universität innsbruck

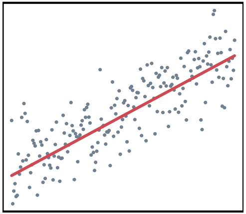
Probability Distribution Forecasts: Learning with Random Forests and Graphical Assessment

Moritz N. Lang, Reto Stauffer, Lisa Schlosser, Achim Zeileis

https://topmodels.R-Forge.R-project.org/

Motivation

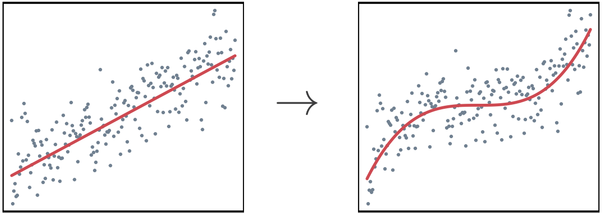
Motivation



LM, GLM

`lm`
`glm`

Motivation



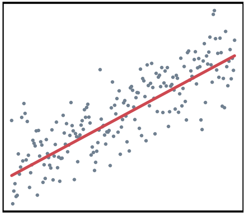
LM, GLM

GAM

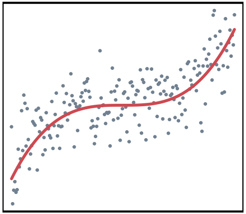
`lm`
`glm`

`mgcv`
`VGAM`

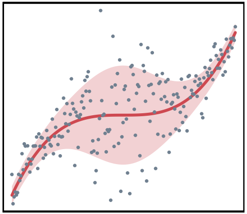
Motivation



→



→



LM, GLM

`lm`
`glm`

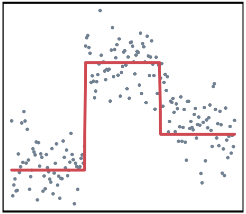
GAM

`mgcv`
`VGAM`

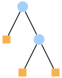
GAMLSS

`gamlss`
`mgcv`
`VGAM`
`gamboostLSS`
`bamlss`

Motivation

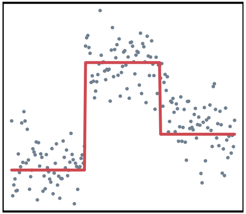


Regression tree

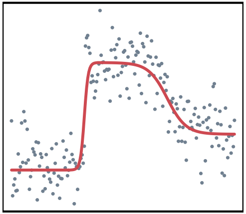


`rpart`
`party(kit)`

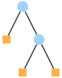
Motivation



→

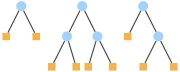


Regression tree



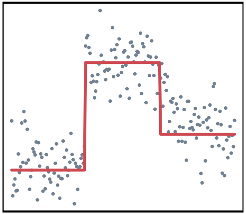
`rpart`
`party(kit)`

Random forest

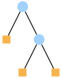


`randomForest`
`ranger`
`party(kit)`

Motivation

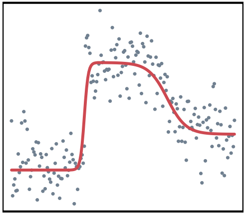


Regression tree

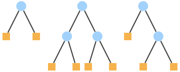


`rpart`
`party(kit)`

→

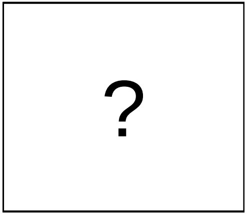


Random forest



`randomForest`
`ranger`
`party(kit)`

→



Distributional trees and forests

`disttree`
based on `partykit`

Motivation

Distributional:

- Specify the complete probability distribution (location, scale, shape, ...).

Tree:

- Automatic detection of steps and abrupt changes.
- Capture non-linear and non-additive effects and interactions.

Forest:

- Smoother effects.
- Stabilization and regularization of the model.

Learning distributional trees and forests

Tree:

Learning distributional trees and forests

Tree:

Learning distributional trees and forests

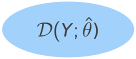
Tree:

1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.

Learning distributional trees and forests

Tree:

1. Fit global distributional model $\mathcal{D}(Y;\theta)$:
   Estimate model parameters $\hat{\theta}$.



Learning distributional trees and forests

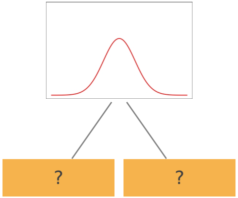
Tree:

1. Fit global distributional model $\mathcal{D}(Y;\theta)$:
   Estimate model parameters $\hat{\theta}$.

Learning distributional trees and forests

Tree:

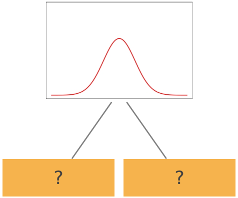
1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.



Learning distributional trees and forests

Tree:

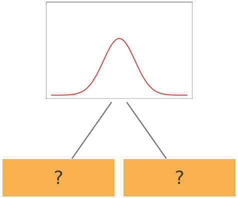
1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).



Learning distributional trees and forests

Tree:

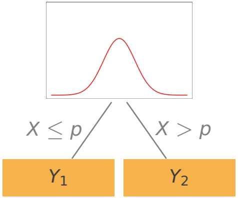
1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).
3. Choose covariate X with strongest influence on goodness of fit of $\mathcal{D}(Y; \hat{\theta})$ as split variable.
4. Find the split point p which leads to the highest improvement.



Learning distributional trees and forests

Tree:

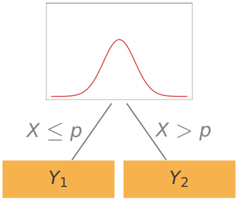
1. Fit global distributional model $\mathcal{D}(Y;\theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).
3. Choose covariate X with strongest influence on goodness of fit of $\mathcal{D}(Y;\hat{\theta})$ as split variable.
4. Find the split point p which leads to the highest improvement.



Learning distributional trees and forests

Tree:

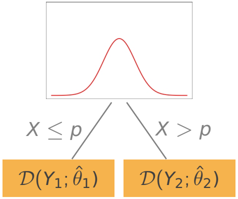
1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).
3. Choose covariate X with strongest influence on goodness of fit of $\mathcal{D}(Y; \hat{\theta})$ as split variable.
4. Find the split point p which leads to the highest improvement.
5. Repeat steps 1–4 recursively in the subgroups until some stopping criterion is met.



Learning distributional trees and forests

Tree:

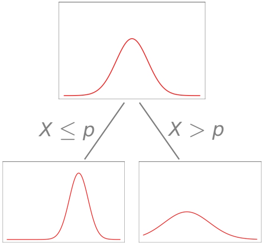
1. Fit global distributional model $\mathcal{D}(Y;\theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).
3. Choose covariate X with strongest influence on goodness of fit of $\mathcal{D}(Y;\hat{\theta})$ as split variable.
4. Find the split point p which leads to the highest improvement.
5. Repeat steps 1–4 recursively in the subgroups until some stopping criterion is met.



Learning distributional trees and forests

Tree:

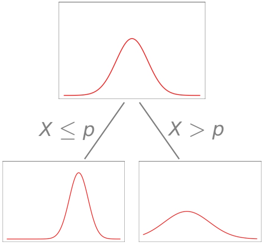
1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).
3. Choose covariate X with strongest influence on goodness of fit of $\mathcal{D}(Y; \hat{\theta})$ as split variable.
4. Find the split point p which leads to the highest improvement.
5. Repeat steps 1–4 recursively in the subgroups until some stopping criterion is met.



Learning distributional trees and forests

Tree:

1. Fit global distributional model $\mathcal{D}(Y; \theta)$:
   Estimate model parameters $\hat{\theta}$.
2. Evaluate goodness of fit
   (for each parameter and each observation).
3. Choose covariate X with strongest influence on goodness of fit of $\mathcal{D}(Y; \hat{\theta})$ as split variable.
4. Find the split point p which leads to the highest improvement.
5. Repeat steps 1–4 recursively in the subgroups until some stopping criterion is met.



Forest: Ensemble of T trees.

- Bootstrap or subsamples.
- Random input variable sampling.

Application

Goal: Probabilistic precipitation forecasting.

Application

Goal: Probabilistic precipitation forecasting.

Observation data:

- Daily 24h precipitation sums from July over 28 years (1985–2012).
- Observation station “Axams” in Tyrol, Austria.

Application

Goal: Probabilistic precipitation forecasting.

Observation data:

- Daily 24h precipitation sums from July over 28 years (1985–2012).
- Observation station “Axams” in Tyrol, Austria.

Covariates:

- Numeric ensemble weather predictions of precipitation, temperature, air pressure, convective available potential energy, . . .
- 80 covariates based on ensemble min/max/mean/standard deviation.

Application

Goal: Probabilistic precipitation forecasting.

Observation data:

- Daily 24h precipitation sums from July over 28 years (1985–2012).
- Observation station “Axams” in Tyrol, Austria.

Covariates:

- Numeric ensemble weather predictions of precipitation, temperature, air pressure, convective available potential energy, ...
- 80 covariates based on ensemble min/max/mean/standard deviation.

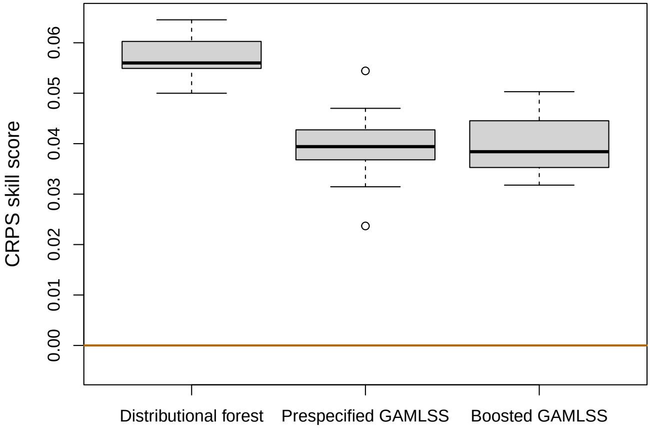
Distribution assumption: Power-transformed Gaussian, censored at 0.

$$(\text{precipitation})^{\frac{1}{1.6}} \sim c\mathcal{N}(\mu, \sigma^2)$$

cc by

Application

Predictive performance: Distributional forests improve CRPS skill score compared to heteroscedastic linear model (EMOS) and competing GAMLSS.



Graphical assessment

However: Is the distributional fit calibrated?

cc by

Graphical assessment

However: Is the distributional fit calibrated?

Graphical assessments: Various possibilities suggested in different parts of the literature.

- (Randomized) quantile-quantile residuals plot.
- Probability integral transform (PIT) histogram.
- Rootogram.
- Reliability diagram at prespecified thresholds.
- Worm plot.

Graphical assessment

However: Is the distributional fit calibrated?

Graphical assessments: Various possibilities suggested in different parts of the literature.

- (Randomized) quantile-quantile residuals plot.
- Probability integral transform (PIT) histogram.
- Rootogram.
- Reliability diagram at prespecified thresholds.
- Worm plot.

In R: Different bits in various packages but no unifying and flexible infrastructure.

Graphical assessment

However: Is the distributional fit calibrated?

Graphical assessments: Various possibilities suggested in different parts of the literature.

- (Randomized) quantile-quantile residuals plot.
- Probability integral transform (PIT) histogram.
- Rootogram.
- Reliability diagram at prespecified thresholds.
- Worm plot.

In R: Different bits in various packages but no unifying and flexible infrastructure.

Now: `topmodels` (on R-Forge).

Graphical assessment

Packages and data:

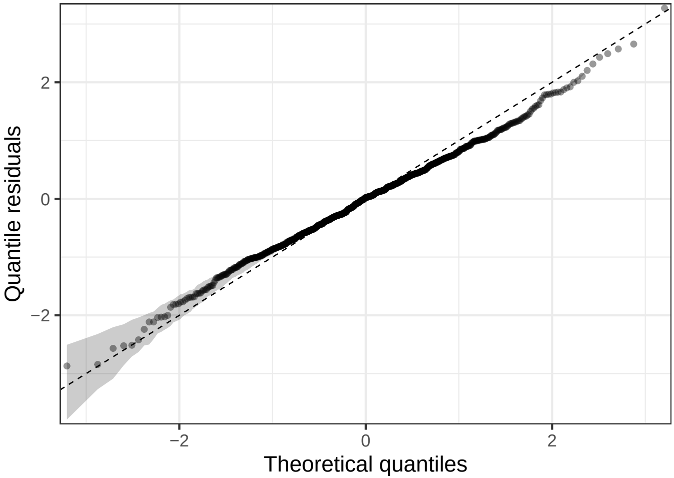
```
R> install.packages("disttree", repos = "https://R-Forge.R-project.org")
R> install.packages("topmodels", repos = "https://R-Forge.R-project.org")
R> library("disttree")
R> library("topmodels")
R> data("RainAxams", package = "disttree")
```

Random forest:

```
R> forest <- distforest(robs ~ .,
+                       family = dist_list_cens_normal,
+                       data = RainAxams, ...)
```

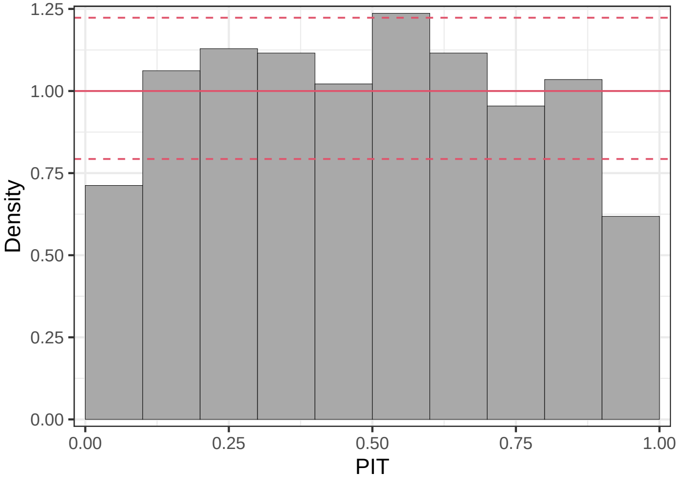
Graphical assessment

Q-Q residuals plot: `qqrplot(forest)`



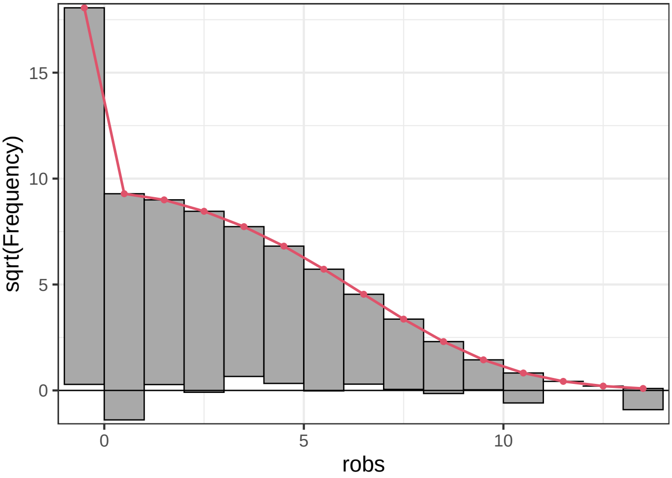
Graphical assessment

PIT histogram: `pithist(forest)`



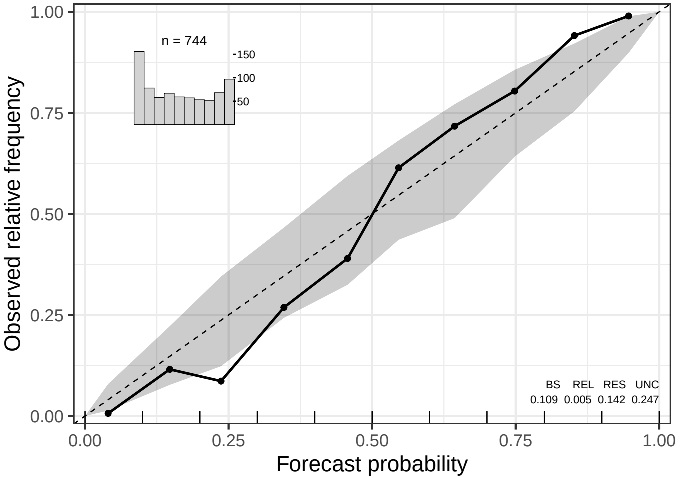
Graphical assessment

Rootogram: `rootogram(forest)`



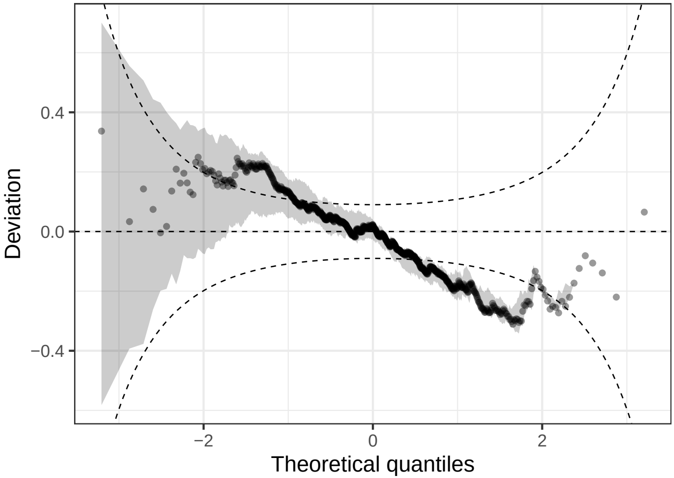
Graphical assessment

Reliability diagram: `reliagram(forest)`



Graphical assessment

Worm plot: `wormplot(forest)`



Graphical assessment

In contrast: Linear Gaussian model.

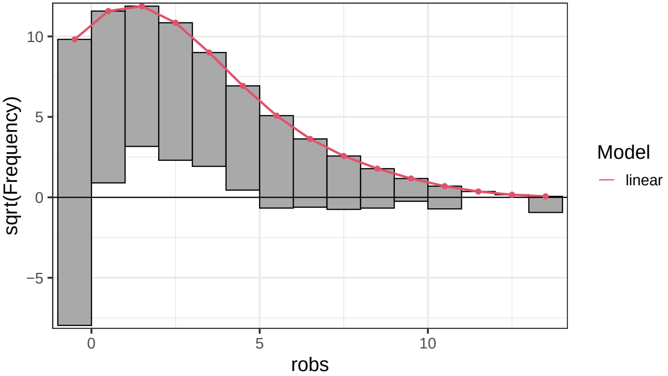
- Homoscedastic.
- Not accounting for excess zeros.
- Incorrect assumption of underlying response distribution.

```
R> linear <- lm(robs ~ tppow_mean, data = RainAxams)
```

Graphical assessment

Model comparison: Rootogram

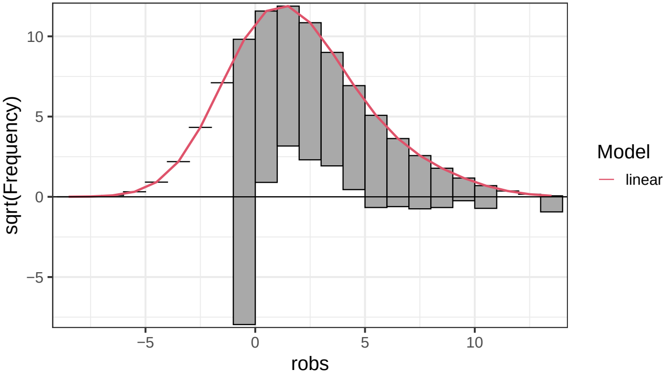
```
R> rootogram(linear, plot = FALSE) |>
+    autoplot(legend = TRUE)
```



Graphical assessment

Model comparison: Rootogram

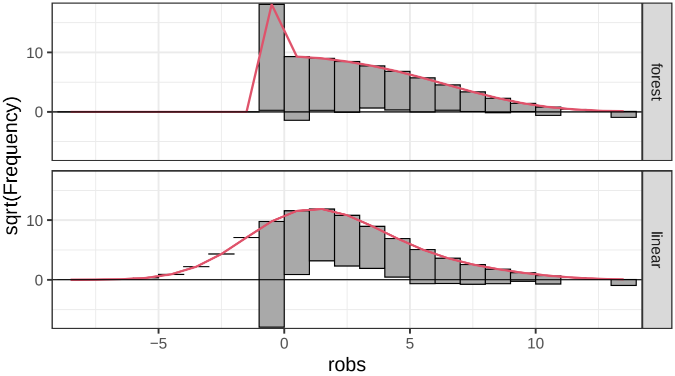
```
R> rootogram(linear, plot = FALSE, breaks = -9:14) |>
+    autoplot(legend = TRUE)
```



Graphical assessment

Model comparison: Rootogram

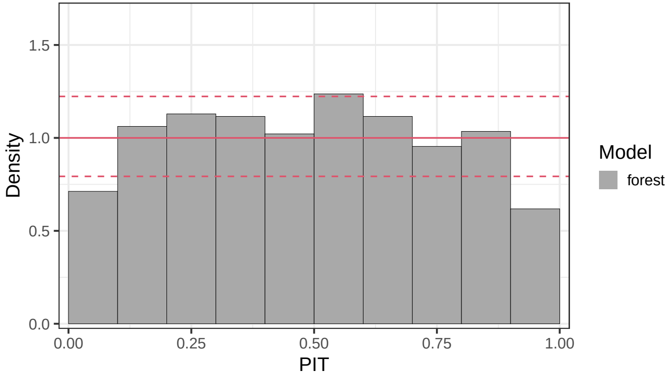
```
R> c(rootogram(forest, breaks = -9:14), rootogram(linear, breaks = -9:14)) |>
+    autoplot(legend = TRUE)
```



Graphical assessment

Model comparison: PIT histogram

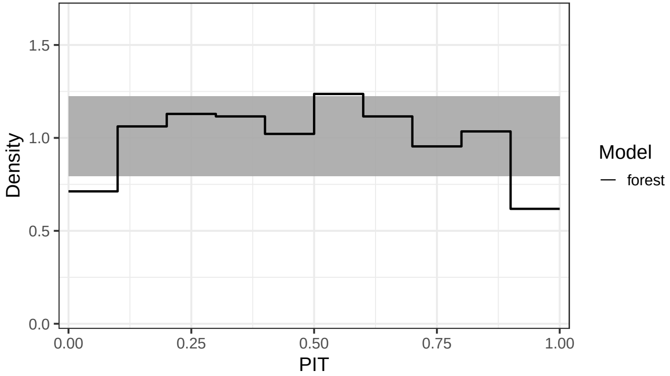
```
R> pithist(forest, plot = FALSE) |>
+    autoplot(legend = TRUE)
```



Graphical assessment

Model comparison: PIT histogram

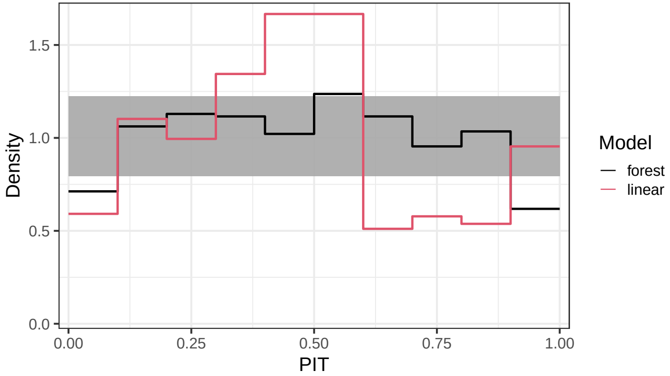
```
R> pithist(forest, plot = FALSE) |>
+    autoplot(legend = TRUE, style = "lines")
```



Graphical assessment

Model comparison: PIT histogram

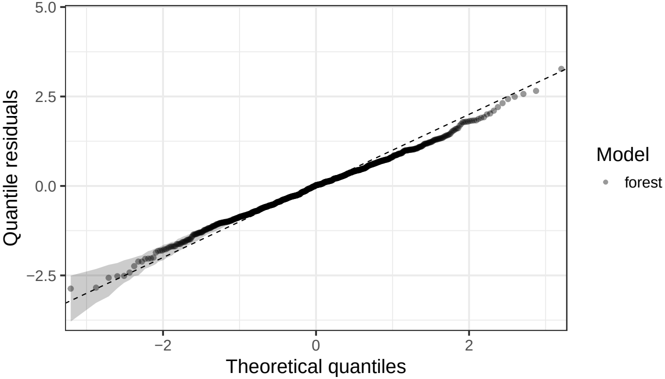
```
R> c(pithist(forest, plot = FALSE), pithist(linear, plot = FALSE)) |>
+    autoplot(legend = TRUE, style = "lines", single_graph = TRUE, col = 1:2)
```



Graphical assessment

Model comparison: Q-Q residuals plot

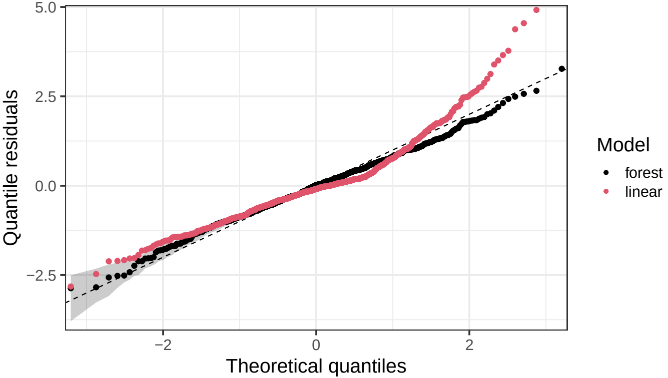
```
R> qqrplot(forest, plot = FALSE) |>
+     autoplot(legend = TRUE)
```



Graphical assessment

Model comparison: Q-Q residuals plot

```
R> c(qqrplot(forest, plot = FALSE), qqrplot(linear, plot = FALSE)) |>
+    autoplot(legend = TRUE, single_graph = TRUE, col = 1:2)
```



Software

disttree: available on R-Forge at

`https://R-Forge.R-project.org/projects/partykit/pkg/disttree/`

Concept: Fusion of tree-based models with distributional modeling.

Main functions:

`distfit`	Distributional fits (ML, `gamlss.family`/custom `list`). No covariates.
`disttree`	Distributional trees (`ctree`/`mob` + `distfit`). Covariates as partitioning variables.
`distforest`	Distributional forests (ensemble of `disttrees`). Covariates as partitioning variables.

Software

topmodels: available on R-Forge at

`https://topmodels.R-Forge.R-project.org/`

Concept: Unifying toolbox for probabilistic forecasts and graphical model assessment.

Main functions:

`procast`	Probabilistic forecasts (`(g)lm`, `crch`, `disttree`, more to come).
	Computation of probabilities, densities, scores, and Hessians.
`rootogram`, `pithist`, ...	Plotting rootograms, PIT histograms, ...
`plot, autoplot`	Generic `plot`, `autoplot` function.

References

Schlosser L, Hothorn T, Stauffer R, Zeileis A (2019). "Distributional Regression Forests for Probabilistic Precipitation Forecasting in Complex Terrain." *The Annals of Applied Statistics*, **13**(3), 1564–1589. doi:10.1214/19-AOAS1247

Lang MN, Zeileis A *et al.* (2021). "topmodels: Infrastructure for Inference and Forecasting in Probabilistic Models." *R package version 0.1-0*. https://topmodels.R-Forge.R-project.org/

Hothorn T, Zeileis A (2015). "partykit: A Modular Toolkit for Recursive Partytioning in R." *Journal of Machine Learning Research*, **16**, 3905–3909. http://www.jmlr.org/papers/v16/hothorn15a

https://topmodels.R-Forge.R-project.org/

moritz.lang@uibk.ac.at MoritzNLang