Graphics with ggplot2

```
library(ggplot2)
```

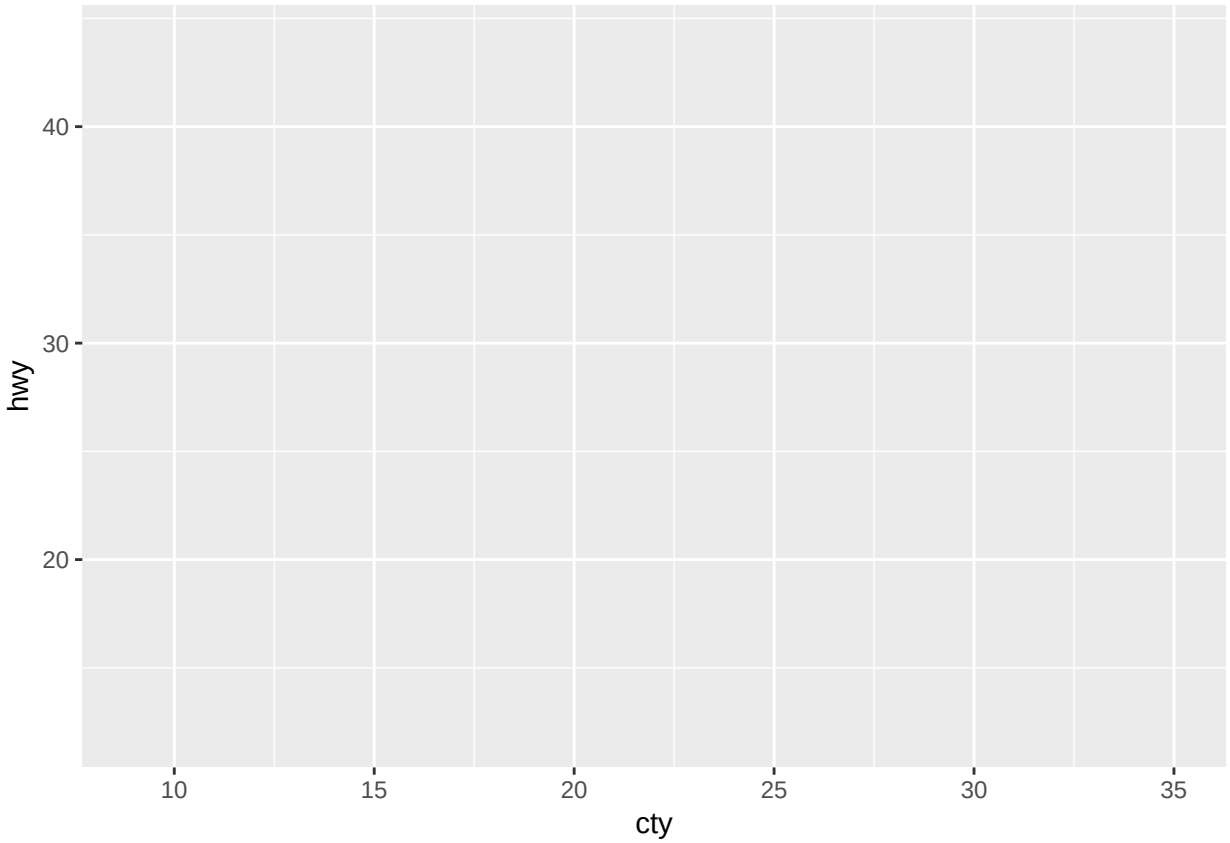
Consider the `mpg` dataset.

```
tail(mpg)
```

```
## # A tibble: 6 x 11
##   manufacturer model displ  year   cyl trans drv     cty   hwy fl    class
##   <chr>        <chr> <dbl> <int> <int> <chr> <chr> <int> <int> <chr> <chr>
## 1 volkswagen   pass~   1.8  1999     4 auto~ f        18    29 p     mids~
## 2 volkswagen   pass~   2    2008     4 auto~ f        19    28 p     mids~
## 3 volkswagen   pass~   2    2008     4 manu~ f        21    29 p     mids~
## 4 volkswagen   pass~   2.8  1999     6 auto~ f        16    26 p     mids~
## 5 volkswagen   pass~   2.8  1999     6 manu~ f        18    26 p     mids~
## 6 volkswagen   pass~   3.6  2008     6 auto~ f        17    26 p     mids~
```

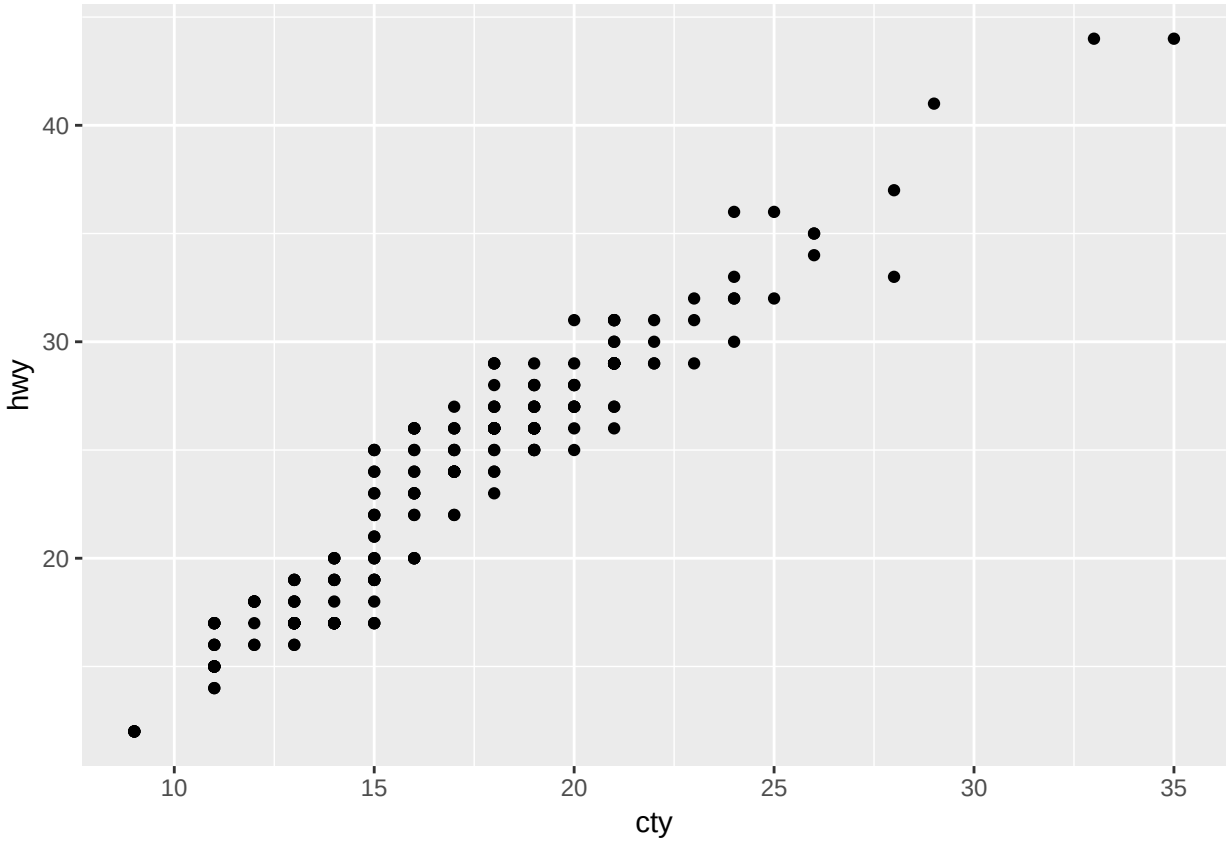
Make an empty plot with the data prepared.

```
g <- ggplot(mpg, aes(cty, hwy))
print(g)
```



Actually plot some points.

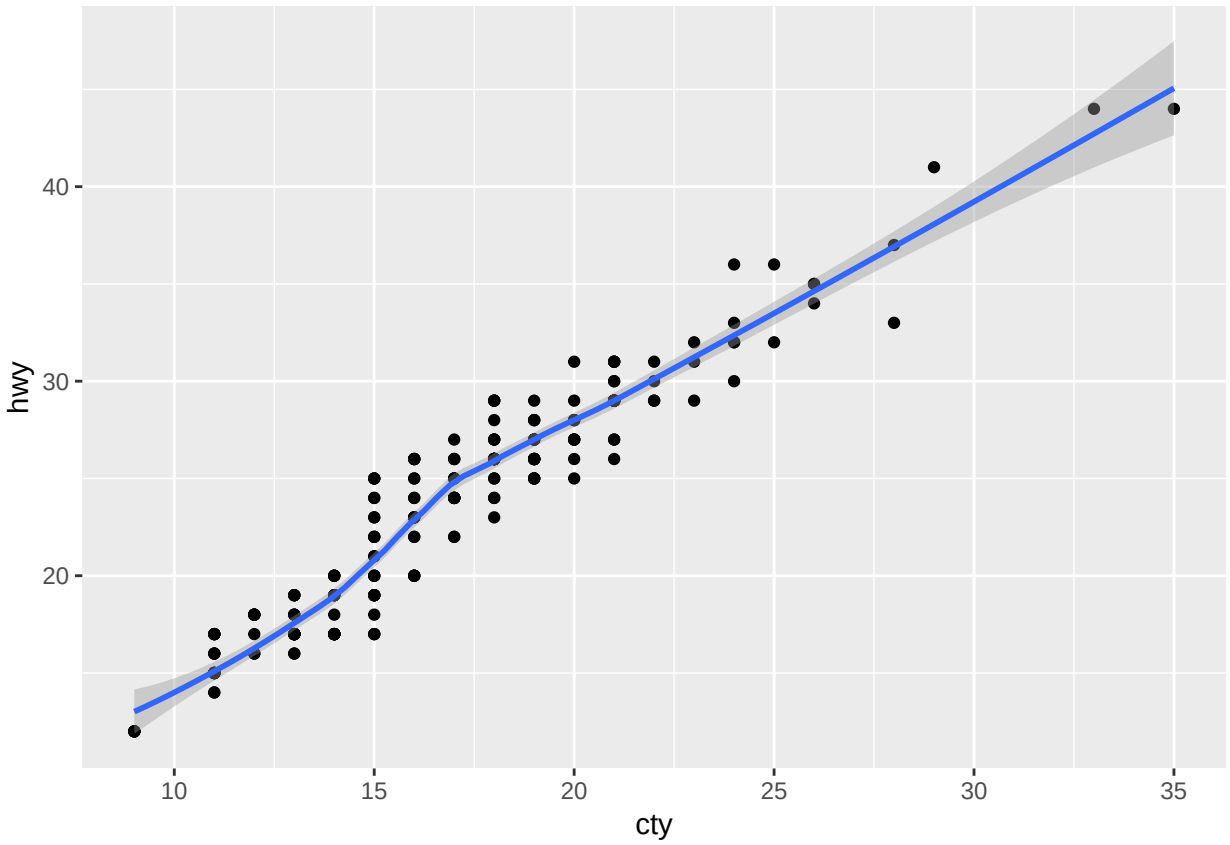
```
g <- ggplot(mpg, aes(cty, hwy)) +
    geom_point()
print(g)
```



Add a loess smoother.

```
g <- ggplot(mpg, aes(cty, hwy)) +
    geom_point() +
    geom_smooth()
print(g)
```

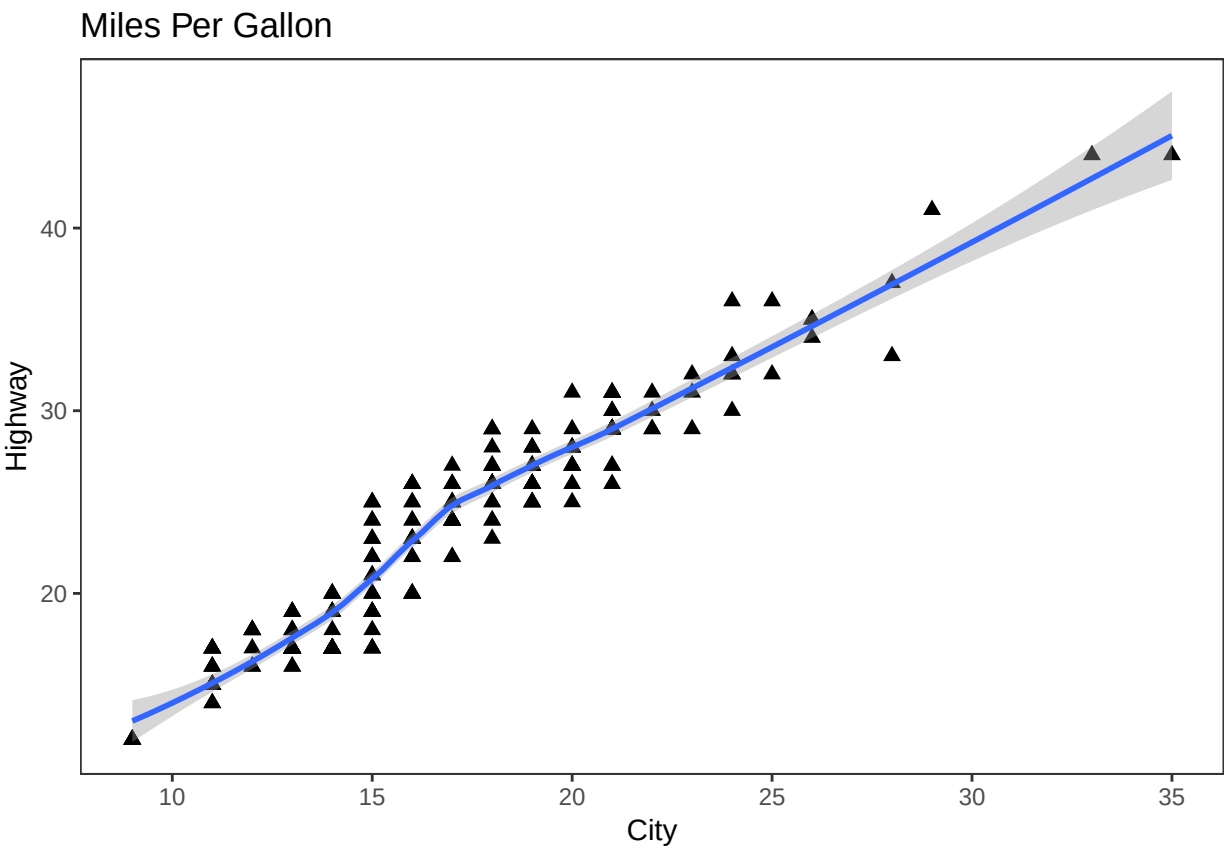
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Customize axes, theme, etc.

```
g <- ggplot(mpg, aes(cty, hwy)) +
    geom_point(shape = 17, size = 2) +
    geom_smooth() +
    labs(x = "City", y = "Highway", title = "Miles Per Gallon") +
    theme_bw() +
    theme(panel.grid.major = element_blank(),
          panel.grid.minor = element_blank())
print(g)
```

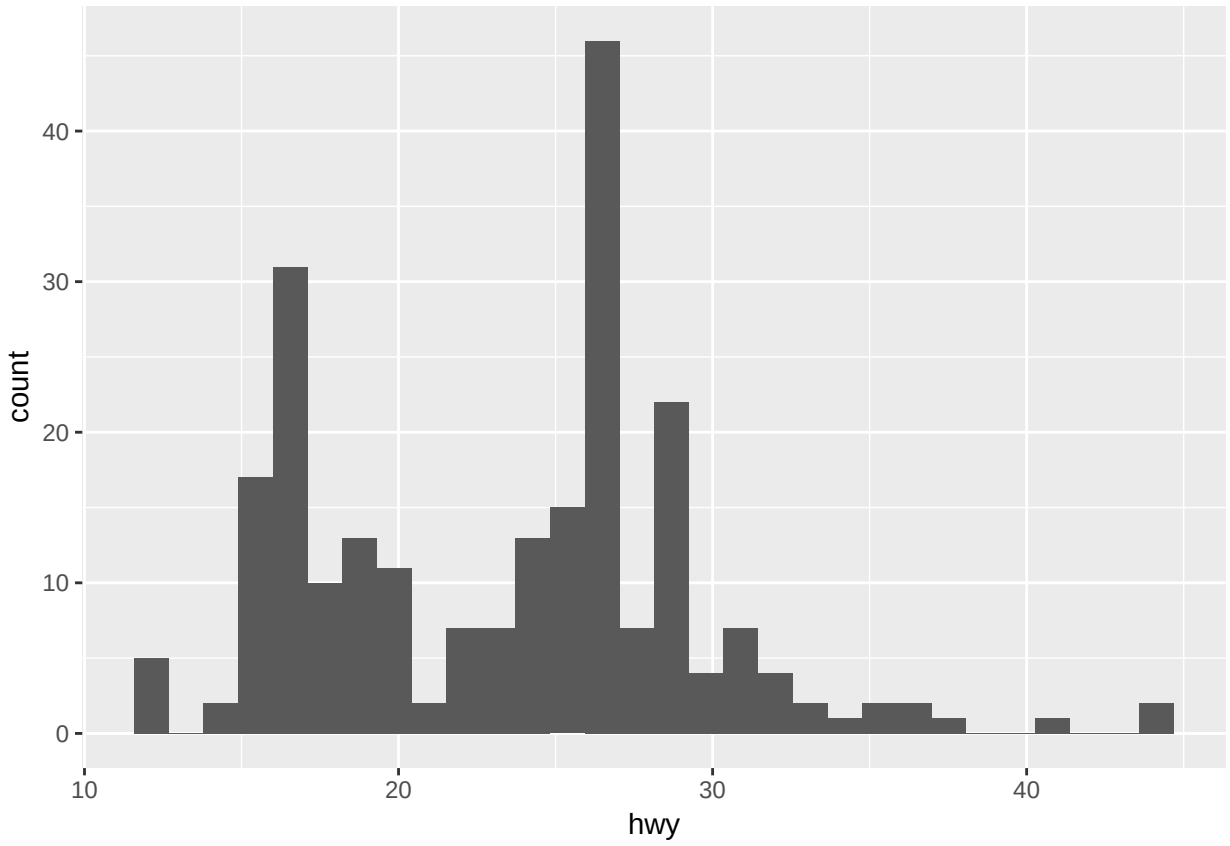
```
## `geom_smooth()` using method = 'loess' and formula 'y ~ x'
```



Now plot a histogram.

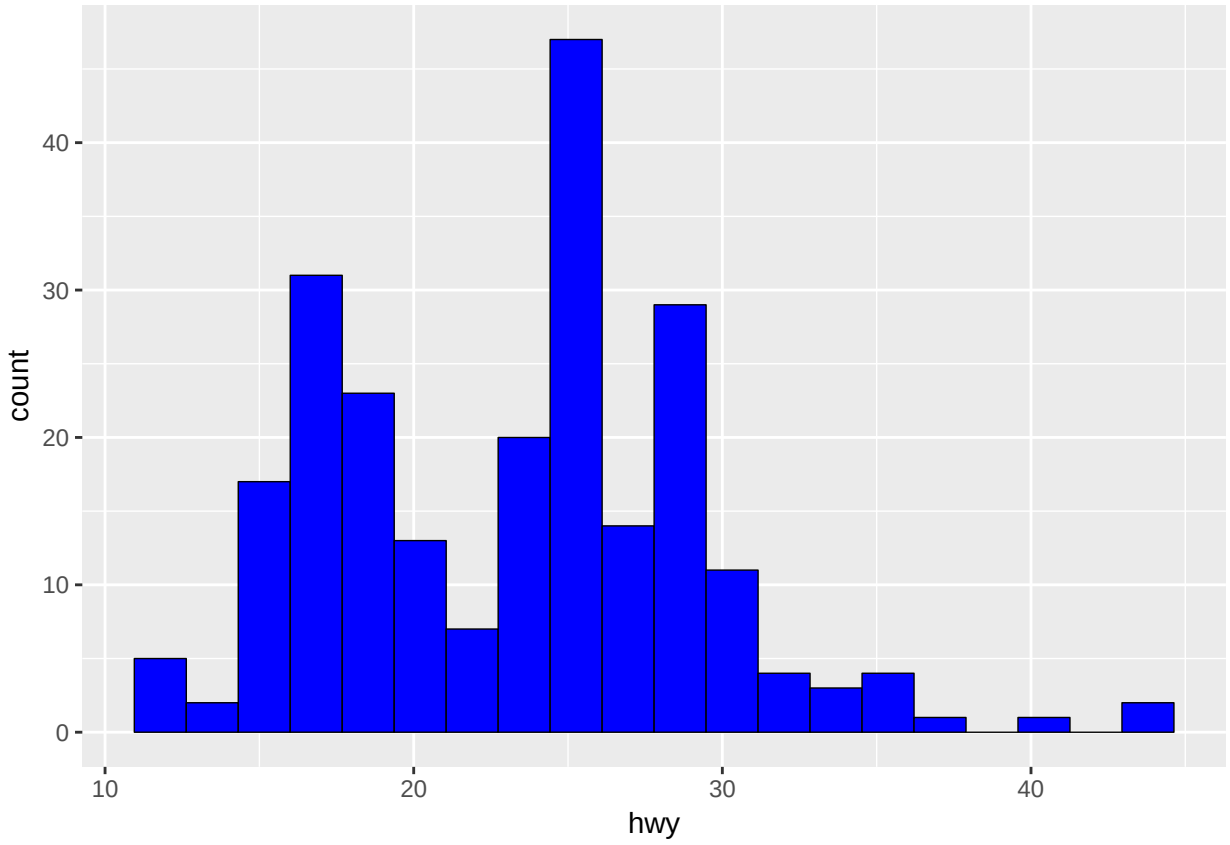
```
g <- ggplot(mpg, aes(hwy)) +
    geom_histogram()
print(g)
```

```
## `stat_bin()` using `bins = 30`. Pick better value with `binwidth`.
```



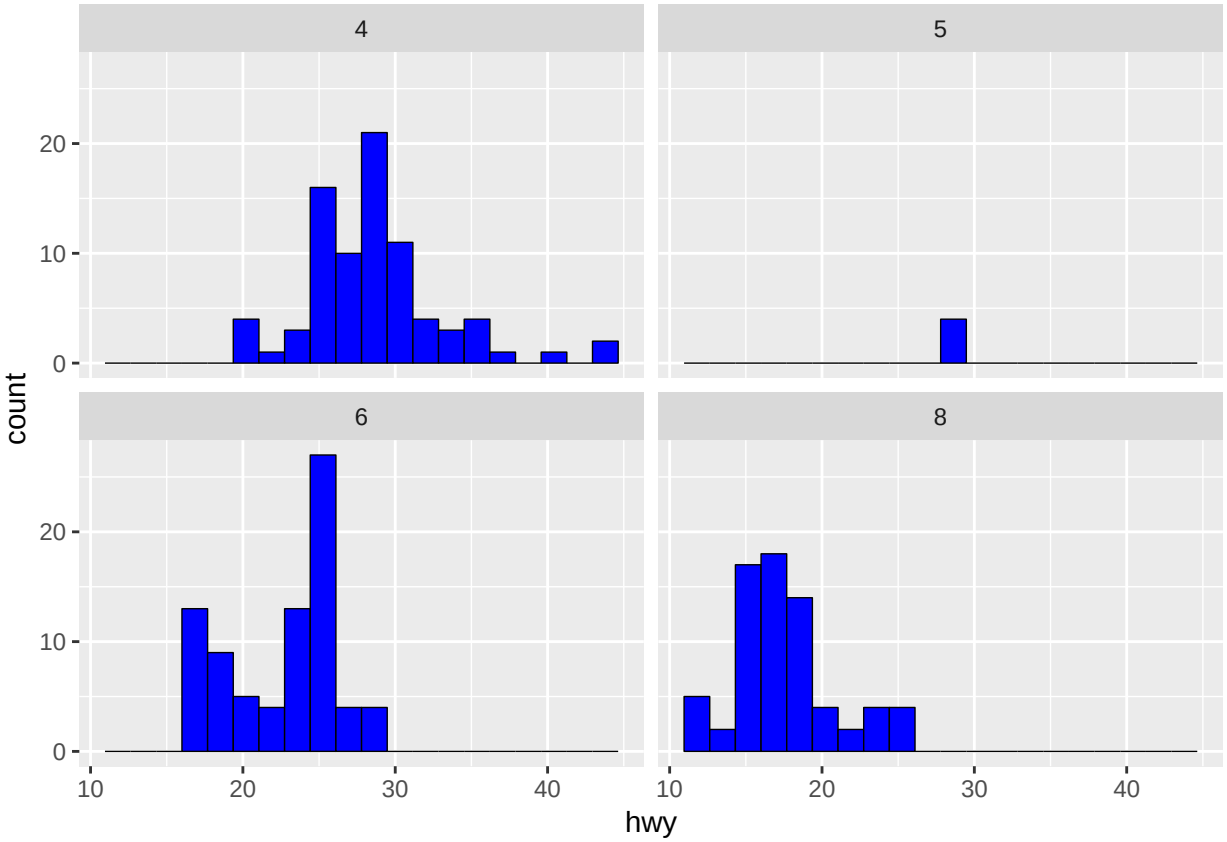
Customize the histogram.

```
g <- ggplot(mpg, aes(hwy)) +
    geom_histogram(bins=20, fill="blue", color="black", size=0.25)
print(g)
```



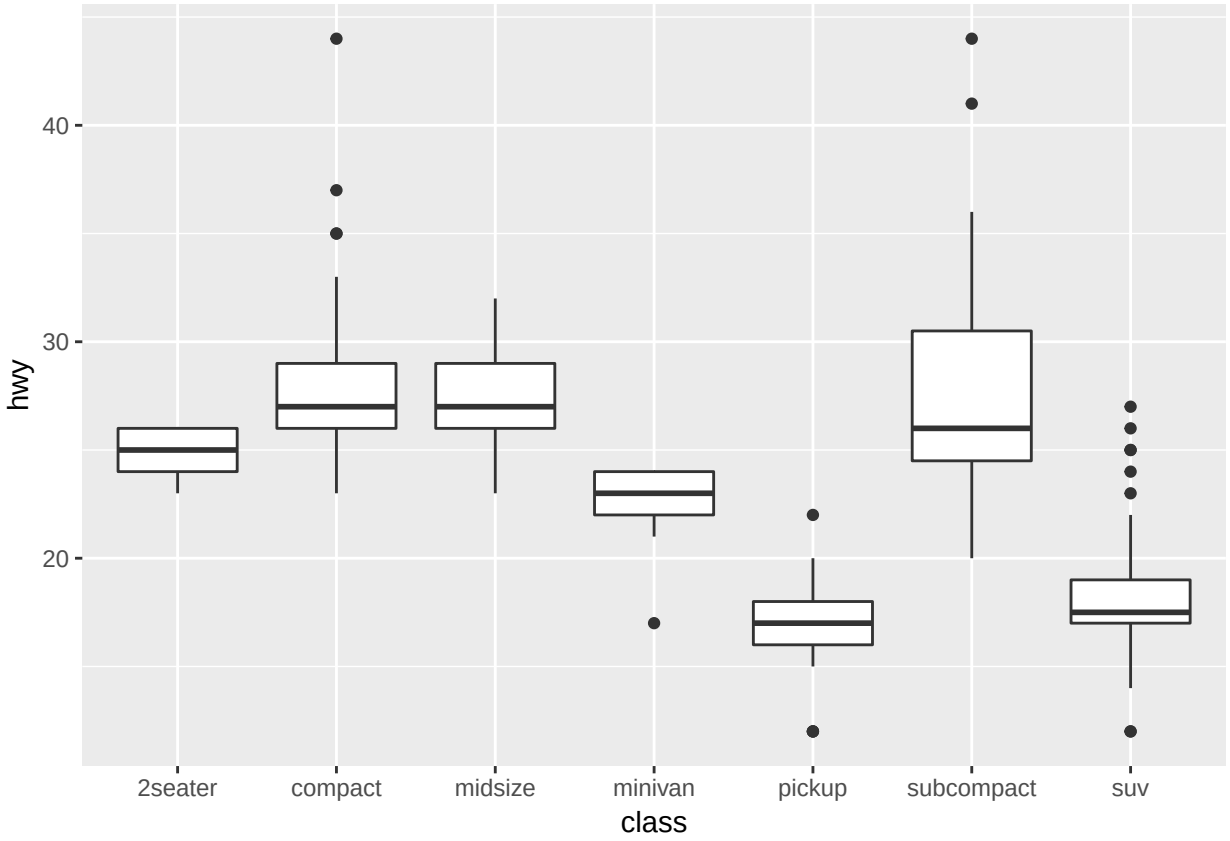
Plot a histogram for each value of `cyl`.

```
g <- ggplot(mpg, aes(hwy)) +
    geom_histogram(bins=20, fill="blue", color="black", size=0.25) +
    facet_wrap(~ cyl)
print(g)
```



Plot boxplots for each value of `class`.

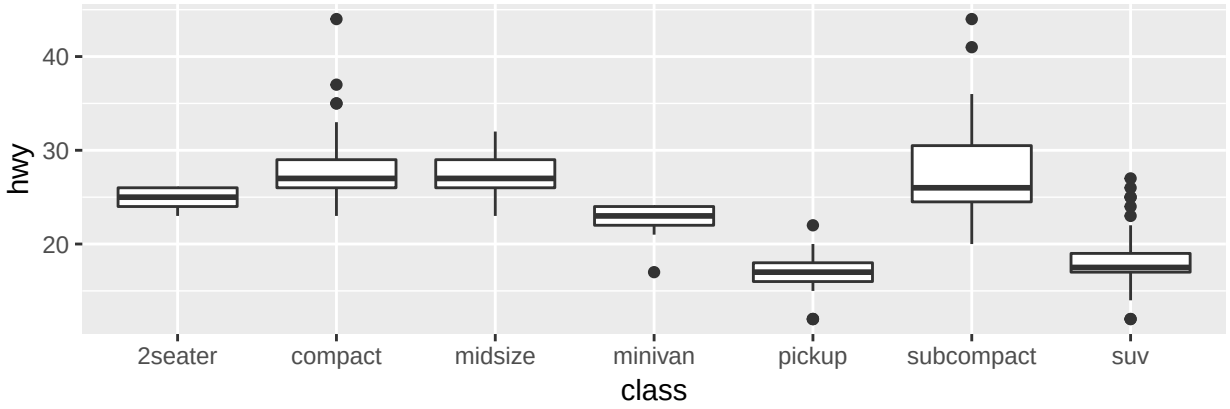
```
g <- ggplot(mpg, aes(class, hwy)) +
    geom_boxplot()
print(g)
```

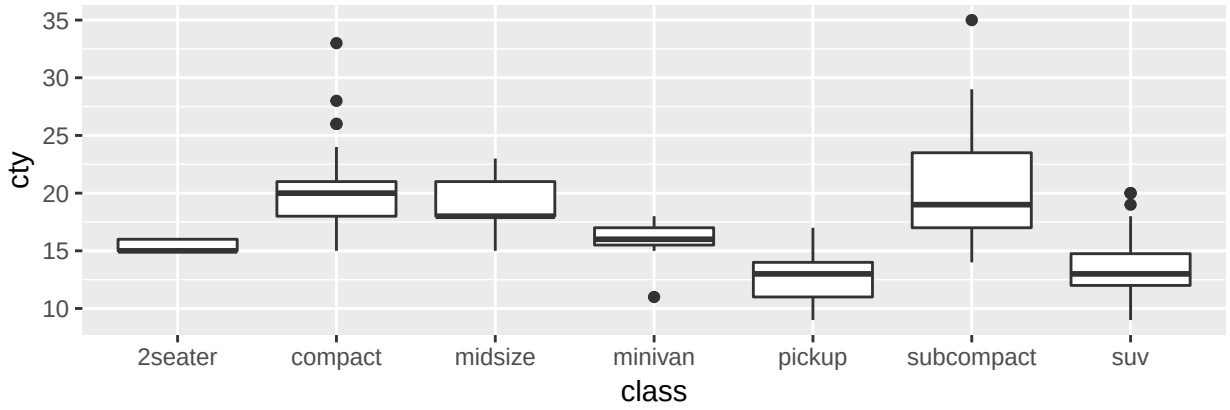


Plot two boxplots together horizontally.

```
library(gridExtra)

g <- ggplot(mpg, aes(class, hwy)) +
    geom_boxplot()
h <- ggplot(mpg, aes(class, cty)) +
    geom_boxplot()
grid.arrange(g, h)
```



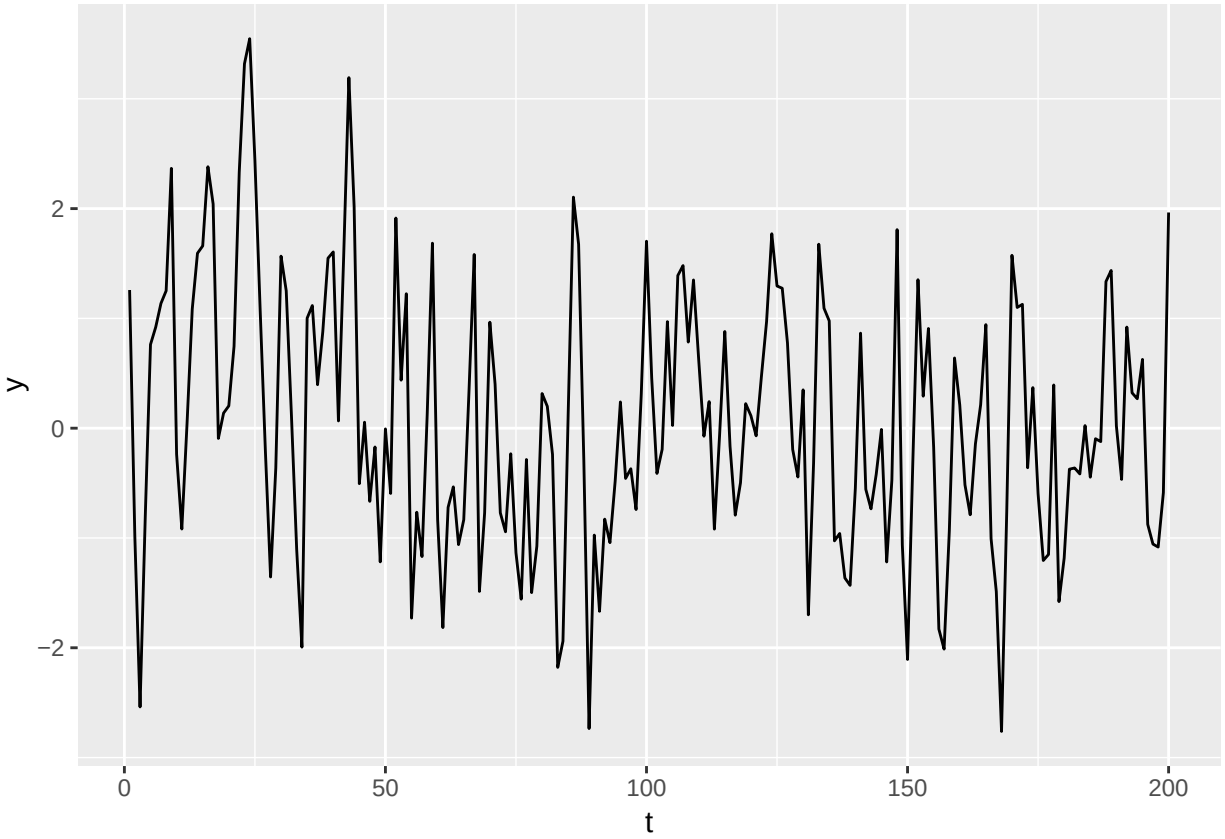


Recall the AR(2) time series model

$$y_t = \phi_1 y_{t-1} + \phi_2 y_{t-2} + \epsilon_t, \quad \epsilon_t \overset{\text{iid}}{\sim} \mathrm{N}(0, \sigma^2), \quad t = 1, \ldots, n.$$

Generate an AR(2) series and plot it.

```
y <- arima.sim(n = 200, list(ar = c(0.5, -0.2), sd = sqrt(0.25)))
dat <- data.frame(t = 1:200, y = as.numeric(y))
g <- ggplot(dat, aes(t, y)) + geom_line()
print(g)
```



What about plotting multiple series on one plot? First draw the series.

```
n <- 200
y1 <- 0 + arima.sim(n = n, list(ar = c(0.5, -0.2), sd = sqrt(0.25)))
y2 <- 3 + arima.sim(n = n, list(ar = c(0.1, -0.2), sd = sqrt(0.25)))
y3 <- -3 + arima.sim(n = n, list(ar = c(0.7, -0.2), sd = sqrt(0.5)))
```

Make the series columns of a `data.frame`.

```
dat <- data.frame(t = 1:n, y1 = as.numeric(y1), y2 = as.numeric(y2), y3 = as.numeric(y3))
head(dat, 3)
```

```
##   t        y1       y2        y3
## 1 1 0.1980709 2.308180 -3.227193
## 2 2 0.6781526 2.886922 -3.645960
## 3 3 0.9911207 2.943533 -2.800056
```

Reshape the `data.frame` by stacking the series vertically.

```
library(reshape2)
newdat <- melt(dat, 't')
head(newdat)
```

```
##   t variable     value
## 1 1       y1 0.1980709
## 2 2       y1 0.6781526
## 3 3       y1 0.9911207
## 4 4       y1 0.6671004
## 5 5       y1 0.7243362
## 6 6       y1 1.1166981
```

```
tail(newdat)
```

```
##       t variable       value
## 595 195       y3 -2.78112396
## 596 196       y3 -2.57448082
## 597 197       y3 -1.41161612
## 598 198       y3 -0.92854100
```

```
## 599 199       y3 -0.05117302
## 600 200       y3 -1.71619736
```

Here is one way to plot the series together.

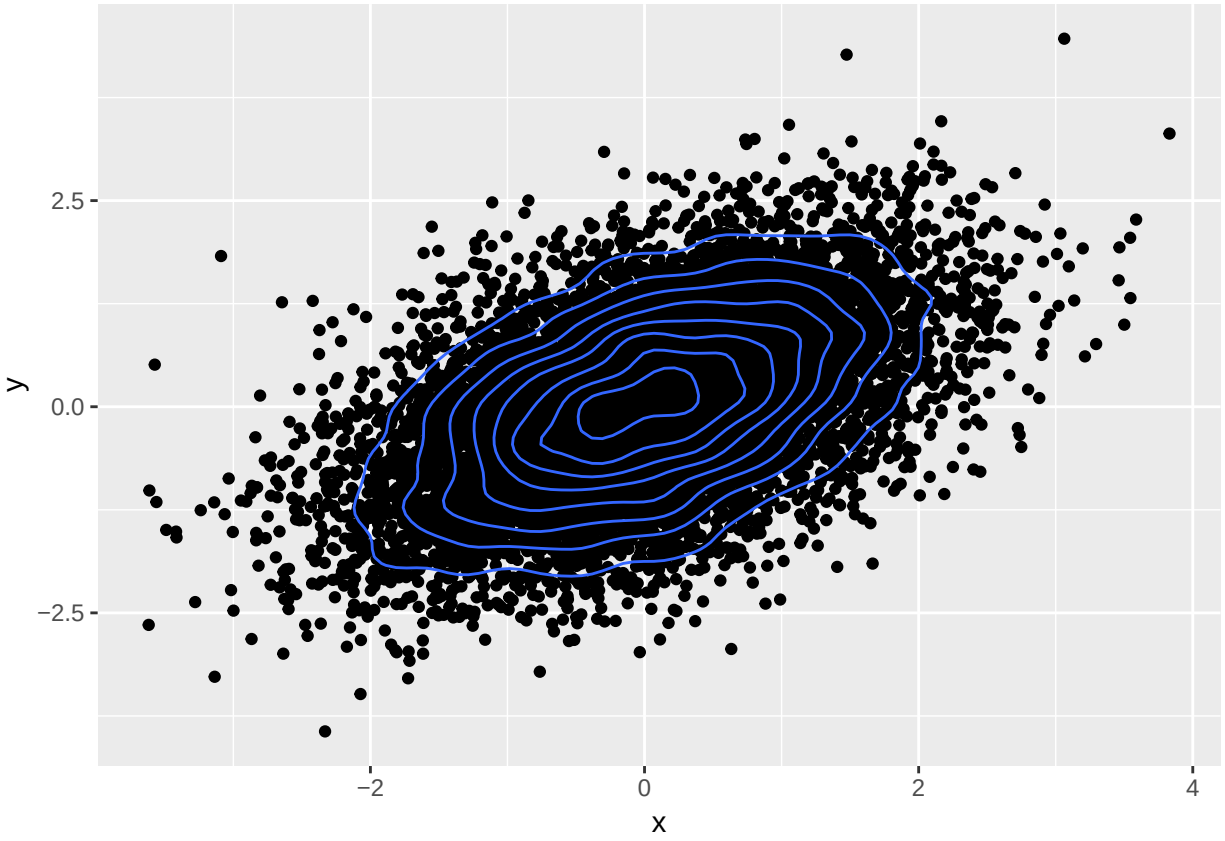
```
g <- ggplot(newdat, aes(x = t, y = value,
            group = variable,
            color = variable,
            linetype = variable)) +
    geom_line() +
    theme(legend.position = "bottom") +
    ggtitle("Multiple Series")
print(g)
```

Draw from bivariate normal.

```
library(mvtnorm)
Sigma <- matrix(c(1, 1/2, 1/2, 1), 2, 2)
x <- rmvnorm(n = 10000, mean = c(0,0), sigma = Sigma)
dat <- data.frame(x)
colnames(dat) <- c("x", "y")
```

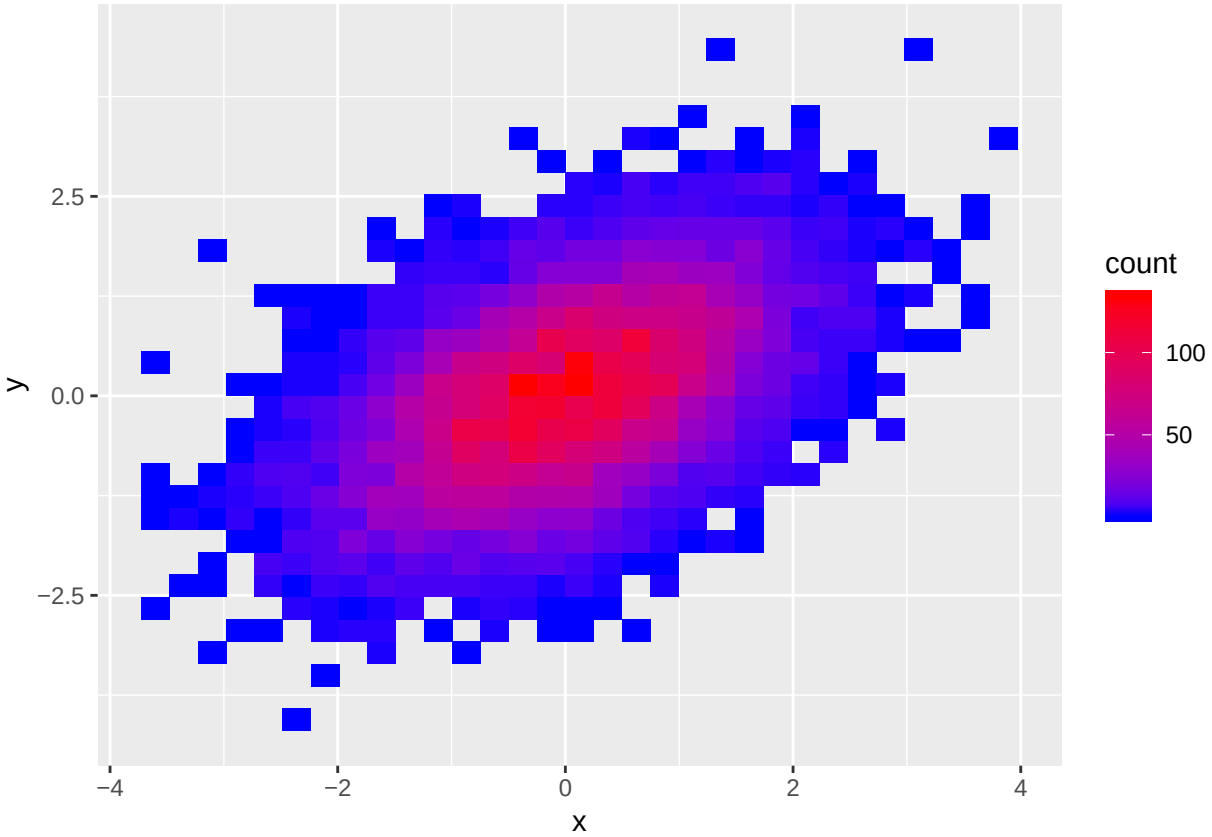
Plot the points and superimpose contours.

```
g <- ggplot(dat, aes(x=x, y=y)) +
    geom_point() +
    geom_density2d()
print(g)
```



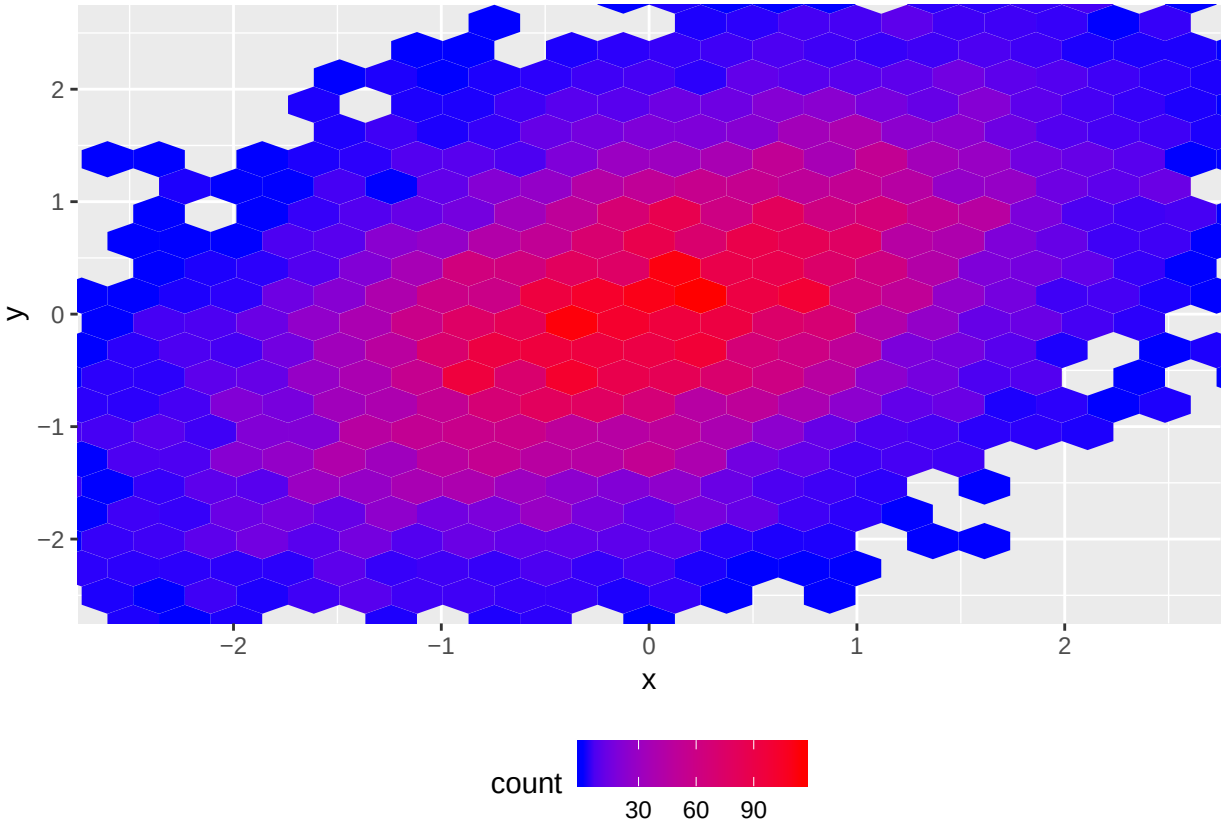
Plot bins instead to display density values.

```
g <- ggplot(dat, aes(x=x, y=y)) +
    geom_bin2d() +
    scale_fill_gradient(low = "blue", high = "red")
print(g)
```



Plot hexigonal bins instead.

```
library(hexbin)
g <- ggplot(dat, aes(x=x, y=y)) +
    geom_hex() +
    scale_fill_gradient(low = "blue", high = "red") +
    coord_cartesian(xlim = c(-2.5, 2.5), ylim = c(-2.5, 2.5)) +
    theme(legend.position = "bottom")
print(g)
```

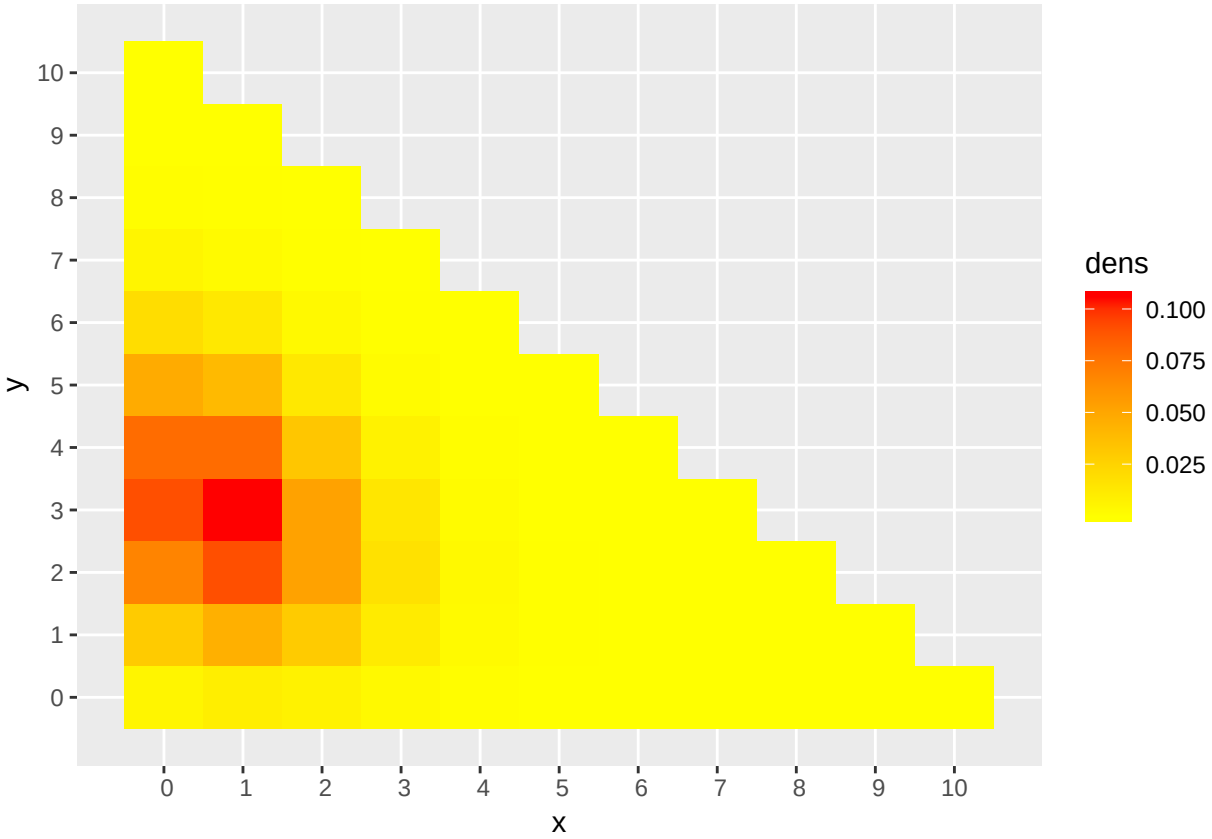


Generate data from a trinomial distribution.

```
m <- 10
grid <- expand.grid(x = 0:m, y = 0:m)
grid <- grid[grid$x + grid$y <= m,]
grid$z <- m - grid$x - grid$y
```

Plot the trinomial data on a 2-d grid (no need to display the redundant third coordinate).

```
grid$dens <- apply(grid, 1, dmultinom, size=m, prob=c(0.1, 0.3, 0.6))
g <- ggplot(grid, aes(x=x, y=y)) +
    geom_raster(aes(fill = dens)) +
    scale_fill_gradient(low = "yellow", high = "red") +
    scale_x_discrete(limits = 0:m) +
    scale_y_discrete(limits = 0:m)
print(g)
```



To save the last plot, use `ggsave`. File type is determined by extension (pdf, png, jpg, etc).

```
ggsave("plot.pdf", width = 5, height = 5)
```