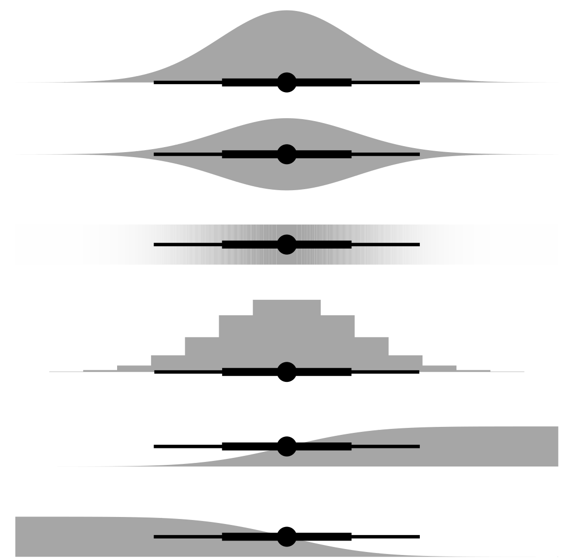
Shortcut slab+interval geometries

The stat_sample_slabinterval() and stat_dist_slabinterval() stats are flexible meta-geometries for visualizing sample data or analytical distributions.
With that flexibility comes a cost in remembering particular combinations of parameters that yield specific visualization types.
Thus, ggdist also provides several **shortcut stats** with sensible default parameters:

This geometry ...	... uses these defaults: mapping =	slab_type =	side =	justification =	normalize =
	aesthetic mapping	*function assigned to the computed aesthetic* **f**	*side to draw the slab on*	*position of interval relative to slab*	*What groups to normalize max height of slab thickness within*
stat_sample_slabinterval() stat_dist_slabinterval()	aes(thickness = f)	"pdf"	"topright"	0	"all"
stat_halfeye() stat_dist_halfeye()	aes(thickness = f)	"pdf"	"topright"	0	"all"
stat_eye() stat_dist_eye()	aes(thickness = f)	"pdf"	"both"	0	"all"
stat_gradientinterval() stat_dist_gradientinterval()	aes(slab_alpha = f)	"pdf"	"topright"	0.5	"all"
stat_histinterval()	aes(thickness = f)	"histogram"	"topright"	0	"all"
stat_cdfinterval() stat_dist_cdfinterval()	aes(thickness = f)	"cdf"	"topleft"	0.5	"none"
stat_ccdfinterval() stat_dist_ccdfinterval()	aes(thickness = f)	"ccdf"	"topleft"	0.5	"none"

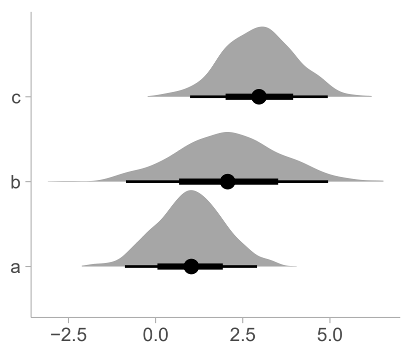


Example from stat_sample_slabinterval() sub-family

Using sample data

```
df = data.frame(
  group = c("a", "b", "c"),
  value = rnorm(
    3000,
    mean = c(1, 2, 3),
    sd = c(1, 1.5, 1)
  )
)

ggplot(df) +
  aes(y = group, x = value) +
  stat_halfeye()
```

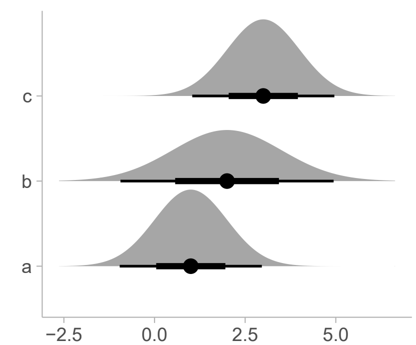


Example from stat_dist_slabinterval() sub-family

Using analytical distributions

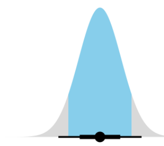
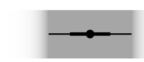
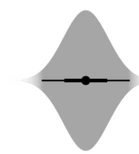
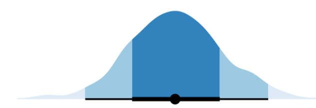
```
df = data.frame(
  group = c("a", "b", "c"),
  mean = c(1, 2, 3),
  sd = c(1, 1.5, 1)
)

ggplot(df) +
  aes(
    y = group,
    dist = dist_normal(mean, sd)
  ) +
  stat_dist_halfeye()
```



Custom slab+interval geometries using computed variables: x/y, CDF, and PDF mappings

The stat_sample_slabinterval() and stat_dist_slabinterval() stats compute cdf and pdf variables representing the cumulative distribution function and the probability density function of the underlying data. Along with x/y position, after the stats are computed these can be mapped onto aesthetics like fill, alpha, or color, or combined with functions like cut_cdf_qi() to create more esoteric visualization types.

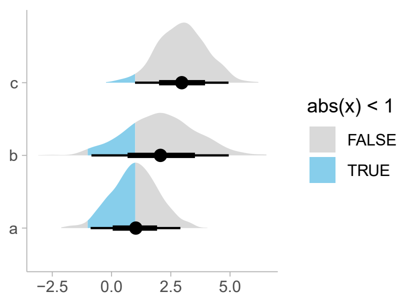
This geometry ...	... combined with this mapping ...	... does this:	
stat_gradientinterval() stat_dist_gradientinterval()	aes(slab_alpha = stat(cdf))	Encodes the **CDF** using opacity. Can be thought of as many transparent bars overlapping each other to build up a "fuzzy" bar chart. Similar to a *fuzzygram*, per Wilkinson (Graphical displays, *Stat Meth in Med Res*, 1992)	
stat_halfeye() stat_dist_halfeye()	aes(fill = stat(abs(x) < 1.5))	Uses a **logical condition** to select a fill region of the slab to color differently. Useful for highlighting a *region of practical equivalence*, or ROPE, per Kruschke (Bayesian estimation supersedes the t test, *JEP*, 2013)	
stat_gradientinterval() stat_dist_gradientinterval()	aes(slab_alpha = stat(−pmax(abs(1 − 2*cdf), .95)))	Fades the tails of the slab outside a desired **interval (here 95%)** in proportion to $\|1-2F(x)\|$ where $F(x)$ is the CDF. Correll & Gleicher (Error bars considered harmful, *TVCG*, 2014) argue that this might reduce dichotomous interpretations, though evidence is unclear.	
stat_eye() stat_dist_eye()	aes(slab_alpha = stat(−pmax(abs(1 − 2*cdf), .95)))	Fades the tails of the slab as in the previous example, but combines this with an eye plot, per Helske *et al.* (Are You Sure You're Sure?, *arXiv*:2002.07671)	
stat_halfeye() stat_dist_halfeye()	aes(fill = stat(cut_cdf_qi(cdf, .width = c(.66, .95, 1))))	Bins the **CDF** into an arbitrary number of **intervals (here 66% and 95%)** and highlights the intervals using the fill color of the slab. Similar in spirit to bayesplot::mcmc_areas().	

Example from stat_sample_slabinterval() sub-family

Using sample data

```r
df = data.frame(
  group = c("a", "b", "c"),
  value = rnorm(
    3000,
    mean = c(1, 2, 3),
    sd = c(1, 1.5, 1)
  )
)

ggplot(df) +
  aes(
    y = group,
    x = value,
    fill = stat(abs(x) < 1)
  ) +
  stat_halfeye() +
  scale_fill_manual(values = c("gray85", "skyblue"))
```



Example from stat_dist_slabinterval() sub-family

Using analytical distributions

```r
df = data.frame(
  group = c("a", "b", "c"),
  mean = c(1, 2, 3),
  sd = c(1, 1.5, 1)
)

ggplot(df) +
  aes(
    y = group,
    dist = dist_normal(mean, sd)
    fill = stat(abs(x) < 1)
  ) +
  stat_dist_halfeye() +
  scale_fill_manual(values = c("gray85", "skyblue"))
```