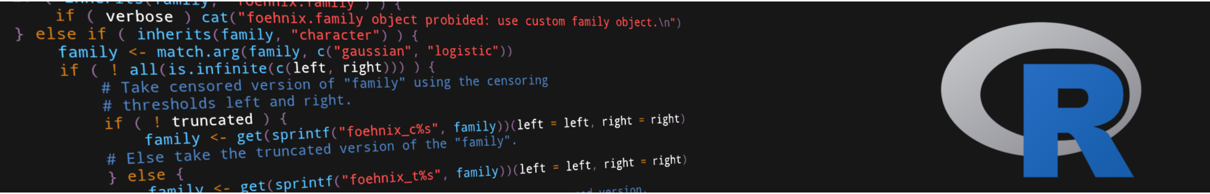
universität innsbruck



distributions3

From Basic Probability to Probabilistic Regression

Achim Zeileis, Moritz N. Lang, Alex Hayes

https://alexpghayes.github.io/distributions3/

Background

distributions3: Probability distributions as S3 objects.

- Started by Alex Hayes in 2019.
- Early contributions from Ralph Moller-Trane, Daniel Jordan, Paul Northrop, ...
- Geared towards introductory statistics courses.
- Beginner-friendly, well-documented, and lightweight interface.

Background

distributions3: Probability distributions as S3 objects.

- Started by Alex Hayes in 2019.
- Early contributions from Ralph Moller-Trane, Daniel Jordan, Paul Northrop, ...
- Geared towards introductory statistics courses.
- Beginner-friendly, well-documented, and lightweight interface.

Recently:

- Contributions from Moritz N. Lang and Achim Zeileis.
- Extension to vectors of distributions (of the same class).
- Extract probability distributions from models: `lm()`, `glm()`, `arima()`, ...
- Infrastructure for assessing goodness of fit in *topmodels* package.

Design

Class constructors: For many distributions, e.g., `Normal()`, `Poisson()`, ...

S3 objects: Distributions are essentially data frames of parameters.

Methods: For standard tasks, e.g., `mean()`, `quantile()`, `cdf()`, `random()`, ...

Under the hood: Rely on the usual `d`/`p`/`q`/`r` distribution functions.

The Poisson distribution

Illustration: Poisson as classic distribution for count data.

The Poisson distribution

Illustration: Poisson as classic distribution for count data.

Probability mass function: For $y \in \{0, 1, 2, \dots\}$ and parameter $\lambda > 0$.

$$\Pr(Y = y) = \frac{\exp(-\lambda) \cdot \lambda^y}{y!}.$$

The Poisson distribution

Illustration: Poisson as classic distribution for count data.

Probability mass function: For $y \in \{0, 1, 2, \dots\}$ and parameter $\lambda > 0$.

$$\Pr(Y = y) = \frac{\exp(-\lambda) \cdot \lambda^y}{y!}.$$

Example: $Y \sim Poisson(\lambda = 1.5)$.

```
R> library("distributions3")
R> Y <- Poisson(lambda = 1.5)
R> print(Y)
```

```
[1] "Poisson distribution (lambda = 1.5)"
```

```
R> pdf(Y, 0:5)
```

```
[1] 0.22313 0.33470 0.25102 0.12551 0.04707 0.01412
```

The Poisson distribution

Moments:

```
R> mean(Y)
[1] 1.5
R> variance(Y)
[1] 1.5
```

The Poisson distribution

Moments:

```
R> mean(Y)
[1] 1.5
R> variance(Y)
[1] 1.5
```

Cumulative probabilities and quantiles:

```
R> cdf(Y, 0:5)
[1] 0.2231 0.5578 0.8088 0.9344 0.9814 0.9955
R> quantile(Y, c(0.1, 0.5, 0.9))
[1] 0 1 3
```

The Poisson distribution

Moments:

```
R> mean(Y)
[1] 1.5
R> variance(Y)
[1] 1.5
```

Cumulative probabilities and quantiles:

```
R> cdf(Y, 0:5)
[1] 0.2231 0.5578 0.8088 0.9344 0.9814 0.9955
R> quantile(Y, c(0.1, 0.5, 0.9))
[1] 0 1 3
```

Random numbers:

```
R> set.seed(0)
R> random(Y, 5)
[1] 3 1 1 2 3
```

Goals in the 2018 FIFA World Cup

Illustration: Goals scored by the two teams in all 64 matches.

Covariates: Basic match information and prediction of team (log-)abilities.

```
R> data("FIFA2018", package = "distributions3")
R> head(FIFA2018)
  goals team match type stage logability difference
1     5  RUS     1    A group     0.1531     0.8638
2     0  KSA     1    A group    -0.7108    -0.8638
3     0  EGY     2    A group    -0.2066    -0.4438
4     1  URU     2    A group     0.2372     0.4438
5     3  RUS     3    A group     0.1531     0.3597
6     1  EGY     3    A group    -0.2066    -0.3597
```

Goals in the 2018 FIFA World Cup

Basic fitted distribution:

```
R> p_const <- Poisson(lambda = mean(FIFA2018$goals))
R> p_const
[1] "Poisson distribution (lambda = 1.3)"
```

Goals in the 2018 FIFA World Cup

Basic fitted distribution:

```
R> p_const <- Poisson(lambda = mean(FIFA2018$goals))
R> p_const
[1] "Poisson distribution (lambda = 1.3)"
```

Alternatively:

```
R> p_const <- fit_mle(Poisson(lambda = 1), FIFA2018$goals)
```

Goals in the 2018 FIFA World Cup

Basic fitted distribution:

```
R> p_const <- Poisson(lambda = mean(FIFA2018$goals))
R> p_const
[1] "Poisson distribution (lambda = 1.3)"
```

Alternatively:

```
R> p_const <- fit_mle(Poisson(lambda = 1), FIFA2018$goals)
```

Observed and expected frequencies:

```
R> observed <- proportions(table(FIFA2018$goals))
R> expected <- pdf(p_const, 0:6)
```

Goals in the 2018 FIFA World Cup

Basic fitted distribution:

```
R> p_const <- Poisson(lambda = mean(FIFA2018$goals))
R> p_const
[1] "Poisson distribution (lambda = 1.3)"
```

Alternatively:

```
R> p_const <- fit_mle(Poisson(lambda = 1), FIFA2018$goals)
```

Observed and expected frequencies:

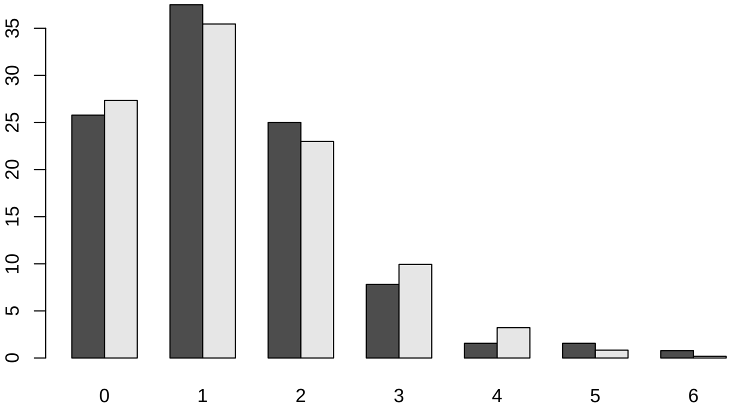
```
R> observed <- proportions(table(FIFA2018$goals))
R> expected <- pdf(p_const, 0:6)
```

Comparison:

```
R> tab <- 100 * rbind(observed, expected)
R> tab
             0     1     2     3     4      5      6
observed 25.78 37.50 25.00 7.812 1.562 1.5625 0.7812
expected 27.34 35.45 22.99 9.938 3.222 0.8358 0.1806
```

Goals in the 2018 FIFA World Cup

```
R> barplot(tab, beside = TRUE)
```



Probabilistic regression

Extension: Poisson generalized linear model (with log link).

Regression: Number of goals per team explained by ability difference (based on bookmakers odds).

Probabilistic regression

Extension: Poisson generalized linear model (with log link).

Regression: Number of goals per team explained by ability difference (based on bookmakers odds).

```
R> m <- glm(goals ~ difference, data = FIFA2018, family = poisson)
R> lmtest::coeftest(m)
```

```
z test of coefficients:

            Estimate Std. Error z value Pr(>|z|)
(Intercept)   0.2127     0.0813    2.62   0.0088 **
difference    0.4134     0.1058    3.91  9.3e-05 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Probabilistic regression

Fitted probability distributions:

```
R> p_reg <- Poisson(lambda = fitted(m))
R> length(p_reg)
```

```
[1] 128
```

```
R> head(p_reg)
```

```
                                       1                                        2 
"Poisson distribution (lambda = 1.768)" "Poisson distribution (lambda = 0.866)" 
                                       3                                        4 
"Poisson distribution (lambda = 1.030)" "Poisson distribution (lambda = 1.486)" 
                                       5                                        6 
"Poisson distribution (lambda = 1.435)" "Poisson distribution (lambda = 1.066)" 
```

Probabilistic regression

Fitted probability distributions:

```
R> p_reg <- Poisson(lambda = fitted(m))
R> length(p_reg)
```

```
[1] 128
```

```
R> head(p_reg)
```

```
                                       1                                        2 
"Poisson distribution (lambda = 1.768)" "Poisson distribution (lambda = 0.866)" 
                                       3                                        4 
"Poisson distribution (lambda = 1.030)" "Poisson distribution (lambda = 1.486)" 
                                       5                                        6 
"Poisson distribution (lambda = 1.435)" "Poisson distribution (lambda = 1.066)" 
```

Convenience function:

```
R> p_reg <- prodist(m)
```

Probabilistic regression

Opportunities: Unification and simplification of many computations.

Probabilistic regression

Opportunities: Unification and simplification of many computations.

Domain-specific:

- Probabilities for match results (assuming independence of goals).
- Corresponding probabilities for win/draw/lose.
- Also for more refined predictions of expected goals.

Probabilistic regression

Opportunities: Unification and simplification of many computations.

Domain-specific:

- Probabilities for match results (assuming independence of goals).
- Corresponding probabilities for win/draw/lose.
- Also for more refined predictions of expected goals.

General modeling:

- Probabilistic forecasts.
- Scoring rules.
- Goodness-of-fit assessments.

Graphical model assessment

Question: Is the model calibrated?

Graphical model assessment

Question: Is the model calibrated?

Idea: Compare observed and average expected frequencies.

```
R> expected <- pdf(p_reg, 0:6)
R> head(expected, 4)
     d_0    d_1    d_2     d_3     d_4      d_5       d_6
1 0.1707 0.3017 0.2667 0.15721 0.06949 0.024571 0.0072403
2 0.4208 0.3642 0.1576 0.04548 0.00984 0.001703 0.0002457
3 0.3571 0.3677 0.1893 0.06498 0.01673 0.003444 0.0005911
4 0.2262 0.3362 0.2498 0.12377 0.04599 0.013669 0.0033857
R> expected <- colMeans(expected)
```

Graphical model assessment

Question: Is the model calibrated?

Idea: Compare observed and average expected frequencies.

```
R> expected <- pdf(p_reg, 0:6)
R> head(expected, 4)
```

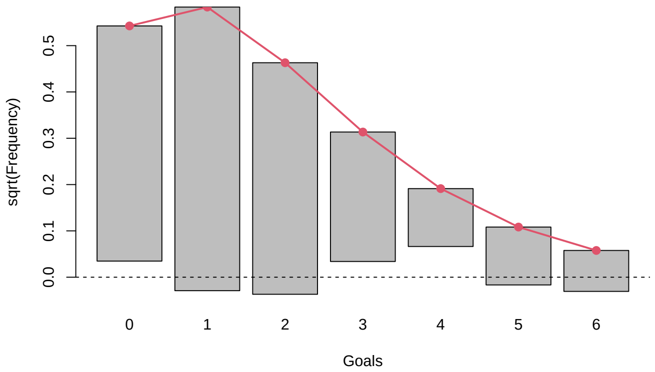
```
     d_0    d_1    d_2     d_3     d_4      d_5       d_6
1 0.1707 0.3017 0.2667 0.15721 0.06949 0.024571 0.0072403
2 0.4208 0.3642 0.1576 0.04548 0.00984 0.001703 0.0002457
3 0.3571 0.3677 0.1893 0.06498 0.01673 0.003444 0.0005911
4 0.2262 0.3362 0.2498 0.12377 0.04599 0.013669 0.0033857
```

```
R> expected <- colMeans(expected)
```

Rootogram: Visualize frequencies and their deviations on a square root scale.

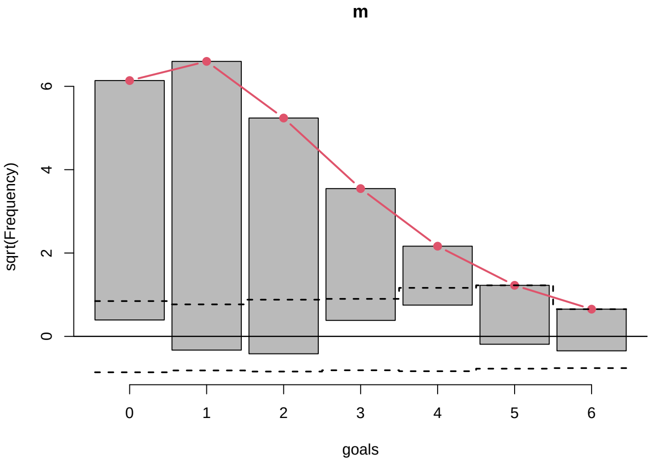
Graphical model assessment

```
R> bp <- barplot(sqrt(observed), offset = sqrt(expected) - sqrt(observed),
+    xlab = "Goals", ylab = "sqrt(Frequency)")
R> lines(bp, sqrt(expected), type = "o", pch = 19, lwd = 2, col = 2)
R> abline(h = 0, lty = 2)
```



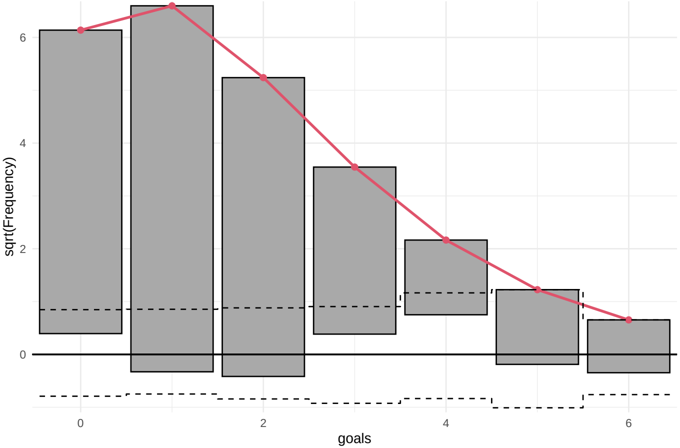
Graphical model assessment

```
R> library("topmodels")
R> rootogram(m)
```



Graphical model assessment

```
R> library("ggplot2")
R> theme_set(theme_minimal())
R> rootogram(m)
```



Graphical model assessment

Furthermore: Other visualizations supported in *topmodels*.

- Rootogram.
- PIT (probability integral transform) histogram.
- (Randomized) quantile residual Q-Q plot.
- Worm plot.
- Reliagram (reliability diagram).

Outlook

distributions3: Support for more distributions and models.

topmodels: Fully leverage *distributions3* infrastructure, introductory vignettes.

Moreover: Interface scoring rules from *scoringRules*.

References

Hayes A, Moller-Trane R, Jordan D, Northrop P, Lang MN, Zeileis A, *et al.* (2022). "distributions3: Probability Distributions as S3 Objects." *R package version 0.2.0*. `https://alexpghayes.github.io/distributions3/`

Lang MN, Zeileis A, Stauffer R, *et al.* (2022). "topmodels: Infrastructure for Inference and Forecasting in Probabilistic Models." *R package version 0.2-0*. `https://topmodels.R-Forge.R-project.org/`

Twitter: `@AchimZeileis`

Web: `https://www.zeileis.org/`