

# Package ‘fitur’

September 1, 2018

**Title** Fit Univariate Distributions

**Version** 0.6.1

**Description** Wrapper for computing parameters for univariate distributions using MLE. It creates an object that stores d, p, q, r functions as well as parameters and statistics for diagnostics. Currently supports automated fitting from base and actuar packages. A manually fitting distribution fitting function is included to support directly specifying parameters for any distribution from ancillary packages.

**URL** <https://github.com/tomroh/fitur>

**BugReports** <https://github.com/tomroh/fitur/issues>

**Depends** R (>= 3.3.0)

**Imports** stats, fitdistrplus, actuar, e1071, ggplot2, goftest, shiny  
(>= 0.13), miniUI (>= 0.1.1), rstudioapi (>= 0.5), DT

**Suggests** knitr, rmarkdown

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**VignetteBuilder** knitr

**RoxygenNote** 6.0.1

**NeedsCompilation** no

**Author** Thomas Roh [aut, cre]

**Maintainer** Thomas Roh <thomas@roh.engineering>

**Repository** CRAN

**Date/Publication** 2018-09-01 19:20:03 UTC

## R topics documented:

build_dist . . . . .	2
calc_moments . . . . .	3
DiscreteUniform . . . . .	3

fit_dist_addin . . . . .	4
fit_empirical . . . . .	4
fit_univariate . . . . .	6
fit_univariate_man . . . . .	7
gen_dist_fun . . . . .	8
GOFTests . . . . .	8
gof_tests . . . . .	9
is.distfun . . . . .	10
Mode . . . . .	10
plot_density . . . . .	11
plot_pp . . . . .	11
plot_qq . . . . .	12

<b>Index</b>	<b>13</b>
--------------	-----------

---

build_dist	<i>Build Distribution Functions</i>
------------	-------------------------------------

---

### Description

A wrapper for building function families given a numeric vector and the distribution

### Usage

```
build_dist(x, distribution)
```

### Arguments

x	numeric vector
distribution	distribution character name

### Value

list of distribution functions for d, p, q, r, and parameters

### Examples

```
fittedDists <- build_dist(rpois(100,5), 'pois')
dpois(x = 5, lambda = 5)
fittedDists$dpois(5)
ppois(5, 5)
fittedDists$ppois(5)
qpois(.5, 5)
fittedDists$qpois(.5)
set.seed(8257)
rpois(100, 5)
set.seed(8257)
fittedDists$rpois(100)
fittedDists$parameters
```

---

calc_moments	<i>Calculate moments of a numeric vector</i>
--------------	--

---

**Description**

Calculate moments of a numeric vector

**Usage**

```
calc_moments(x)
```

**Arguments**

x                    a numeric vector

**Value**

a named vector of descriptive statistics

**Examples**

```
x <- rexp(1000, 2)
calc_moments(x)
```

---

DiscreteUniform	<i>The Discrete Uniform Distribution</i>
-----------------	--

---

**Description**

The Discrete Uniform Distribution

**Usage**

```
ddunif(x, min = 0, max = 1)
```

```
pdunif(q, min = 0, max = 1)
```

```
qdunif(p, min = 0, max = 1)
```

```
rdunif(n, min = 0, max = 1)
```

**Arguments**

x	vector of (non-negative integer) quantiles
min	minimum value of distribution (integer)
max	maximum value of distribution (integer)
q	vector of quantiles
p	vector of probabilities
n	number of random values to return

**Value**

ddunif gives the density, pdunif gives the distribution function, qdunif gives the quantile function, rdunif generates random deviates

**Examples**

```
ddunif(0:1)
pdunif(1)
qdunif(.5)
rdunif(10)
```

---

fit\_dist\_addin

*Fit Univariate Distributions Addin*


---

**Description**

Interactively submit a numeric vector and choose what distributions that you want to run fit diagnostics. Click done to have the desired distribution code put into your cursor position.

**Usage**

```
fit_dist_addin()
```

---

fit\_empirical

*Fit Empirical Distribution*


---

**Description**

Fit Empirical Distribution

**Usage**

```
fit_empirical(x)
```

**Arguments**

x integer or double vector

**Value**

if integer vector then list of family functions for d, p, q, r, and parameters based on each integer value. if it is a double vector then list of family functions for d, p, q, r, and parameters based on Freedman-Diaconis rule for optimal number of histogram bins.

**Examples**

```
set.seed(562)
x <- rpois(100, 5)
empDis <- fit_empirical(x)

# probability density function
plot(empDis$dempDis(0:10),
     xlab = 'x',
     ylab = 'dempDis')
# cumulative distribution function
plot(x = 0:10,
     y = empDis$pempDis(0:10),
     #type = 'l',
     xlab = 'x',
     ylab = 'pempDis')
# quantile function
plot(x = seq(.1, 1, .1),
     y = empDis$qempDis(seq(.1, 1, .1)),
     type = 'p',
     xlab = 'x',
     ylab = 'qempDis')
# random sample from fitted distribution
summary(empDis$r(100))

empDis$parameters

set.seed(562)
x <- rexp(100, 1/5)
empCont <- fit_empirical(x)

# probability density function
plot(x = 0:10,
     y = empCont$dempCont(0:10),
     xlab = 'x',
     ylab = 'dempCont')
# cumulative distribution function
plot(x = 0:10,
     y = empCont$pempCont(0:10),
     #type = 'l',
     xlab = 'x',
     ylab = 'pempCont')
# quantile function
```

```

plot(x = seq(.5, 1, .1),
     y = empCont$qempCont(seq(.5, 1, .1)),
     type = 'p',
     xlab = 'x',
     ylab = 'qempCont')
# random sample from fitted distribution
summary(empCont$r(100))

empCont$parameters

```

---

fit\_univariate

*Fit Univariate Distribution*


---

## Description

Fit Univariate Distribution

## Usage

```
fit_univariate(x, distribution, type = "continuous")
```

## Arguments

x	numeric vector
distribution	character name of distribution
type	discrete or continuous data

## Value

a fitted list object of d, p, q, r distribution functions and parameters, MLE for probability distributions, custom fit for empirical

## Examples

```

# Fit Discrete Distribution
set.seed(42)
x <- rpois(1000, 3)
fitted <- fit_univariate(x, 'pois', type = 'discrete')
# density function
plot(fitted$dpois(x=0:10),
     xlab = 'x',
     ylab = 'dpois')
# distribution function
plot(fitted$ppois(seq(0, 10, 1)),
     xlab = 'x',
     ylab = 'ppois')
# quantile function
plot(fitted$qpois,
     xlab = 'x',

```

```
      ylab = 'qpois')
# sample from theoretical distribution
summary(fitted$rpois(100))
# estimated parameters from MLE
fitted$parameters

set.seed(24)
x <- rweibull(1000, shape = .5, scale = 2)
fitted <- fit_univariate(x, 'weibull')
# density function
plot(fitted$dweibull,
     xlab = 'x',
     ylab = 'dweibull')
# distribution function
plot(fitted$pweibull,
     xlab = 'x',
     ylab = 'pweibull')
# quantile function
plot(fitted$qweibull,
     xlab = 'x',
     ylab = 'qweibull')
# sample from theoretical distribution
summary(fitted$rweibull(100))
# estimated parameters from MLE
fitted$parameters
```

---

fit\_univariate\_man      *Fit Univariate Distributions by Specifying Parameters*

---

## Description

Fit Univariate Distributions by Specifying Parameters

## Usage

```
fit_univariate_man(distribution, parameters)
```

## Arguments

distribution	distribution character name
parameters	named vector of parameters to set

## Value

list of distribution functions for d, p, q, r, and parameters

**Examples**

```

manFun <- fit_univariate_man('norm', c(mean = 2, sd = 5))
set.seed(5)
m1 <- mean(manFun$rnorm(100000))
set.seed(5)
m2 <- mean(rnorm(100000, 2, 5))
identical(m1, m2)

```

---

gen_dist_fun	<i>Generate Single Distribution Function</i>
--------------	--

---

**Description**

Generate Single Distribution Function

**Usage**

```
gen_dist_fun(f, parameters, ...)
```

**Arguments**

f	one of distribution functions
parameters	new parameters for distribution
...	arguments to pass on to distribution function

**Value**

one of parameterized distribution functions in d, p, q, r

---

GOFTests	<i>Wrappers to compute goodness of fit test froms distfun objects</i>
----------	---

---

**Description**

Wrappers to compute goodness of fit test froms distfun objects

**Usage**

```

ks_test(distfun, x, ...)

## S3 method for class 'distfun'
ad_test(distfun, x)

ad_test(distfun, x)

## S3 method for class 'distfun'
cvm_test(distfun, x)

cvm_test(distfun, x)

```

**Arguments**

distfun	a distfun object
x	numeric vector
...	arguments to be passed on to test function

**Value**

goodness of fit object

**Examples**

```

x <- rgamma(100, 1, 1)
fit <- fit_univariate(x, 'gamma')
ks_test(fit, x)
ad_test(fit, x)
cvm_test(fit, x)

```

---

gof\_tests

*Goodness of Fit Testing*


---

**Description**

Apply all goodness of fit tests and return a data.frame with the results

**Usage**

```
gof_tests(fits, x)
```

**Arguments**

fits	a list object produced from fit_univariate, fit_empirical, or fit_univariate_man
x	numeric vector of sample data

**Value**

a data.frame of test statistic results for each distribution

**Examples**

```
set.seed(84)
x <- rgamma(100, 1, 1)
dists <- c('gamma', 'lnorm', 'weibull')
multipleFits <- lapply(dists, fit_univariate, x = x)
gof_tests(multipleFits, x)
```

---

is.distfun	<i>Test if object is a distfun object</i>
------------	---

---

**Description**

Test if object is a distfun object

**Usage**

```
is.distfun(x)
```

**Arguments**

x                    an R object to be tested

**Value**

TRUE if x is a disfun object, FALSE otherwise

---

Mode	<i>Find Mode</i>
------	------------------

---

**Description**

Find Mode

**Usage**

```
Mode(x)
```

**Arguments**

x                    vector of data

**Value**

mode of data

---

plot_density	<i>Density Comparison Plot</i>
--------------	--------------------------------

---

**Description**

Density Comparison Plot

**Usage**

```
plot_density(x, fits, nbins)
```

**Arguments**

x	numeric vector of sample data
fits	a list object produced from fit_univariate, fit_empirical, or fit_univariate_man
nbins	number of bins for histogram

**Value**

ggplot of empirical histogram of x compared to theoretical density distributions

**Examples**

```
library(ggplot2)
set.seed(37)
x <- rgamma(10000, 5)
dists <- c('gamma', 'lnorm', 'weibull')
fits <- lapply(dists, fit_univariate, x = x)
plot_density(x, fits, 30) +
  theme_bw()
```

---

plot_pp	<i>P-P Plot</i>
---------	-----------------

---

**Description**

P-P Plot

**Usage**

```
plot_pp(x, fits)
```

**Arguments**

x	numeric vector of sample data
fits	a list object produced from fit_univariate, fit_empirical, or fit_univariate_man

**Value**

ggplot of percentile-percentile comparison of theoretical distribution

**Examples**

```
library(ggplot2)
set.seed(37)
x <- rgamma(10000, 5)
dists <- c('gamma', 'lnorm', 'weibull')
fits <- lapply(dists, fit_univariate, x = x)
plot_pp(x, fits) +
  theme_bw()
```

---

plot\_qq

*Q-Q Plot*

---

**Description**

Q-Q Plot

**Usage**

```
plot_qq(x, fits)
```

**Arguments**

**x** numeric vector of sample data  
**fits** a list object produced from `fit_univariate`, `fit_empirical`, or `fit_univariate_man`

**Value**

ggplot of quantile-quantile comparison of theoretical distribution

**Examples**

```
library(ggplot2)
set.seed(37)
x <- rgamma(10000, 5)
dists <- c('gamma', 'lnorm', 'weibull')
fits <- lapply(dists, fit_univariate, x = x)
plot_qq(x, fits) +
  theme_bw()
```

# Index

`ad_test` (GOFTests), 8

`build_dist`, 2

`calc_moments`, 3

`cvm_test` (GOFTests), 8

`ddunif` (DiscreteUniform), 3

DiscreteUniform, 3

`fit_dist_addin`, 4

`fit_empirical`, 4

`fit_univariate`, 6

`fit_univariate_man`, 7

`gen_dist_fun`, 8

`gof_tests`, 9

GOFTests, 8

`is.distfun`, 10

`ks_test` (GOFTests), 8

Mode, 10

`pdunif` (DiscreteUniform), 3

`plot_density`, 11

`plot_pp`, 11

`plot_qq`, 12

`qdunif` (DiscreteUniform), 3

`rdunif` (DiscreteUniform), 3