How to create a facet in ggplot, except with different variables

I have a data frame with 3 variables, which are all wind speeds. I want to check how well the hardware was calibrated by plotting all the variables against each other. Although there are three in this instance, it may be that there are up to 6.

This would result in 3 different graphs, where the `x` and `y` parameters keep changing. I'd really like to plot these using facets- or something with the same appearance.

Here is some sample data, in a data frame called `wind`:

```
wind <- structure(list(speed_60e = c(3.029, 3.158, 2.881, 2.305, 2.45,
2.358, 2.325, 2.723, 2.567, 1.972, 2.044, 1.745, 2.1, 2.08, 1.914,
2.44, 2.356, 1.564, 1.942, 1.413, 1.756, 1.513, 1.263, 1.301,
1.403, 1.496, 1.828, 1.8, 1.841, 2.014), speed_60w = c(2.981,
3.089, 2.848, 2.265, 2.406, 2.304, 2.286, 2.686, 2.511, 1.946,
2.004, 1.724, 2.079, 2.058, 1.877, 2.434, 2.375, 1.562, 1.963,
1.436, 1.743, 1.541, 1.256, 1.312, 1.402, 1.522, 1.867, 1.837,
1.873, 2.055), speed_40 = c(2.726, 2.724, 2.429, 2.028, 1.799,
1.863, 1.987, 2.445, 2.282, 1.938, 1.721, 1.466, 1.841, 1.919,
1.63, 2.373, 2.22, 1.576, 1.693, 1.185, 1.274, 1.421, 1.071,
1.163, 1.166, 1.504, 1.77, 1.778, 1.632, 1.545)), .Names = c("speed_60e",
"speed_60w", "speed_40"), class = "data.frame", row.names = c("1",
"2", "3", "4", "5", "6", "7", "8", "9", "10", "11", "12", "13",
"14", "15", "16", "17", "18", "19", "20", "21", "22", "23", "24",
"25", "26", "27", "28", "29", "30"))

R> head(wind)
  speed_60e speed_60w speed_40
1     3.029     2.981    2.726
2     3.158     3.089    2.724
3     2.881     2.848    2.429
4     2.305     2.265    2.028
5     2.450     2.406    1.799
6     2.358     2.304    1.863
```

I wish to plot three square graphs. An individual one can be plotted by calling

```
ggplot() + geom_point(data=wind, aes(wind[,1],wind[,3]), alpha=I(1/30),
                      shape=I(20), size=I(1))
```

Any idea how I can do this?

r ggplot2

edited Apr 1 '11 at 10:24
Gavin Simpson
94.5k 9 170 276

asked Apr 1 '11 at 10:17
Chris
478 8 19

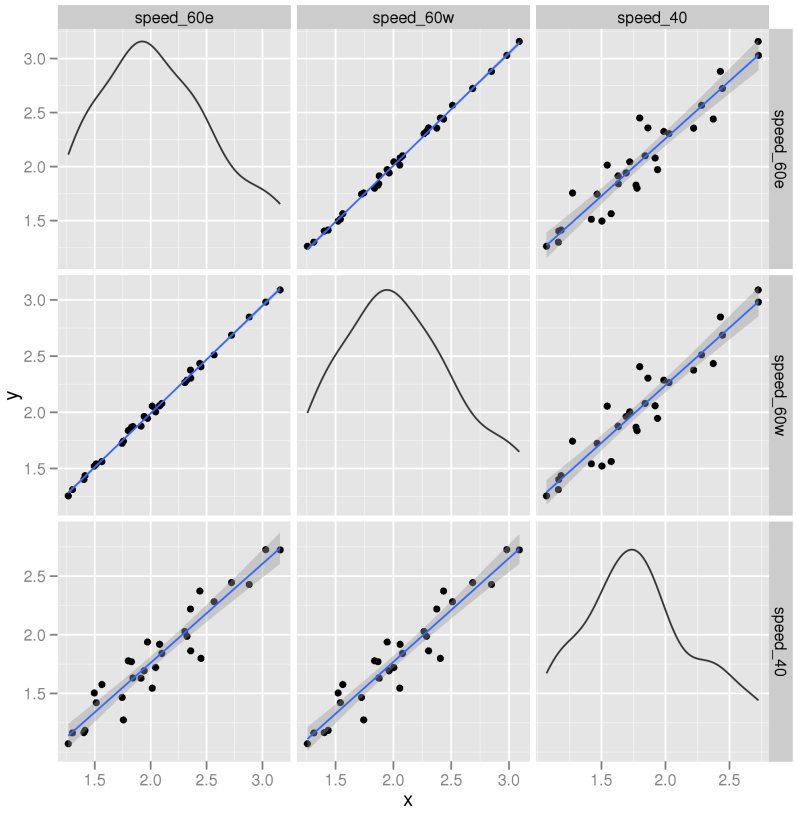
+1 For a very clear question and using the results of dput() to share your data – Andrie Apr 1 '11 at 12:11

3 Answers

Will something like this do?

```
plotmatrix(data = wind) + geom_smooth(method="lm")
```

Which gives:



Hadley calls this a "Crude experimental scatterplot matrix", but it might suffice for your needs?

Edit: Currently, `plotmatrix()` isn't quite flexible enough to handle all of @Chris' requirements regarding specification of the `geom_point()` layer. However, we can cut the guts out of `plotmatrix()` as use Hadley's nice code to create the data structure needed for plotting, but plot it however we like using standard `ggplot()` calls. This function also drops the densities but you can look into the code for `plotmatrix()` to see how to get them.

First, a function that expands the data from the wide format to the repeated format required for a pairs plot where we plot each variables against every other, but not itself.

```
Expand <- function(data) {
    grid <- expand.grid(x = 1:ncol(data), y = 1:ncol(data))
    grid <- subset(grid, x != y)
    all <- do.call("rbind", lapply(1:nrow(grid), function(i) {
        xcol <- grid[i, "x"]
        ycol <- grid[i, "y"]
        data.frame(xvar = names(data)[ycol], yvar = names(data)[xcol],
                   x = data[, xcol], y = data[, ycol], data)
    }))
    all$xvar <- factor(all$xvar, levels = names(data))
    all$yvar <- factor(all$yvar, levels = names(data))
    all
}
```

Note: *all* this does is steal Hadley's code from `plotmatrix()` - I have done nothing fancy here.

Expand the data:

```
wind2 <- Expand(wind)
```

Now we can plot this as any other long-format data object required by `ggplot()`:

```
ggplot(wind2, aes(x = x, y = y)) +
    geom_point(alpha = I(1/10), shape = I(20), size = I(1)) +
    facet_grid(xvar ~ yvar, scales = "free")
```

If you want the densities, then we can pull out that bit of code two into a helper function:

```
makeDensities <- function(data) {
    densities <- do.call("rbind", lapply(1:ncol(data), function(i) {
        data.frame(xvar = names(data)[i], yvar = names(data)[i],
                   x = data[, i])
    }))
    densities
}
```

Then compute the densities for the **original** data:

```
dens <- makeDensities(wind)
```

and then add then using the same bit of code from `plotmatrix()`:

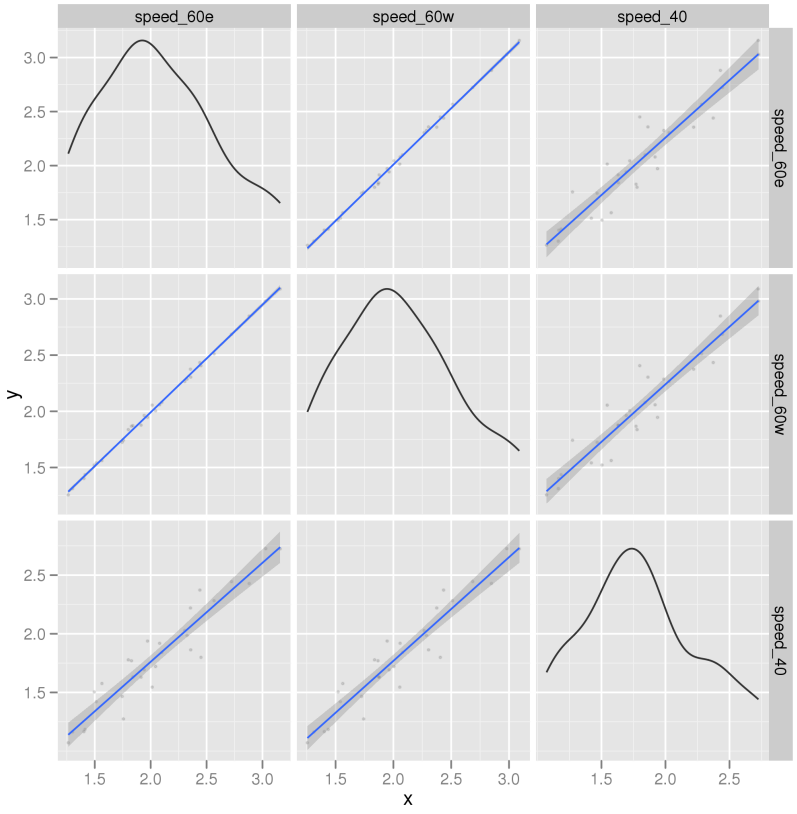
```
ggplot(wind2, aes(x = x, y = y)) +
```

```
        geom_point(alpha = I(1/10), shape = I(20), size = I(1)) +
        facet_grid(xvar ~ yvar, scales = "free")+
        stat_density(aes(x = x, y = ..scaled.. * diff(range(x)) + min(x)),
                     data = dens, position = "identity", colour = "grey20",
                     geom = "line")
```

A complete version of the original figure I showed above but using the extracted code would be:

```
ggplot(wind2, aes(x = x, y = y)) +
        geom_point(alpha = I(1/10), shape = I(20), size = I(1)) +
        facet_grid(xvar ~ yvar, scales = "free")+
        stat_density(aes(x = x, y = ..scaled.. * diff(range(x)) + min(x)),
                     data = dens, position = "identity", colour = "grey20",
                     geom = "line") +
        geom_smooth(method="lm")
```

giving:



edited Apr 1 '11 at 11:11

answered Apr 1 '11 at 10:32

Gavin Simpson
94.5k 9 170 276

+1 For telling me about plotmatrix() – Andrie Apr 1 '11 at 10:41

If I could up-vote further, I would have. This amount of work deserves some extra points. Next best thing: I'll contact you separately and buy you a beer next time I am in London. – Andrie Apr 1 '11 at 11:27

I wish I could upvote this forever. – Erik Shilts Feb 14 '12 at 18:51

Melt the data first (convert it to long form).

```
mwind <- melt(wind)
ggplot(mwind, aes(value)) + geom_histogram() + facet_wrap(~ variable)
```

If you want to plot points, you need to add an index variable for the x axis.

answered Apr 1 '11 at 10:39

Richie Cotton
50.6k 13 108 206

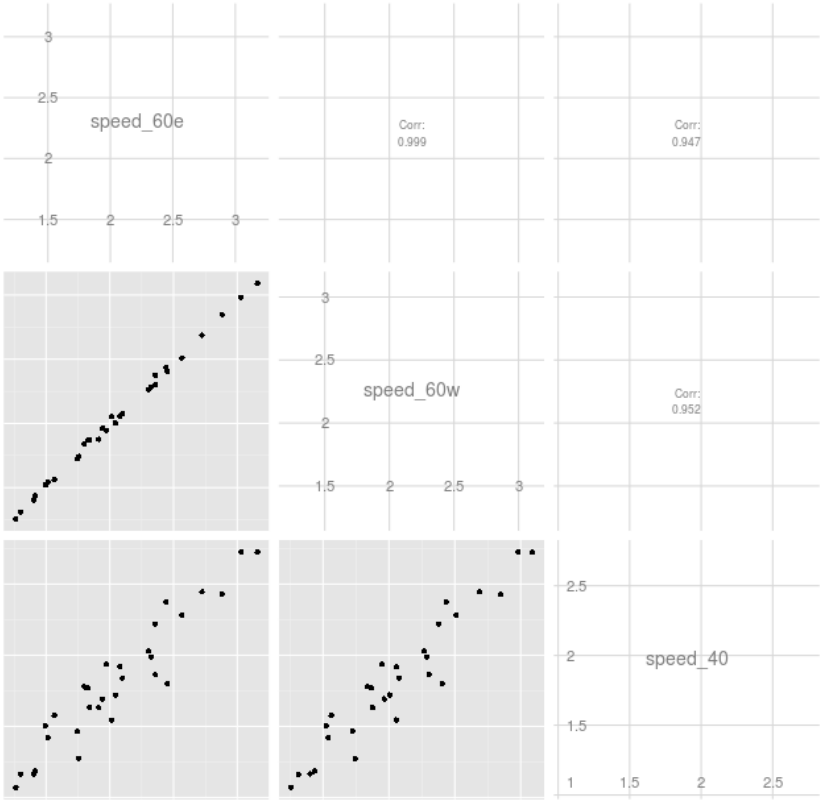
This was also useful for me- I had tried to melt the data but wasn't sure what to use as an id, so thanks. – Chris Apr 1 '11 at 12:55

@Chris part of the problem is that this is a non-standard melt. You are in effect needing to

duplicate/replicate the data in the long format to allow for plotting of one variable against the others. A simple melt won't work - hence the efforts Hadley went to in the code in `plotmatrix()` . If a melt would have worked, he'd have used that instead. – Gavin Simpson Apr 2 '11 at 1:33

`ggpairs` from the `GGally` package is quite nice for quick comparison of each variable in a dataframe:

```
ggpairs(wind)
```



It will also handle comparisons of numeric and factor data.

answered Mar 27 '13 at 5:32

naught101
4,593 4 40 69