Applied Statistics in R

Elena Parilina

Master's Program
Game Theory and Operations Research

Saint Petersburg State University

2019

Agenda

1. One-Sample Tests for Normality
2. One-Sample Tests for Exponentiality
3. One-Sample Tests for Discrete Probability Distributions
4. Two-Sample Tests
5. Tests Related to Distribution Parameters

One-Sample Tests for Normality

One-sample Kolmogorov–Smirnov test

$$D^{+} = \max_{i=1,\ldots,n}\left[\frac{i}{n} - F(x_{(i)})\right], \quad D^{-} = \max_{i=1,\ldots,n}\left[F(x_{(i)}) - \frac{i-1}{n}\right],$$

$$D = \max\{D^{+}, D^{+}\}.$$

Test statistics: $D\sqrt{n}$.

ks.test

```
ks.test(x, y, ...,
    alternative = c("two.sided", "less", "greater"),
    exact = NULL)
```

ks.test

- Distribution is continuous.
- Missing values are silently omitted.
- The possible values "two.sided", "less" and "greater" of alternative specify the null hypothesis that the true distribution function of x is equal to, not less than or not greater than the hypothesized distribution function.
- If exact = NULL (the default), an exact p-value is computed if the sample size is less than 100 in the one-sample case.

Examples

```
 x <- rnorm(50)
# Does x come from a shifted gamma distribution with shape
3 and rate 2?
ks.test(x+2, "pgamma", 3, 2) # two-sided, exact
ks.test(x+2, "pgamma", 3, 2, exact = FALSE)
ks.test(x+2, "pgamma", 3, 2, alternative = "gr")
```

Lilliefors (one-sample Kolmogorov–Smirnov) test

$$D^{+} = \max_{i=1,\ldots,n}\left[\frac{i}{n} - F_0\left(\frac{x_{(i)} - \bar{x}}{s}\right)\right], \quad D^{-} = \max_{i=1,\ldots,n}\left[F_0\left(\frac{x_{(i)} - \bar{x}}{s}\right) - \frac{i-1}{n}\right],$$

$$D = \max\{D^{+}, D^{+}\}.$$

Test statistics: $D\left(\sqrt{n} - 0.01 + \frac{0.85}{\sqrt{n}}\right)$.

lillie.test

```
library(nortest)
   lillie.test(x)
```

uniNorm

```
library(MVN)
   uniNorm(data, type = "Lillie", desc = TRUE)
```

lillie.test

- Argument: a numeric vector of data values, the number of which must be greater than 4. Missing values are allowed.
- Although the test statistic obtained from `lillie.test(x)` is the same as that obtained from `ks.test(x, "pnorm", mean(x), sd(x))`, it is not correct to use the p-value from the latter for the composite hypothesis of normality (mean and variance unknown), since the distribution of the test statistic is different when the parameters are estimated.

Examples

```
lillie.test(rnorm(100, mean = 5, sd = 3))
lillie.test(runif(100, min = 2, max = 4))
```

uniNorm

- This function performs univariate normality tests, including Shapiro-Wilk, Cramer-von Mises, Lilliefors (Kolmogorov-Smirnov), Shapiro-Francia and Anderson-Darling.
- Arguments: `data` is a vector, data frame or matrix; `type` select one of the univariate normality tests: `SW` Shapiro-Wilk, `CVM` Cramer-von Mises, `Lillie` Lilliefors (Kolmogorov-Smirnov), `SF` Shapiro-Francia, `AD` Anderson-Darling; `desc` if TRUE, it displays descriptive statistics including mean, standard deviation, median, minimum, maximum, 25th and 75th percentiles, skewness and kurtosis.

```
uniNorm(x, type = "SW", desc = TRUE)
```

Anderson–Darling test

$$A = -n - \frac{1}{n}\sum_{i=1}^{n}(2i-1)\left[\ln F_0\left(\frac{x_{(i)}-\bar{x}}{s}\right) + \ln(1 - F_0\left(\frac{x_{(n-i+1)}-\bar{x}}{s}\right)\right]$$

Test statistics: $A\left(1 + \frac{0.75}{n} + \frac{2.25}{n^2}\right)$.

ad.test

```
library(nortest)
   ad.test(x)
```

uniNorm

```
library(MVN)
   uniNorm(data, type = "AD", desc = TRUE)
```

ad.test

- Argument: x which is a numeric vector of data values, the number of which must be greater than 7. Missing values are allowed.
- Compared to the Cramer-von Mises test (as second choice) it gives more weight to the tails of the distribution.

Examples

```
ad.test(rnorm(100, mean = 5, sd = 3))
ad.test(runif(100, min = 2, max = 4))
```

Cramer–von Mises test

$$\omega^2 = \frac{1}{12n} + \sum_{i=1}^{n} \left(F_0 \left(\frac{x_{(i)} - \bar{x}}{s} \right) - \frac{2i-1}{2n} \right)^2.$$

Test statistics: $\omega^2 \left(1 + \frac{0.5}{n}\right)$.

cvm.test

```
library(nortest)
   cvm.test(x)
cvm.test(rnorm(100, mean = 5, sd = 3))
cvm.test(runif(100, min = 2, max = 4))
```

uniNorm

```
library(MVN)
   uniNorm(data, type = "CVM", desc = TRUE)
```

Pearson chi-square test

Test statistics:

$$\sum_{i=1}^{r} \frac{\left(n_i - np_i^{(0)}(\hat{\theta})\right)^2}{np_i^{(0)}(\hat{\theta})}.$$

pearson.test

```
library(nortest)
   pearson.test(x, n.classes = ceiling(2 *(n^(2/5))),
       adjust = TRUE)
```

- `n.classes` is a number of classes. The default is due to Moore (1986).
- `adjust` is logical; if `TRUE` (default), the p-value is computed from a chi-square distribution with `n.classes-3` degrees of freedom, otherwise from a chi-square distribution with `n.classes-1` degrees of freedom.

pearson.test

The Pearson chi-square test is usually not recommended for testing the composite hypothesis of normality due to its inferior power properties compared to other tests. It is common practice to compute the p-value from the chi-square distribution with n.classes - 3 degrees of freedom, in order to adjust for the additional estimation of two parameters. (For the simple hypothesis of normality (mean and variance known) the test statistic is asymptotically chi-square distributed with n.classes - 1 degrees of freedom.) This is, however, not correct as long as the parameters are estimated by mean(x) and var(x) (or sd(x)), as it is usually done, see Moore (1986) for details.
Since the true p-value is somewhere between the two, it is suggested to run pearson.test twice, with adjust = TRUE (default) and with adjust = FALSE. It is also suggested to slightly change the default number of classes, in order to see the effect on the p-value. Eventually, it is suggested not to rely upon the result of the test.

Shapiro–Wilk test:

shapiro.test

```
shapiro.test(x)
```

uniNorm

```
library(MVN)
    uniNorm(data, type = "SW", desc = TRUE)
```

Shapiro–Francia test:

sf.test

```
library(nortest)
    sf.test(x)
```

uniNorm

```
library(MVN)
    uniNorm(data, type = "SF", desc = TRUE)
```

One-Sample Tests for Exponentiality

One-sample Kolmogorov–Smirnov test

Test statistics:

$$KS = \sup_{x>0} \left| F_n(x) - (1 - e^{-x}) \right|,$$

where F_n is the ecdf of the scaled data $y_i = \frac{x_i}{\bar{x}}$.

The test uses Monte Carlo simulations.

ks.exp.test

```
library(exptest)
ks.exp.test(x, nrepl=2000)
ks.exp.test(rexp(100))
ks.exp.test(runif(100, min = 50, max = 100))
```

`nrepl` is the number of replications in Monte Carlo simulation.

Cramer–von Mises test

$$\omega^2 = \int_0^\infty \left[F_n(x) - (1 - e^{-x})\right]^2 e^{-x}\, dx,$$

where F_n is the ecdf of the scaled data $y_i = \frac{x_i}{\bar{x}}$.
The test uses Monte Carlo simulations.

cvm.exp.test

```
library(exptest)
cvm.exp.test(x, nrepl=2000)
cvm.exp.test(rexp(100))
cvm.exp.test(runif(100, min = 50, max = 100))
```

`nrepl` is the number of replications in Monte Carlo simulation.

Atkinson test test

Test statistics:

$$T(p) = \sqrt{n}\left|\frac{\left(\frac{1}{n}\sum_{i=1}^{n} x_i^p\right)^{1/p}}{\bar{x}} - (\Gamma(1+p))^{1/p}\right|$$

The test uses Monte Carlo simulations.

atkinson.exp.test

```
 library(exptest)
atkinson.exp.test(x, p=0.99, simulate.p.value=FALSE,
nrepl=2000)
atkinson.exp.test(rexp(100))
atkinson.exp.test(rchisq(100,3))
```

`simulate.p.value` is a logical value indicating whether to compute p-values by Monte Carlo simulation.

Lorenz test

$$L = \frac{\sum_{i=1}^{np} x_{(i)}}{\sum_{i=1}^{n} x_{(i)}}.$$

Test statistics: $\sqrt{n}(L - p - (1-p)\ln(1-p))$.

The test uses Monte Carlo simulations.

lorenz.exp.test

```
library(exptest)
   lorenz.exp.test(x, p=0.5, simulate.p.value=FALSE,
       nrepl=2000)
```

Shapiro-Wilk test for exponentiality:

shapiro.exp.test

```
library(exptest)
   shapiro.exp.test(x, nrepl=2000)
```

Kimber-Michael test for exponentiality:

shapiro.exp.test

```
library(exptest)
   kimber.exp.test(x, nrepl=2000)
```

One-Sample Tests for Discrete Probability Distributions

Pearson chi-square test

goodfit

```
 library(vcd)
gf<-goodfit(x,type="poisson", "MinChisq")
summary(gf)

# type = c("poisson", "binomial", "nbinomial")
# method = c("ML", "MinChisq")
```

The negative binomial distribution with `size=n` and `prob=p` has density

$$\frac{\Gamma(x+n)}{\Gamma(n)x!}p^n(1-p)^x$$

for $x = 0, 1, 2, \ldots, n > 0$ and $0 < p \leqslant 1$.
This represents the number of failures which occur in a sequence of Bernoulli trials before a target number of successes is reached. The mean is $\mu = n(1-p)/p$ and variance $n(1-p)/p^2$. This definition allows non-integer values of `size`.

goodfit

Examples

```
 # Simulated data examples:
dummy <- rnbinom(200, size = 5, prob = 0.8)
gf <- goodfit(dummy, type = "nbinomial", method =
"MinChisq")

dummy <- rbinom(100, size = 6, prob = 0.5)
gf1 <- goodfit(dummy, type = "binomial", par = list(size =
6))
gf2 <- goodfit(dummy, type = "binomial", par = list(prob =
0.6, size = 6))
```

Two-Sample Tests

Two-sample Kolmogorov–Smirnov test

$$D^+ = \max_{r=1,\dots,m} \left[\frac{r}{m} - F_n(y_{(r)})\right] = \max_{s=1,\dots,n} \left[G_m(x_{(s)}) - \frac{s-1}{n}\right],$$

$$D^- = \max_{r=1,\dots,m} \left[F_n(y_{(r)}) - \frac{r-1}{m}\right] = \max_{s=1,\dots,n} \left[\frac{s}{n} - G_m(x_{(s)})\right],$$

$$D = \max\{D^+, D^-\}.$$

Test statistics: $D\sqrt{\frac{mn}{m+n}}$.

ks.test

```
ks.test(x, y, ...,
    alternative = c("two.sided", "less", "greater"),
    exact = NULL)
```

Wilcoxon rank sum test / Mann–Whitney U-test

ks.test

```
wilcox.test(x, y = NULL,
    alternative = c("two.sided", "less", "greater"),
    mu = 0, paired = FALSE, exact = NULL, correct = TRUE,
    conf.int = FALSE, conf.level = 0.95, ...)
```

Tests Related to Distribution Parameters

F-test

Test statistics:

$$F = \frac{s_1^2\, n/(n-1)}{s_2^2\, m/(m-1)} = \frac{\tilde{s}_1^2}{\tilde{s}_2^2}.$$

var.test

```
var.test(x, y, ratio = 1,
   alternative = c("two.sided", "less", "greater"),
   conf.level = 0.95, ...)
```

Student's t-test

One-sample t-test: $\tau = \dfrac{\bar{x} - a_0}{s}\sqrt{n-1} = \dfrac{\bar{x} - a_0}{\tilde{s}}\sqrt{n}$

t.test

```
t.test(x, alternative = c("two.sided", "less", "greater"),
    mu = 0, conf.level = 0.95, ...)
```

Two-sample t-test:

$$\tau_1 = \frac{\bar{x} - \bar{y}}{\sqrt{\sigma_1^2/n + \sigma_2^2/m}}, \qquad \tau_2 = \frac{\bar{x} - \bar{y}}{\hat{s}\sqrt{1/n + 1/m}},$$

$$\hat{s} = \sqrt{\frac{ns_1^2 + ms_2^2}{n + m - 2}} = \sqrt{\frac{(n-1)\tilde{s}_1^2 + (m-1)\tilde{s}_2^2}{n + m - 2}}.$$

t.test

```
t.test(x, y = NULL, alternative = c("two.sided", "less",
    "greater"), mu = 0, paired = FALSE, var.equal = FALSE,
    conf.level = 0.95, ...)
```

Probability of success in a Bernoulli experiment

Test statistics: $B = \sum_{i=1}^{n} x_i$.

binom.test

```
binom.test(x, n, p = 0.5,
   alternative = c("two.sided", "less", "greater"),
   conf.level = 0.95)
```

Sign Test for the population median

Test statistics:

$$ST = \sum_{i=1}^{n} s(x_i - \theta_0),$$

where

$$s(x_i - \theta_0) = \begin{cases} 1, & x_i > \theta_0, \\ 0, & x_i \leqslant \theta_0. \end{cases}$$

binom.test

```
library(BSDA) SIGN.test(x, md = 0,
   alternative = c("two.sided", "less", "greater"),
   conf.level = 0.95)
```