Bayesplot :: CHEAT SHEET

```r
library("bayesplot")
library("rstanarm")
options(mc.cores = parallel::detectCores())
library("ggplot2")
library("dplyr")
```
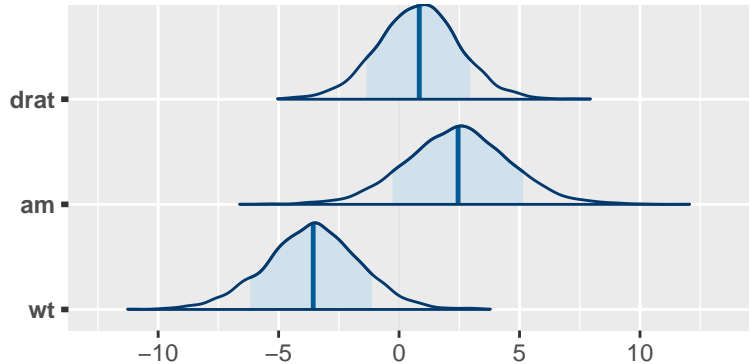
Model Parameters

To showcase `bayesplot`, we'll fit linear regression using `rstanarm::stan_glm` and use this model throughout.

```r
model <- stan_glm(mpg ~ ., data=mtcars, chains=4)
posterior <- as.matrix(model)
```

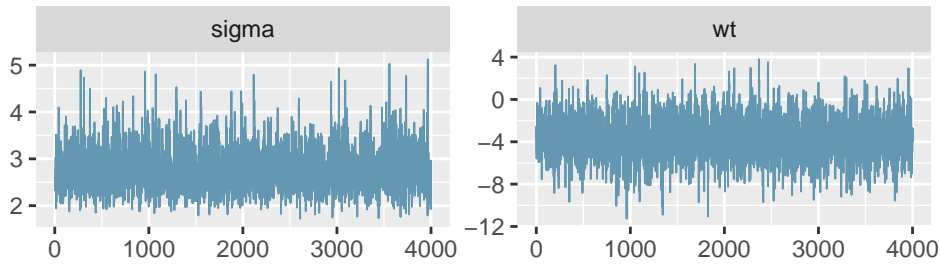Chances are good you're most interested in the posterior distributions for select parameters.

```r
plot_title <- ggtitle("Posterior distributions",
                      "medians and 80% intervals")
mcmc_areas(posterior,
          pars = c("drat", "am", "wt"),
          prob = 0.8) + plot_title
```


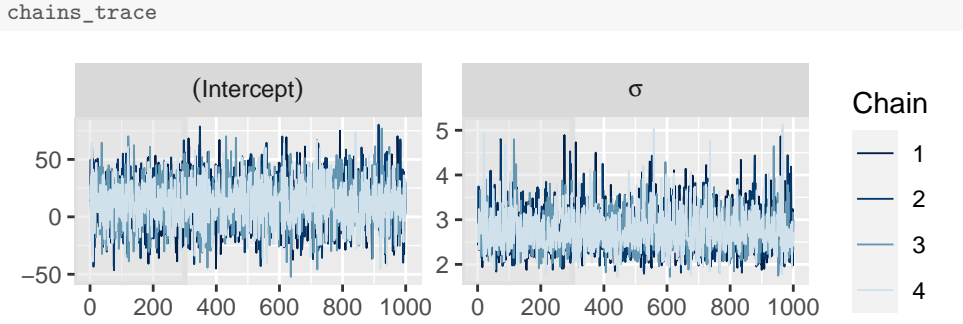Posterior distributions
medians and 80% intervals

Diagnosing convergence with traceplots is simple.

```r
mcmc_trace(posterior, pars=c("sigma", "wt"))
```
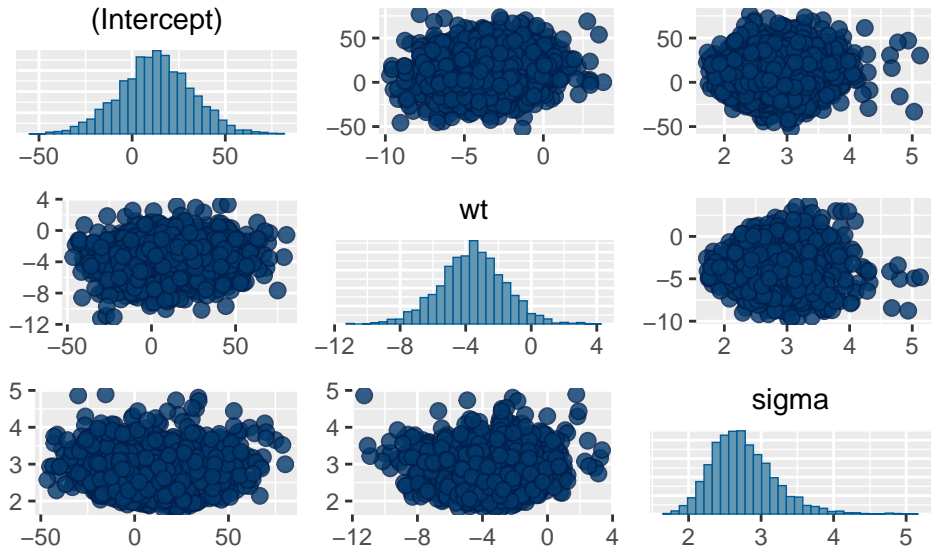

Using `as.array`, you can extract each of the four chain's posterior draws, different from above. This allows you to see each chain's traceplot for selected parameters.

```r
posterior_chains <- as.array(model)
fargs <- list(ncol = 2, labeller = label_parsed)
pars <- c("(Intercept)", "sigma")
chains_trace <- mcmc_trace(posterior_chains, pars = pars,
                           n_warmup = 300, facet_args = fargs)
```

```r
chains_trace
```


The pairs plot is helpful in determining if you have any highly correlated parameters.

```r
posterior_chains %>%
  mcmc_pairs(pars = c("(Intercept)", "wt", "sigma"))
```
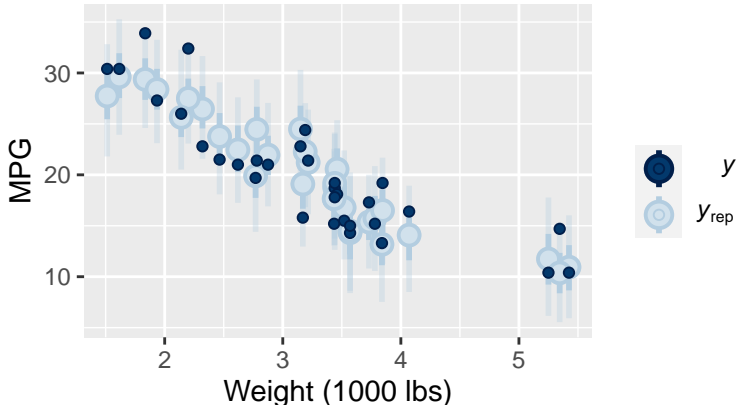

Posterior Predictive Checks
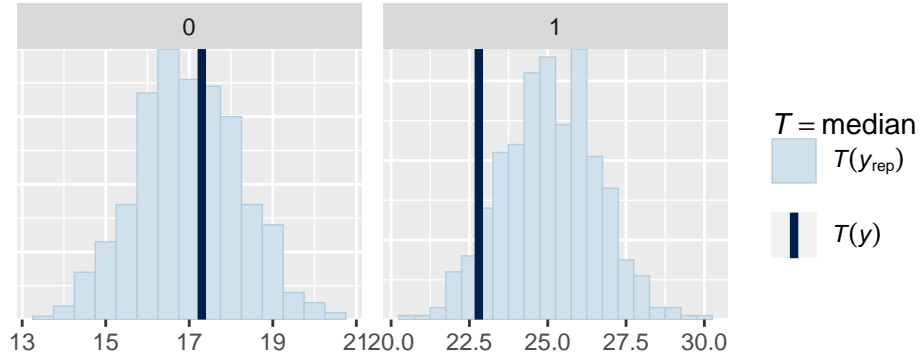
Check how well the model covers your data with draws from the posterior predictive density.

```r
ppd <- posterior_predict(model, draws=500)
ppd %>%
  ppc_intervals(y = mtcars$mpg, yrep = ., x = mtcars$wt, prob = 0.5) +
  labs(x = "Weight (1000 lbs)", y = "MPG",
     title = "50% posterior predictive intervals of MPG by weight")
```
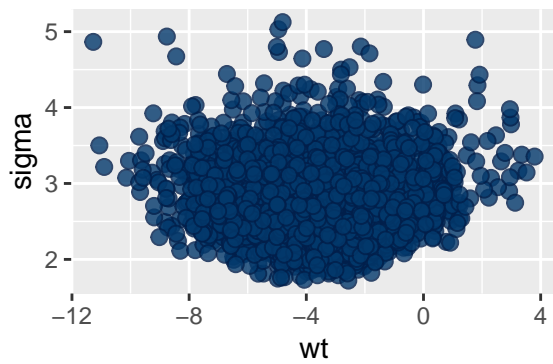

50% posterior predictive intervals of MPG by

```r
ppd %>%
  ppc_stat_grouped(y = mtcars$mpg, group = mtcars$am,
                   stat = "median", binwidth=0.5)
```


T = median

$T(y_{rep})$

$T(y)$

Diagnostics

Bayesplot makes it easy to check diagnostics specific to the NUTS sampling method that `rstanarm` uses by default.

```r
mcmc_scatter(posterior, pars = c("wt", "sigma"),
   np = nuts_params(model$stanfit))
```



```r
np <- nuts_params(model$stanfit)
mcmc_nuts_energy(np, binwidth=1) +
     ggtitle("NUTS Energy Diagnostic")
```

NUTS Energy Diagnostic

π_E

$\pi_{\Delta E}$