	Method	Argument Name	Argument Value	Argument info	Method info
CONSTRUCTOR – Object creation and assignment, first step					
g= **g(ind_row,ind_col)=**	**gramm(**	'x'	x variable	1D array/cellstr of length N, Matrix of size (N,M) , (N,1) cell of 1D arrays	Constructor for the class. Must be called first and result assigned to a variable. Use to provide the data to be plotted
		'y'	y variable	1D array of length N, Matrix of size (N,M) , (N,1) cell of 1D arrays	
		'z'	z variable	1D array of length N, Matrix of size (N,M) , (N,1) cell of 1D arrays	
		'label'	label text	1D array/cellstr of length N	
		'color'	color grouping/continuous variable	1D array/cellstr of length N	
		'lightness'	lightness grouping variable	1D array/cellstr of length N	
		'linestyle'	linestyle grouping variable	1D array/cellstr of length N	
		'marker'	marker grouping variable	1D array/cellstr of length N	
		'size'	size grouping variable	1D array/cellstr of length N	
		'row'	subplot row grouping variable	1D array/cellstr of length N | Use facet_ functions for more control	
		'column'	subplot column grouping variable	1D array/cellstr of length N	
		'group'	subgrouping variable	1D array/cellstr of length N	
		'subset'	selection variable	1D Logical array of length N	
		'ymin'	upper y interval (absolute)	1D array of length N	
		'ymax'	lower y interval (absolute)	1D array of length N	
SUBPLOTS/FACETING AND MULTIPLE FIGURES – Method calls, order indifferent					
g. **g(ind_row,ind_col).**	**facet_grid(**		row grouping variable	1D array/cellstr of length N	Use to provide data that will determine separation between subblots rows and columns. First argument provided will separate along rows, second will separate along columns
			column grouping variable	1D array/cellstr of length N	
		'scale'	'fixed'	Same x and y limits on all subplots	
			'free_x'	Same y limits on all subplots, same x limits within columns	
			'free_y'	Same x limits on all subplots, same y limits within rows	
			'free'	Same x limits within columns, same y limits within rows	
			'independent'	Independent limits on each plot	
		'space'	'fixed'	Same x and y axe size on all subplots	
			'free_x'	Axis width proportional to x limits (requires 'scale', 'free_x' or 'free')	
			'free_y'	Axis height proportional to y limits (requires 'scale', 'free_y' or 'free')	
			'free'	Axis width and height proportional to x and y limits (requires 'scale','free')	
		'column_labels'	true/false	Do we label subplot columns	
		'row_labels'	true/false	Do we label subplot rows	
		'force_ticks'	true/false	Do we override defaults and force ticks on all subplots	
	facet_wrap(		column grouping variable	1D array/cellstr of length N	Use to provide data that will determine separation between subblots columns, with a wrapping: a new row of subplots is created when ncols is reached
		'ncols'	4	After how many columns do we wrap and create a new row	
		'scale'	...	Same as argument in gramm facet_grid()	
		'column_labels'	true/false	Do we label subplot columns	
		'force_ticks'	true/false	Do we override defaults and force ticks on all subplots	
	fig(		figure grouping variable	1D array/cellstr of length N	Use to provide data that will determine separation between figures
DIRECT DATA VISUALIZATIONS – geom_ method calls, order indifferent					
	geom_point(	'dodge'	0.5		Represent raw data as points (supports color, lightness, marker, size)
		'alpha'	1	Set the alpha of points (0:fully transparent, 1: solid; no export)	
	geom_jitter(	'width'	0.2	How much are the points jittered in horizontal direction (in data units)	Represent raw data as jittered points, useful when lots of overlapping points, e.g. with discrete values (supports color, lightness, marker, size)
		'height'	0	How much are the points jittered in vertical direction (in data units)	
		'dodge'	0.5	When using multiple colors, use to dodge graphical elements between colors with the same x value	
		'alpha'	1	Set the alpha of points (0:fully transparent, 1: solid; no export)	
	geom_swarm(	'type'	'up'	Points are added to the swarm from low to high values	Represent raw data poins as a swarm / beeswarm : points are displayed at their correct Y position but are moved to the side so that they don't overlap
			'down'	Points are added to the swarm from high to low values	
			'fan'	Points are added to the swarm starting from the center out	
			'hex'	Points are added to the swarm on an hexagonal grid	
			'square'	Points are added to the swarm on a square grid	
		'corral'	'none'	The swarm can go beyond the width	
			'gutter'	Points of the swarm are all added but can't go beyond the width	
			'wrap'	Points of the swarm beyond width are mirrored horizontally around the limit	
			'random'	Points of the swarm beyond the width are added at a random horizontal position	
			'omit'	Points of the swarm beyond the width are not added (dangerous !)	
		'alpha'	1	Set the alpha of points (0:fully transparent, 1: solid)	
		'point_size'	3	Set the point size within the swarm, due to the high dependecy of swarm shape on point size this is separate from set_point_options()	
		'dodge'	0.7	When using multiple colors, use to dodge graphical elements between colors with the same x value	
		'width'	0.9	What is the width of the swarm (interacts with the 'corral' parameter)	
	geom_line(	'dodge'	0.5	When using multiple colors, use to dodge graphical elements between colors with the same x value	Represent raw data with lines (supports color, lightness, marker, size). If x and y are 1D arrays, all points within a group will be connected !
		'alpha'	1	Set the alpha of lines (0:fully transparent, 1: solid)	
		'stacked'	true/false	Stack lines from different groups and shade the area between lines (only supports simple array input for x and y)	
		'fill'	[] or numeric value	If a numeric value is set, shade the area between the provided value and lines	
		'fill_alpha'	0.4	Set the alpha of shaded areas	
		'ordered'	false/true	If set to true, x values within each line are ordered before plotting so that the line doesn't go back on itself	
	geom_raster(	'geom'	'point'	raster elements are points	Represents raw x data as a raster plot
			'line'	raster elements are lines	
	geom_bar(	'width'	0.6	Provide to set the width of errorbars	
		'dodge'	0.8	When using multiple colors, use to dodge graphical elements between colors with the same x value	
		'stacked'	true/false	Se to true to have bars placed at the same x stacked	
		'FaceColor'	'auto'	Any property of a patch() object. 'FaceColor' and 'EdgeColor' can be set to 'auto' in order to use gramm color	
		'fill'	'face'		
			'edge'		
			'all'		

Method	Argument Name	Argument Value	Argument info	Method info
		`'transparent'`		
geom_interval(	`'geom'`	`'area'`	Same 'geom' as in stat_summary()	
		`...`		Represent intervals provided by 'ymin' and 'ymax' data (error bars, area)
	`'width'`	`0.6`	Provide to set the width of bars and errorbars	
	`'dodge'`	`0.7`	When using multiple colors, use to dodge graphical elements between colors with the same x value	
geom_label(	`'dodge'`	`0`	When using multiple colors, use to dodge graphical elements between colors with the same x value	
	`'Color'`	`'auto'`	Color of the text, default is 'auto' in order for the text color to follow gramm color	
	`...`		Any property of a text() object. 'Color','BackgroundColor' and 'EdgeColor' can be set to 'auto' in order to use gramm color	

STATISTICAL VISUALIZATIONS – stat_ method calls, order indifferent

Method	Argument Name	Argument Value	Argument info	Method info
stat_summary(	`'type'`	`'ci'`	mean & 95% CI of the mean (assumes normal data)	
		`'bootci'`	mean & bootstrapped 95%CI of the mean	
		`'sem'`	mean and standard error of the mean	
		`'std'`	mean and standard deviation	
		`'quartile'`	median and quartiles	
		`'95percentile'`	median and 95% percentiles	
		`'fitnormalci'`	mean and 95% CI of the mean from fitted normal distribution	
		`'fitpoissonci'`	mean and 95% CI of the mean from fitted Poisson distribution	
		`'fitbinomialci'`	mean and 95% CI of the mean from fitted binomial distribution	
		`function handle`	Provide a function to compute custom values (see doc)	Represents summarized Y data per unique values of X. By default, it will group all Y values that have the same X value, compute the summary variables of interest ('type' argument), and plot it according to the 'geom' argument.
	`'geom'`	`'area'`	means connected by a line, CI as shaded transparent area	
		`'lines'`	means connected by a line, CI as thin lines	
		`'line'`	means connected by a line	If X and Y are provided as 1D arrays but X values are not discrete enough, it is possible to compute the Y summaries over X bins with the 'bin_in' argument
		`'solid_area'`	means connected by a line, CI as solid shaded area (use for vector exports in pre 2014b versions)	
		`'black_errorbar'`	CI as black errorbar	If X is provided as a matrix or a cell of arrays but every element has non-aligned X values, the argument 'interp_in' must be used to create aligned X values by interpolation over X.
		`'errorbar'`	CI as colored errorbar	
		`'bar'`	means as colored bars	
		`'point'`	means as points	
		`'area_only'`	CI as shaded transparent area, no line	
	`'setylim'`	`true/false`	Do we set the YLim for the subplot according to the summary or the data?	
	`'interp'`	`'linear'`	Provide to interpolate the output (corresponds to the methods argument of interp1). Use 'polar' for circular data.	
	`'interp_in'`	`100`	Provide to linearly interpolate the input over x (corresponds to number of x points). ⚠️ **Must be used when X and Y are given as a cell and X values are not aligned** ⚠️	
	`'bin_in'`	`10`	Provide to bin inputs over x values (corresponds to number of bins)	
	`'width'`	`0.6`	Provide to set the width of bars and errorbars	
	`'dodge'`	`0.7`	When using multiple colors, use to dodge graphical elements between colors with the same x value	
stat_smooth(	`'method'`	`'eilers'`	Smoother described in Eilers 2003 (default, fast)	
		`'smoothingspline'`	uses fit() from the curve fitting toolbox	
		`'moving' 'lowess' 'sgolay' …`	uses smooth() from the curve fitting toolbox	Represents smoothed Y data with confidence interval.
	`'lambda'`	`1000`	Smoothing parameter, depends on method, see documentation	
	`'npoints'`	`200`	Number of points over which the smooth is evaluated	
	`'geom'`	`...`	Same geom as in gramm stat_summary()	
stat_glm(	`'distribution'`	`'normal'`	Same argument as fitglm()	
		`...`		
	`'geom'`	`...`	Same geom as in gramm stat_summary()	Fits and displays generalized linear models to the data.
	`'fullrange'`	`true/false`	Do we display the fit over the whole x axis, or just on the range of the value used for the fit	
	`'disp_fit'`	`true/false`	Do we display the fitted equations (with pvals stars)	
stat_fit(	`'fun'`	`@(param1,param2,x)x.^param1+param2`	Anonymous function with parameters to fit as first arguments and x as last argument	
		`@(params,x)x.^params(1)+params(2)`	When using the statistics toolbox the format is different, parameters are given as a single vector	
	`'stats'`	`false/true`	Use the fitnlm() statistics toolbox when set to true instead of fit() from the curve fitting toolbox	
	`'StartPoint'`	`[param1_start param2_start]`	Array with starting values of parameters	Fits and displays a provided custom function to the data
	`'intopt'`	`'observation'`	95% bounds on a new observation (see option of predint())	
		`'functional'`	95% bounds for the fitted function	
	`'fullrange'`	`true/false`	Do we display the fit over the whole x axis, or just on the range of the value used for the fit	
	`'disp_fit'`	`true/false`	Do we display the fitted equations	
	`'geom'`	`...`	Same geom as in gramm stat_summary()	
stat_bin(	`'nbins'`	`30`	Number of bins	
	`'edges'`	`-20 : 0.5 : 20`	Edges ovf bins (overrides 'nbins')	
	`'geom'`	`'bar'`	Results as dodged bars	
		`'line'`	Results connected by a line	
		`'overlaid_bar'`	Results as overlaid bars (use transparency)	
		`'stacked_bars'`	Results as stacked bars	
		`'stairs'`	Results as stair line	
		`'point'`	Results as points	
	`'normalization'`	`'count'`		
		`...`	Same as 'Normalization' argument of histcounts()	
	`'fill'`	`'face'`		
		`'edge'`		
		`'all'`		
		`'transparent'`		
	`'width'`	`0.6`	Provide to specify width of bars	
	`'dodge'`	`0.7`	Provide to specify dodging between elements	
stat_cornerhist(	`'location'`		x (or y) location of the inset axis on the unity line of the parent	
	`'aspect'`	`0.3`	Aspect ratio (y/x) of the inset axis	Display an histogram of the x-y difference in an inset axis
	`'edges'`	`...`	Same options as stat_bin(). 'specifying edges is recommended, stacked_bar geom unsupported	
stat_density(	`'bandwidth'`		Same argument as ksdensity()	
	`'function'`	`'pdf'`		
		`...`	Same argument as ksdensity()	
	`'kernel'`	`'normal'`		
		`...`	Same argument as ksdensity()	
	`'npoints'`	`100`	How many points are used to plot the density	

2

Method	Argument Name	Argument Value	Argument info	Method info
	`'extra_x'`	`10`	Extend the x value range over which the density is evaluated	
stat_bin2d(	`'nbins'`	`[n_xbins n_ybins]`		
	`'edges'`	`{x_edges_array, y_edges_array}`		
	`'geom'`	`'image'`		
		`'contour'`		
stat_ellipse(	`'type'`	`'95percentile'`	Fit ellipse that contains 95% of the points (assuming bivariate normal)	
		`'ci'`	Fit ellipse that contains 95% of the bootstrapped xy means	
	`'geom'`	`'area'`	Plot the ellipse as a shaded area with outline	
		`'line'`	Just plot the outline of the ellipse	
	`patch_opts`			
stat_qq(	`'distribution'`	`makedist('Normal',0,1)`	Provide a theoretical distribution to plot x against using Matlab's makedist() function. Set to 'y' to plot x against y densities.	Quantile-quantile plot
stat_boxplot(	`'width'`	`0.6`	Width of boxes	Box and whisker plots of y data for each unique x value
	`'dodge'`	`0.7`	Dodging between boxes of different colors within unique x values	
	`'notch'`	`false`	Add notches at median ± 1.58 IQR /sqrt(N) to the boxplot	
stat_violin(	`'normalization'`	`'area'`	Equal violin areas	
		`'count'`	Areas proportional to point count	
		`'width'`	Equal violin widths	
	`'half'`	`false`	Same argument as stat_density()	
	`'bandwidth'`		Same argument as stat_density()	
	`'kernel'`	`'normal'`	Same argument as stat_density()	
	`'npoints'`	`100`	Same argument as stat_density()	
	`'extra_y'`	`0`	Same argument as stat_density()	
	`'fill'`	`'face'`	Same argument as stat_bin()	
	`'width'`	`0.6`		
	`'dodge'`	`0.7`		

ADDITIONAL GRAPHICAL ELEMENTS – geom_ method calls, order indifferent

Method	Argument Name	Argument Value	Argument info	Method info
geom_abline(	`'intercept'`	`0`	Single value or 1D array of length P	
	`'slope'`	`1`	Single value or 1D array of size P	
	`'style'`	`'k--'`	Single string or 1D cellstr of size P	
geom_vline(	`'xintercept'`	`1`	Single value or 1D array of size P	
	`'style'`	`'k--'`	Single string or 1D cellstr of size P	
geom_hline(	`'yintercept'`	`1`	Single value or 1D array of size P	
	`'style'`	`'k--'`	Single string or 1D cellstr of size P	
geom_funline(	`'fun'`	`@(x)exp(sin(x-pi))`	Anonymous function or cell of anonymous functions	
	`'style'`	`'k--'`	Single string or 1D cellstr of size P	
geom_polygon(	`'x'`	`{}`	Cell of vectors with vertices x coordinates, or cell of vectors with x polygon limits if y omitted. Length P	
	`'y'`	`{}`	Cell of vectors with vertices y coordinates, or cell of vectors with y polygon limits if x omitted. Length P	
	`'alpha'`	`0.2`	Single value or 1D array of length P	
	`'color'`	`[0 0 0]`	RGB: 1x3 vector or matrix of size Px3. Or color indices	
	`'line_color'`	`[0 0 0]`	RGB: 1x3 vector or matrix of size Px3. Or color indices	
	`'line_style'`	`{'none'}`	1D cell of length 1 or P	

OPTIONS AND CUSTOMIZATIONS – Method calls, order indifferent

Method	Argument Name	Argument Value	Argument info	Method info
set_names(	`'x'`	`'x axis legend'`	Legend for the x axes	
	`'y'`	`'y axis legend'`	Legend for the y axes	
	`'row'`	`'row legend'`	Title of the row legends (actual titles will be a combination of title and value)	
	`'column'`	`'column legend'`	Title of the column legends (actual titles will be a combination of title and value)	
	`'color'`	`'color legend'`	Title of the color legend (actual legend will use the values)	
	`...`		All other titles for the gramm() arguments	
set_title(		`'Title'`	Desired title	Call on individual gramm objects to set title. Call on array of gramm objects to set global title
	`'FontSize'`	`16`	Any text property 'Name',value pair	
set_polar(	`'closed'`	`true/false`	Do we connect the first and last points ?	
	`'maxy'`	`10`	Impose the max of the radial scale (default corresponds to the max of y values)	
set_stat_options(	`'alpha'`	`0.05`	Alpha-level for confidence intervals	
	`'nboot'`	`200`	Number of boostrap samples	
set_color_options(	`'map'`	`'lch'`	Default HCL-based colormap	
		`'matlab'`	Matlab's own post 2014b map	
		`'brewer1' 'brewer2' 'brewer3' 'brewer_pastel' 'brewer_dark' 'brewer_paired'`	colorbrewer2.org colormaps	
		`'d3_10' 'd3_20' 'd3_20b' 'd3_20c'`	d3.js colormaps	
		`[0.1 0 0 ; 0 0.2 0.9]`	Custom RGB colormap as Nx3 matrix N = n_colors x n_lightness Row ordering should be color#1/lightness#1 ; color#1/lightness#2 ; ... ; color#1/lightness#n ; color#2/lightness#1 ; ... ; color#n/lightness#n	
	`'n_color'`		number of color categories when using a custom RGB colormap	
	`'n_lightness'`		number of color categories when using a custom RGB colormap	
	`'legend'`	`'separate_gray'`	default for LCH colormap, shows colors and lightness in separate legends, lightness is displayed in a gray scale	
		`'separate'`	default for other colormaps, shows colors and lightness in separate legends, lightness is displayed using the first color	
		`'expand'`	displays all color/lightness combinations	
		`'merge'`	merge color legends with marker/line/size legends if the categories are the same	
	`'lightness_range'`	`[85 15]`		
	`'chroma_range'`	`[30 90]`		
	`'hue_range'`	`[25 385]`	Options for the HCL colormap generation	
	`'lightness'`	`65`		
	`'chroma'`	`75`		
set_point_options(	`'markers'`	`{'o' 's' 'd' '^' 'v' '>' '<' 'p' 'h' '*' '+' 'x'}`	Set order for marker categories	
	`'base_size'`	`5`	Set marker base size	
	`'step_size'`	`2`	Set size categories size increment	
	`'use_input'`	`false`	Set to true to use the actual values of size categories as marker	
	`'input_fun'`	`@(s)s`	when 'use_input' is set to true, provide a function to map category value to marker size	
set_line_options(	`'styles'`	`{'-' '--' ':' '-.'}`	Set order for line style categories	
	`...`		Same size options as set_point_options()	

Method	Argument Name	Argument Value	Argument info	Method info
set_order_options(	`'x'`	`1`	Values sorted in ascending order (default)	
		`0`	Keep order of appearance of values in the input	
		`-1`	Values sorted in descending order	This method allows to reorder each grouping variable. Supports all variables provided in the main gramm() call except y, also supports reordering of facets with 'row' and 'column'
		`[value1 value2 value3 …]` `{'value1' 'value2' 'value3' …}`	Values ordered according to the provided array/cell. If the provided data is a cell of strings, provide a cell of strings containing the unique categories in the desired order. Extra categories provided here will be ignored, missing categories will truncate the data.	
		`[index1 index2 index3 …]`	Values ordered according to the provided indices (indices correspond to indices in the **sorted** values array/cell)	
	`'color'`			
	`...`			
set_continuous_color(	`'colormap'`	`'viridis'`	Set continuous colormap by name (Matlab defaults available)	
	`'active'`	`false`	Force continuous colors on or off if possible	
	`'LCH_colormap'`	`[L_start L_end ;` `C_start C_end ;` `H_start H_end ]`	Set continuous colormap definition in LCH colorspace	
	`'CLim'`	`[color_min color_max]`	Force color axis limits (automatic by default)	
set_text_options(	`'font'`	`'Helvetica'`	Font to use for all text	
	`'interpreter'`	`'none'`	Interpretation of text characters ('tex' / 'latex' / 'none')	
	`'base_size'`	`10`	Base text size, corresponds to axis ticks text size	
	`'label_scaling'`	`1`	Scaling of axis label sizes relative to base	
	`'legend_scaling'`	`1`	Scaling of legend label sizes relative to base	
	`'legend_title_scaling'`	`1.2`	Scaling of legend title sizes relative to base	
	`'facet_scaling'`	`1.2`	Scaling of facet title sizes relative to base	
	`'title_scaling'`	`1.4`	Scaling of facet title sizes relative to base	
	`'big_title_scaling'`	`1.4`	Scaling of overarching figure title size relative to base	
set_layout_options(	`'position'`	`'auto'` `[left bottom width height]`	Position of the plot in the figure. when set to 'auto', the position is set according to the indices of the gramm object in the matrix i.e. g(ind_row,ind_col). When set manually the indices of the gramm objects don't matter.	
	`'legend'`	`true/false`	Side legend (colors, markers, etc) on or off	
	`'legend_width'`	`'auto'` `0.2`	Proportion of the width of the plot occupied by the side legend	
	`'legend_position'`	`'auto'` `[left bottom width height]`	Detach side legend and place in the figure	
	`'title_centering'`	`'axes'` `'plot'`	Centering of plot title relative to axes or axes+legend	
	`'redraw'`	`true/false`	If 'redraw' is true, spacing is adjusted automatically after drawing and figure resizing in order to keep the plots tight. This can cause misaligments	
	`'redraw_gap'`	`0.04`	gap to use for automatic spacing	
	`'margin_height'`	`[bottom top]`		
	`'margin_width'`	`[left right]`	Adjust margins and gaps when 'redraw' is set to false	
	`'gap'`	`'auto'` `[width height]`		
axe_property(	`'axe_property'`	`axe_property_value`	Pass one or multiple name,value pairs for Axes Properties (XLim,XGrid, DataAspectRatio…)	
no_legend(				color/size/line/marker legend are not displayed
set_limit_extra(		`[0.05 0.05]`	How much do we extend limits of x axis (ratio wrt original limits)	
		`[0.05 0.05]`	How much do we extend limits of y axis (ratio wrt original limits)	
set_datetick(	`'x'`	`1`	Same arguments as datetick(): tickaxis,dateformat	
	`'y'`	`2`		
coord_flip(				Exchange the X and Y axes: use to generate horizontal plot elements (boxplots, violins…)

<div align="center">

DRAWING – Last method call

</div>

g.

Method	Argument Name	Argument Value	Argument info	Method info
draw(		`false`	Give false as (optional) argument to disable automatic setting of redraw() as resizing callback	Draw the plot ! Call on an array of gramm objects to draw all elements on the same figure. The plots are then located according to the row and column indices in the array)
redraw(		`0.05`	Redraw with custom spacing between elements (facets, legends)	

<div align="center">

SUPERIMPOSING MULTIPLE GRAMM PLOTS – After draw() call, allows new visualizations with new data

</div>

Method	Argument Name	Argument Value	Argument info	Method info
update(	`'color'`	`new color grouping variable`	update() takes the same type of arguments as gramm(). Provide the variables you want to change or add for the following layers. All the other variables will stay as defined by the first call to gramm().	Call update() after a first draw() call in order to change grouping variables for the next layers. Note that after an update() call it is also possible to update facets with facet_grid() or facet_wrap(). for facet updates, the only supported update is going from one facet to multiple ones, or from multiple facets to one: in each case, the layers drawn on the single facet will be copied to the other facets.
	`...`			

<div align="center">

FIGURE EXPORT – After draw() call

</div>

Method	Argument Name	Argument Value	Argument info	Method info
export(	`'file_name'`	`'gramm_export'`	Name of the exported file	
	`'export_path'`	`''`	Path of the destination folder (default is current folder)	
	`'file_type'`	`'svg'`	Format of the saved image	
		`'pdf' 'eps' 'png' 'jpg'`		
	`'width'`	`desired width`	Width of the saved image in 'units'	
	`'height'`	`desired height`	Height of the saved image in 'units'	
	`'units'`	`'centimeters'`	Units for the saved image dimensions	
		`'inches'`		