Fitting distributions with R

Prof. Anja Feldmann, Ph.D.
Dr. Nikolaos Chatzis
Fitting distributions

- Concept: finding a mathematical function that represents a statistical variable, e.g. delay
- E.g. modelling hopcount from traceroute measurements
- How to proceed?
  1. Guess the distribution from which the data might be drawn
  2. Estimate the parameters of that distribution
  3. Evaluate the quality of fit
Data manipulation

Loading data
- vector = read.table("hopcount",nrows=1000)
- vector = vector[,1]

Playing with data
- Vector Length: length(vector)
- Element n: vector[n]
- First k elements: vector[1:k]
- Last k elements: vector[(length(vector)-k+1), length(vector)]
- All elements larger than x: vector(vector>x)
- Trimming: mean(vector,trim=1/x)
Basic stats

- **Summary statistics:**
  - Mean: `mean(vector)`
  - Median: `median(vector)`
  - Standard deviation: `sd(vector)`
  - Variance: `var(vector)`
  - Summary: `summary(vector)`

- **Plotting densities:**
  - Basic plot: `plot(table(vector))`
  - Histogram: `hist(vector,x)`
  - CDF: `plot(ecdf(vector))`
  - Quantile plot: `boxplot(vector)`
Fitting distributions

- Choosing a model
  - Graphics, e.g., qqplot
    - Subjective
  - Matching the empirical distribution
    - Mean: \[ \mu = \frac{\sum_{i=1}^{n} x_i}{n} \]
    - Variability: \[ \text{var} = \sigma^2 = \frac{\sum_{i=1}^{n} (x_i - \mu)^2}{n} \]
    - Skewness: \[ \gamma_1 = \frac{\sum_{i=1}^{n} (x_i - \mu)^3}{n\sigma^3} \]
    - Kurtosis: \[ \gamma_2 = \frac{\sum_{i=1}^{n} (x_i - \mu)^4}{n\sigma^4} \]
R modules

- Loading R modules:
  - library(fBasics)
  - library(VarianceGamma)
  - library(stats4)
  - library(MASS)

- New statistics available:
  - Skewness
  - Kurtosis
The normal distribution

\[ f(x, \mu, \sigma) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-\frac{(x-\mu)^2}{2\sigma^2}} \]

- Generating a normal distribution with same mean and standard deviation as data:
  - `x.normal = rnorm(n=1000, m=mean(vector), sd=sd(vector))`
- Plotting the CDF:
  - `hist(x.normal)`
  - `plot(ecdf(x.normal))`
  - `qqplot(vector, x.normal)`
The Poisson distribution

\[ f(x, \lambda) = \frac{\lambda^x}{x!} e^{-\lambda} \]

Generating a Poisson distribution with same mean as data:

- `x.poisson = rpois(n=1000, lambda=mean(vector))`

Plotting the CDF:

- `hist(x.poisson)`
- `plot(ecdf(x.poisson))`
- `qqplot(vector, x.poisson)`
The Gamma distribution

\[ f(x, \alpha, \beta) = \alpha \beta^{-\alpha} x^{\alpha-1} e^{-(x/\beta)^\alpha} \]

- Gamma distribution: sum of alpha exponential distributions
  - `x.gamma = rgamma`
    - `n=1000, scale=0.83, shape=10.59`
  - `hist(x.gamma)`
  - `qqplot(vector, x.gamma)`
The Weibull distribution

\[ f(x, \lambda, k) = \frac{k}{\lambda} \left( \frac{x}{\lambda} \right)^{k-1} e^{-\left(\frac{x}{\lambda}\right)^k} \]

- Weibull distribution: distribution of failures
  - \( x\text{.weibull} = \text{rweibull} \)
    - (n=1000, scale=3.5, shape=14.1)
  - \( \text{hist}(x\text{.weibull}) \)
  - \( \text{qqplot}(\text{vector}, x\text{.weibull}) \)
Parameter estimation

- Estimating parameters of a model:
  - Analogic: assume that the estimate from the data is valid, e.g. mean
  - Moments: build estimator based on moments of the data, e.g. match first n moments of the distribution
  - Maximum likelihood: infer the value of the parameter that maximizes the probability of observing it based on the data
Moment estimation

- \( t^{th} \) moment of a distribution: 
  \[ m_t = \sum_{i=1}^{n} x_i^t y_i \]

- Example:
  - Estimating parameters of a gamma distribution using the first 2 moments:
    \[
    \frac{\beta}{\alpha} = \bar{x} \\
    \frac{\beta}{\alpha^2} = s^2
    \]
  - Gives estimates for parameters:
    \[
    \hat{\alpha} = \frac{\bar{x}}{s^2} \\
    \hat{\beta} = \frac{\bar{x}^2}{s^2}
    \]
Moment estimation: Gamma

\[ f(x, \alpha, \beta) = \alpha \beta^{-\alpha} x^{\alpha-1} e^{-(x/\beta)^\alpha} \]

- Estimating parameters:
  - Alpha = mean(vector)/var(vector)
  - Beta = (mean(vector))**2/var(vector)
  - x.gamma = rgamma
    (n=1000,scale=alpha,shape=beta)
  - hist(x.gamma)
  - qqplot(vector,x.gamma)
Maximum Likelihood Estimation

Given data $x_i$ and a parameter theta to be estimated, what is the most likely value of theta?

$$L(x_1, x_2, \ldots, x_n, \theta) = \prod_{i=1}^{n} f(x_i, \theta)$$

Example:

- Estimating parameters of distributions with `fitdistr()`:
  - library(MASS)
  - Gamma: `fitdistr(vector, "gamma")`
    - Shape: 10.59, Scale: 0.83
  - Weibull: `fitdistr(vector, "weibull")`
    - Shape: 3.5, Scale: 14.14
  - Normal: `fitdistr(vector, "normal")`
    - Mean: 12.74, Sd: 3.87
Goodness of fit

- Test: Is it reasonable to assume that the random sample comes from a specific distribution?

- 2 hypotheses:
  - $H_0$: Sample data comes from the stated distribution
  - $H_A$: Sample data does not come from the stated distribution

- Example: Kolmogorov-Smirnov test
  - Compares empirical distribution against theoretical one
  - Given $n$ data points $x_1,...,x_n$, define $F_n(x_i) = N(i)/n$
  - Test statistic: $D_n = \sup_i |F(x_i) - F_n(x_i)|$
  - If $D_n$ is too large for a given significance level, $H_0$ is rejected.
Goodness of fit (2)

- Testing goodness of generated samples:
  - Gamma: ks.test(x.gamma, "pgamma", scale=0.83, shape=10.59)
  - Weibull: ks.test(x.weibull,"pweibull",scale=3.5,shape=14.14)

- Testing for normality:
  - Shapiro-Wilk test
  - Shapiro.test(x.normal)

- Now test the data against Normal, Gamma and Weibull
  - Normal?
  - Gamma?
  - Weibull?