Outlook on 3D data visualization

Tony Hsu

11/22/2017

```
#Make sure you install the package plot3D in order to reproduce this post
#install.packages("plot3D")

#Loading Packages
library("plot3D") #version 1.1.1

#I am using Version 1.0.153 of RStudio
```

Introduction and Motivation

Data visualization has always been an integral part of data science. If we want to demonstrate the result of our findings, then we must be able to use an effective data visualization tool. I am sure most people are familiar with 2D data visualizations. Whether it be histogram, scatterplot, barchart, or piechart, chances are that you have dealt with these techniques before in terms of a 2D manner. In Statistics 133, we examined the relationship between points and salary. In that case, we looked at how points affected salary.

However, let's suppose that we want to add an additional variable to our examination, for example, years of experience. Then, we would need a data visualization that allows us to look at three variables at a time.

There are many 3D data visualization packages in R; however, the one that I found to be most useful is `plot3D` due to its simplicity and effectiveness.

In this post, I will be going over some of the main functions of the R package `plot3D`:

- `scatter3D()`
- `text3D()`
- `arrows3D()`
- `rect3D()`

Moreover, I will utilizing some of the functions parameter such as `clab` to make the visualization more effective.

Background

Big data has been a hot topic in the recent years. Many companies are catching on this trend. 3D visualization techniques benefit companies by offering them a new way to "analyze, manage, and interact" with their data.

Analyzing data in the usual 2D format means that there is a limit on how much information we are able to gain from plots. 3D visualizations allow us to put an additional layer on data visualization.

Just this year on August 28, Karline Soetaert published the R package `plot3D`. Dr. Karline Soetaert currently serves as a senior scientist at Royal Netherlands Institute for Sea Research, focusing on areas such as biogeochemical cycles and open source software.

Examples

In this section, we will be using provided R dataset `mtcars`. The data came from the 1974 Motor Trend US magazine. It details various automobile design and performance for 32 automobiles. Some of the attributes of the dataset that we will looking are:

- `drat` Rear axle ratio
- `wt` Weight (1000 lbs)
- `qsec` 1/4 mile time

Here are some sample rows of the dataset

```
#showing head rows of mtcars
head(mtcars)
```

```
##                   mpg cyl disp  hp drat    wt  qsec vs am gear carb
## Mazda RX4         21.0   6  160 110 3.90 2.620 16.46  0  1    4    4
## Mazda RX4 Wag     21.0   6  160 110 3.90 2.875 17.02  0  1    4    4
## Datsun 710        22.8   4  108  93 3.85 2.320 18.61  1  1    4    1
## Hornet 4 Drive    21.4   6  258 110 3.08 3.215 19.44  1  0    3    1
## Hornet Sportabout 18.7   8  360 175 3.15 3.440 17.02  0  0    3    2
## Valiant           18.1   6  225 105 2.76 3.460 20.22  1  0    3    1
```

Data preparation

```
#assigning the variables x, y, z to be vectors from mtcars' columns
x = mtcars$drat
y = mtcars$wt
z = mtcars$qsec
```

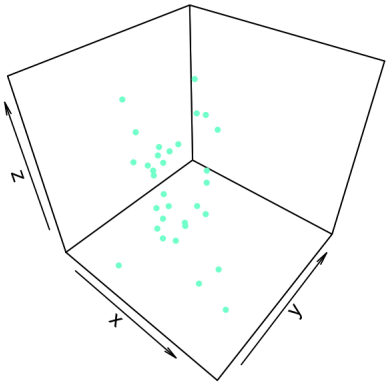
Scatter3D

Let's first take a look at the function `scatter3D`. Here are the functions' most important parameters:

- `x,y,z` = vectors for point coordinates

- `colvar` = the variable used for coloring
- `col` = color palette used to color the colvar variable

```
#ploting a basic 3D scatter plot
scatter3D(
x = x,
y = y,
z = z,
colvar = NULL,
col = "aquamarine" ,
pch = 19,
cex = 0.5
)
```

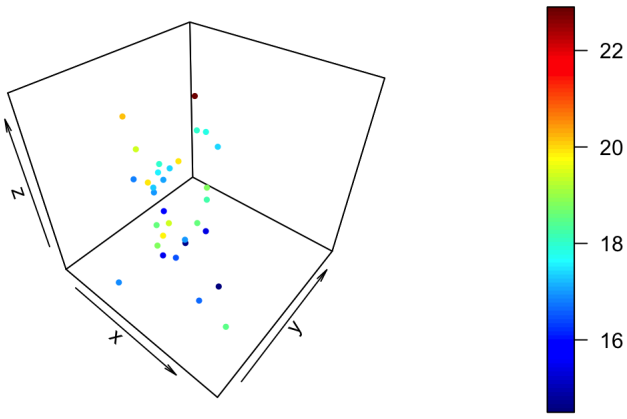


Now let's change some other parameters to make the plot more effective and "fancier".

Changing the color of the points based on some variable

We want to, for example, change the color based on the `z` variable, then we can put `z` in the `colvar` parameter.

```
#ploting a scatter plot with color variable dependent on z
scatter3D(
x = x,
y = y,
z = z,
colvar = z,
pch = 19,
cex = 0.5
)
```



As you can see, the color of the point changes based on the point's z-coordinate. There is also a color key showing the corresponding height

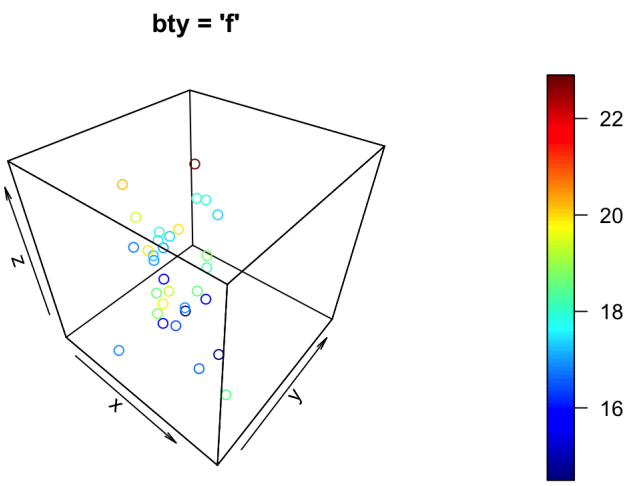
and color.

Changing the box around the plot

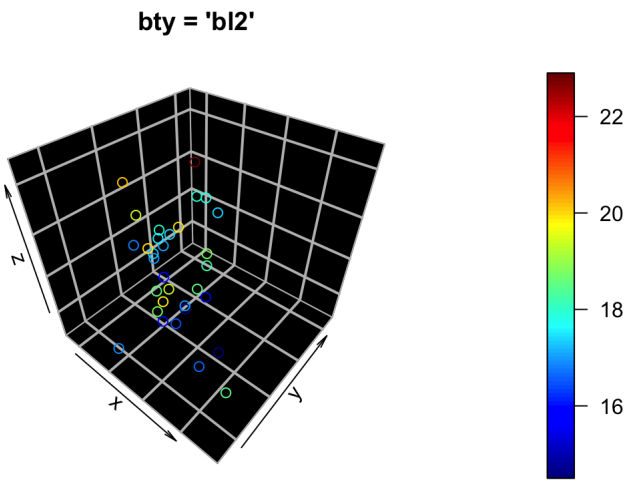
To change the box around the plot, we use the parameter `bty`. It takes values such as:

- `f` : full box
- `b` : default value. Only the back panels are visible
- `u` : means that the user will specify the arguments col.axis, col.panel, lwd.panel, col.grid, lwd.grid manually
- `n` : no box will be drawn. This is the same as setting box = FALSE
- `b2` : back panels and grid lines are visible
- `g` : grey background with white grid lines
- `bl` : black background
- `bl2` : black background with grey lines

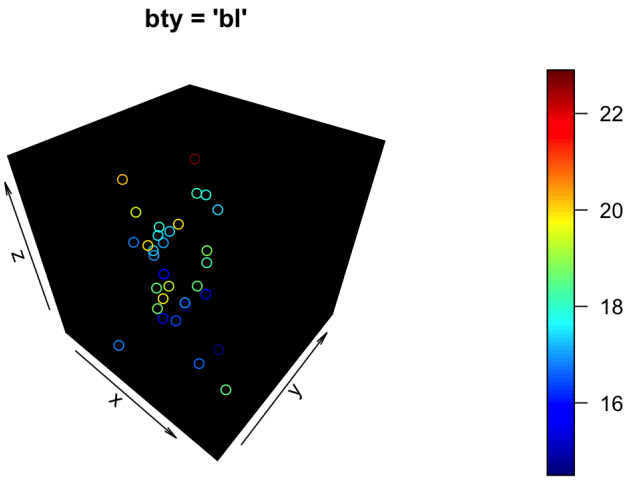
```
#Plotting scatter plots with different boxes
scatter3D(x, y, z, bty = "f", main = "bty = 'f'")
```



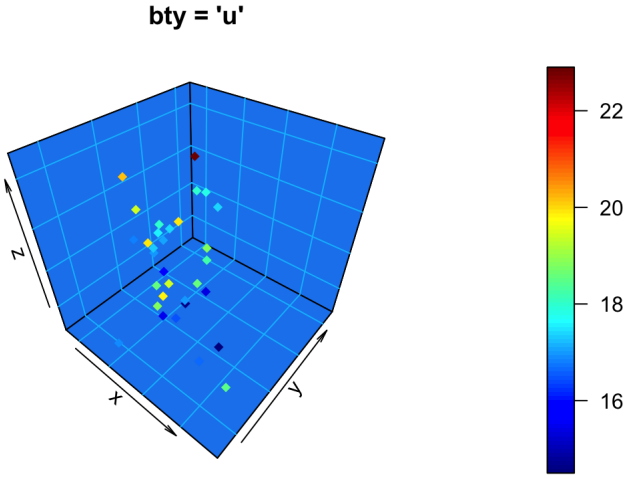

```
scatter3D(x, y, z, bty = "bl2", main = "bty = 'bl2'")
```




```
scatter3D(x, y, z, bty = "bl", main = "bty = 'bl'")
```



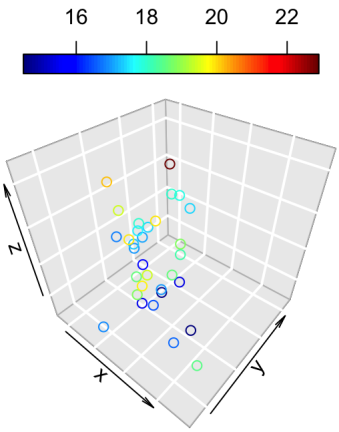

```
scatter3D(
x,
y,
z,
pch = 18,
bty = "u",
main = "bty = 'u'",
col.panel = "dodgerblue2",
col.grid = "deepskyblue"
)
```



Changin the position of the legend

For the parameter colkey, we can pass in a list that specifies the which side we want and how long we want the key to be.

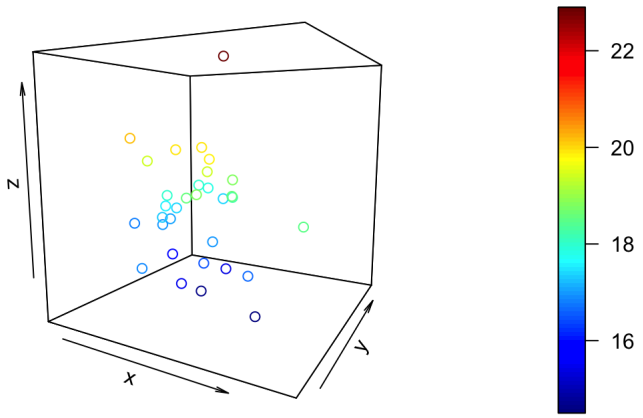
```
# Top colkey and a scatterplot
scatter3D(x, y, z, bty = "g",
          colkey = list(side = 3, length = 0.4))
```



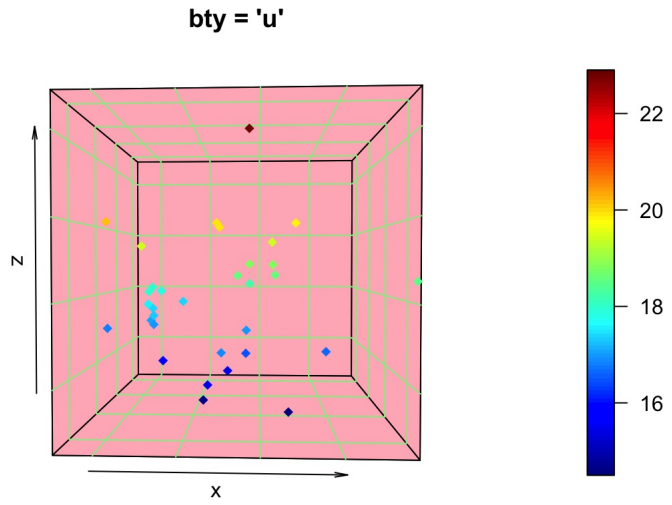
3D viewing direction

The parameters `phi` and `theta` can be used together to change the angles for the viewing direction. `phi` is the co-latitude and `theta` is the azimuthal direction.

```
#A scatterplot with phi and theta to change view direction
scatter3D(x, y, z, theta = 29, phi = 8)
```



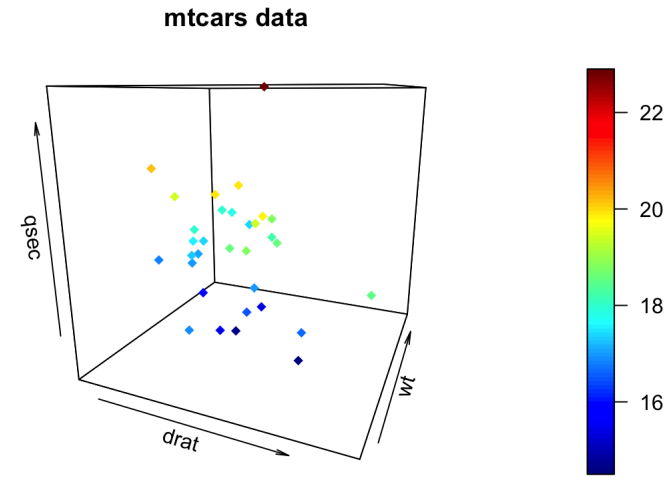

```
scatter3D(
x,
y,
z,
pch = 18,
bty = "u",
main = "bty = 'u'",
col.panel = "lightpink",
col.grid = "lightgreen",
phi = 1,
theta = 2
)
```



Axis labels

In order to add axis labels, we use the arguments: `xlab`, `ylab`, and `zlab`.

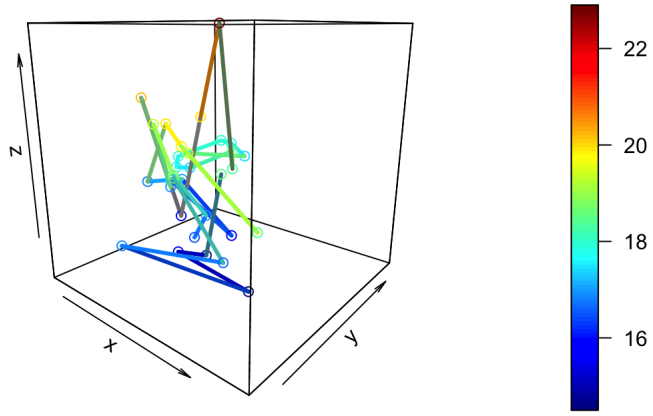
```
#Scatterplot with axises labeled
scatter3D(
  x,
  y,
  z,
  pch = 18,
  theta = 23,
  phi = 15,
  main = "mtcars data",
  xlab = "drat",
  ylab = "wt",
  zlab = "qsec"
  )
```



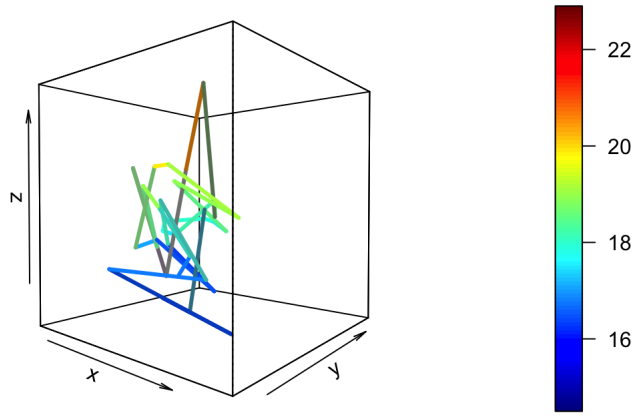
Line plots

If you want to add line to the plot, you have to specify `type` as one of the argument. Adding would be `l` for the type to plot only the line. `b` would be for both the line and points

```
#Graphing a line plot
scatter3D(x, y, z, phi = 15, bty = "f", type = "b", lwd = 3)
```



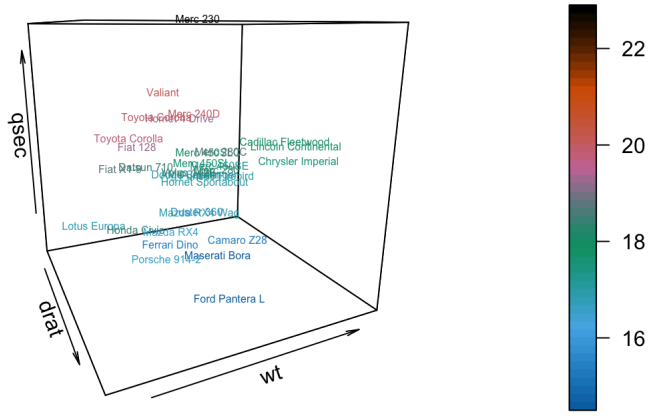

```
scatter3D(x, y, z, phi = 1, bty = "f", type = "l", lwd = 3)
```



text3D

`text3D()` can be used to plot the label of the point instead of just a plain point. Its arguments are very similar to that of the `scatter3D`.

```
#Plotting a text3D chart
text3D(
x,
y,
z,
labels = rownames(mtcars),
xlab = "drat",
ylab = "wt",
zlab = "qsec",
theta = 65,
phi = 15,
cex = 0.5,
colvar = z,
col = gg.col(50)
)
```



arrows3D

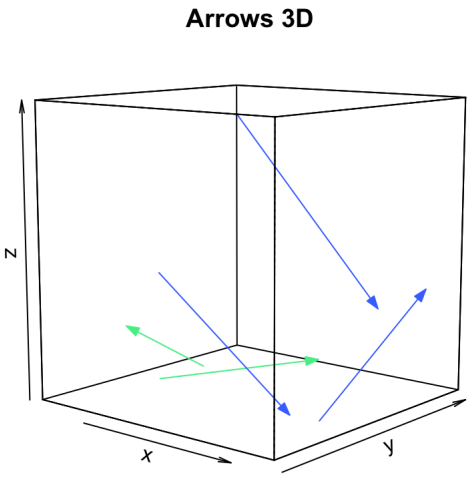
With the function `arrows3D`, the package `plot3D` allows us to plot arrows in 3D, and also in 2D.

`arrows3D` has the parameters of `x0, y0, z0, x1, y1, z1`. These can also be also be passed in the form of vectors to plot multiple vectors.

Data preparation

```
#Assigning the coordinates for points. First element of each vector correspond to the coordinates for
#each point.
x0 <- c(0, 0, 2, 0, 2)
y0 <- c(0, 1, 0, 0, 2)
z0 <- c(2, 0, 5, 0, -1)
x1 <- c(2.89, -3.46, 2.49, 1.96, 2)
y1 <- c(0.36,  2, 3.02, 2, 5)
z1 <- c(-0.28, 0.09, 1.05, 0.2, 1)
```

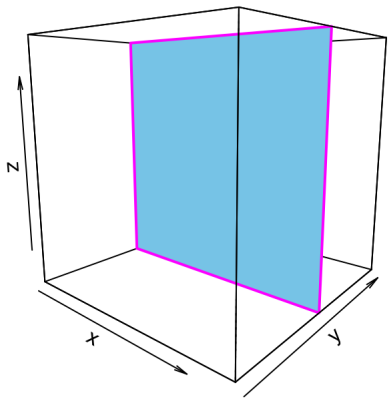
```
#Plotting arrows in a 3D setting
arrows3D(
  x0,
  y0,
  z0,
  x1,
  y1,
  z1,
  phi = 10,
  lwd = 1,
  d = 4,
  col = c("royalblue1", "seagreen2"),
  main = "Arrows 3D",
  bty = "f"
  )
```



react3D

Since we can plot arrows, you should not be surprised that we are also able to plot rectangles. This can be done using the function `rect3D`. `rect3D` also takes in the arguments `x0, y0, z0, x1, y1, z1`, which specifies where the rectangle should begin and stop in each dimension. It can takes the argument `border` to fill its border color.

```
rect3D(
  x0 = 0,
  y0 = 0.2,
  z0 = 0.2,
  x1 = 1.78,
  z1 = 5.9,
  bty = "f",
  facets = TRUE,
  border = "magenta",
  col = "skyblue",
  lwd = 2,
  phi = 10
  )
```



Discussion and Conclusions

Prior to learning about the package `plot3D`, I did not know how much potential 3D data visualization can bring. After doing much research and walking through different types of examples, I realize that 3D visualizations can be applied to various areas. For instance, in the beginning of post, I talked about how we can examine the relationship between points, years of experience, and salary. Moreover, this technique can be applied in many areas such as medicine or finance. The arrows plot can be used in applications of physics to represent vectors. Vector fields can certainly put arrows plot in good use. Lastly, in machine learning, we can use the rectangle in 3D to represent decision boundaries.

Thank you so much for reading for my post about `plot3D`. I hope after this post, you're more comfortable with 3D data visualizations.

References

- https://cran.r-project.org/web/packages/plot3D/plot3D.pdf
- https://stat.ethz.ch/R-manual/R-devel/library/datasets/html/mtcars.html
- http://www.sthda.com/english/wiki/impressive-package-for-3d-and-4d-graph-r-software-and-data-visualization
- https://www.youtube.com/watch?v=OgWe98EHsQc
- http://www.sthda.com/english/wiki/a-complete-guide-to-3d-visualization-device-system-in-r-r-software-and-data-visualization
- https://cran.r-project.org/web/packages/plot3D/vignettes/plot3D.pdf
- http://www.truecadd.com/news/the-importance-of-3d-animation-and-visualization-in-construction-and-manufacturing