

# Package ‘deptime’

September 20, 2022

**Title** Plotting Tools for Anyone Working in Deep Time

**Version** 0.2.3

**Maintainer** William Gearty <willgearty@gmail.com>

**Description** Extends the functionality of other plotting packages like 'ggplot2' and 'lattice' to help facilitate the plotting of data over long time intervals, including, but not limited to, geological, evolutionary, and ecological data. The primary goal of 'deptime' is to enable users to add highly customizable timescales to their visualizations. Other functions are also included to assist with other areas of deep time visualization.

**URL** <https://github.com/willgearty/deptime>

**BugReports** <https://github.com/willgearty/deptime/issues>

**Depends** R (>= 3.4)

**License** GPL (>= 2)

**Encoding** UTF-8

**RoxygenNote** 7.2.1

**LazyData** true

**biocViews**

**Imports** ggplot2, ggnewscale, utils, ggforce, grid, gridExtra, gtable, methods, stats, lattice, rlang, scales, ggfittext, curl

**Suggests** tidyverse, divDyn, gsloid, phytools, paleotree, dispRity, ggtree, testthat (>= 3.0.0), vdiffr (>= 1.0.0),

**Config/testthat/edition** 3

**NeedsCompilation** no

**Author** William Gearty [aut, cre]

**Repository** CRAN

**Date/Publication** 2022-09-20 21:00:02 UTC

**R topics documented:**

coord_geo . . . . .	2
coord_trans_flip . . . . .	5
coord_trans_xy . . . . .	6
disparity_through_time . . . . .	7
eons . . . . .	9
epochs . . . . .	10
eras . . . . .	10
getScaleData . . . . .	11
ggarrange2 . . . . .	12
gggeo_scale . . . . .	13
gggeo_scale_old . . . . .	17
gtable_frame2 . . . . .	20
panel.disparity . . . . .	21
periods . . . . .	22
stages . . . . .	22

<b>Index</b>	<b>24</b>
--------------	-----------

---

coord_geo	<i>Transformed coordinate system with geological timescale</i>
-----------	--

---

**Description**

coord\_geo behaves similarly to [coord\\_trans](#) in that it occurs after statistical transformation and will affect the visual appearance of geoms. The main difference is that it also adds a geological timescale to the specified side of the plot.

**Usage**

```
coord_geo(
  pos = "bottom",
  dat = "periods",
  xlim = NULL,
  ylim = NULL,
  xtrans = identity_trans(),
  ytrans = identity_trans(),
  clip = "on",
  expand = FALSE,
  fill = NULL,
  color = "black",
  alpha = 1,
  height = unit(2, "line"),
  lab = TRUE,
  lab_color = NULL,
  rot = 0,
  abbrev = TRUE,
```

```

skip = c("Quaternary", "Holocene", "Late Pleistocene"),
size = 5,
lwd = 0.25,
neg = FALSE,
bord = c("left", "right", "top", "bottom"),
center_end_labels = FALSE,
dat_is_discrete = FALSE,
fittext_args = list()
)

```

### Arguments

pos	Which side to add the scale to (left, right, top, or bottom). First letter may also be used.
dat	Either A) a string indicating a built-in dataframe with interval data from the ICS ("periods", "epochs", "stages", "eons", or "eras"), B) a string indicating a timescale from macrostrat (see list here: <a href="https://macrostrat.org/api/defs/timescales?all">https://macrostrat.org/api/defs/timescales?all</a> ), or C) a custom data.frame of time interval boundaries (see Details).
xlim, ylim	Limits for the x and y axes.
xtrans, ytrans	Transformers for the x and y axes. For more information see <a href="#">coord_trans</a> .
clip	Should drawing be clipped to the extent of the plot panel? For more information see <a href="#">coord_trans</a> .
expand	If 'FALSE', the default, limits are taken exactly from the data or 'xlim'/'ylim'. If 'TRUE', adds a small expansion factor to the limits to ensure that data and axes don't overlap.
fill	The fill color of the boxes. The default is to use the color column included in dat. If a custom dataset is provided with dat without a color column and without fill, a greyscale will be used. Custom fill colors can be provided with this option (overriding the color column) and will be recycled if/as necessary.
color	The outline color of the interval boxes.
alpha	The transparency of the fill colors.
height	The height (or width if pos is left or right) of the scale.
lab	Whether to include labels.
lab_color	The color of the labels. The default is to use the lab_color column included in dat. If a custom dataset is provided with dat without a lab_color column and without fill, all labels will be black. Custom label colors can be provided with this option (overriding the lab_color column) and will be recycled if/as necessary.
rot	The amount of counter-clockwise rotation to add to the labels (in degrees).
abbrv	If including labels, whether to use abbreviations instead of full interval names.
skip	A vector of interval names indicating which intervals should not be labeled. If abbrv is TRUE, this can also include interval abbreviations.
size	Label size. Either a number as you would specify in <a href="#">geom_text</a> or "auto" to use <a href="#">geom_fit_text</a> .

lwd	Line width.
neg	Set this to true if your x-axis is using negative values.
bord	A vector specifying on which sides of the scale to add borders (same options as pos).
center_end_labels	Should labels be centered within the visible range of intervals at the ends of the axis?
dat_is_discrete	Are the ages in dat already converted for a discrete scale?
fittext_args	A list of named arguments to provide to <code>geom_fit_text</code> . Only used if size is set to "auto".

## Details

Transforming the side with the scale is not currently implemented. If a custom data.frame is provided (with `dat`), it should consist of at least 3 columns of data. See `data( periods )` for an example.

- The `name` column lists the names of each time interval. These will be used as labels if no abbreviations are provided.
- The `max_age` column lists the oldest boundary of each time interval.
- The `min_age` column lists the youngest boundary of each time interval.
- The `abbr` column is optional and lists abbreviations that may be used as labels.
- The `color` column is also optional and lists a `color` for the background for each time interval.
- The `lab_color` column is also optional and lists a `color` for the label for each time interval.

If the axis of the time scale is discrete, `max_age` and `min_age` will automatically be converted to the discrete scale. In this case, the categories of the discrete axis should match the values in the `name` column. If the ages within `dat` are already discretized, you can set `dat_is_discrete` to `TRUE` to prevent this automatic conversion. This can be useful for adding a time scale where categories and time intervals are not 1:1.

`pos` may also be a list of sides (including duplicates) if multiple time scales should be added to the plot. In this case, `dat`, `fill`, `color`, `alpha`, `height`, `lab`, `lab_color`, `rot`, `abbrv`, `skip`, `size`, `lwd`, `neg`, `bord`, `center_end_labels`, and `dat_is_discrete` can also be lists. If these lists are not as long as `pos`, the elements will be recycled. If individual values (or vectors) are used for these parameters, they will be applied to all time scales (and recycled as necessary).

## Examples

```
library(ggplot2)
#single scale on bottom
ggplot() +
  geom_point(aes(y = runif(1000, 0, 8), x = runif(1000, 0, 1000))) +
  scale_x_reverse() +
  coord_geo(xlim = c(1000, 0), ylim = c(0,8)) +
  theme_classic()

#stack multiple scales
```

```
ggplot() +
  geom_point(aes(y = runif(1000, 0, 8), x = runif(1000, 0, 100))) +
  scale_x_reverse() +
  coord_geo(xlim = c(100, 0), ylim = c(0,8), pos = as.list(rep("bottom", 3)),
  dat = list("stages", "epochs", "periods"),
  height = list(unit(4, "lines"), unit(4, "lines"), unit(2, "line")),
  rot = list(90, 90, 0), size = list(2.5, 2.5, 5), abbrev = FALSE) +
  theme_classic()
```

---

 coord\_trans\_flip

*Transformed and flipped Cartesian coordinate system*


---

### Description

coord\_trans\_flip behaves similarly to [coord\\_trans](#) in that it occurs after statistical transformation and will affect the visual appearance of geoms. The main difference is that it also flips the x and y coordinates like [coord\\_flip](#).

### Usage

```
coord_trans_flip(
  x = "identity",
  y = "identity",
  xlim = NULL,
  ylim = NULL,
  clip = "on",
  expand = TRUE
)
```

### Arguments

x, y	Transformers for x and y axes or their names.
xlim	Limits for the x and y axes.
ylim	Limits for the x and y axes.
clip	Should drawing be clipped to the extent of the plot panel? A setting of "on" (the default) means yes, and a setting of "off" means no. In most cases, the default of "on" should not be changed, as setting clip = "off" can cause unexpected results. It allows drawing of data points anywhere on the plot, including in the plot margins. If limits are set via xlim and ylim and some data points fall outside those limits, then those data points may show up in places such as the axes, the legend, the plot title, or the plot margins.
expand	If TRUE, the default, adds a small expansion factor to the limits to ensure that data and axes don't overlap. If FALSE, limits are taken exactly from the data or xlim/ylim.

**Examples**

```
library(ggplot2)
ggplot(mtcars, aes(displ, wt)) +
  geom_point() +
  coord_trans_flip(x = "log10", y = "log10")
```

---

 coord\_trans\_xy

*Transformed XY Cartesian coordinate system*


---

**Description**

coord\_trans\_xy behaves similarly to [coord\\_trans](#) in that it occurs after statistical transformation and will affect the visual appearance of geoms. The main difference is that it takes a single transformer that is applied to the x and y axes simultaneously. Any transformers produced by [linear\\_trans](#) that have x and y arguments should work, but any other transformers produced using [trans\\_new](#) that take x and y arguments should also work. Axis limits will be adjusted to account for transformation unless limits are specified with 'xlim' or 'ylim'. This only works with geoms where all points are defined with x and y coordinates (e.g. [geom\\_point](#), [geom\\_polygon](#)). This does not currently work with geoms where point coordinates are extrapolated (e.g. [geom\\_rect](#)).

**Usage**

```
coord_trans_xy(
  trans = NULL,
  xlim = NULL,
  ylim = NULL,
  expand = TRUE,
  default = FALSE,
  clip = "on"
)
```

**Arguments**

trans	Transformer for x and y axes.
xlim, ylim	Limits for the x and y axes.
expand	If 'TRUE', the default, adds a small expansion factor to the limits to ensure that data and axes don't overlap. If 'FALSE', limits are taken exactly from the data or 'xlim'/'ylim'.
default	Is this the default coordinate system? If 'FALSE' (the default), then replacing this coordinate system with another one creates a message alerting the user that the coordinate system is being replaced. If 'TRUE', that warning is suppressed.
clip	Should drawing be clipped to the extent of the plot panel? A setting of "on" (the default) means yes, and a setting of "off" means no. In most cases, the default of "on" should not be changed, as setting 'clip = "off"' can cause unexpected results. It allows drawing of data points anywhere on the plot, including in the plot margins. If limits are set via 'xlim' and 'ylim' and some data points fall outside those limits, then those data points may show up in places such as the axes, the legend, the plot title, or the plot margins.

**Examples**

```

#make transformer
library(ggforce)
trans <- linear_trans(shear(2, 0), rotate(-pi / 3))

#set up data to be plotted
square <- data.frame(x = c(0, 0, 4, 4), y = c(0, 1, 1, 0))
points <- data.frame(x = runif(100, 0, 4), y = runif(100, 0, 1))

#plot data normally
library(ggplot2)
ggplot(data = points, aes(x = x, y = y)) +
  geom_polygon(data = square, fill = NA, color = "black") +
  geom_point(color = 'black') +
  coord_cartesian(expand = FALSE) +
  theme_classic()

#plot data with transformation
ggplot(data = points, aes(x = x, y = y)) +
  geom_polygon(data = square, fill = NA, color = "black") +
  geom_point(color = 'black') +
  coord_trans_xy(trans = trans, expand = FALSE) +
  theme_classic()

```

---

disparity\_through\_time

*Disparity through time plot using lattice*

---

**Description**

Plots points on 2-D surfaces within a 3-D framework. See [wireframe](#) and [panel.cloud](#) for customization options.

**Usage**

```

disparity_through_time(
  x,
  data,
  groups,
  pch = 16,
  col.point = c("blue"),
  scales = list(arrows = FALSE, distance = 1, col = "black", z = list(rot = 90)),
  colorkey = FALSE,
  screen = list(z = 90, x = 70, y = 180),
  aspect = c(1.5, 4),
  drape = TRUE,
  col.regions = c("white"),
  alpha.regions = c(1),

```

```

perspective = FALSE,
R.mat = matrix(c(1, 1, 0, 0, 0, 1, 0, 0, 0, 0, 1, 0, 0, 0, 0, 1), 4, 4),
par.settings = list(axis.line = list(col = "transparent"), layout.heights =
  list(top.padding = 0, main.key.padding = 0, key.axis.padding = 0, axis.xlab.padding =
    0, xlab.key.padding = 0, key.sub.padding = 0, bottom.padding = 0), layout.widths =
  list(left.padding = 0, key.ylab.padding = 0, ylab.axis.padding = 0, axis.key.padding
    = 0, right.padding = 0)),
lattice.options = list(axis.padding = list(factor = 0)),
...
)

```

### Arguments

<code>x</code>	a formula (most likely of the form $z \sim x * y$ )
<code>data</code>	a data frame in which variables in the formula are to be evaluated
<code>groups</code>	a variable in data to be used as a grouping variable (this is probably the z variable)
<code>pch</code>	the point type
<code>col.point</code>	color(s) for points on surfaces
<code>scales</code>	a list specifying how the axes are drawn (see <a href="#">xyplot</a> for details)
<code>colorkey</code>	logical, should a legend be drawn (or a list describing the legend; see <a href="#">levelplot</a> for details)
<code>screen</code>	a list of the rotations that should be applied to each axis
<code>aspect</code>	a numeric vector of length 2, giving the relative aspects of the y-size/x-size and z-size/x-size of the enclosing cube
<code>drape</code>	logical, whether the surfaces should be colored based on <code>col.regions</code> and <code>alpha.regions</code>
<code>col.regions</code>	color(s) for surfaces
<code>alpha.regions</code>	alpha value(s) for surfaces
<code>perspective</code>	logical, whether to plot a perspective view
<code>R.mat</code>	a transformational matrix that is applied to the orientation of the axes
<code>par.settings</code>	plotting settings (see <a href="#">trellis.par.set</a> )
<code>lattice.options</code>	lattice settings (see <a href="#">lattice.options</a> )
<code>...</code>	Other arguments passed to <a href="#">wireframe</a>

### Value

An object of class "trellis", as output by [wireframe](#).



## Examples

```
g <- data.frame(x = runif(100, 0, 60), y = runif(100,0,10),
               z = factor(rep( periods$name[1:5], each=20),
                           levels = periods$name[1:5]))
disparity_through_time(z~x*y, data = g, groups = z, aspect = c(1.5,2),
                      xlim = c(0,60), ylim = c(0,10), col.regions = "lightgreen",
                      col.point = c("red","blue"))
```

---

eons	<i>Eon data from the International Commission on Stratigraphy (v2022/02)</i>
------	--

---

## Description

A dataset containing the boundary ages, abbreviations, and colors for the eons of the Geologic Time Scale. Based on The ICS International Chronostratigraphic Chart (v2022/02), by Cohen, Finney, Gibbard, and Fan.

## Usage

```
eons
```

## Format

A data frame with 3 rows and 5 variables:

**name** eon name

**max\_age** maximum age, in millions of years

**min\_age** minimum age, in millions of years

**abbr** eon name abbreviations

**color** the colors for each eon, according to the Commission for the Geological Map of the World

## Source

<https://stratigraphy.org> via <https://macrostrat.org/api/v2/defs/intervals?timescale=international%20eons>

## See Also

Other timescales: [epochs](#), [eras](#), [periods](#), [stages](#)

---

epochs *Epoch data from the International Commission on Stratigraphy (v2022/02)*

---

### Description

A dataset containing the boundary ages, abbreviations, and colors for the epochs of the Geologic Time Scale. Based on The ICS International Chronostratigraphic Chart (v2022/02), by Cohen, Finney, Gibbard, and Fan.

### Usage

epochs

### Format

A data frame with 34 rows and 5 variables:

**name** epoch name

**max\_age** maximum age, in millions of years

**min\_age** minimum age, in millions of years

**abbr** epoch name abbreviations

**color** the colors for each epoch, according to the Commission for the Geological Map of the World

### Source

<https://stratigraphy.org> via <https://macrostrat.org/api/v2/defs/intervals?timescale=international%20epochs>

### See Also

Other timescales: [eons](#), [eras](#), [periods](#), [stages](#)

---

eras *Era data from the International Commission on Stratigraphy (v2022/02)*

---

### Description

A dataset containing the boundary ages, abbreviations, and colors for the eras of the Geologic Time Scale. Based on The ICS International Chronostratigraphic Chart (v2022/02), by Cohen, Finney, Gibbard, and Fan.

### Usage

eras

**Format**

A data frame with 10 rows and 5 variables:

**name** era name

**max\_age** maximum age, in millions of years

**min\_age** minimum age, in millions of years

**abbr** era name abbreviations

**color** the colors for each era, according to the Commission for the Geological Map of the World

**Source**

<https://stratigraphy.org> via <https://macrostrat.org/api/v2/defs/intervals?timescale=international%20eras>

**See Also**

Other timescales: [eons](#), [epochs](#), [periods](#), [stages](#)

---

getScaleData	<i>Get geological timescale data</i>
--------------	--------------------------------------

---

**Description**

This function takes a name of a geological timescale and returns data for the timescale. Valid names include those of built-in data.frames ([periods](#), [epochs](#), [stages](#), [eons](#), or [eras](#)), partial matches of those names (e.g., "per" or "stage"), and exact matches to those hosted by Macrostrat (see full list here: <https://macrostrat.org/api/defs/timescales?all>).

**Usage**

```
getScaleData(name)
```

**Arguments**

**name** The name of the desired timescale.

**Value**

A data.frame with the following columns:

**name** the names of the time intervals.

**max\_age** the oldest boundaries of the time intervals, in millions of years.

**min\_age** the youngest boundaries of the time intervals, in millions of years.

**abbr** either traditional abbreviations of the names of the time intervals (if they exist) or custom abbreviations created with R.

**color** hex color codes associated with the time intervals (if applicable).

---

 ggarrange2

*ggarrange2*


---

### Description

Arrange multiple ggplot, grobified ggplot, or geo\_scale objects on a page, aligning the plot panels, axes, and axis titles.

### Usage

```
ggarrange2(
  ...,
  plots = list(...),
  layout = NULL,
  nrow = NULL,
  ncol = NULL,
  widths = NULL,
  heights = NULL,
  byrow = TRUE,
  top = NULL,
  bottom = NULL,
  left = NULL,
  right = NULL,
  padding = unit(0.5, "line"),
  margin = unit(0.5, "line"),
  clip = "on",
  draw = TRUE,
  newpage = TRUE,
  debug = FALSE,
  labels = NULL,
  label.args = list(gp = gpar(font = 4, cex = 1.2))
)
```

### Arguments

...	ggplot, grobified ggplot (gtable), or geo_scale objects
plots	list of ggplot, gtable, or geo_scale objects
layout	a matrix of integers specifying where each plot should go, like mat in <a href="#">layout</a> ; NA or a value less than 0 or greater than the number of plots indicates a blank plot; overrides nrow/ncol/byrow
nrow	number of rows
ncol	number of columns
widths	list of requested widths
heights	list of requested heights
byrow	logical, fill by rows

top	optional string, or grob
bottom	optional string, or grob
left	optional string, or grob
right	optional string, or grob
padding	unit of length one, margin around annotations
margin	vector of units of length 4: top, right, bottom, left (as in <a href="#">gtable_add_padding</a> )
clip	argument of gtable
draw	logical: draw or return a grob
newpage	logical: draw on a new page
debug	logical, show layout with thin lines
labels	character labels used for annotation of subfigures (should be in the same order as plots)
label.args	label list of parameters for the formatting of labels

**Value**

gtable of aligned plots

**Examples**

```
library(ggplot2)
p1 <- ggplot(mtcars, aes(mpg, wt, colour = factor(cyl))) +
  geom_point()
p2 <- ggplot(mtcars, aes(mpg, wt, colour = factor(cyl))) +
  geom_point() + facet_wrap(~ cyl, ncol=2, scales = 'free') +
  guides(colour='none') +
  theme()
ggarrange2(p1, p2, widths = c(2,1), labels = c('a', 'b'))

p3 <- ggplot() +
  geom_point(aes(y = runif(1000, 0, 8), x = runif(1000, 0, 1000))) +
  scale_x_reverse() +
  coord_geo(xlim = c(1000, 0), ylim = c(0,8)) +
  theme_classic()
ggarrange2(ggarrange2(p1, p2, widths = c(2,1), draw = FALSE), p3, nrow = 2)
```

---

gggeo\_scale

---

*Add a geologic scale to ggplots*


---

**Description**

This function takes a ggplot object and adds a geologic time scale at the specified side.

**Usage**

```
gggeo_scale(obj, ...)  
  
## S3 method for class 'gtable'  
gggeo_scale(  
  obj,  
  lims,  
  dat = "periods",  
  fill = NULL,  
  color = "black",  
  alpha = 1,  
  height = unit(2, "line"),  
  pos = "bottom",  
  lab = TRUE,  
  rot = 0,  
  abbrev = TRUE,  
  skip = c("Quaternary", "Holocene", "Late Pleistocene"),  
  size = 5,  
  lwd = 0.25,  
  margin = unit(0, "line"),  
  neg = FALSE,  
  bord = c("left", "right", "top", "bottom"),  
  center_end_labels = FALSE,  
  ...  
)  
  
## S3 method for class 'ggplot'  
gggeo_scale(  
  obj,  
  dat = "periods",  
  fill = NULL,  
  color = "black",  
  alpha = 1,  
  height = unit(2, "line"),  
  pos = "bottom",  
  lab = TRUE,  
  rot = 0,  
  abbrev = TRUE,  
  skip = c("Quaternary", "Holocene", "Late Pleistocene"),  
  size = 5,  
  lwd = 0.25,  
  margin = unit(0, "line"),  
  neg = FALSE,  
  bord = c("left", "right", "top", "bottom"),  
  center_end_labels = FALSE,  
  ...  
)
```

```

## S3 method for class 'geo_scale'
gggeo_scale(
  obj,
  dat = "periods",
  fill = NULL,
  color = "black",
  alpha = 1,
  height = unit(2, "line"),
  pos = "bottom",
  lab = TRUE,
  rot = 0,
  abbrev = TRUE,
  skip = c("Quaternary", "Holocene", "Late Pleistocene"),
  size = 5,
  lwd = 0.25,
  margin = unit(0, "line"),
  neg = FALSE,
  bord = c("left", "right", "top", "bottom"),
  center_end_labels = FALSE,
  ...
)

## S3 method for class 'geo_scale'
print(x, ...)

```

### Arguments

obj	An object of class ggplot, gtable, or geo_scale (as produced by this function).
...	further arguments passed to <code>grid.draw</code> .
lims	The limits of the axis of the desired side of the plot. Only required if using a gtable object not created by this function.
dat	Either A) a string indicating a built-in dataframe with interval data from the ICS ("periods", "epochs", "stages", "eons", or "eras"), B) a string indicating a timescale from macrostrat (see list here: <a href="https://macrostrat.org/api/defs/timescales?all">https://macrostrat.org/api/defs/timescales?all</a> ), or C) a custom dataframe of time interval boundaries (see Details).
fill	The fill color of the boxes. The default is to use the colors included in dat. If a custom dataset is provided with dat without color and without fill, a greyscale will be used. Custom fill colors can be provided with this option and will be recycled if/as necessary.
color	The outline color of the interval boxes.
alpha	The transparency of the fill colors.
height	The height (or width if pos is left or right) of the scale.
pos	Which side to add the scale to (left, right, top, or bottom). First letter may also be used.

lab	Whether to include labels.
rot	The amount of counter-clockwise rotation to add to the labels (in degrees).
abbrv	If including labels, whether to use abbreviations instead of full interval names.
skip	A vector of interval names indicating which intervals should not be labeled.
size	Label size.
lwd	Line width.
margin	The width of the margin around the returned object (can be a vector of length 4).
neg	Set this to true if your x-axis is using negative values.
bord	A vector specifying on Which sides of the scale to add borders (same options as pos).
center_end_labels	Should labels be centered within the visible range of intervals at the ends of the axis?
x	An object of class <code>geo_scale</code> .

### Details

If custom data is provided (with `dat`), it should consist of at least 3 columns of data. See `data( periods )` for an example. The `name` column lists the names of each time interval. These will be used as labels if no abbreviations are provided. The `max_age` column lists the oldest boundary of each time interval. The `min_age` column lists the youngest boundary of each time interval. The `abbr` column is optional and lists abbreviations that may be used as labels. The `color` column is also optional and lists a hex color code (which can be obtained with `rgb()`) for each time interval.

### Value

A `geo_scale` object. Basically a `gtable` object but with the axis limits included.

### Examples

```
library(ggplot2)
# bottom scale by default
p <- ggplot() +
  geom_point(aes(y = runif(1000, 0, 8), x = runif(1000, 0, 1000))) +
  scale_x_reverse() +
  coord_cartesian(xlim = c(1000, 0), ylim = c(0, 8), expand = FALSE) +
  theme_classic()
gggeo_scale(p)

# can specify any side of the plot
p <- ggplot() +
  geom_point(aes(x = runif(1000, 0, 8), y = runif(1000, 0, 1000))) +
  scale_y_reverse() +
  coord_cartesian(xlim = c(0, 8), ylim = c(1000, 0), expand = FALSE) +
  theme_classic()
gggeo_scale(p, pos = "left", rot = 90)
```



```

# can add multiple scales
p <- ggplot() +
  geom_point(aes(y = runif(1000, 0, 8), x = runif(1000, 0, 100))) +
  scale_x_reverse() +
  coord_cartesian(xlim = c(100, 0), ylim = c(0, 8), expand = FALSE) +
  theme_classic()
p <- gggeo_scale(p, abbrv = FALSE)
p <- gggeo_scale(p, dat = "epochs", height = unit(4, "lines"), rot = 90, size = 2.5, abbrv = FALSE)
gggeo_scale(p, dat = "stages", height = unit(4, "lines"), rot = 90, size = 2.5, abbrv = FALSE)

# intervals on both sides for different timescales (ICS stages vs North American Land Mammal Ages)
p <- ggplot() +
  geom_point(aes(x = runif(1000, 0, 10), y = runif(1000, 0, 65))) +
  scale_y_reverse() +
  coord_cartesian(xlim = c(0, 10), ylim = c(65, 0), expand = FALSE) +
  theme_classic()
p <- gggeo_scale(p, dat = "stages", pos = "left", height = unit(4, "lines"), size = 2.5,
  abbrv = FALSE)
gggeo_scale(p, dat = "North American Land Mammal Ages", pos = "right", height = unit(4, "lines"),
  size = 2.5, abbrv = FALSE)

#can add scales to a faceted plot
#use gggeo_scale_old() if you have more than one column
df <- data.frame(x = runif(1000, 0, 541), y = runif(1000, 0, 8),
  z = sample(c(1, 2, 3, 4), 1000, TRUE))
p <- ggplot(df) +
  geom_point(aes(x, y)) +
  scale_x_reverse() +
  coord_cartesian(xlim = c(541, 0), ylim = c(0, 8), expand = FALSE) +
  theme_classic() +
  facet_wrap(~z, ncol = 1)
gggeo_scale(p)

#can even add a scale to a phylogeny (using ggtree)

library(phytools)
library(ggtree)
tree <- pbtree(b = .03, d = .01, n=100)
p <- ggtree(tree) +
  coord_cartesian(xlim = c(-500, 0), ylim = c(-2, Ntip(tree)), expand = FALSE) +
  scale_x_continuous(breaks=seq(-500, 0, 100), labels=abs(seq(-500, 0, 100))) +
  theme_tree2()
p <- revts(p)
gggeo_scale(p, neg = TRUE)

```

---

gggeo\_scale\_old

Add a geologic scale to ggplots (old version)

---

## Description

This function takes a ggplot object and adds a geologic time scale at the specified side.

**Usage**

```
gggeo_scale_old(
  gg,
  dat = "periods",
  fill = NULL,
  color = "black",
  alpha = 1,
  height = 0.05,
  gap = 0,
  pos = "bottom",
  lab = TRUE,
  rot = 0,
  abbrev = TRUE,
  skip = c("Quaternary", "Holocene", "Late Pleistocene"),
  size = 5,
  neg = FALSE
)
```

**Arguments**

gg	The ggplot object.
dat	Either A) a string indicating a built-in dataframe with interval data from the ICS ("periods", "epochs", "stages", "eons", or "eras"), B) a string indicating a timescale from macrostrat (see list here: <a href="https://macrostrat.org/api/defs/timescales?all">https://macrostrat.org/api/defs/timescales?all</a> ), or C) a custom dataframe of time interval boundaries (see Details).
fill	The fill color of the boxes. The default is to use the colors included in dat. If a custom dataset is provided with dat without color and without fill, a greyscale will be used. Custom fill colors can be provided with this option and will be recycled if/as necessary.
color	The outline color of the interval boxes.
alpha	The transparency of the fill colors.
height	The proportional height (or width if pos is left or right) of the entire plot to use for the scale.
gap	The proportional height (or width) of the entire plot to use as a gap between the axis and the scale.
pos	Which side to add the scale to (left, right, top, or bottom). First letter may also be used.
lab	Whether to include labels.
rot	The amount of counter-clockwise rotation to add to the labels (in degrees).
abbrev	If including labels, whether to use abbreviations instead of full interval names.
skip	A vector of interval names indicating which intervals should not be labeled.
size	Label size.
neg	Set this to true if your x-axis is using negative values.

**Details**

If custom data is provided (with `dat`), it should consist of at least 3 columns of data. See `data( periods )` for an example. The `name` column lists the names of each time interval. These will be used as labels if no abbreviations are provided. The `max_age` column lists the oldest boundary of each time interval. The `min_age` column lists the youngest boundary of each time interval. The `abbr` column is optional and lists abbreviations that may be used as labels. The `color` column is also optional and lists a hex color code (which can be obtained with `rgb()`) for each time interval.

**Value**

A ggplot object.

**Examples**

```
library(ggplot2)
# bottom scale by default
p <- ggplot() +
  geom_point(aes(y = runif(1000, .5, 8), x = runif(1000, 0, 1000))) +
  scale_x_reverse() +
  coord_cartesian(xlim = c(0, 1000), ylim = c(0,8), expand = FALSE) +
  theme_classic()
gggeo_scale_old(p)

# can specify any side of the plot
p <- ggplot() +
  geom_point(aes(x = runif(1000, .5, 8), y = runif(1000, 0, 1000))) +
  scale_y_reverse() +
  coord_cartesian(xlim = c(0, 8), ylim = c(0,1000), expand = FALSE) +
  theme_classic()
gggeo_scale_old(p, pos = "left", rot = 90)

# can add multiple scales
p <- ggplot() +
  geom_point(aes(y = runif(1000, 1, 8), x = runif(1000, 0, 1000))) +
  scale_x_reverse() +
  coord_cartesian(xlim = c(0, 100), ylim = c(0,8), expand = FALSE) +
  theme_classic()
p <- gggeo_scale_old(p, height = .03, abbrv = FALSE)
p <- gggeo_scale_old(p, dat = "epochs", gap = .03, height = .1, rot = 90, size = 2.5, abbrv = FALSE)

# intervals on both sides for different timescales (ICS stages vs North American Land Mammal Ages)
p <- ggplot() +
  geom_point(aes(x = runif(1000, 1, 9), y = runif(1000, 0, 65))) +
  scale_y_reverse() +
  coord_cartesian(xlim = c(0, 10), ylim = c(0,65), expand = FALSE) +
  theme_classic()
p <- gggeo_scale_old(p, dat = "stages", pos = "left", height = .1, size = 2.5, abbrv = FALSE)
gggeo_scale_old(p, dat = "North American Land Mammal Ages", pos = "right", height = .1, size = 2.5,
  abbrv = FALSE)

#can add scales to a faceted plot
```

```

df <- data.frame(x = runif(1000,0,541), y = runif(1000,.5,8), z = sample(c(1,2,3,4), 1000, TRUE))
p <- ggplot(df) +
  geom_point(aes(x, y)) +
  scale_x_reverse() +
  coord_cartesian(xlim = c(0, 541), ylim = c(0,8), expand = FALSE) +
  theme_classic() +
  facet_wrap(~z, nrow = 2)
gggeo_scale_old(p)

#can even add a scale to a phylogeny (using ggtree)
library(phytools)
library(ggtree)
tree <- pbtree(b = .03, d = .01, n=100)
p <- ggtree(tree) +
  coord_cartesian(xlim = c(0,-500), ylim = c(-10,Ntip(tree)), expand = FALSE) +
  scale_x_continuous(breaks=seq(-500,0,100), labels=abs(seq(-500,0,100))) +
  theme_tree2()
p <- revts(p)
gggeo_scale_old(p, neg = TRUE)

```

---

gtable\_frame2

*gtable\_frame2*


---

## Description

Reformat the `gtable` associated with a `ggplot` object into a 7x7 `gtable` where the central cell corresponds to the plot panel(s), the rectangle of cells around that corresponds to the axes, and the rectangle of cells around that corresponds to the axis titles.

## Usage

```

gtable_frame2(
  g,
  width = unit(1, "null"),
  height = unit(1, "null"),
  debug = FALSE
)

```

## Arguments

<code>g</code>	<code>gtable</code>
<code>width</code>	requested width
<code>height</code>	requested height
<code>debug</code>	logical draw <code>gtable</code> cells

**Value**

7x7 gtable wrapping the plot

**Examples**

```
library(grid)
library(gridExtra)
library(ggplot2)
p1 <- ggplot(mtcars, aes(mpg, wt, colour = factor(cyl))) +
  geom_point()

p2 <- ggplot(mtcars, aes(mpg, wt, colour = factor(cyl))) +
  geom_point() + facet_wrap(~ cyl, ncol=2, scales = 'free') +
  guides(colour='none') +
  theme()

p3 <- ggplot(mtcars, aes(mpg, wt, colour = factor(cyl))) +
  geom_point() + facet_grid(. ~ cyl, scales = 'free')

g1 <- ggplotGrob(p1);
g2 <- ggplotGrob(p2);
g3 <- ggplotGrob(p3);
fg1 <- gtable_frame2(g1)
fg2 <- gtable_frame2(g2)
fg12 <- gtable_frame2(gtable_rbind(fg1,fg2), width=unit(2,'null'), height=unit(1,'null'))
fg3 <- gtable_frame2(g3, width=unit(1,'null'), height=unit(1,'null'))
grid.newpage()
combined <- gtable_cbind(fg12, fg3)
grid.draw(combined)
```

---

panel.disparity

*Combined wireframe and cloud panel*

---

**Description**

Plots the provided data on 2-D surfaces within a 3-D framework. See [disparity\\_through\\_time](#).

**Usage**

```
panel.disparity(x, y, z, groups, subscripts, ...)
```

**Arguments**

x, y, z, groups, subscripts, ...  
 Same as for [panel.cloud](#)

**Value**

No return value, plots the results of both [panel.cloud](#) and [panel.wireframe](#).

---

periods	<i>Period data from the International Commission on Stratigraphy (v2022/02)</i>
---------	---

---

### Description

A dataset containing the boundary ages, abbreviations, and colors for the periods of the Geologic Time Scale. Based on The ICS International Chronostratigraphic Chart (v2022/02), by Cohen, Finney, Gibbard, and Fan.

### Usage

periods

### Format

A data frame with 22 rows and 5 variables:

**name** period name

**max\_age** maximum age, in millions of years

**min\_age** minimum age, in millions of years

**abbr** period name abbreviations

**color** the colors for each period, according to the Commission for the Geological Map of the World

### Source

<https://stratigraphy.org> via <https://macrostrat.org/api/v2/defs/intervals?timescale=international%20periods>

### See Also

Other timescales: [eons](#), [epochs](#), [eras](#), [stages](#)

---

stages	<i>Stage data from the International Commission on Stratigraphy (v2022/02)</i>
--------	--

---

### Description

A dataset containing the boundary ages, abbreviations, and colors for the stages of the Geologic Time Scale. Based on The ICS International Chronostratigraphic Chart (v2022/02), by Cohen, Finney, Gibbard, and Fan.

### Usage

stages

### Format

A data frame with 102 rows and 5 variables:

**name** stage name

**max\_age** maximum age, in millions of years

**min\_age** minimum age, in millions of years

**abbr** stage name abbreviations

**color** the colors for each stage, according to the Commission for the Geological Map of the World

### Source

<https://stratigraphy.org> via <https://macrostrat.org/api/v2/defs/interval%20ages>

### See Also

Other timescales: [eons](#), [epochs](#), [eras](#), [periods](#)

# Index

## \* datasets

- coord\_geo, 2
- coord\_trans\_flip, 5
- coord\_trans\_xy, 6
- eons, 9
- epochs, 10
- eras, 10
- periods, 22
- stages, 22

## \* timescales

- eons, 9
- epochs, 10
- eras, 10
- periods, 22
- stages, 22

color, 4

coord\_flip, 5

coord\_geo, 2

coord\_trans, 2, 3, 5, 6

coord\_trans\_flip, 5

coord\_trans\_xy, 6

CoordGeo (coord\_geo), 2

CoordTransFlip (coord\_trans\_flip), 5

CoordTransXY (coord\_trans\_xy), 6

disparity\_through\_time, 7, 21

eons, 9, 10, 11, 22, 23

epochs, 9, 10, 11, 22, 23

eras, 9, 10, 10, 11, 22, 23

geom\_fit\_text, 3, 4

geom\_point, 6

geom\_polygon, 6

geom\_rect, 6

geom\_text, 3

getScaleData, 11

ggarrange2, 12

gggeo\_scale, 13

gggeo\_scale\_old, 17

grid.draw, 15

gtable\_add\_padding, 13

gtable\_frame2, 20

lattice.options, 8

layout, 12

levelplot, 8

linear\_trans, 6

panel.cloud, 7, 21

panel.disparity, 21

panel.wireframe, 21

periods, 9–11, 22, 23

print.geo\_scale (gggeo\_scale), 13

stages, 9–11, 22, 22

trans\_new, 6

trellis.par.set, 8

wireframe, 7, 8

xyplot, 8