Package ‘ffmanova’

March 28, 2019

Type Package
Title Fifty-Fifty MANOVA
Version 1.1.0
Date 2019-03-28
Author Øyvind Langsrud [aut, cre],
    Bjørn-Helge Mevik [aut]
Maintainer Øyvind Langsrud <oyl@ssb.no>
Encoding UTF-8

Description General linear modeling with multiple responses (MANCOVA). An overall p-value for each model term is calculated by the 50-50 MANOVA method by Langsrud (2002) <doi:10.1111/1467-9884.00320>, which handles collinear responses. Rotation testing, described by Langsrud (2005) <doi:10.1007/s11222-005-4789-5>, is used to compute adjusted single response p-values according to familywise error rates and false discovery rates (FDR). The approach to FDR is described in the appendix of Moen et al. (2005) <doi:10.1128/AEM.71.4.2086-2094.2005>. Unbalanced designs are handled by Type II sums of squares as argued in Langsrud (2003) <doi:10.1023/A:1023260610025>. Furthermore, the Type II philosophy is extended to continuous design variables as described in Langsrud et al. (2007) <doi:10.1080/02664760701594246>. This means that the method is invariant to scale changes and that common pitfalls are avoided.

Imports stats, utils
Suggests car, testthat
License GPL-2

URL https://github.com/olangsrd/ffmanova

BugReports https://github.com/olangsrd/ffmanova/issues

RoxygenNote 6.1.1

NeedsCompilation no
Repository CRAN

Date/Publication 2019-03-28 14:50:03 UTC
R topics documented:

dressing .......................... 2
ffAnova .................................. 3
ffmanova .................................. 5
fixModelMatrix .......................... 8
manova5050 ................................. 10
matlabColon ................................. 11
multiPvalues ................................. 12
multiStatistics .............................. 13
predict.ffmanova .......................... 13
rotationtests ............................... 15
stdize .................................. 16
unitests ................................. 17

Index 19

dressing

Dressing data

Description

A dataset from an experiment studying structural and rheological properties of a full fat dressing.

Usage

data(dressing)

Format

A data frame with 29 observations on the following 7 variables.

press  a numeric vector with values 75, 125 and 225. The homogenisation pressure.
stab  a numeric vector with values 0.3, 0.4 and 0.5. Amount of stabiliser.
emul  a numeric vector with values 0.1, 0.2 and 0.35. Amount of emulsifier.
day  a factor with levels 1, ..., 5. The day the experimental run was performed on.
visc  a numeric vector. The measured viscosity of the dressing.
rhoe  a matrix with 9 columns. Nine different response-parameters derived from rheological measuring. These parameters contain information about the physics of the dressing (more general that viscosity).
pvol  a matrix with 241 columns. Particle-volume curves. Using a coulter-counter instrument fat particles were counted and their volumes were registered. These data are presented as smoothed histograms (equally spaced bins on log-scale). The total area under the curve represents the total volume of the counted fat particles. The shape of the curve reflects how the total fat volume is distributed among the different particle sizes.
Details

The data comes from an experiment in which full fat dressings were produced with different amount of stabiliser and emulsifier, and with different homogenisation pressure (see above).

A full factorial 3^3 design with two additional center points was used. The experiment was run over five days. It was unknown up front how many experimental runs could be performed each day, so the order of the runs was randomised.

For each dressing, viscosity, rheology and particle volume measurements were taken (see above).

The day is stored as a factor. The other design variables are stored as numerical variables. If one wants to use them as factors, one can use e.g. `factor(press)` in the model formula, or `dressing$press <- factor(dressing$press)` prior to calling the modelling function.

Source

The data is taken from a research project financed by a grant (131472/112) from the Norwegian Research Council. The project was managed by Stabburet, which is a major manufacturer of dressing in Norway.

---

### ffAnova

**Type II* Anova**

#### Description

Analysis of variance table for a linear model using type II* sums of squares, which are described in Langsrud et al. (2007). Type II* extends the type II philosophy to continuous variables. The results are invariant to scale changes and pitfalls are avoided.

#### Usage

```r
ffAnova(formula, data = NULL)
```

#### Arguments

- `formula` A model formula or an R object (preferably an `lm` object).
- `data` An optional data frame, list or environment.

#### Details

This function is a variant of `ffmanova` for the univariate special case. The two input parameters will be interpreted by `model.frame`.

#### Value

An object of class "anova" (see `anova`).

#### Author(s)

Øyvind Langsrud and Bjørn-Helge Mevik
References


Examples

```r
# Generate example data
set.seed(123)
a <- c(0, 0, 0, 10, 10, 10, 1, 1, 1)
A <- as.character(a)  # A is categorical
b <- 1:9
y <- rnorm(9)/10 + a  # y depends strongly on a (and A)
a100 <- a + 100  # change of scale (origin)
b100 <- b + 100  # change of scale (origin)

# Four ways of obtaining the same results
ffAnova(y ~ A * b)
ffAnova(y ~ A * b100)
ffAnova(lm(y ~ A * b))
ffAnova(y ~ A * b, data.frame(A = A, y = y, b = 1:9))

# Second order continuous variable
ffAnova(y ~ a + I(a^2))

# Model equivalent to 'y ~ A * b'
ffAnova(y ~ (a + I(a^2)) * b)

# Demonstrating similarities and differences using package car
if (!require(car)) {  # Package car is loaded if available
  Anova <- function(x) {  # Replacement function if car not available
    warning("No results since package car is not available")
  }
}

lm_Ab <- lm(y ~ A * b)
lm_Ab100 <- lm(y ~ A * b100)

# Type II same as type II* in this case
Anova(lm_Ab)  # Type II
Anova(lm_Ab100)  # Type II
ffAnova(lm_Ab)  # Type II*
ffAnova(lm_Ab100)  # Type II*

# Type III depends on scale
Anova(lm_Ab, type = 3)
Anova(lm_Ab100, type = 3)

lm_a <- lm(y ~ a + I(a^2))
lm_a100 <- lm(y ~ a100 + I(a100^2))

# Now Type II depends on scale
Anova(lm_a)  # Type II
Anova(lm_a100)  # Type II
ffAnova(lm_a)  # Type II*
```

**ffAnova**
ffmanova

ffAnova(lm_a100) # Type II*

ffmanova  Fifty-fifty MANOVA

Description

General linear modeling of fixed-effects models with multiple responses is performed. The function calculates 50-50 MANOVA p-values, ordinary univariate p-values and adjusted p-values using rotation testing.

Usage

ffmanova(formula, data = NULL, stand = TRUE, nSim = 0, verbose = TRUE, returnModel = TRUE, returnY = FALSE, returnYhat = FALSE, returnYhatStd = FALSE, newdata = NULL, linComb = NULL, nonEstimableAsNA = TRUE, outputClass = "ffmanova")

Arguments

formula  Model formula. See "Note" below.
data  An optional data frame or list.
stand  Logical. Standardization of responses. This option has effect on the 50-50 MANOVA testing and the calculation of exVarSS.
nSim  nonnegative integer. The number of simulations to use in the rotation tests. Can be a single nonnegative integer or a list of values for each term.
verbose  Logical. If TRUE, the rotation tests print trace information.
returnModel  When TRUE, and object, ffModel, with output from ffModelObj is included in output. Must be TRUE to enable predictions by predict.ffmanova.
returnY  Response matrix, Y, in output when TRUE.
returnYhat  Matrix Yhat of fitted values corresponding to Y in output when TRUE.
returnYhatStd  Standard errors, YhatStd, in output when TRUE.
n newData  Possible input to predict.ffmanova. When non-NULL, prediction results will be included output.
linComb  Possible input to predict.ffmanova in addition to newData.
nonEstimableAsNA  Will be used as input to predict.ffmanova when newData and/or linComb is non-NULL.
outputClass  When set to, "anova", ffAnova results will be produced.
Details

An overall p-value for all responses is calculated for each model term. This is done using the 50-50 MANOVA method, which is a modified variant of classical MANOVA made to handle several highly correlated responses.

Ordinary single response p-values are produced. By using rotation testing these can be adjusted for multiplicity according to familywise error rates or false discovery rates. Rotation testing is a Monte Carlo simulation framework for doing exact significance testing under multivariate normality. The number of simulation repetitions (nSim) must be chosen.

Unbalance is handled by a variant of Type II sums of squares, which has several nice properties:

1. Invariant to ordering of the model terms.
2. Invariant to scale changes.
3. Invariant to how the overparameterization problem of categorical variable models is solved (how constraints are defined).
4. Whether two-level factors are defined to be continuos or categorical does not influence the results.
5. Analysis of a polynomial model with a single experimental variable produce results equivalent to the results using an orthogonal polynomial.

In addition to significance testing an explained variance measure, which is based on sums of sums of squares, is computed for each model term.

Value

An object of class "ffmanova", which consists of the concatenated results from the underlying functions `manova5050`, `rotationtests` and `unitests`:

- `termNames` model term names
- `exVarSS` explained variances calculated from sums of squares summed over all responses
- `df` degrees of freedom - adjusted for other terms in model
- `df_om` degrees of freedom - adjusted for terms contained in actual term
- `nPC` number of principal components used for testing
- `nBU` number of principal components used as buffer components
- `exVarPC` variance explained by `nPC` components
- `exVarBU` variance explained by `(nPC+nBU)` components
- `pValues` 50-50 MANOVA p-values
- `stand` logical. Whether the responses are standardised.
- `stat` The test statistics as t-statistics (when single degree of freedom) or F-statistics
- `pRaw` matrix of ordinary p-values from F- or t-testing
- `pAdjusted` matrix of adjusted p-values according to familywise error rates
- `pAdjFDR` matrix of adjusted p-values according to false discovery rates
- `simN` number of simulations performed for each term (same as input)

The matrices `stat`, `pRaw`, `pAdjusted` and `pAdjFDR` have one row for each model term and one column for each response.

According to the input parameters, additional elements can be included in output.
Note
The model is specified with `formula`, in the same way as in `lm` (except that offsets are not supported). See `lm` for details. Input parameters `formula` and `data` will be interpreted by `model.frame`.

Author(s)
Øyvind Langsrud and Bjørn-Helge Mevik

References

See also https://www.langsrud.com/statprogram.htm.

See Also
`ffanova` and `predict.ffanova`.

Examples

```r
data(dressing)

# An ANOVA model with all design variables as factors
# and with visc as the only response variable.
# Classical univariate Type II test results are produced.
ffanova(visc ~ (factor(press) + factor(stab) + factor(emul))^2 + day,
         data = dressing)

# A second order response surface model with day as a block factor.
# The properties of the extended Type II approach is utilized.
ffanova(visc ~ (press + stab + emul)^2 + I(press^2) + I(stab^2) + I(emul^2) + day,
         data = dressing)

# 50-50 MANOVA results with the particle-volume curves as
# multivariate responses. The responses are not standardized.
ffanova(pvol ~ (press + stab + emul)^2 + I(press^2) + I(stab^2) + I(emul^2) + day,
        stand = FALSE, data = dressing)

# 50-50 MANOVA results with 9 rheological responses (standardized).
# 99 rotation simulation repetitions are performed.
res <- ffanova(rheo ~ (press + stab + emul)^2 + I(press^2) + I(stab^2) + I(emul^2) + day,
               stand = TRUE, data = dressing)
```

说到这里，我们可以继续讨论具体的例子。我们考虑一个模型，其中的自变量为压力、阻力和环境变化。为了确保模型的层次结构得到保留，我们可以使用`fixModelMatrix`函数来处理因素矩阵。这个函数简单地接受一个模型矩阵作为输入，并返回一个经过调整的矩阵，其中模型的层次结构得到了保持。

```r
fixModelMatrix(mOld)
```

### Arguments

- `mOld` The factor matrix (i.e. the "factor" attribute) of a terms object.

### Description

The function takes the factor matrix of the terms object corresponding to a model formula and changes it so that model hierarchy is preserved also for powers of terms (e.g., `I(a^2)`).

### Usage

```r
fixModelMatrix(mOld)
```
Details

The ordinary model handling functions in do not treat powers of terms \(a^n\) as being higher order terms (like interaction terms). `fixModelMatrix` takes the "factor" attribute of a terms object (usually created from a model formula) and changes it such that power terms can be treated hierarchically just like interaction terms.

The factor matrix has one row for each variable and one column for each term. Originally, an entry is 0 if the term does not contain the variable. If it contains the variable, the entry is 1 if the variable should be coded with contrasts, and 2 if it should be coded with dummy variables. See `terms.object` for details.

The changes performed by `fixModelMatrix` are:

- Any 2’s are changed to 1.
- In any column corresponding to a term that contains \(I(a^n)\), where \(a\) is the name of a variable and \(n\) is a positive integer, the element in the row corresponding to \(a\) is set to \(n\). For instance, the entry of row \(D\) and column \(C: I(D^2)\) is set to 2.
- Rows corresponding to \(I(a^n)\) are deleted.

Note that this changes the semantics of the factor matrix: 2 no longer means ‘code via dummy variables’.

Value

A factor matrix.

Author(s)

Øyvind Langsrud and Bjørn-Helge Mevik

See Also

`terms, terms.object`

Examples

```r
mt <- terms(y ~ a + b + a:b + a:c + I(a^2) + I(a^3) + I(a^2):b)
print(mOld <- attr(mt, "factor"))
fixModelMatrix(mOld)
```
Computation of 50-50 MANOVA results

Description

The function takes a design-with-responses object created by `xy_obj` and produces 50-50 MANOVA output. Results are produced for each term in the model.

Usage

```r
manova5050(xyobj, stand)
```

Arguments

- `xyobj`: design-with-responses object
- `stand`: standardisation of responses (0 or 1)

Details

Classical multivariate ANOVA (MANOVA) are useless in many practical cases. The tests perform poorly in cases with several highly correlated responses and the method collapses when the number of responses exceeds the number of observations. 50-50 MANOVA is made to handle this problem. Principal component analysis (PCA) is an important part of this methodology. Each test is based on a separate PCA.

Value

A list with components

- `termNames`: model term names
- `exVarSS`: explained variances calculated from sums of squares summed over all responses
- `df`: degrees of freedom - adjusted for other terms in model
- `df_om`: degrees of freedom - adjusted for terms contained in actual term
- `nPC`: number of principal components used for testing
- `nBU`: number of principal components used as buffer components
- `exVarPC`: variance explained by `nPC` components
- `exVarBU`: variance explained by `(nPC+nBU)` components
- `pValues`: 50-50 MANOVA p-values
- `stand`: logical. Whether the responses are standardised.

Note

The 50-50 MANOVA p-values are based on the Hotelling-Lawley Trace Statistic. The number of components for testing and the number of buffer components are chosen according to default rules.
matlabColon

Author(s)
Øyvind Langsrud and Bjørn-Helge Mevik

References

See Also
ffmanova

matlabColon  Simulate Matlab’s ‘:’

Description
A function to simulate Matlab’s ‘:’ operator.

Usage
matlabColon(from, to)

Arguments
from numeric. The start value.
to numeric. The end value.

Details
matlabCode(a, b) returns a:b (‘s version) unless a > b, in which case it returns integer(0).

Value
A numeric vector, possibly empty.

Author(s)
Bjørn-Helge Mevik

See Also
seq

Examples
identical(3:5, matlabColon(3, 5)) ## => TRUE
3:1 ## => 3 2 1
matlabColon(3, 1) ## => integer(0)
**Description**

$p$-values from the four MANOVA test statistics are calculated according to the traditional F-distribution approximations (exact in some cases).

**Usage**

```r
multipvalues(D, E, A, M, dim, dimX, dimY)
```

**Arguments**

- `D` Wilks' Lambda
- `E` Roy's Largest Root
- `A` Hotelling-Lawley Trace Statistic
- `M` Pillay-Bartlett Trace Statistic
- `dim` Number of observations
- `dimX` Number of x-variables
- `dimY` Number of y-variables

**Details**

The parameters `dim`, `dimX` and `dimY` corresponds to a situation where the test statistics are calculated from two data matrices with zero mean (test of independence).

**Value**

- `pD` $p$-value: Wilks’ Lambda
- `pE` $p$-value: LOWER BOUND for Roy’s Largest Root
- `pA` $p$-value: Hotelling-Lawley Trace Statistic
- `pM` $p$-value: Pillay-Bartlett Trace Statistic

**Author(s)**

Øyvind Langsrud and Bjørn-Helge Mevik

**See Also**

`ffmanova`
**multiStatistics**

**MANOVA test statistics**

**Description**

The four classical MANOVA test statistics are calculated from a set of eigenvalues.

**Usage**

`multiStatistics(ss)`

**Arguments**

- `ss` A list of eigenvalues

**Details**

These eigenvalues are also known as the squared canonical correlation coefficients.

**Value**

A list with elements

- `D` Wilks’ Lambda
- `E` Roy’s Largest Root
- `A` Hotelling-Lawley Trace Statistic
- `M` Pillay-Bartlett Trace Statistic

**Author(s)**

Øyvind Langsrud and Bjørn-Helge Mevik

---

**predict.ffmanova**

**Predictions, mean predictions, adjusted means and linear combinations**

**Description**

The same predictions as `lm` can be obtained. With some variables missing in input, adjusted means or mean predictions are computed (Langsrud et al., 2007). Linear combinations of such predictions, with standard errors, can also be obtained.

**Usage**

```r
## S3 method for class 'ffmanova'
predict(object, newdata = NULL, linComb = NULL,
       nonEstimableAsNA = TRUE, ...)
```
Arguments

- **object**: Output from `ffmanova`.
- **newdata**: Data frame or list. Missing values and missing variables are possible.
- **linComb**: A matrix defining linear combinations.
- **nonEstimableAsNA**: When TRUE, missing values are returned when predictions cannot be made. When FALSE, predictions are made anyway, but the logical vector, `estimable`, is added to output in cases of non-estimable results.

Value

A list of two matrices:

- **YnewPred**: Predictions, mean predictions, adjusted means or linear combinations of such predictions.
- **YnewStd**: Corresponding standard errors.

References


Examples

```r
# Generate data
x1 <- 1:6
x2 <- rep(c(100, 200), each = 3)
y1 <- x1 + rnorm(6)/10
y2 <- y1 + x2 + rnorm(6)/10

# Create ffmanova object
ff <- ffmanova(cbind(y1, y2) ~ x1 + x2)

# Predictions from the input data
predict(ff)

# Rows 1 and 5 from above predictions
predict(ff, data.frame(x1 = c(1, 5), x2 = c(100, 200)))

# Rows 1 as above and row 2 different
predict(ff, data.frame(x1 = c(1, 5), x2 = 100))

# Three ways of making the same mean predictions
predict(ff, data.frame(x1 = c(1, 5), x2 = 150))
predict(ff, data.frame(x1 = c(1, 5), x2 = NA))
predict(ff, data.frame(x1 = c(1, 5)))

# Using linComb input specified to produce regression coefficients
# with std. As produced by summary(lm(cbind(y1, y2) ~ x1 + x2))
```
### rotationtests

The functions perform rotation testing based on a matrix of hypothesis observations and a matrix of error observations. Adjusted p-values according to familywise error rates and false discovery rates are calculated.

#### Usage

```r
rotationtests(xyObj, nSim, verbose = TRUE)
```

```r
rotationtest(modelData, errordata, simN = 999, dfe = -1,
             dispsim = TRUE)
```

#### Arguments

- **xyObj**: a design-with-response object created by `xy_Obj`
- **nSim**: vector of nonnegative integers. The number of simulations to use for each term.
- **verbose**: logical. Whether `rotationtests` (and `rotationtest`) should be verbose.
- **modelData**: matrix of hypothesis observations
- **errordata**: matrix of error observations
- **simN**: Number of simulations for each test. Can be a single value or a list of values for each term.
- **dfe**: Degrees of freedom for error needs to be specified if `errorData` is incomplete
- **dispsim**: When TRUE, dots are displayed to illustrate simulation progress.

#### Details

- **modelData and errorObs** correspond to `hypObs` and `errorObs` calculated by `xy_Obj`. These matrices are efficient representations of sums of squares and cross-products (see `xy_Obj` for details). This means that `rotationtest` can be viewed as a generalised F-test function.
- **rotationtests** is a wrapper function that calls `rotationtest` for each term in the `xyObj` and collects the results.
Value

Both functions return a list with components

- `pAdjusted`: adjusted $p$-values according to familywise error rates
- `pAdjFDR`: adjusted $p$-values according to false discovery rates
- `simN`: number of simulations performed for each term

Author(s)

Øyvind Langsrud and Bjørn-Helge Mevik

References


See Also

`unitest`, `unitests`

---

**stdize**

*Centering and scaling of matrices*

**Description**

Function to center and/or scale the columns of a matrix in various ways. The columns can be centered with their means or with supplied values, and they can be scaled with their standard deviations or with supplied values.

**Usage**

```r
stdize(x, center = TRUE, scale = TRUE, avoid.zero.divisor = FALSE)
stdize3(x, center = TRUE, scale = TRUE, avoid.zero.divisor = FALSE)
```

**Arguments**

- `x`: A matrix.
- `center`: A logical, or a numeric vector. The values to subtract from each column. If center is TRUE, the mean values are used.
- `scale`: A logical, or a numeric vector. The values to divide each column with. If scale is TRUE, the standard deviations are used.
- `avoid.zero.divisor`: A logical. If TRUE, each occurrence of 0 in scale is replaced with a 1.
**Details**

`stdize` standardizes the columns of a matrix by subtracting their means (or the supplied values) and dividing by their standard deviations (or the supplied values).

If `avoid.zero.divisor` is TRUE, division-by-zero is guarded against by substituting any 0 in center (either calculated or supplied) with 1 prior to division.

The main difference between `stdize` and `scale` is that `stdize` divides by the standard deviations even when center is not TRUE.

**Value**

A matrix.

**Note**

`stdize3` is a variant with a three-element list as output (`x`, `center`, `scale`) and where `avoid.zero.divisor` is also used to avoid centring (constant term in model matrix is unchanged).

**Author(s)**

Bjørn-Helge Mevik and Øyvind Langsrud

**See Also**

`scale`

**Examples**

```r
A <- matrix(rnorm(15, mean = 1), ncol = 3)
stopifnot(all.equal(stdize(A), scale(A), check.attributes = FALSE))

## These are different:
stdize(A, center = FALSE)
scale(A, center = FALSE)
```

**Description**

The functions perform `F` or `t` testing for several responses based on a matrix of hypothesis observations and a matrix of error observations.

**Usage**

```r
unitests(xyObj)
```

```r
unitest(modelData, errorData, dfError = dim(errorData)[1])
```
Arguments

- `xyObj` a design-with-responses object created by `xy_Obj`
- `modelData` matrix of hypothesis observations
- `errorData` matrix of error observations
- `dfError` Degrees of freedom for error needs to be specified if `errorData` is incomplete

Details

`modelData` and `errorObs` correspond to `hypObs` and `errorObs` calculated by `xy_Obj`. These matrices are efficient representations of sums of squares and cross-products (see `xy_Obj` for details). This means the univariate $F$-statistics can be calculated straightforwardly from these input matrices. Furthermore, in the single-degree-of-freedom case, $t$-statistics with correct sign can be obtained.

`unitests` is a wrapper function that calls `unitest` for each term in the `xyObj` (see `xy_Obj` for details) and collects the results.

Value

`unitest` returns a list with components

- `pValues` $p$-values
- `stat` The test statistics as $t$-statistics (when single degree of freedom) or $F$-statistics

`unitests` returns a list with components

- `pRaw` Matrix of $p$-values from `unitest`, one row for each term.
- `stat` Matrix of test statistics from `unitest`, one row for each term.

Note

The function calculates the $p$-values by making a call to `pf`.

Author(s)

Øyvind Langsrud and Bjørn-Helge Mevik

See Also

- `rotationtest`, `rotationtests`
Index

*Topic **array**
  - stdize, 16
*Topic **datasets**
  - dressing, 2
*Topic **design**
  - ffmanova, 5
  - manova5050, 10
*Topic **htest**
  - multiPvalues, 12
  - multiStatistics, 13
  - rotationtests, 15
  - unitests, 17
*Topic **models**
  - ffmanova, 5
  - fixModelMatrix, 8
  - manova5050, 10
*Topic **multivariate**
  - ffmanova, 5
  - manova5050, 10
*Topic **programming**
  - matlabColon, 11

anova, 3
dressing, 2
ffAnova, 3, 5, 7
ffmanova, 3, 5, 11, 12, 14
ffModelObj, 5
fixModelMatrix, 8
formula, 3

lm, 3, 7, 13
manova5050, 6, 10
matlabColon, 11
model.frame, 3, 7
multiPvalues, 12
multiStatistics, 13
predict.ffmanova, 5, 7, 13

rotationtest, 18
rotationtest (rotationtests), 15
rotationtests, 6, 15, 18
scale, 17
seq, 11
stdize, 16
stdize3 (stdize), 16
terms, 9
terms.object, 9
unitest, 16
untest (untests), 17
untests, 6, 16, 17
xy_Obj, 15, 18