

INTRODUCTION

gganimate extends the grammar of graphics as implemented by ggplot2 to include the description of animation. It does this by providing a range of new grammar classes that can be added to the plot object in order to customise how it should change with time.

BASIC FUNCTION

- ```
p <- ggplot()+
 geom_point()+ // or other kinds of graph
 transition_states(states, transition_length,
 state_length)+
 view_follow(fixed_x,fixed_y)+
 shadow_wake(wake_length,size,alpha)
 enter_fade()+
 exit_shrink()+
 ease_aes(default='linear')
```
- animate(p)  
anim\_save(filename,path)
- *transition\_\**():  
defines how the data should be spread out and how it relates to itself across time.
  - *view\_\**():  
defines how the positional scales should change along the animation.
  - *shadow\_\**():  
defines how data from other points in time should be presented in given point in time.
  - *enter\_\**()/ *exit\_\**():  
defines how new data should appear and how old data should disappear during the course of the animation.
  - *ease\_aes*():  
defines how different aesthetics should be eased during transitions.
  - *animate*():  
render a gganimate object.
  - *anim\_save*():  
save an animation to a file.

## INSTALLATION

```
install.packages('devtools')
devtools::install_github('thomasp85/gganimate')
devtools::install_github('thomasp85/transformr')
library(ggplot2), library(gganimate)
NOTE: May also need install 'gifski' and 'av' package.
```

## TRANSITION

**transition\_states:** Between several distinct stages of the data  

```
anim<-ggplot(iris,aes(Sepal.Width,Petal.Width))+
 geom_point()+
 transition_states(Species,transition_length=3,state_length=1)
```

**transition\_filter:** Between different filters  

```
anim<-ggplot(iris,aes(Petal.Width,Petal.Length,
 colour=Species))+
 geom_point()+
 transition_filter(
 transition_length=2,filter_length = 1,
 Setosa=Species=='setosa',
 Long = Petal.Length>4,
 Wide = Petal.Width>2)
```

**transition\_layers:** Build up plot layer by layer  

```
transition_layers(layer_length=1,transition_length=1,keep_layers=TRUE,from_blank=TRUE,
 layer_order=NULL,layer_names=NULL)
```

**transition\_reveal:** Time series  

```
anim<-ggplot(airquality,aes(Day,Temp,group=Month))+geom_line()+
 geom_point(aes(group=seq_along(Day))),size=3,color='red')+
 transition_reveal(Day)
```

**transition\_time:** Time series  

```
anim<-ggplot(airquality,aes(Day,Temp))+
 geom_point(aes(colour=factor(Month)))+
 transition_time(Day)
```

## SHADOW

**shadow\_mark:** Show original data as background  

```
anim<-ggplot(airquality,aes(Day,Temp,colour=factor(Month)))+
 geom_point()+
 transition_time(Day)
anim1<-anim+shadow_mark(colour='black',
 size=0.75,past= TRUE, future = FALSE)
```

**shadow\_trail:** A trail of evenly spaced old frames  

```
anim2<-anim+shadow_trail(distance=0.4,alpha=0.3,shape = 2)
```

**shadow\_weak:** Show preceding frames with gradual falloff  

```
anim3<-anim+shadow_wake(wake_length=0.1,
 size=2,alpha=FALSE,colour='grey92')
```

## VIEW

**view\_follow:** Let the view follow the data  

```
anim<-ggplot(iris,aes(Sepal.Length, Sepal.Width))+
 geom_point()+labs(title = "closest_state")+
 transition_states(Species,transition_length=4,state_length=1)
anim1<-anim+view_follow(fixed_x=TRUE,
 fixed_y=FALSE)
anim2<-anim+view_follow(fixed_x=c(4,NA),
 fixed_y=c(2,NA))
```

**view\_step:** Follow the data in steps  

```
anim<-ggplot(iris,aes(Petal.Length,Petal.Width))+
 geom_point()+
 transition_states(Species,transition_length=1)+
 view_step(pause_length=2,step_length=1,
 nsteps =3,pause_first=TRUE)
```

NOTE: The use of view\_step is relative to transition\_states. If the transition doesn't wrap, then the view shouldn't either

## ANIMATION

```
animate(plot, nframes, fps, height, width, duration, detail, renderer, device, ref_frame, start_pause,
 end_pause, rewind,...)
anim_save(filename, animation=last_animation(), path=NULL, ...)
```

## EXAMPLE

```
ggplot(mtcars, aes(factor(cyl), mpg)) +
 geom_boxplot() +
 transition_states(
 gear,
 transition_length = 2,
 state_length = 1
) +
 enter_fade() +
 exit_shrink() +
 ease_aes('sine-in-out')
```

```
library(gapminder)
ggplot(gapminder, aes(gdpPercap, lifeExp, size = pop, colour = country)) +
 geom_point(alpha=0.7,show.legend=FALSE)+
 scale_colour_manual(values=country_colors)+
 scale_size(range = c(2, 12))+scale_x_log10()+
 facet_wrap(~continent) +
 labs(title='Year:frame_time', x='GDP per capita', y='life expectancy') +
 transition_time(year) +
 ease_aes('linear')
```

## REFERENCES

<https://gganimate.com/>  
<https://cran.r-project.org/web/packages/gganimate/gganimate.pdf>

## INFORMATION

See html version including gif from github: [https://github.com/lisiyu98/gganimate\\_cheatsheet](https://github.com/lisiyu98/gganimate_cheatsheet) Email : [sl4826@columbia.edu](mailto:sl4826@columbia.edu)