

# Package ‘autocogs’

February 12, 2019

**Title** Automatic Cognition Summaries

**Version** 0.1.2

**Description** Automatically calculates cognition groups for plot objects and list column plot objects. Results are returned in a nested data frame.

**Depends** R (>= 3.4.0)

**License** MIT + file LICENSE

**Encoding** UTF-8

**LazyData** true

**Imports** dplyr, checkmate, MASS, broom, moments, diptest, mclust, ggplot2, tibble, utils, hexbin, progress

**RoxygenNote** 6.1.1

**Collate** 'autocog.R' 'known\_cog\_groups.R' 'add\_cog\_group.R' 'field\_info.R' 'add\_cog\_group\_.R' 'known\_layer\_cogs.R' 'add\_layer\_cogs.R' 'add\_layer\_cogs\_.R' 'cog\_desc.R' 'cog\_spec.R' 'layer\_count.R' 'layer\_info.R' 'of\_type.R' 'plot\_class.R' 'plot\_cogs.R'

**Suggests** testthat, covr

**URL** <https://github.com/schloerke/autocogs>

**BugReports** <https://github.com/schloerke/autocogs/issues>

**NeedsCompilation** no

**Author** Barret Schloerke [aut],  
Ryan Hafen [ths, cre]

**Maintainer** Ryan Hafen <rhafen@gmail.com>

**Repository** CRAN

**Date/Publication** 2019-02-12 00:03:28 UTC

**R topics documented:**

add_cog_group	2
add_layer_cogs	3
autocog	3
autocog_	4
cog_desc	5
cog_group	5
cog_spec	6
field_info	7
known_cog_groups	8
known_layer_cogs	8
layer_count	8
layer_info	9
panel_cogs	10
plot_class	10
<b>Index</b>	<b>12</b>

---

add_cog_group	<i>Add a cognostic group</i>
---------------	------------------------------

---

**Description**

Add a new cognostic to be used when calculating automatic cognostics.

**Usage**

```
add_cog_group(name, fields, description = NA, fn, ...)
```

**Arguments**

name	Name of cognostic group
fields	data.frame of 'dimension' and 'type' columns. <code>dplyr::bind_rows()</code> of <code>field_info</code> outputs for convenience
description	Description of cognostic group
fn	function to calculate a cognostic group. May return a named list or a single row tibble. Each value of the return data should be the output of <code>cog_desc</code>
...	ignored

---

add_layer_cogs	<i>Add plot layer cognostics</i>
----------------	----------------------------------

---

**Description**

Add a new set of cognostic groups for a given plot layer. If the plot layer is found, the corresponding cognostic groups will be calculated.

**Usage**

```
add_layer_cogs(name, description, cog_groups, kind = "ggplot", ...)
```

**Arguments**

name	Name of plot layer. This should match the output of the "name" values of <a href="#">layer_info</a>
description	Description of cognostic group
cog_groups	A data.frame (or tibble) containing the columns: "cog_group", "cols", "name". "cog_group" column should contain a string value of a known cognostic group. "cols" should be a single value or vector of column names to use from the data supplied by <a href="#">layer_info</a> during calculations. "name" should contain the final storage name of the cognostic group.
kind	String value that will match the output of <a href="#">plot_class</a> of the desired plot object
...	ignored

---

autocog	<i>Auto cognostic function</i>
---------	--------------------------------

---

**Description**

Calculate an auto cognostic function given a name

**Usage**

```
autocog(.name, ..., .fn_only = FALSE)
```

**Arguments**

.name	name of a known cognostic
...	arguments passed onto the found function
.fn_only	boolean that determines if the function should be returned

**Examples**

```
autocog("univariate_continuous", iris$Sepal.Length)
fn <- autocog("univariate_continuous", .fn_only = TRUE)
fn(iris$Sepal.Length)
```

**Description**

These set of functions comprise the default cognition groups. Each function produces it's own cognition information given the required pieces of data.

The functions' print method will display the description. autocog\_\* functions will take the `known_cog_groups()` functions and format the output into a single row tibble. Any new known cognition group function, NAME, will create a function called autocog\_NAME, which may be called.

Default Cognition Group Functions:

- autocog\_bivariate\_continuous
- autocog\_bivariate\_counts
- autocog\_bivariate\_step
- autocog\_boxplot
- autocog\_density\_2d\_continuous
- autocog\_density\_continuous
- autocog\_grouped\_counts
- autocog\_grouped\_testing
- autocog\_hex\_counts
- autocog\_histogram\_counts
- autocog\_linear\_model
- autocog\_loess\_model
- autocog\_pairwise\_counts
- autocog\_quantile\_quantile
- autocog\_scagnostics
- autocog\_smooth\_line
- autocog\_square\_counts
- autocog\_univariate\_continuous
- autocog\_univariate\_counts
- autocog\_univariate\_discrete

**Arguments**

x	data that should appear on an x axis
y	data that should appear on an y axis
...	ignored
direction	step direction. Defaults to "hv"
na.rm	should NA points be removed when performing calculations
h, n, bins, binwidth, clusters, bw, adjust, kernel, trim, group, groups, center, boundary, closed,	parameters usually set by corresponding "geoms" to be used within ggplot2
	Stat* methods

**See Also**

[known\\_cog\\_groups\(\)](#)

**Examples**

```
autocog_bivariate_continuous
autocog_bivariate_continuous(iris$Sepal.Length, iris$Sepal.Width)
```

---

cog_desc	<i>Cognostic</i>
----------	------------------

---

**Description**

Add a description to a cognostic (subset metric)

**Usage**

```
cog_desc(x, desc = NULL)
```

**Arguments**

x	univariate scalar
desc	description of x

**Examples**

```
cog_desc(mean(1:10), "mean of 10 numbers")
```

---

cog_group	<i>Cog group data frame</i>
-----------	-----------------------------

---

**Description**

Make a cog group data frame to be passed into [add\\_layer\\_cogs](#)

**Usage**

```
cog_group(...)
```

**Arguments**

...	sets of three values to fill in 'cog_group', 'cols', and 'name'
-----	---

**Examples**

```

cog_group(
  "univariate_discrete", "x", "_x",
  "univariate_counts", "x", "_n"
)
cog_group(
  "univariate_continuous", "x", "_x",
  "univariate_continuous", "y", "_y",
  "bivariate_continuous", c("x", "y"), "_bivar",
  "scagnostics", c("x", "y"), "_scagnostic",
  "bivariate_counts", c("x", "y"), "_n"
)

```

---

cog\_spec

*Cognostic Specification*


---

**Description**

Cognostic Specification

**Usage**

```

cog_spec(bivariate_continuous = TRUE, bivariate_counts = TRUE,
  bivariate_step = TRUE, boxplot = TRUE,
  density_2d_continuous = TRUE, density_continuous = TRUE,
  grouped_counts = TRUE, grouped_testing = TRUE, hex_counts = TRUE,
  histogram_counts = TRUE, linear_model = TRUE, loess_model = TRUE,
  pairwise_counts = TRUE, quantile_quantile = TRUE,
  scagnostics = TRUE, smooth_line = TRUE, square_counts = TRUE,
  univariate_continuous = TRUE, univariate_counts = TRUE,
  univariate_discrete = TRUE, ..., .keep_layer = TRUE)

```

```
as_cog_specs(p, specs)
```

**Arguments**

`bivariate_continuous`, `bivariate_counts`, `bivariate_step`, `boxplot`, `density_2d_continuous`, `density_continuous`, `grouped_counts`, `grouped_testing`, `hex_counts`, `histogram_counts`, `linear_model`, `loess_model`, `pairwise_counts`, `quantile_quantile`, `scagnostics`, `smooth_line`, `square_counts`, `univariate_continuous`, `univariate_counts`, `univariate_discrete`, `...`, `.keep_layer` names of cognostic groups to calculate. The boolean value (TRUE) supplied to each argument determines if the value should be displayed if possible or removed if possible.

`...` ignored. Will cause error if any are supplied

`.keep_layer` boolean (TRUE) that determines if the layer should be kept at all

`p` plot object in question

`specs` list of cog\_spec outputs for each layer of the plot object

**Value**

cognostic specification that determines which cogs are added or removed if possible

**Examples**

```
# example cog specifications
# display like normal
cog_spec(); TRUE
# remove scagnostics
cog_spec(scagnostics = FALSE)
# remove layer
cog_spec(.keep_layer = FALSE); FALSE

# set up data
p <- ggplot2::qplot(Sepal.Length, Sepal.Width, data = iris, geom = c("point", "smooth"))
dt <- tibble::data_frame(geom = list(p))

# compute cognostics like normal
add_panel_cogs(dt)

# do not compute scagnostics for geom_point cognostics
# compute geom_smooth cognostics
add_panel_cogs(dt, spec = list(cog_spec(scagnostics = FALSE), TRUE))

# do not compute scagnostics for geom_point cognostics
# do not compute geom_smooth cognostics
add_panel_cogs(dt, spec = list(cog_spec(scagnostics = FALSE), FALSE))
```

---

field\_info

*Field Type Information*


---

**Description**

Field Type Information

**Usage**

```
field_info(dimension = c("x", "y", "z", "group", "any"),
           type = c("continuous", "discrete", "date", "any"))
```

**Arguments**

dimension      field name. Use one of the listed options provided  
type            field type. Use one of the listed options provided

---

known_cog_groups	<i>Cognostic Group information</i>
------------------	------------------------------------

---

**Description**

To add more cognostic groups, please see [add\\_cog\\_group\(\)](#)

**Usage**

```
known_cog_groups()
```

**Examples**

```
known_cog_groups()
```

---

known_layer_cogs	<i>Layer Cognostic groups</i>
------------------	-------------------------------

---

**Description**

Display all layer cognostic information to be paired with information from [known\\_cog\\_groups\(\)](#).

**Usage**

```
known_layer_cogs()
```

**Examples**

```
known_layer_cogs()
```

---

layer_count	<i>Plot layer count</i>
-------------	-------------------------

---

**Description**

Retrieves the number of layers in a given plot

**Usage**

```
layer_count(p)
```

```
## Default S3 method:
```

```
layer_count(p)
```

```
## S3 method for class 'ggplot'
```

```
layer_count(p)
```



**Arguments**

p plot object

**Value**

number

**Examples**

```
library(ggplot2)
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) + geom_point()
layer_count(p) # 1
layer_count(p + geom_smooth(method = "lm") + geom_density_2d()) # 3
```

---

layer_info	<i>Data List</i>
------------	------------------

---

**Description**

Data List

**Usage**

```
layer_info(p, keep = TRUE, ...)

## Default S3 method:
layer_info(p, keep = TRUE, ...)

## S3 method for class 'ggplot'
layer_info(p, keep = TRUE, ...)
```

**Arguments**

p plot object

keep boolean vector (size = 1 or length(plot\$layers)). Determines if that layer should have cognostics calculated

... parameters passed on to corresponding layer\_info

**Examples**

```
require(ggplot2)
p <- ggplot(iris, aes(Sepal.Length, Sepal.Width)) +
  geom_point(data = mpg, mapping = aes(cty, hwy))
layer_info(p)
```

---

panel_cogs	<i>Panel diagnostics</i>
------------	--------------------------

---

**Description**

Return or concatenate panel diagnostics. For each panel (plot) in the panel column, diagnostics will be calculated for each panel. The result will be returned in a nested [tibble](#).

**Usage**

```
panel_cogs(dt, panel_col = "panel", ...)
add_panel_cogs(dt, panel_col = "panel", ...)
```

**Arguments**

dt	data to be used
panel_col	panel column to be used in dt
...	parameters passed to <a href="#">layer_info</a>

---

plot_class	<i>Plot class</i>
------------	-------------------

---

**Description**

First class of the plot object. Exception is ggplot2 as many objects are of class 'gg'

**Usage**

```
plot_class(p)

## Default S3 method:
plot_class(p)

## S3 method for class 'gg'
plot_class(p)

## S3 method for class 'ggplot'
plot_class(p)
```

**Arguments**

p	plot object to retrieve class from
---	------------------------------------

**Examples**

```
library(ggplot2)
p <- qplot(Sepal.Length, Sepal.Width, data = iris)
plot_class(p)
```

# Index

add\_cog\_group, [2](#), [8](#)  
add\_layer\_cogs, [3](#), [5](#)  
add\_panel\_cogs (panel\_cogs), [10](#)  
as\_cog\_specs (cog\_spec), [6](#)  
autocog, [3](#)  
autocog\_, [4](#)  
autocog\_bivariate\_continuous  
    (autocog\_), [4](#)  
autocog\_bivariate\_counts (autocog\_), [4](#)  
autocog\_bivariate\_step (autocog\_), [4](#)  
autocog\_boxplot (autocog\_), [4](#)  
autocog\_density\_2d\_continuous  
    (autocog\_), [4](#)  
autocog\_density\_continuous (autocog\_), [4](#)  
autocog\_grouped\_counts (autocog\_), [4](#)  
autocog\_grouped\_testing (autocog\_), [4](#)  
autocog\_hex\_counts (autocog\_), [4](#)  
autocog\_histogram\_counts (autocog\_), [4](#)  
autocog\_linear\_model (autocog\_), [4](#)  
autocog\_loess\_model (autocog\_), [4](#)  
autocog\_pairwise\_counts (autocog\_), [4](#)  
autocog\_quantile\_quantile (autocog\_), [4](#)  
autocog\_scagnostics (autocog\_), [4](#)  
autocog\_smooth\_line (autocog\_), [4](#)  
autocog\_square\_counts (autocog\_), [4](#)  
autocog\_univariate\_continuous  
    (autocog\_), [4](#)  
autocog\_univariate\_counts (autocog\_), [4](#)  
autocog\_univariate\_discrete (autocog\_),  
    [4](#)  
  
bind\_rows, [2](#)  
  
cog\_desc, [2](#), [5](#)  
cog\_group, [5](#)  
cog\_spec, [6](#)  
  
field\_info, [2](#), [7](#)  
  
known\_cog\_groups, [4](#), [5](#), [8](#), [8](#)  
  
known\_layer\_cogs, [8](#)  
  
layer\_count, [8](#)  
layer\_info, [3](#), [9](#), [10](#)  
  
panel\_cogs, [10](#)  
plot\_class, [3](#), [10](#)  
  
tibble, [10](#)