip$R, name=c("power", "gp"),
            init=list("power"=c(-1,1,2), "gp"=c(-1,.5,0.5,0.5)),
            ym=.2, step=.2, R=50)
#> [1] "power"
#> [1] "gp"
```



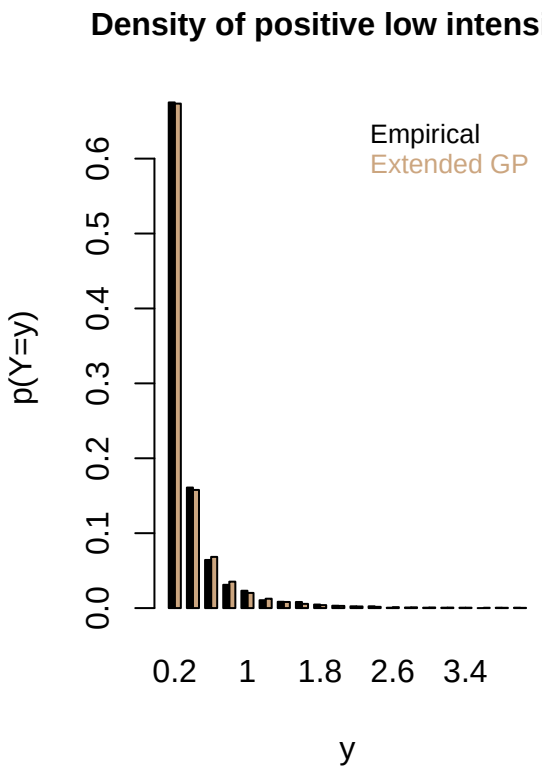


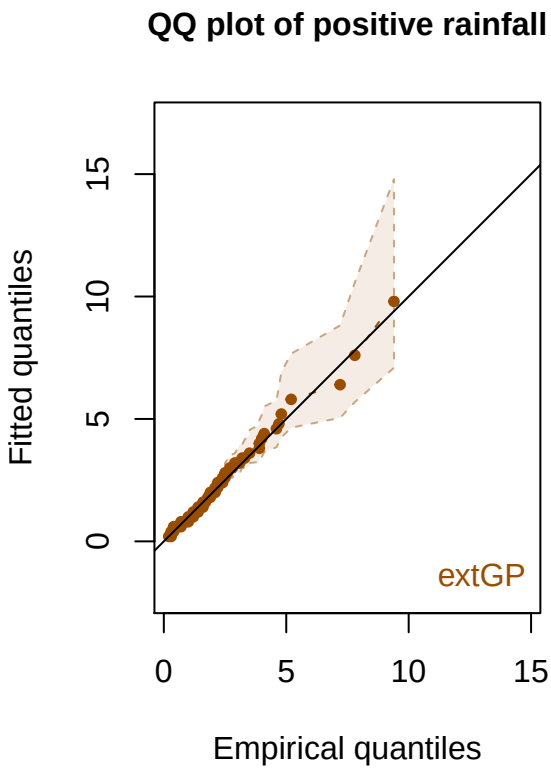

```
# parameters
res$par
#> $power
#> [1] -1.720982  0.794918  1.930151
#>
#> $gp
#> [1] -1.7207684  0.5829441  0.6563083  0.4389557

# AIC
which.min(res$AIC)
#> gp
#>  2
```

Fitting the extended GP model

```
# uniquement sur les mesures positives
res=extGP.fit(guip$R[guip$R>0], init=c(.5,.5,.5), ym=0.2, step=0.2, R=50)
```





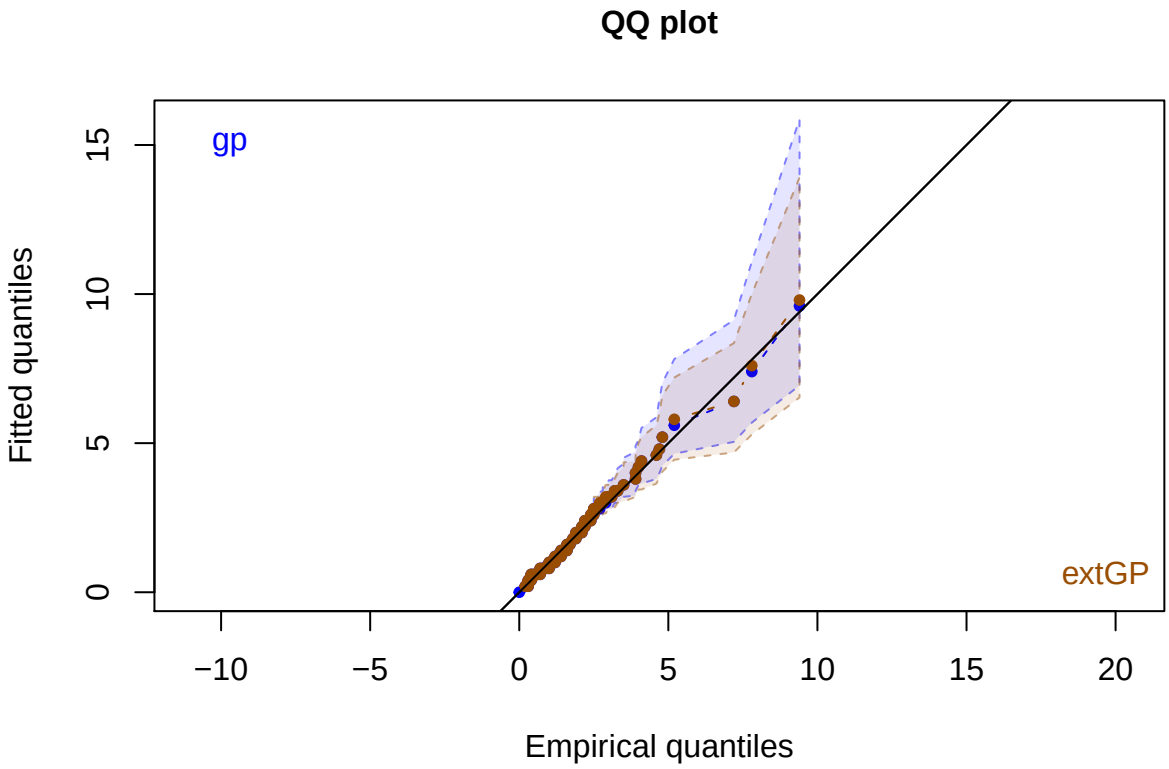

```
res$par
#> [1] 0.2205567 0.6481637 0.3630870
```

Comparing meta-Gaussian and extended GP

We're not going to use the default plots, but the `res.plot` function.

```
res_tcG=tcG.fit(guip$R, name="gp", init=list("gp"=c(-1,.5,0.5,0.5)), ym=0.2, step=0.2, R=50, plots=FALSE
#> [1] "gp"
res_eGP=extGP.fit(guip$R[guip$R>0], init=c(.5,0.5,0.5), ym=0.2, step=0.2, R=50, plots=FALSE)

# plot
res.plot(res.tcG=res_tcG, res.extGP=res_eGP, y=guip$R, ym=0.2, step=0.2, choice="qqplot")
```


QQ plot
gp
extGP
Fitted quantiles
Empirical quantiles
0
5
10
15
−10
−5
0
5
10
15
20