Data Visualization in R

Genevieve Housman

AGAR Workshop

August 3, 2018

Session Outline

- Basics of ggplot2
  - scatter plots, bar plots, histograms, boxplots, heatmaps
- Basics of Gviz
  - ideogram, genome axis, sequence, data, annotation, gene region tracks
- Applying ggplot2 and Gviz to VCF data

Basics of ggplot2

ggplot2

- What
  - ***third*** graphics system for R (along with **base** and **lattice**)
  - implementation of the *Grammar of Graphics* by Leland Wilkinson (2005)
- When
  - Written by Hadley Wickham in 2005
- Why
  - Follows a grammar and supports a continuum of expertise
- How…

Other Graphics Systems

base

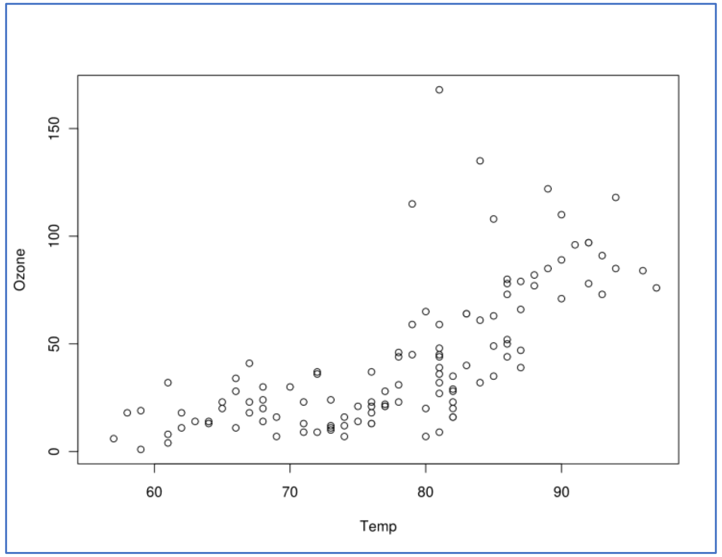
- Start with `plot` function (or similar)
- Use annotation functions to add/modify (`text`, `lines`, `points`, `axis`)
- Convenient, mirrors how we think of building plots and analyzing data
- Cannot go back once plot has started (i.e. to adjust margins); need to plan in advance
- Difficult to “translate” to other plot types once a new plot has been created

lattice

- Plots created with single function call (`xyplot`, `bwplot`, etc.)
- Annotation in plot is not intuitive
- Good for putting many plots on a screen (to see how *y* changes with *x* across levels of *z*)
- Cannot “add” to plot once created; requires intense preparation
- Sometimes awkward to specify entire plot in a single function call

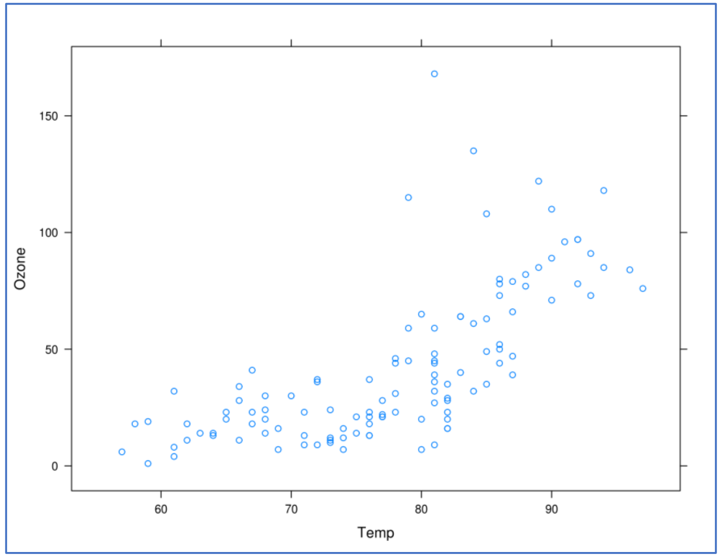
Other Graphics Systems

base



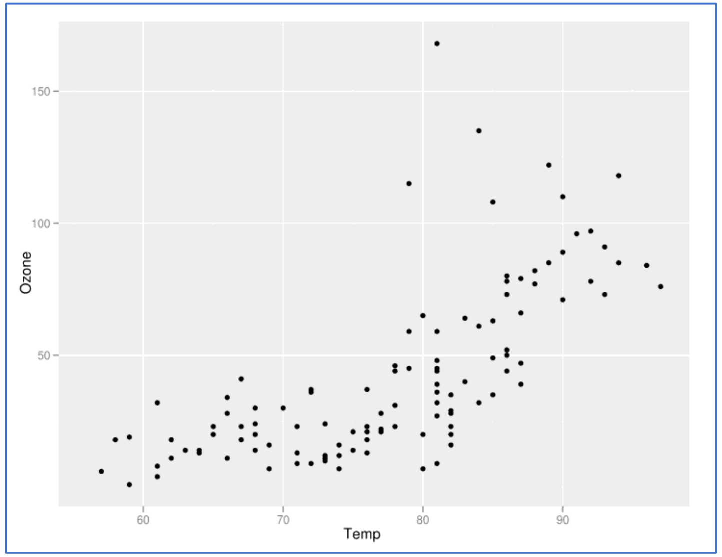
```
with(airquality, plot(Temp, Ozone))
```

lattice



```
xyplot(Ozone ~ Temp, airquality)
```

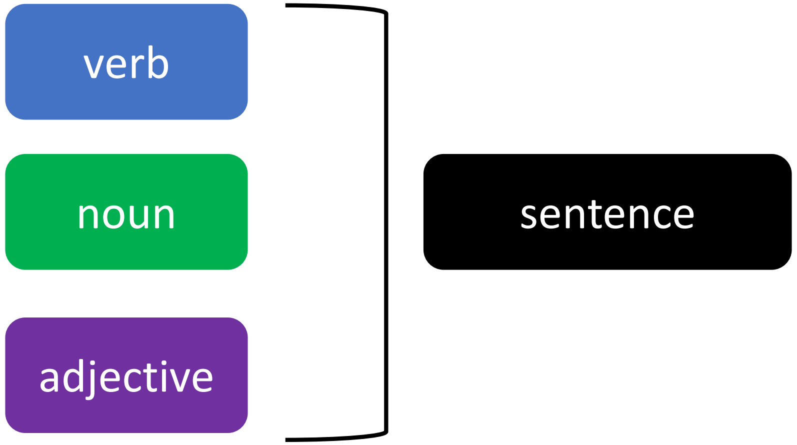
ggplot2:



```
ggplot(airquality, aes(Temp, Ozone)) +
      geom_point( )
```

ggplot2: Grammar of Graphics

- a coherent system for describing and building graphs
- allows for a theory of graphics on which to build new graphics

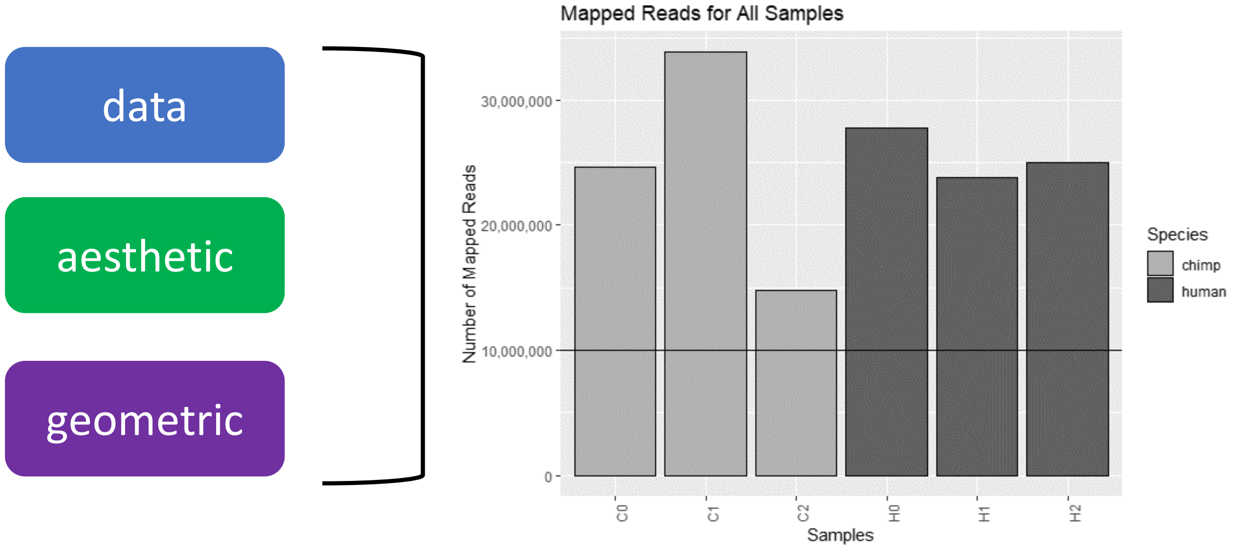


verb

noun

adjective

sentence



data

aesthetic

geometric

Mapped Reads for All Samples

ggplot2: Grammar of Graphics

“In brief, the grammar tells us that a statistical graphic is a **mapping** from data to **aesthetic** attributes (colour, shape, size) of **geometric** objects (points, lines, bars). The plot may also contain statistical transformations of the data and is drawn on a specific coordinate system”

from *ggplot2* book

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(
     mapping = aes(<MAPPINGS>),
     stat = <STAT>,
     position = <POSITION>
  ) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

ggplot2: Building a Plot

- ***ggplot*** is the main function
- supply ***data*** to visualize
- map variables to ***aes***thetic attributes
- ***geom***etric objects define what you see
- ***stat***istical transformations summarize data
- ***positions*** adjust placement of data in space
- ***coord***inate systems put data on plane of graphic
- ***facet***ing subsets the data to show multiple plots

Graphing Template

```
ggplot(data = <DATA>) +
  <GEOM_FUNCTION>(
     mapping = aes(<MAPPINGS>),
     stat = <STAT>,
     position = <POSITION>
  ) +
  <COORDINATE_FUNCTION> +
  <FACET_FUNCTION>
```

Let’s try building some plots!

ggplot2: Preparing Data

sample.details

```
Samples Species TimePoint CellDensity CellViability RNAConcentration
     H0   human         0         0.8         0.890              377
     H1   human         1         1.0         0.875              257
     H2   human         2         0.6         0.810              109
     C0   chimp         0         0.9         0.930              219
     C1   chimp         1         1.0         0.390              160
     C2   chimp         2         0.3         0.715               90
```

prop.reads

```
Samples NumTotal NumMapped PropMapped NumUnmapped
     H0 34275201  27787986   0.810732     6487215
     H1 28978629  23861725   0.823425     5116904
     H2 30053417  25009798   0.832178     5043619
     C0 30406842  24611163   0.809396     5795679
     C1 40051004  33819900   0.844421     6231104
     C2 17178516  14776677   0.860184     2401839
```

gene.counts

```
                  H0   H1   H2   C0   C1   C2
ENSG00000000003 4628 5940 3809 5079 6506 2623
ENSG00000000005  177    4    4   29    0    0
ENSG00000000419 2589 1255 1876 1501 1252  370
ENSG00000000457  309  695  316  582 2451  268
ENSG00000000460  997  591  434 1165 1407  156
ENSG00000000971    1    0    0    0    0    0
```

ggplot2: Preparing Data

sample.details

```
Samples Species TimePoint CellDensity CellViability RNAConcentration
     H0   human         0         0.8         0.890              377
     H1   human         1         1.0         0.875              257
     H2   human         2         0.6         0.810              109
     C0   chimp         0         0.9         0.930              219
     C1   chimp         1         1.0         0.390              160
     C2   chimp         2         0.3         0.715               90
```

scores

```
Samples         PC1       PC2        PC3        PC4       PC5           PC6 Species TimePoint CellDensity CellViability  RNAConcentration
     C0 -282135.133 -37806.44   38659.90   57289.09 -55611.44  2.451884e-08   chimp         0         0.9         0.930               219
     C1  377327.549 -87787.33   36933.51   78383.35  11784.00 -1.178564e-08   chimp         1         1.0         0.390               160
     C2    6222.875 179894.96   91593.70  -26106.48  16969.88  8.236373e-09   chimp         2         0.3         0.715                90
     H0 -285924.073 -64966.96  -34484.24   11975.85  60066.66  3.190043e-09   human         0         0.8         0.890               377
     H1   83237.796 -93318.89    1087.82 -137655.43 -16395.06 -2.644546e-08   human         1         1.0         0.875               257
     H2  101270.985 103984.66 -133790.69   16113.62 -16814.04  2.278867e-09   human         2         0.6         0.810               109
```

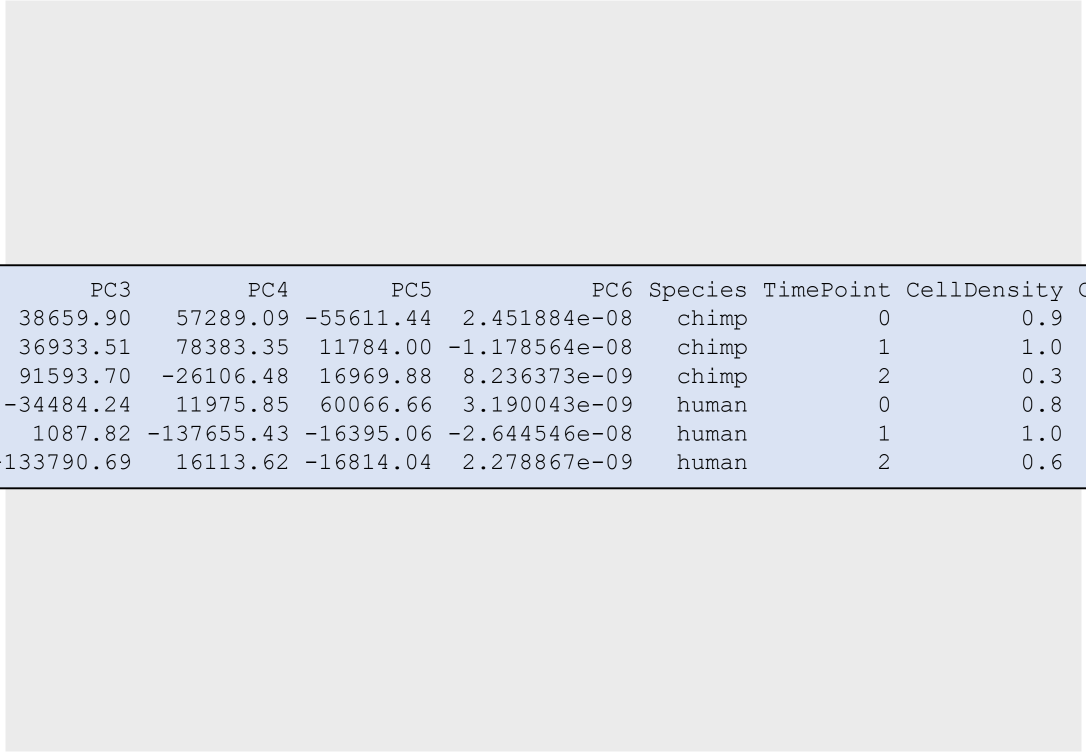
gene.counts

```
                  H0   H1   H2   C0   C1   C2
ENSG00000000003 4628 5940 3809 5079 6506 2623
ENSG00000000005  177    4    4   29    0    0
ENSG00000000419 2589 1255 1876 1501 1252  370
ENSG00000000457  309  695  316  582 2451  268
ENSG00000000460  997  591  434 1165 1407  156
ENSG00000000971    1    0    0    0    0    0
```

ggplot2: Building a Plot

supplying data

```
ggplot(data = scores)
```

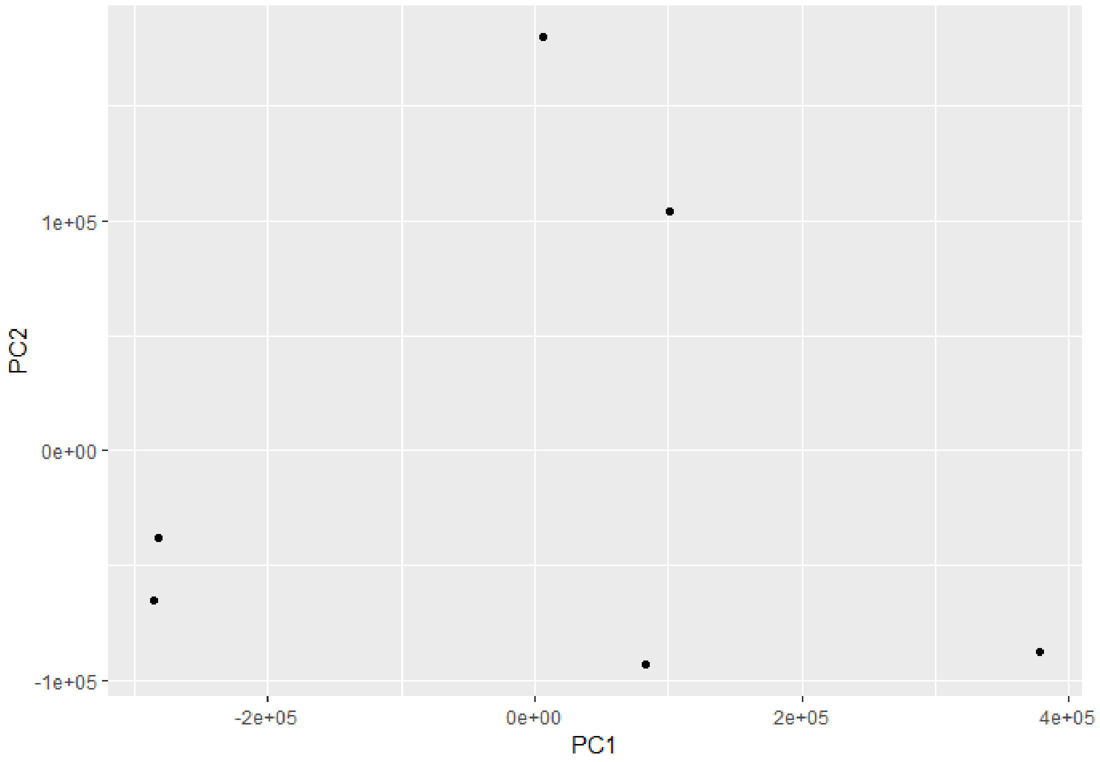


```
Samples         PC1       PC2        PC3        PC4       PC5           PC6 Species TimePoint CellDensity CellViability  RNAConcentration
     C0 -282135.133 -37806.44   38659.90   57289.09 -55611.44  2.451884e-08   chimp         0         0.9         0.930               219
     C1  377327.549 -87787.33   36933.51   78383.35  11784.00 -1.178564e-08   chimp         1         1.0         0.390               160
     C2    6222.875 179894.96   91593.70  -26106.48  16969.88  8.236373e-09   chimp         2         0.3         0.715                90
     H0 -285924.073 -64966.96  -34484.24   11975.85  60066.66  3.190043e-09   human         0         0.8         0.890               377
     H1   83237.796 -93318.89    1087.82 -137655.43 -16395.06 -2.644546e-08   human         1         1.0         0.875               257
     H2  101270.985 103984.66 -133790.69   16113.62 -16814.04  2.278867e-09   human         2         0.6         0.810               109
```

ggplot2: Building a Plot

supplying data
mapping aesthetics

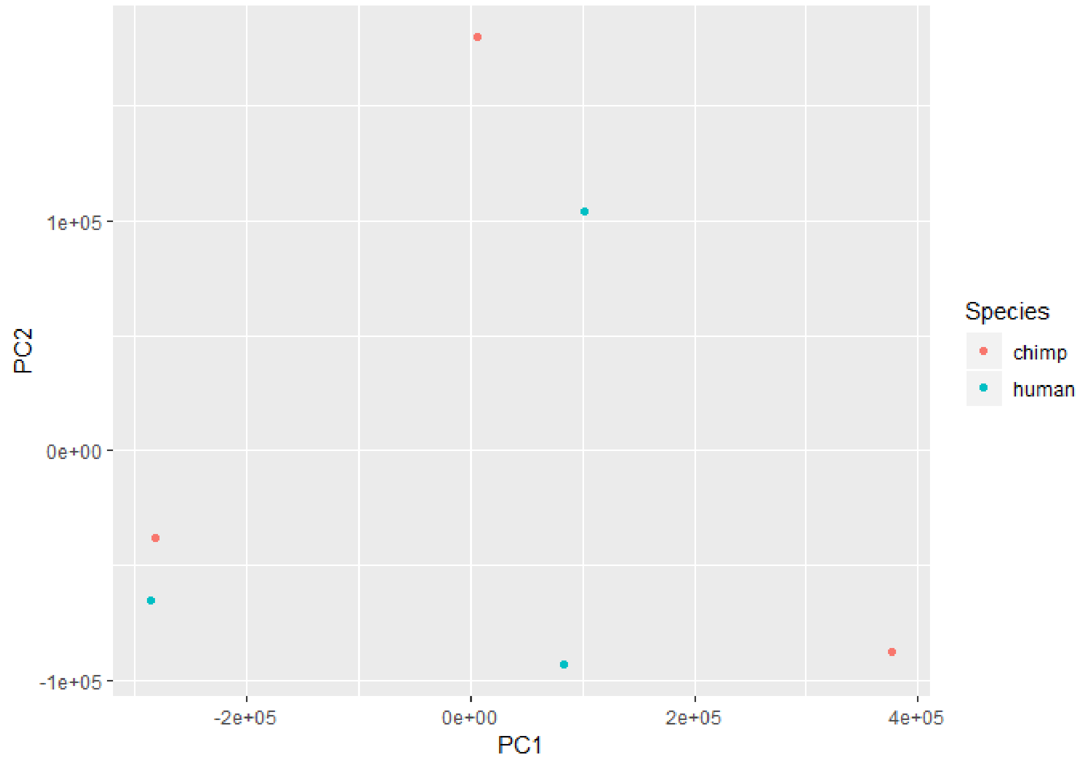
```
ggplot(data = scores) +
  geom_point(
    mapping = aes(x = PC1, y = PC2)
  )
```



ggplot2: Building a Plot

supplying data
mapping aesthetics

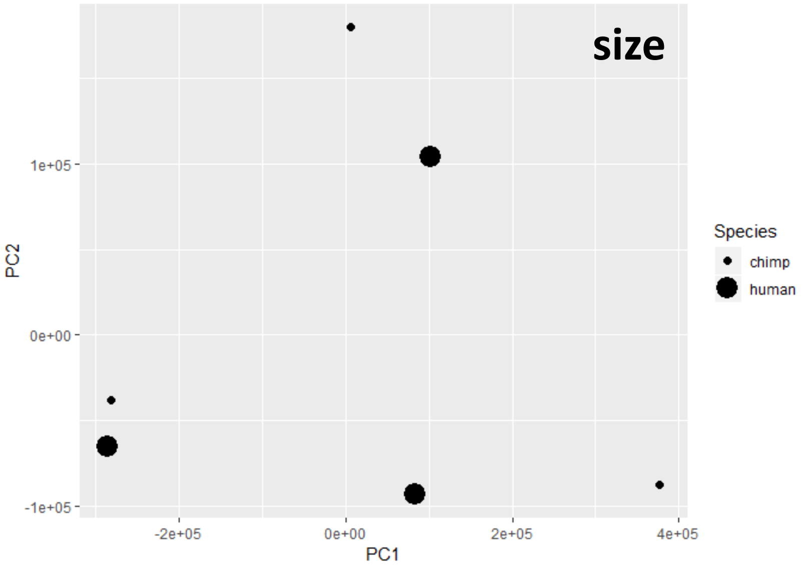
```
ggplot(data = scores) +
  geom_point(
    mapping = aes(x = PC1, y = PC2, color = Species)
  )
```

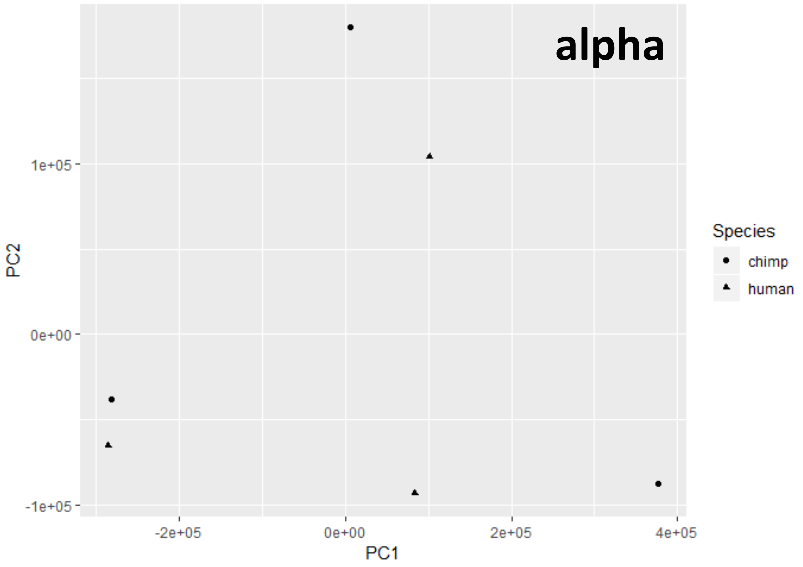


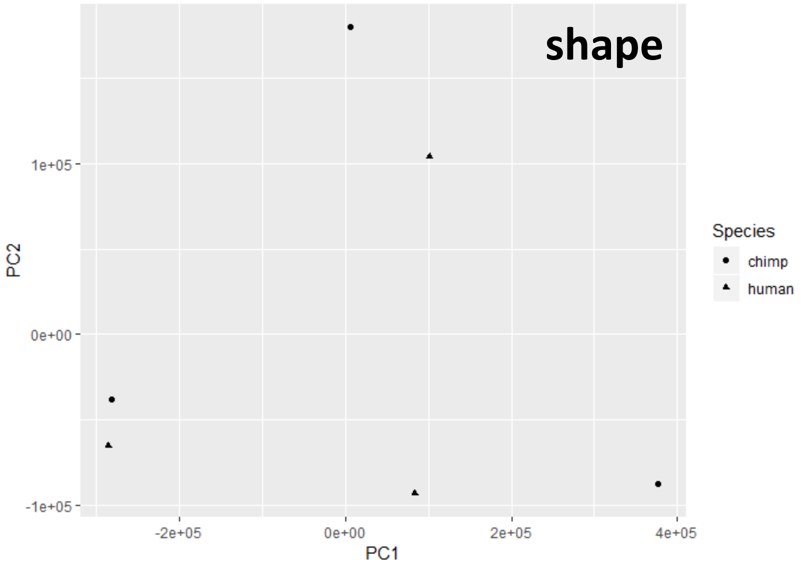
ggplot2: Building a Plot

supplying data
mapping aesthetics

```
ggplot(data = scores) +
  geom_point(
    mapping = aes(x = PC1, y = PC2, <MAPPINGS>)
  )
```


size

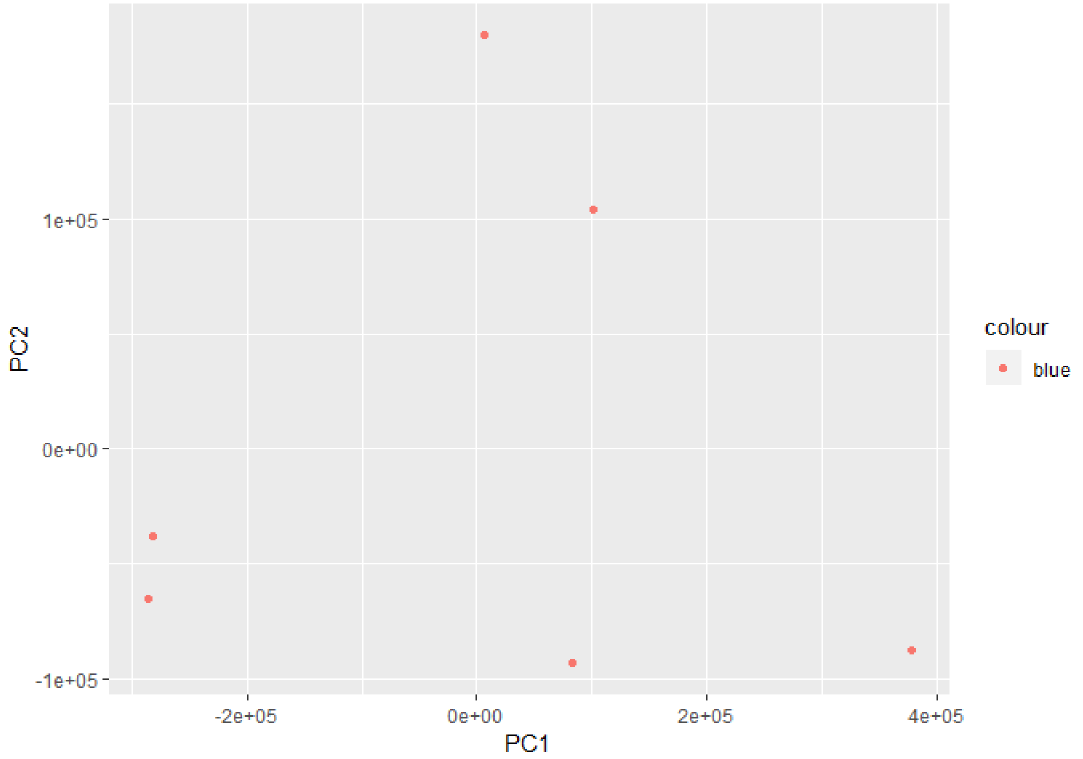

alpha


shape

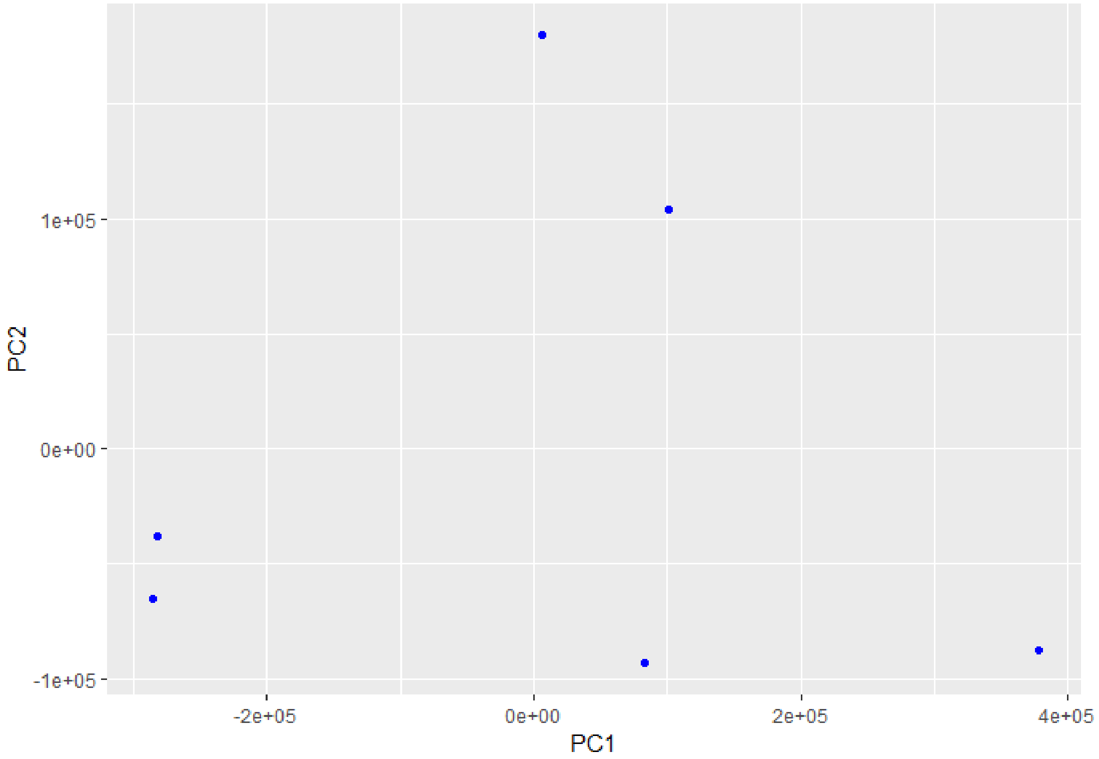
ggplot2: Building a Plot

supplying data
mapping aesthetics

```
ggplot(data = scores) +
  geom_point(
    mapping = aes(x = PC1, y = PC2, <MAPPINGS>)
  )
```



```
mapping = aes(x = PC1, y = PC2, color = "blue")
```

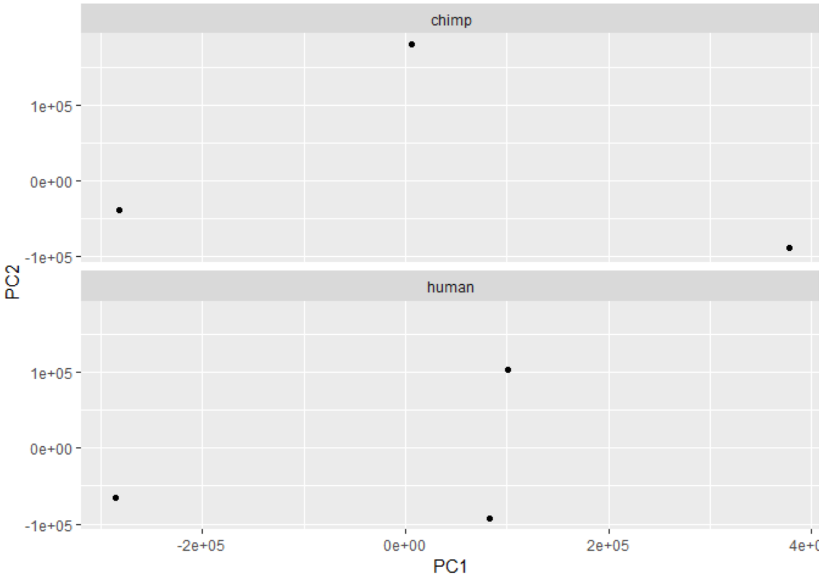


```
mapping = aes(x = PC1, y = PC2), color = "blue"
```

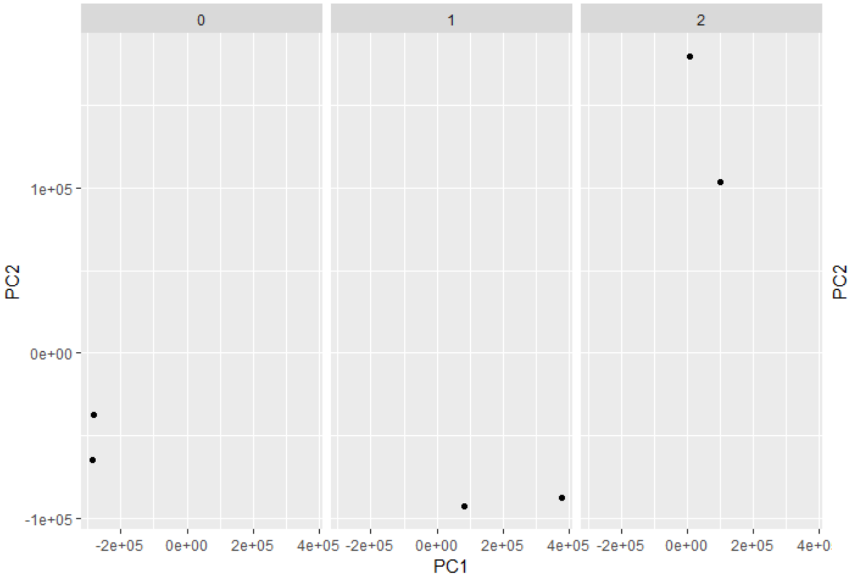
ggplot2: Building a Plot

supplying data
mapping aesthetics
splitting into facets

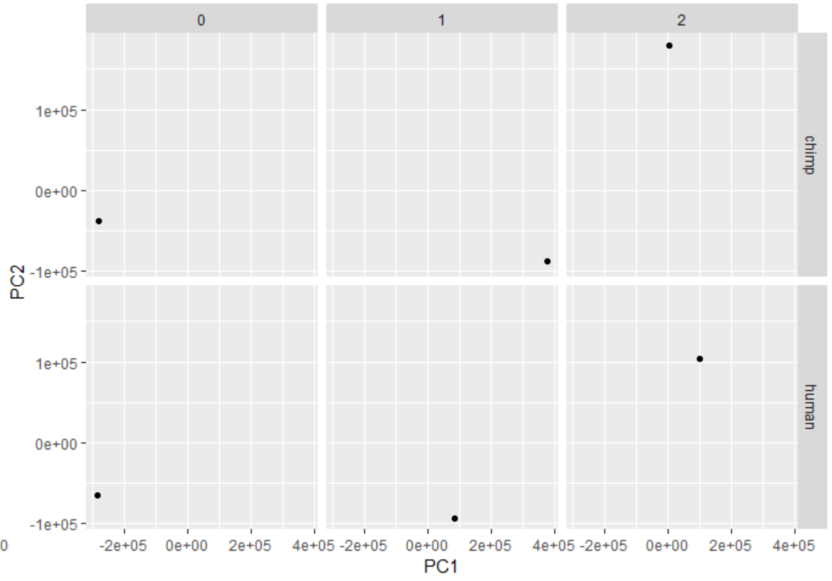
```
ggplot(data = scores) +
  geom_point(
    mapping = aes(x = PC1, y = PC2)
  ) +
  facet_wrap(<FORMULA>)
```



`facet_wrap(~ Species, nrow = 2)`



`facet_wrap(~ TimePoint)`

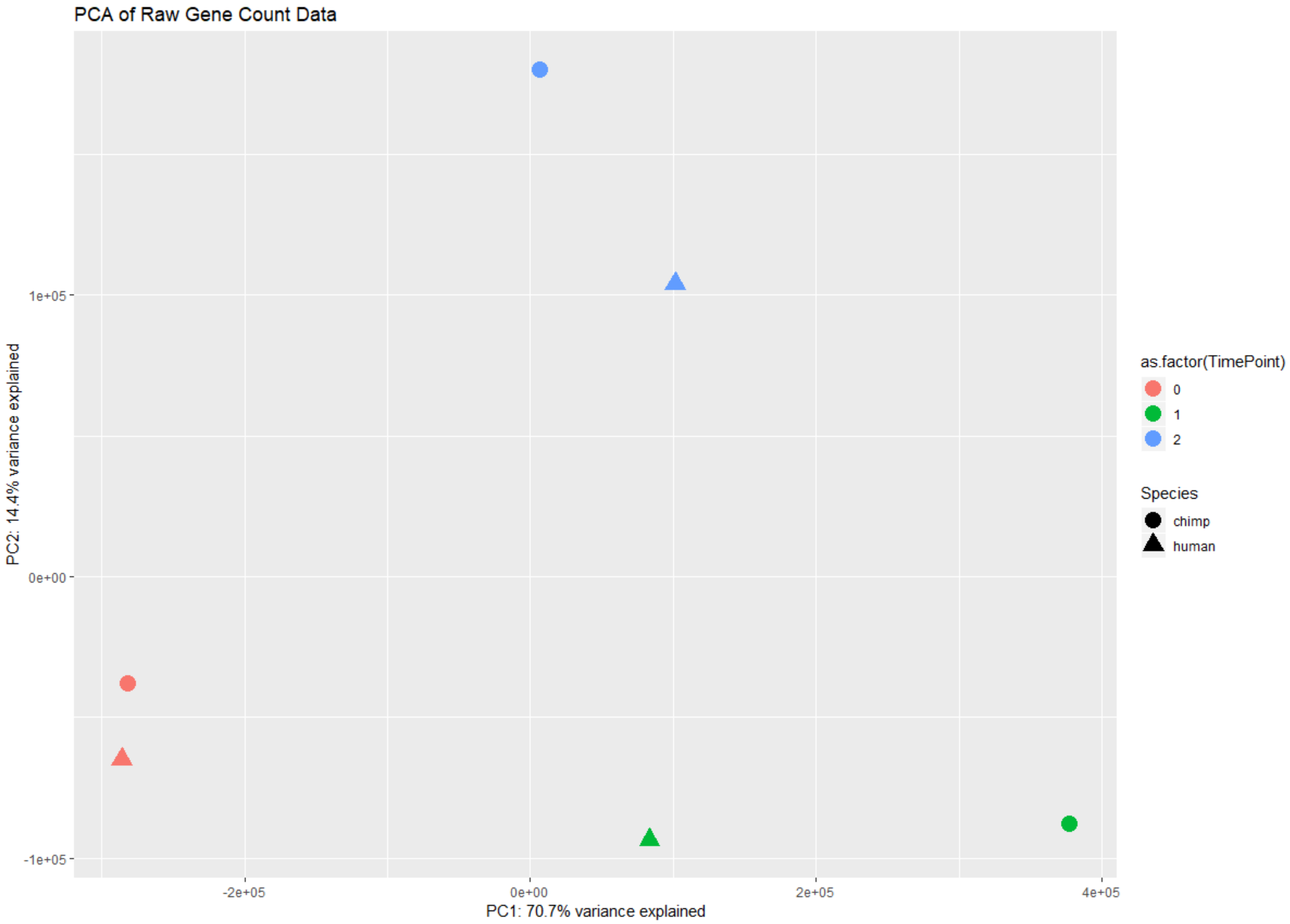


`facet_grid(Species ~ TimePoint)`

ggplot2: Building a Plot

EXERCISES

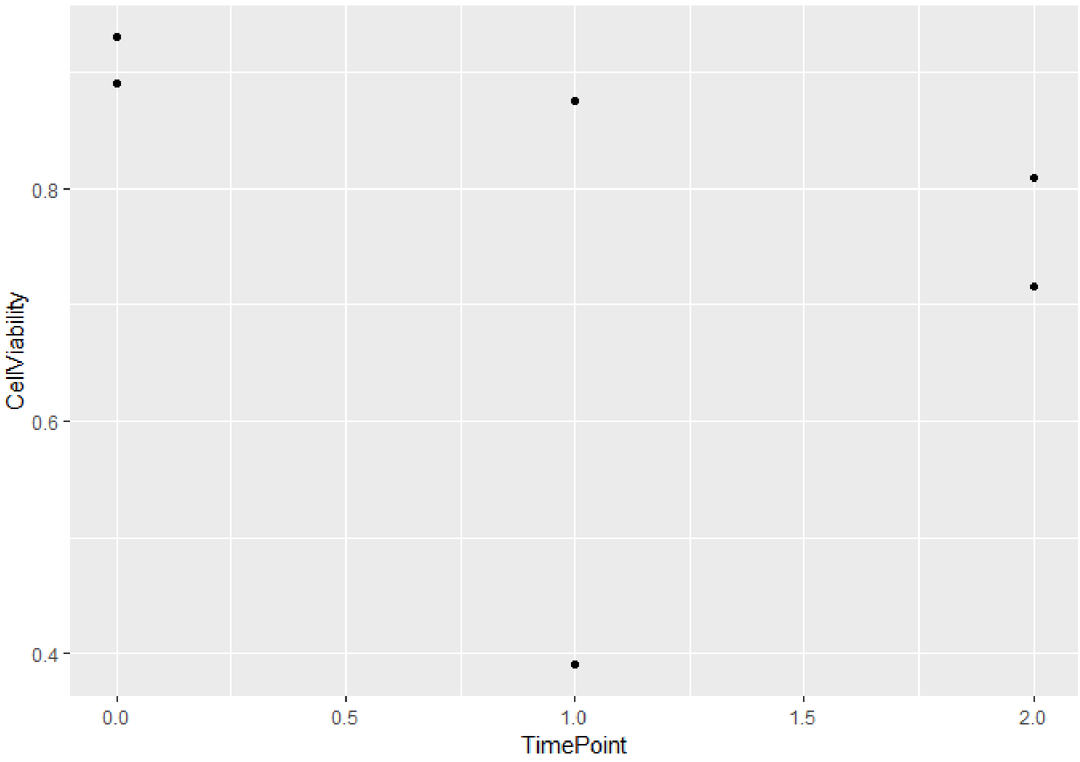
```
ggplot(data = scores) +
  geom_point(mapping = aes(x = PC1, y = PC2, color = as.factor(TimePoint), shape = Species), size = 5) +
  ggtitle("PCA of Raw Gene Count Data") +
  xlab(paste("PC1: ", round(summary(pca_genes)$importance[2,1],3)*100, "% variance explained", sep="")) +
  ylab(paste("PC2: ", round(summary(pca_genes)$importance[2,2],3)*100, "% variance explained", sep=""))
```

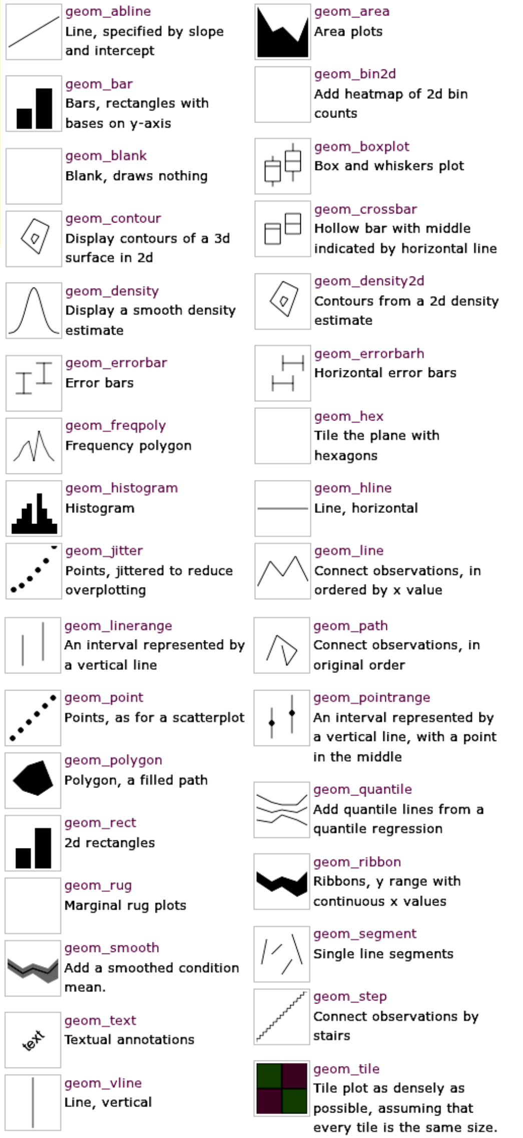


ggplot2: Building a Plot - point

supplying data
mapping aesthetics
defining geometric

```
ggplot(data = scores) +
  geom_point(
    mapping = aes(x = TimePoint, y = CellViability)
  )
```





geom_abline
Line, specified by slope and intercept

geom_bar
Bars, rectangles with bases on y-axis

geom_blank
Blank, draws nothing

geom_contour
Display contours of a 3d surface in 2d

geom_density
Display a smooth density estimate

geom_errorbar
Error bars

geom_freqpoly
Frequency polygon

geom_histogram
Histogram

geom_jitter
Points, jittered to reduce overplotting

geom_linerange
An interval represented by a vertical line

geom_point
Points, as for a scatterplot

geom_polygon
Polygon, a filled path

geom_rect
2d rectangles

geom_rug
Marginal rug plots

geom_smooth
Add a smoothed condition mean.

geom_text
Textual annotations

geom_vline
Line, vertical

geom_area
Area plots

geom_bin2d
Add heatmap of 2d bin counts

geom_boxplot
Box and whiskers plot

geom_crossbar
Hollow bar with middle indicated by horizontal line

geom_density2d
Contours from a 2d density estimate

geom_errorbarh
Horizontal error bars

geom_hex
Tile the plane with hexagons

geom_hline
Line, horizontal

geom_line
Connect observations, in ordered by x value

geom_path
Connect observations, in original order

geom_pointrange
An interval represented by a vertical line, with a point in the middle

geom_quantile
Add quantile lines from a quantile regression

geom_ribbon
Ribbons, y range with continuous x values

geom_segment
Single line segments

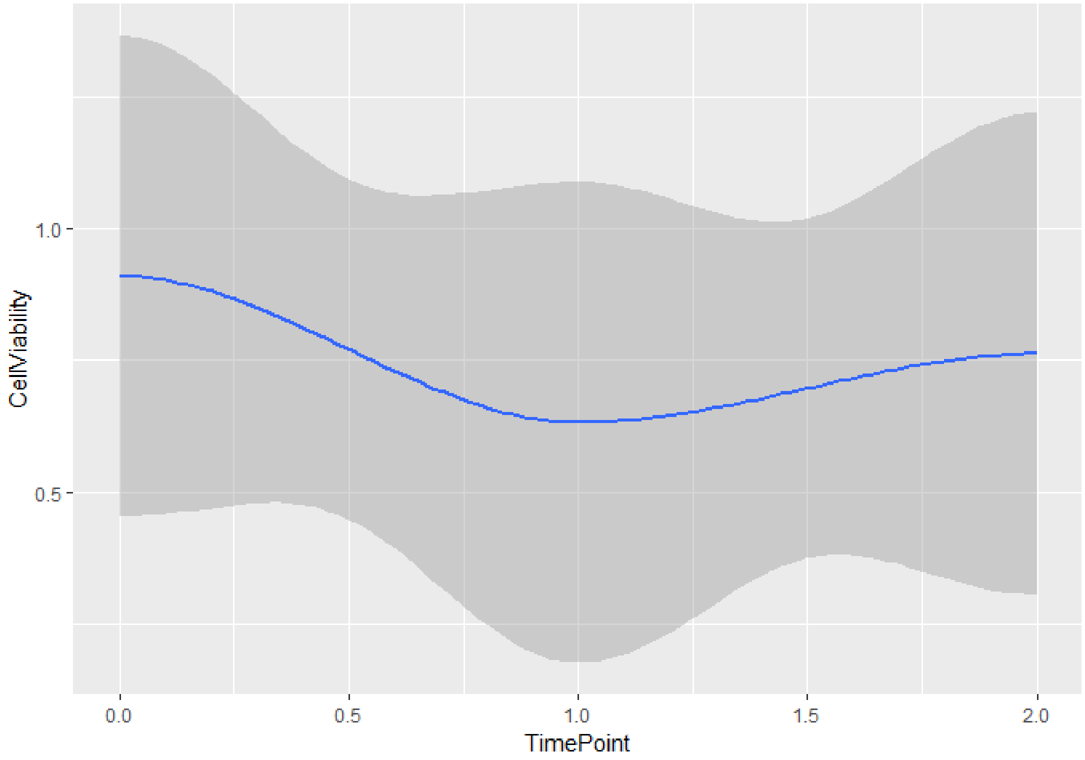
geom_step
Connect observations by stairs

geom_tile
Tile plot as densely as possible, assuming that every tile is the same size.

ggplot2: Building a Plot - smooth

supplying data
mapping aesthetics
defining geometric

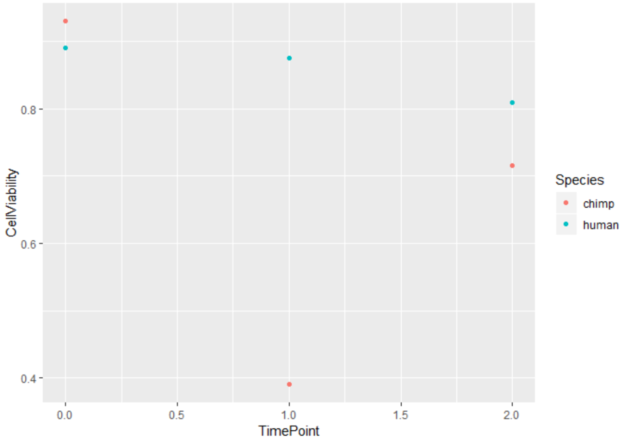
```
ggplot(data = scores) +
  geom_smooth(
    mapping = aes(x = TimePoint, y = CellViability)
  )
```



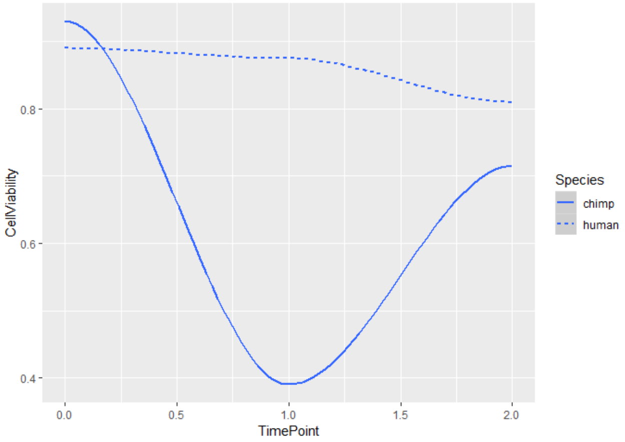
ggplot2: Building a Plot

supplying data
mapping aesthetics
defining geometric

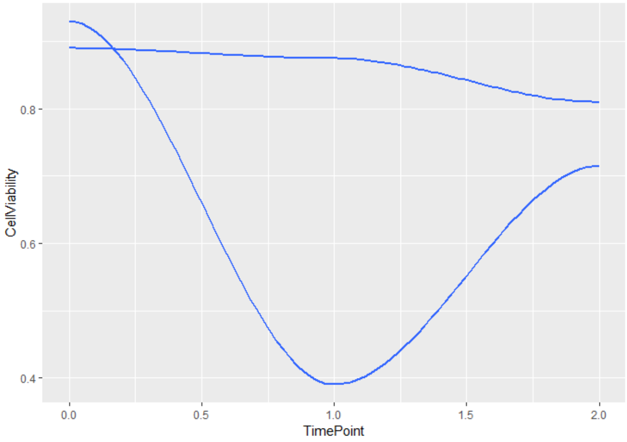
```
ggplot(data = scores) +
  <GEOM_FUNCTION> (
    mapping = aes(x = TimePoint, y = CellViability, <MAPPINGS>)
  )
```



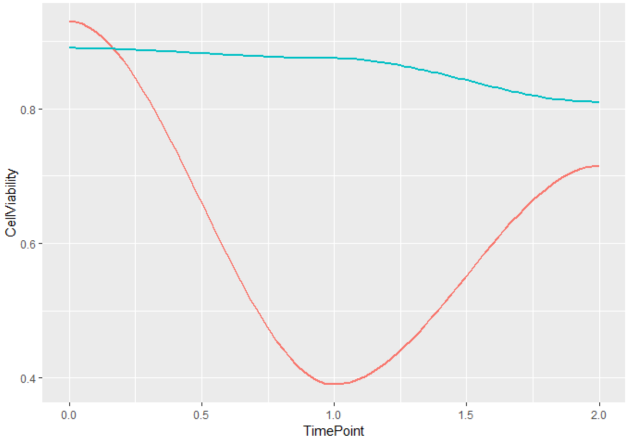
```
geom_point()
color = Species
```



```
geom_smooth()
linetype = Species
```



```
geom_smooth()
group = Species
```

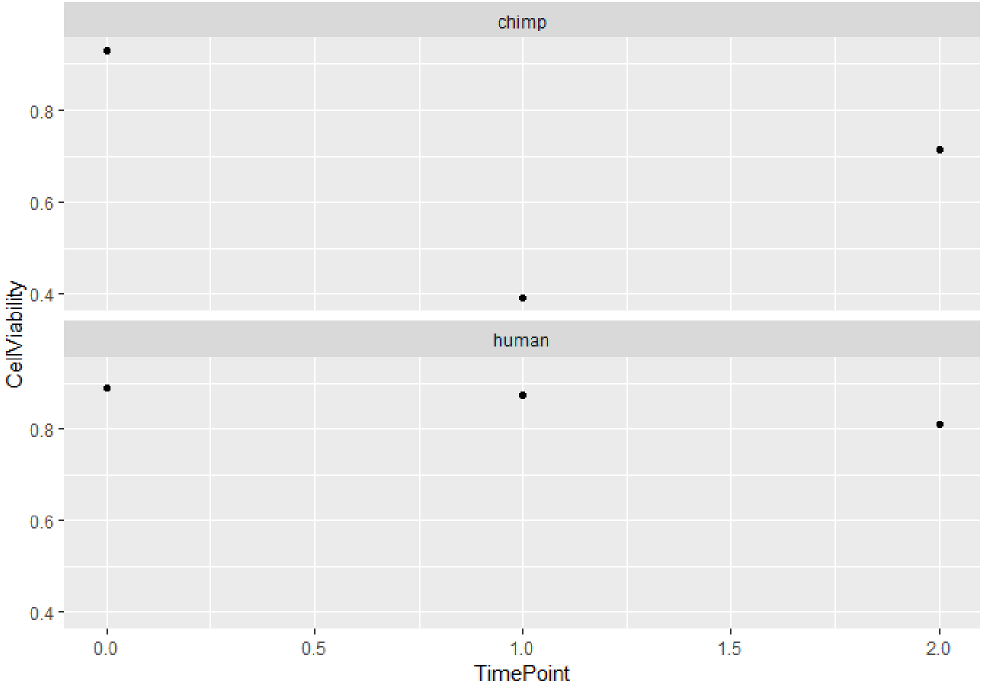


```
geom_smooth()
color = Species
```

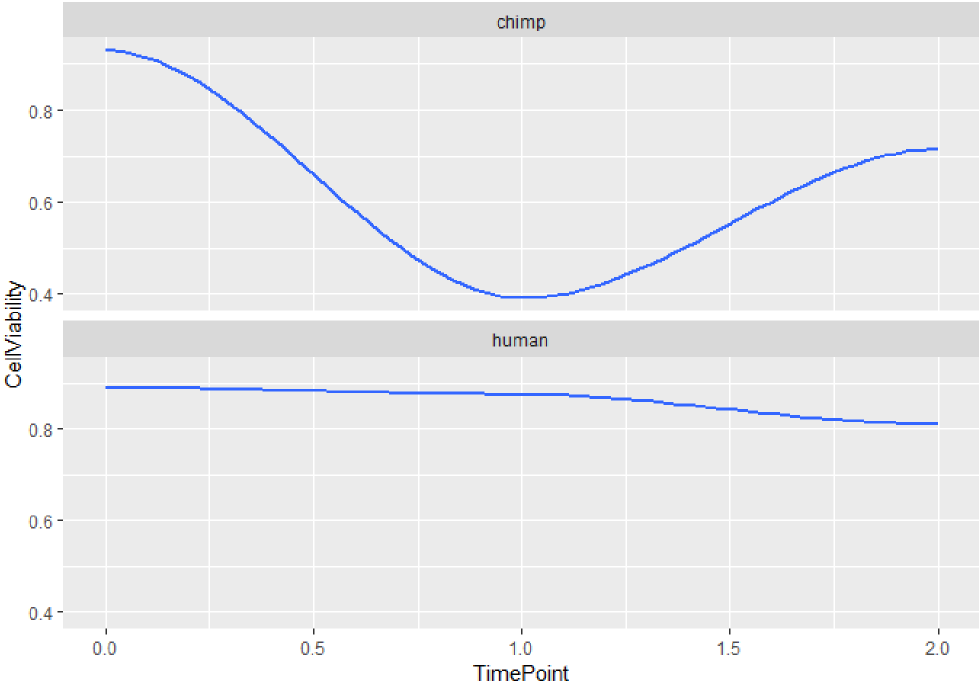
ggplot2: Building a Plot

supplying data
mapping aesthetics
defining geometric

```
ggplot(data = scores) +
  <GEOM_FUNCTION> (
    mapping = aes(x = TimePoint, y = CellViability)
  ) +
  facet_wrap(<FORMULA>)
```



```
geom_point()
facet_wrap(~ Species, nrow = 2)
```



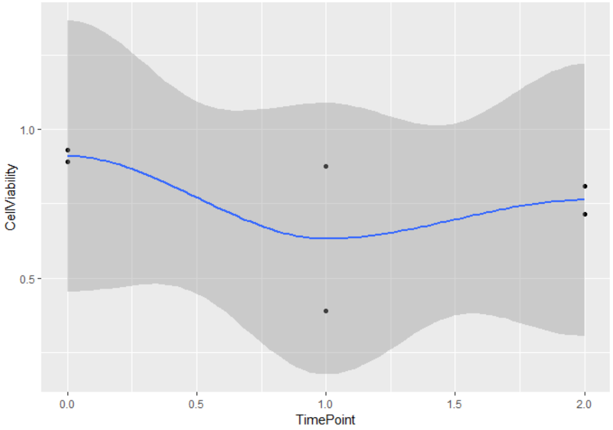
```
geom_smooth()
facet_wrap(~ Species, nrow = 2)
```

ggplot2: Building a Plot - multiple

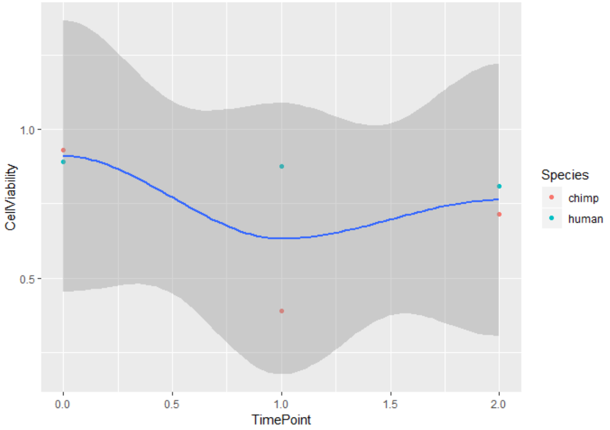
supplying data
mapping aesthetics
defining geometric

```
ggplot(data = scores) +
  <GEOM_FUNCTION_1> (<MAPPINGS>) +
  <GEOM_FUNCTION_2> (<MAPPINGS>) +
```

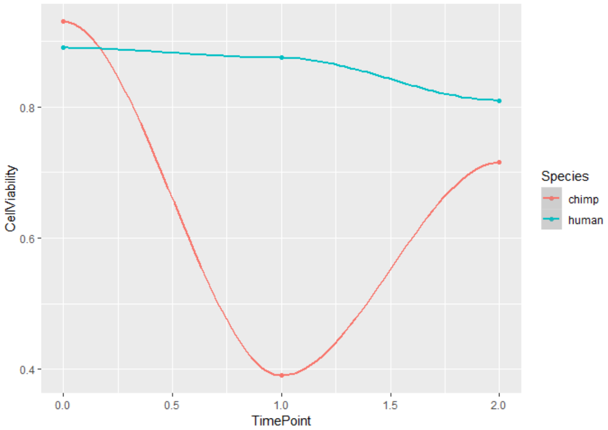
```
ggplot(data = scores, <MAPPINGS>) +
  <GEOM_FUNCTION_1> () +
  <GEOM_FUNCTION_2> () +
```



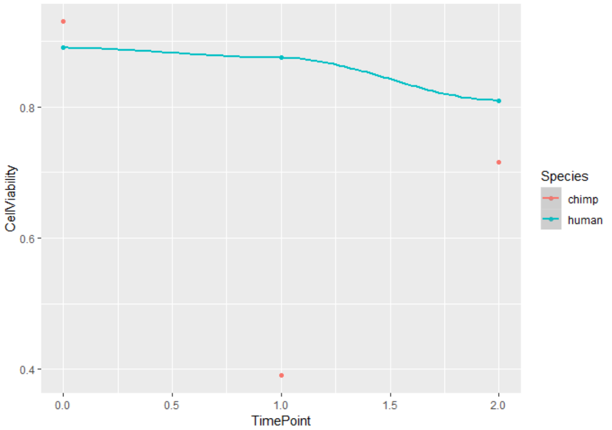
```
geom_point()
geom_smooth()
```



```
geom_point(color = Species)
geom_smooth()
```



```
geom_point(color = Species)
geom_smooth(color = Species)
```

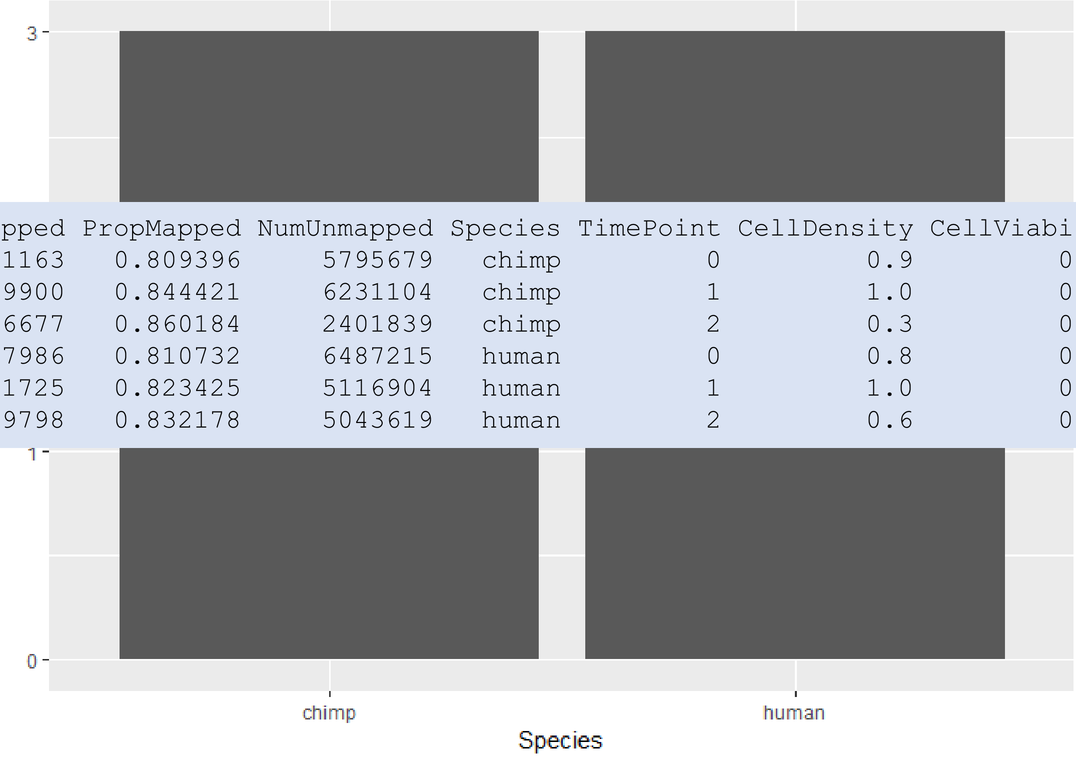


```
geom_point(color = Species)
geom_smooth(newdata, color = Species)
```

ggplot2: Building a Plot - bar

supplying data
mapping aesthetics
defining geometric
describing statistic

```
ggplot(data = prop.reads) +
  stat_count(
    mapping = aes(x = Species),
)
```

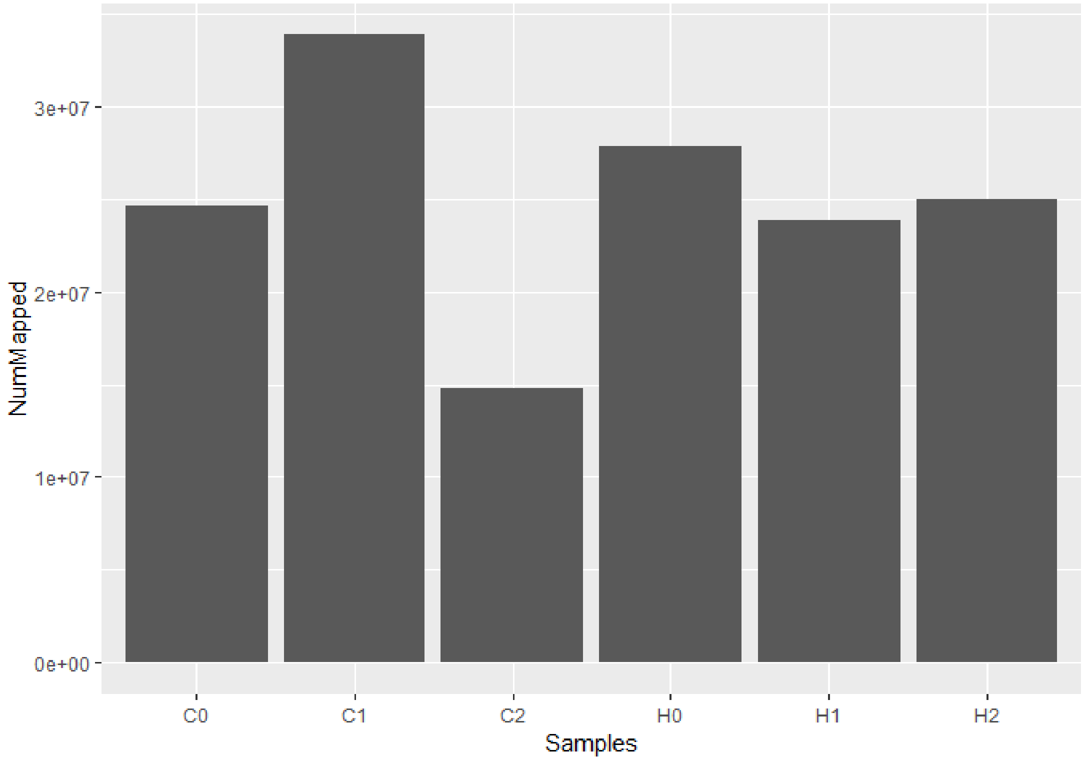


```
Samples NumTotal NumMapped PropMapped NumUnmapped Species TimePoint CellDensity CellViability RNAConcentration
     C0 30406842  24611163   0.809396     5795679   chimp         0         0.9         0.930              219
     C1 40051004  33819900   0.844421     6231104   chimp         1         1.0         0.390              160
     C2 17178516  14776677   0.860184     2401839   chimp         2         0.3         0.715               90
     H0 34275201  27787986   0.810732     6487215   human         0         0.8         0.890              377
     H1 28978629  23861725   0.823425     5116904   human         1         1.0         0.875              257
     H2 30053417  25009798   0.832178     5043619   human         2         0.6         0.810              109
```

ggplot2: Building a Plot - bar

supplying data
mapping aesthetics
defining geometric
describing statistic

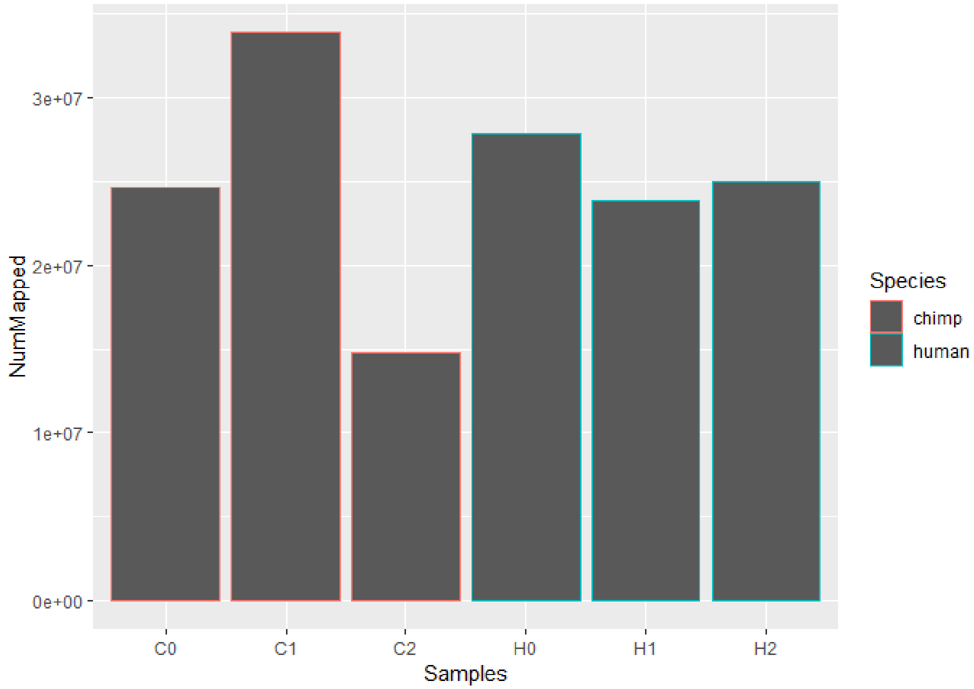
```
ggplot(data = prop.reads) +
  geom_bar(
    mapping = aes(x = Species, y = NumMapped),
    stat = “identity”
  )
```



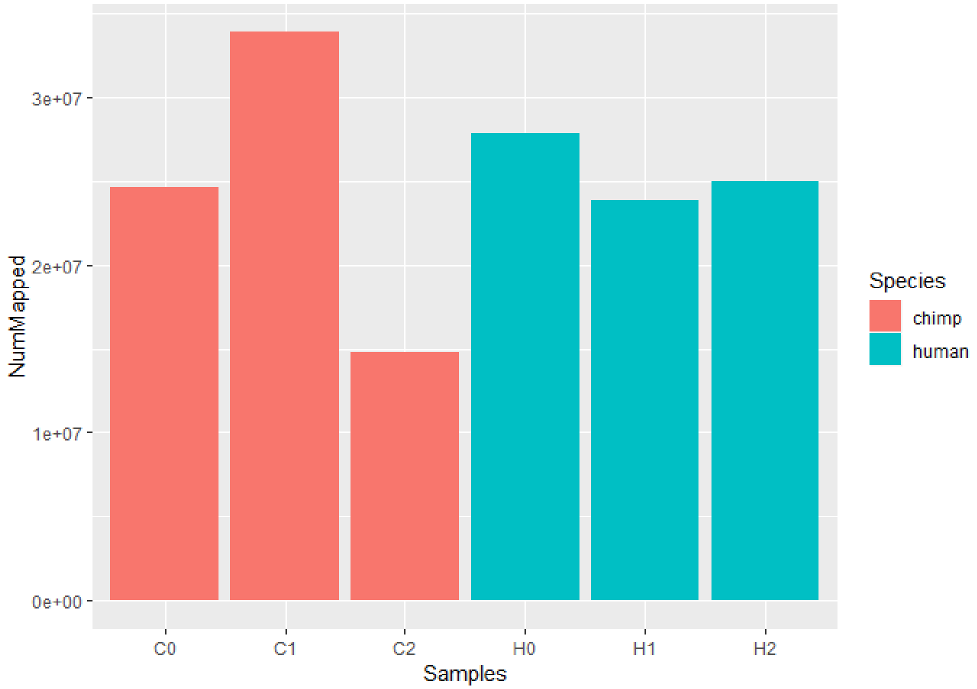
ggplot2: Building a Plot - bar

supplying data
mapping aesthetics
defining geometric
describing statistic

```
ggplot(data = prop.reads) +
  geom_bar(
    mapping = aes(x = Species, y = NumMapped, <MAPPINGS>),
    stat = “identity”
  )
```



`color = Species`

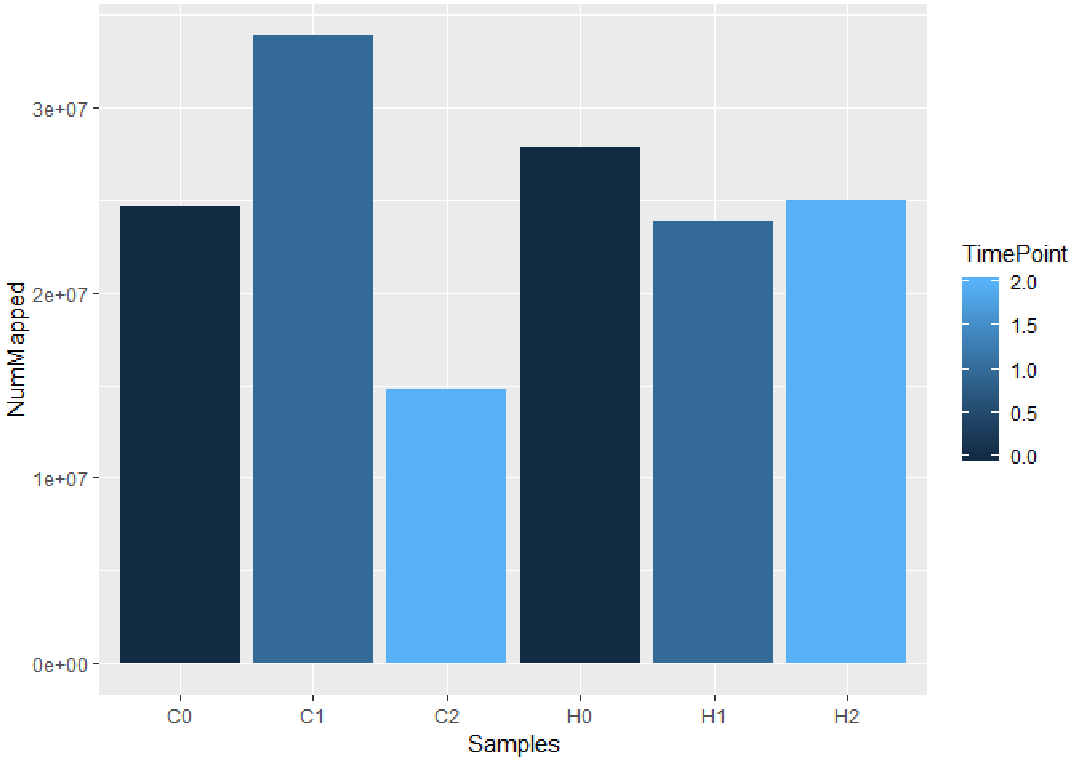


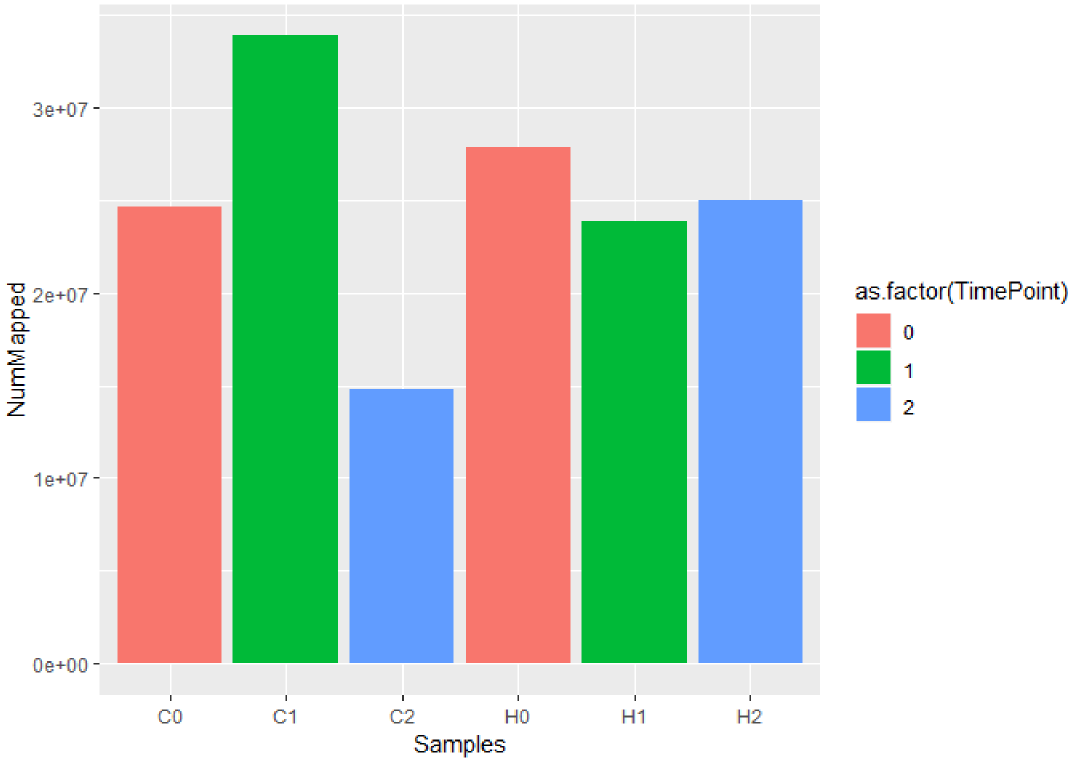
`fill = Species`

ggplot2: Building a Plot - bar

EXERCISES

Try changing the fill to different variables. Instead of Species, try TimePoint.





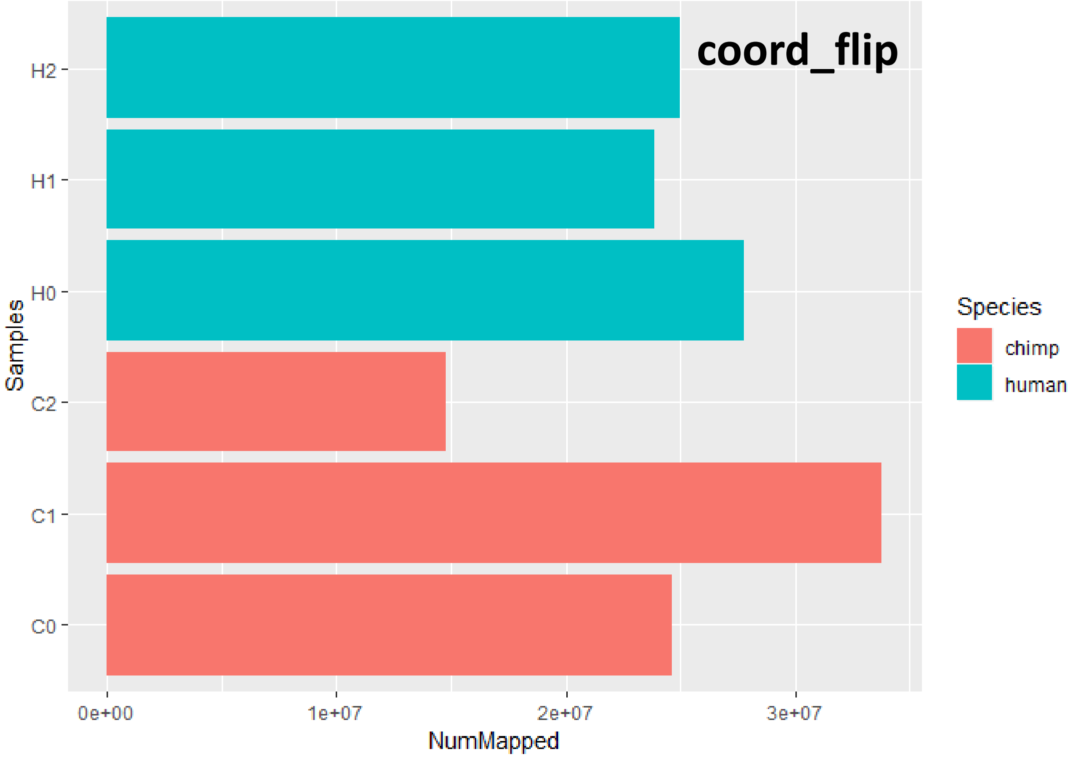
```
ggplot(data = prop.reads) +
  geom_bar(mapping = aes(x = Samples, y = NumMapped, fill = TimePoint), stat = "identity")
```

```
ggplot(data = prop.reads) +
  geom_bar(mapping = aes(x = Samples, y = NumMapped, fill = as.factor(TimePoint)), stat = "identity")
```

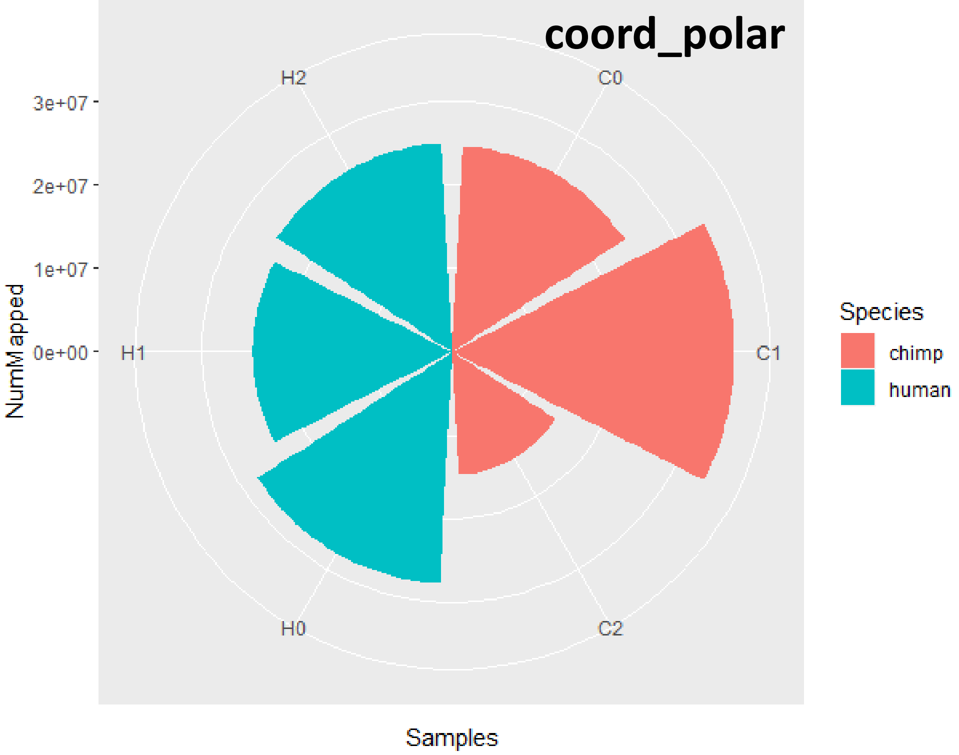
ggplot2: Building a Plot - bar

supplying data
mapping aesthetics
defining geometric
describing statistic
adjusting coordinate

```
ggplot(data = prop.reads) +
  geom_bar(
    mapping = aes(x = Species, y = NumMapped),
    stat = “identity”
  ) +
<COORDINATE_FUNCTION>
```



coord_flip

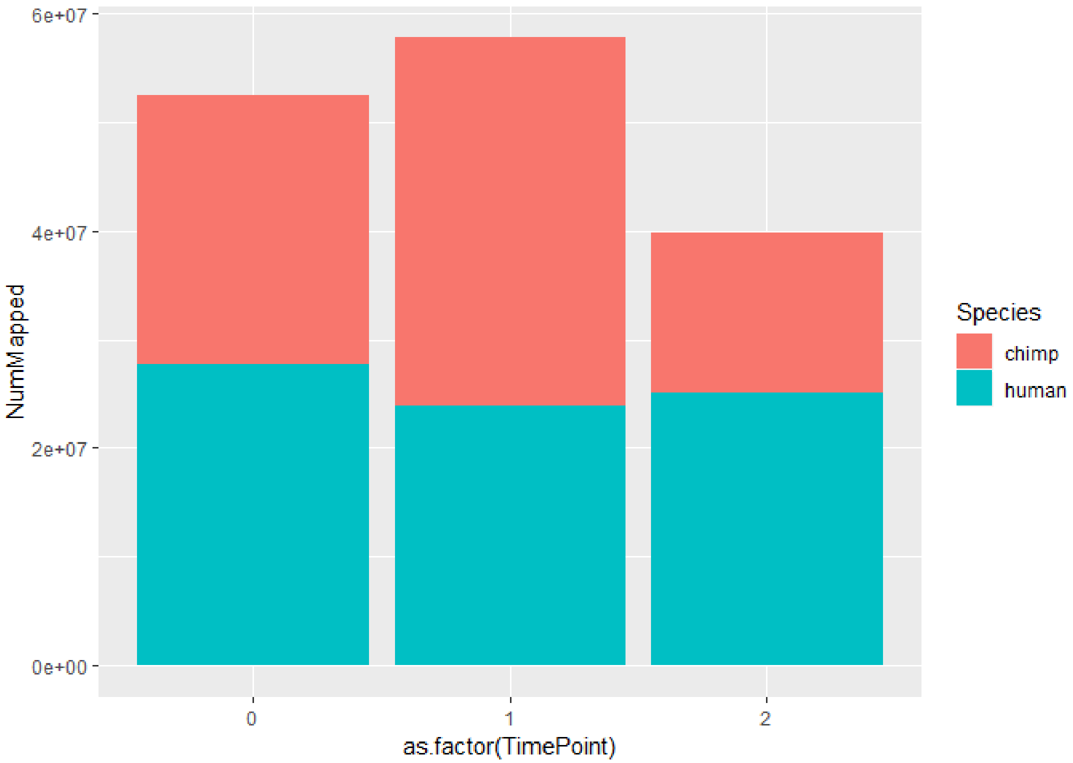


coord_polar

ggplot2: Building a Plot - bar

supplying data
mapping aesthetics
defining geometric
describing statistic

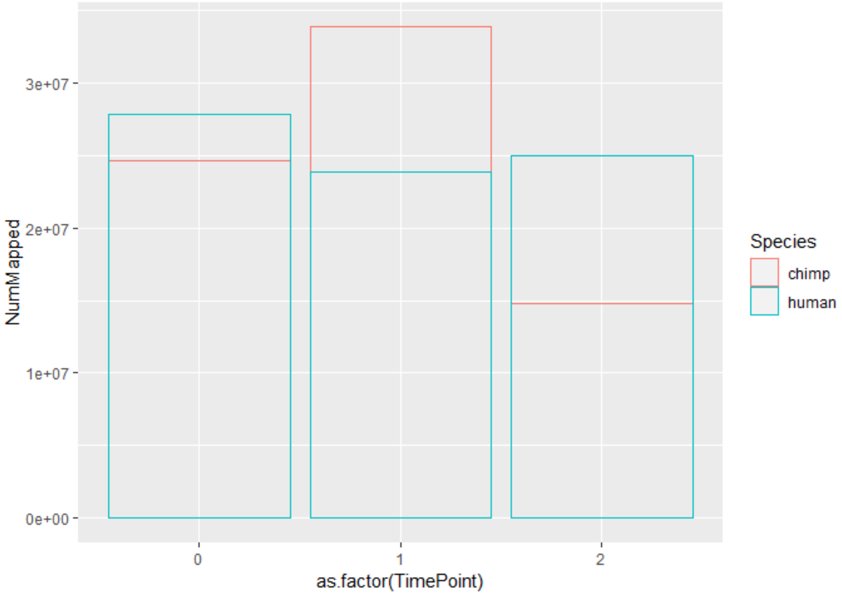
```
ggplot(data = prop.reads) +
  geom_bar(
    mapping = aes(x = as.factor(TimePoint), y = NumMapped,
                  fill = Species),
    stat = “identity”
  )
```



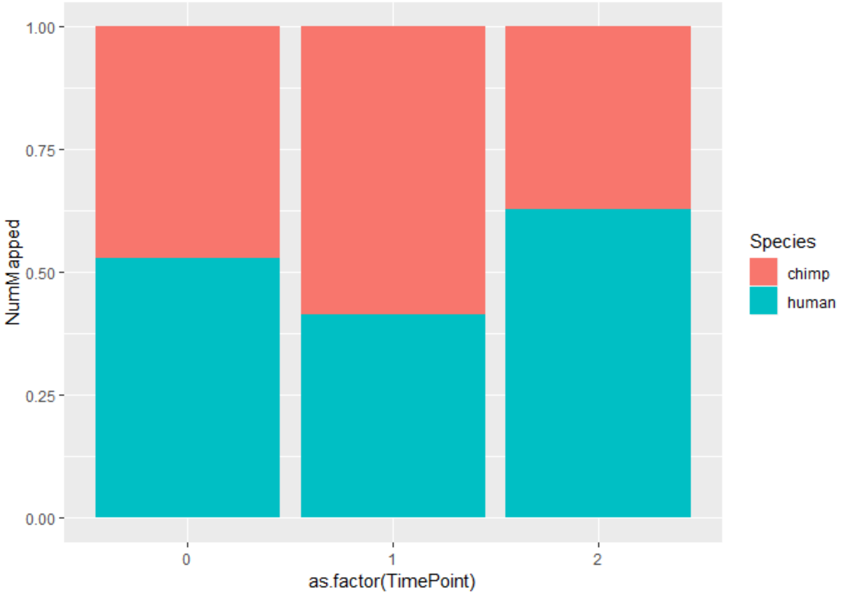
ggplot2: Building a Plot - bar

supplying data
mapping aesthetics
defining geometric
describing statistic
changing position

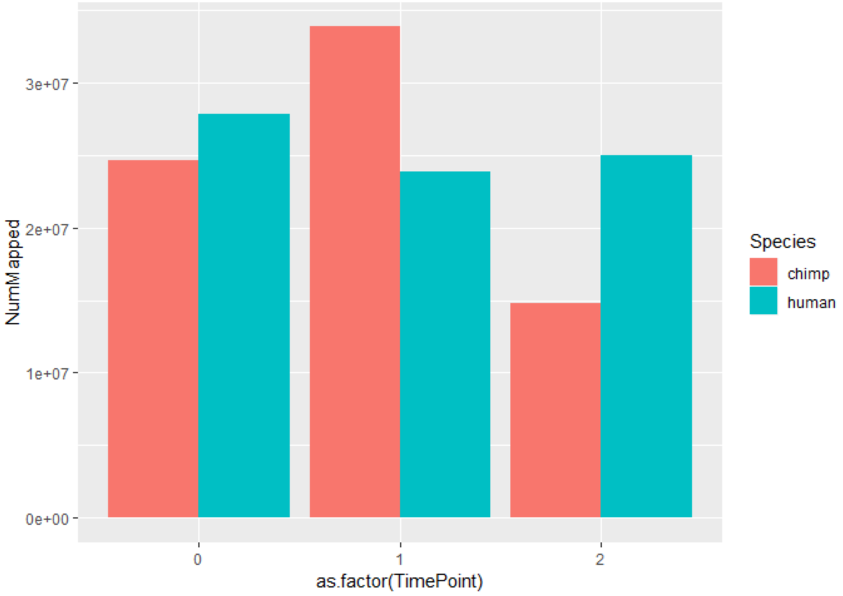
```
ggplot(data = prop.reads) +
  geom_bar(
    mapping = aes(x = as.factor(TimePoint), y = NumMapped,
                  fill = Species),
    stat = “identity”,
    position = <POSITION>
  )
```



`position = “identity”`



`position = “fill”`

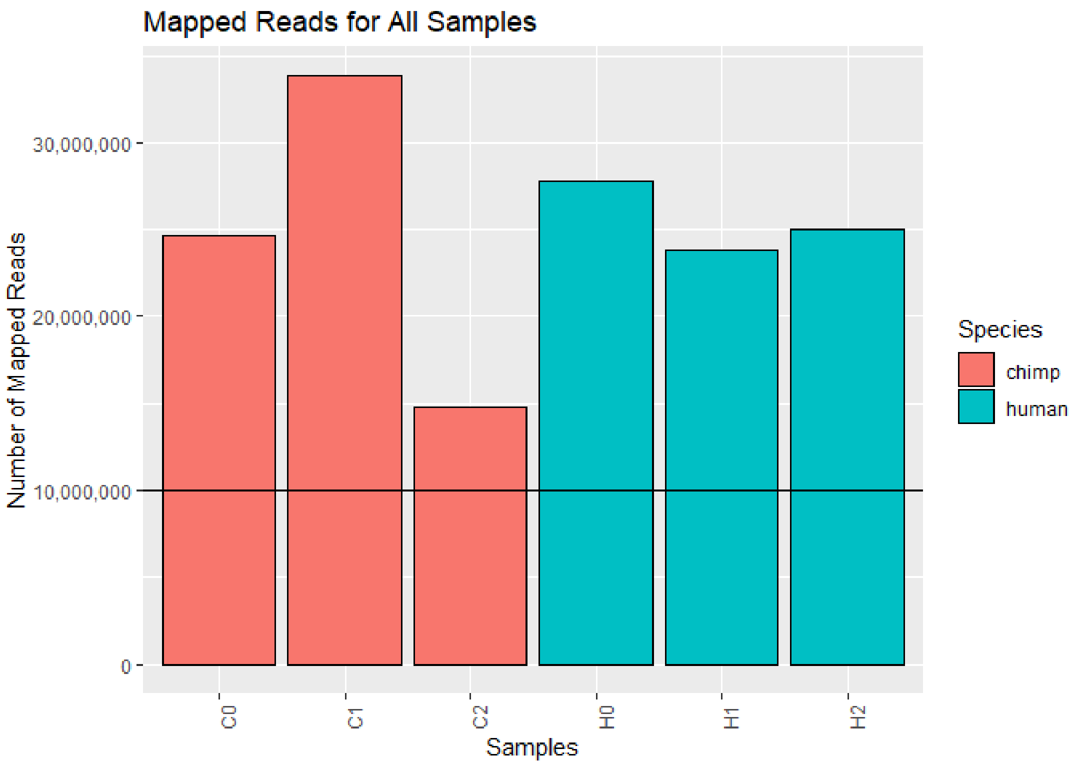


`position = “dodge”`

ggplot2: Building a Plot - bar

EXERCISES

```
ggplot(data = prop.reads) +
  geom_bar(mapping = aes(x = Samples, y = NumMapped, fill = Species), stat = "identity", color= "black") +
  ggtitle("Mapped Reads for All Samples") +
  ylab("Number of Mapped Reads") +
  xlab("Samples") +
  geom_hline(yintercept=10000000) +
  theme(axis.text.x=element_text(angle=90, hjust=1)) +
  scale_y_continuous(labels=comma)
```

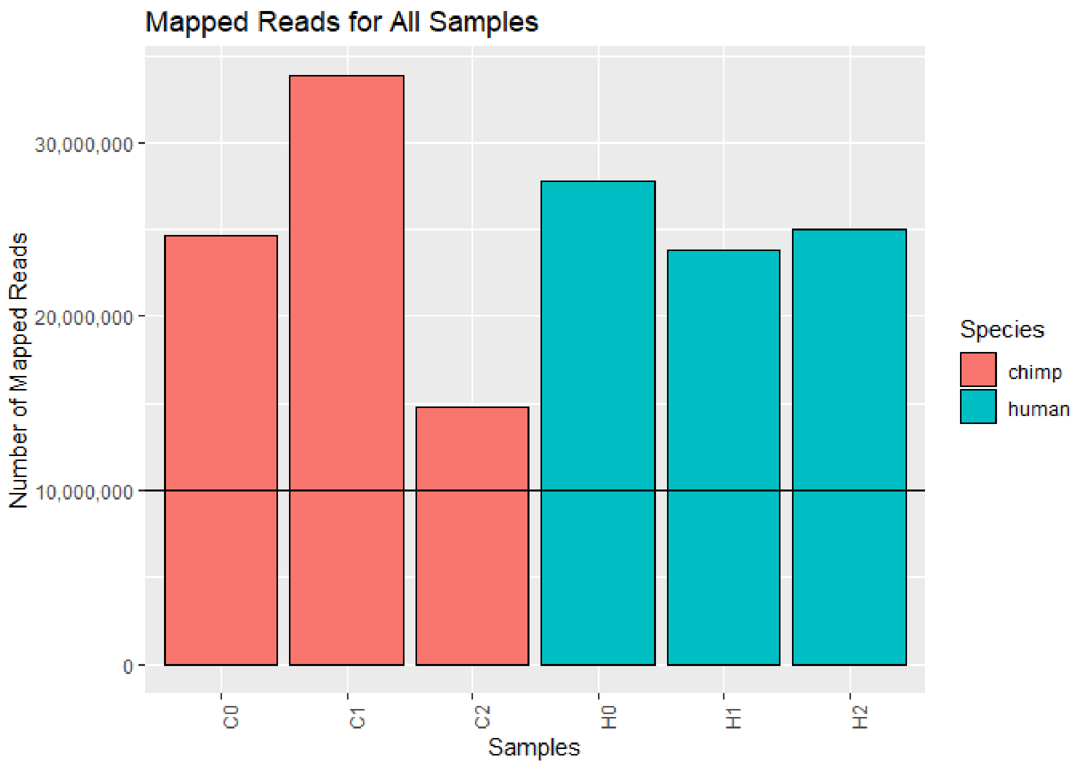


ggplot2: Building a Plot - bar

EXERCISES

```
ggplot(data = prop.reads) +
  geom_bar(mapping = aes(x = Samples, y = NumMapped, fill = Species), stat = "identity", color= "black") +
  ggtitle("Mapped Reads for All Samples") +
  ylab("Number of Mapped Reads") +
  xlab("Samples") +
  geom_hline(yintercept=10000000) +
  theme(axis.text.x=element_text(angle=90, hjust=1)) +
  scale_y_continuous(labels=comma)
```

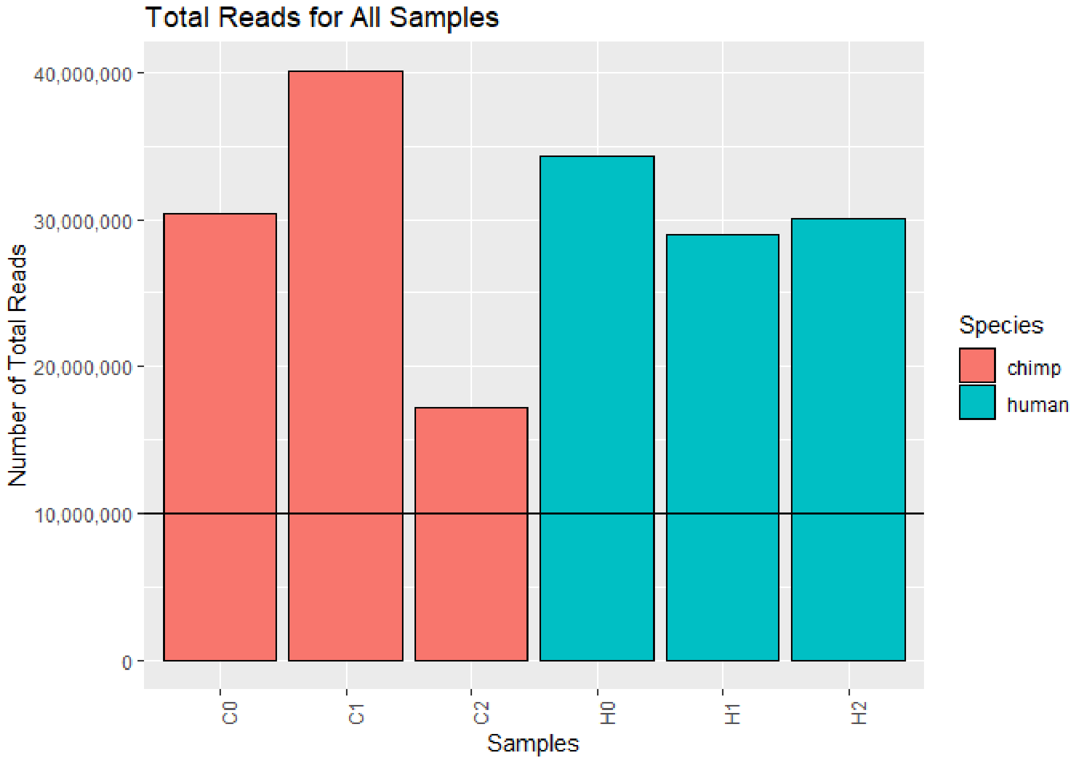
Try other formatting changes to this figure.

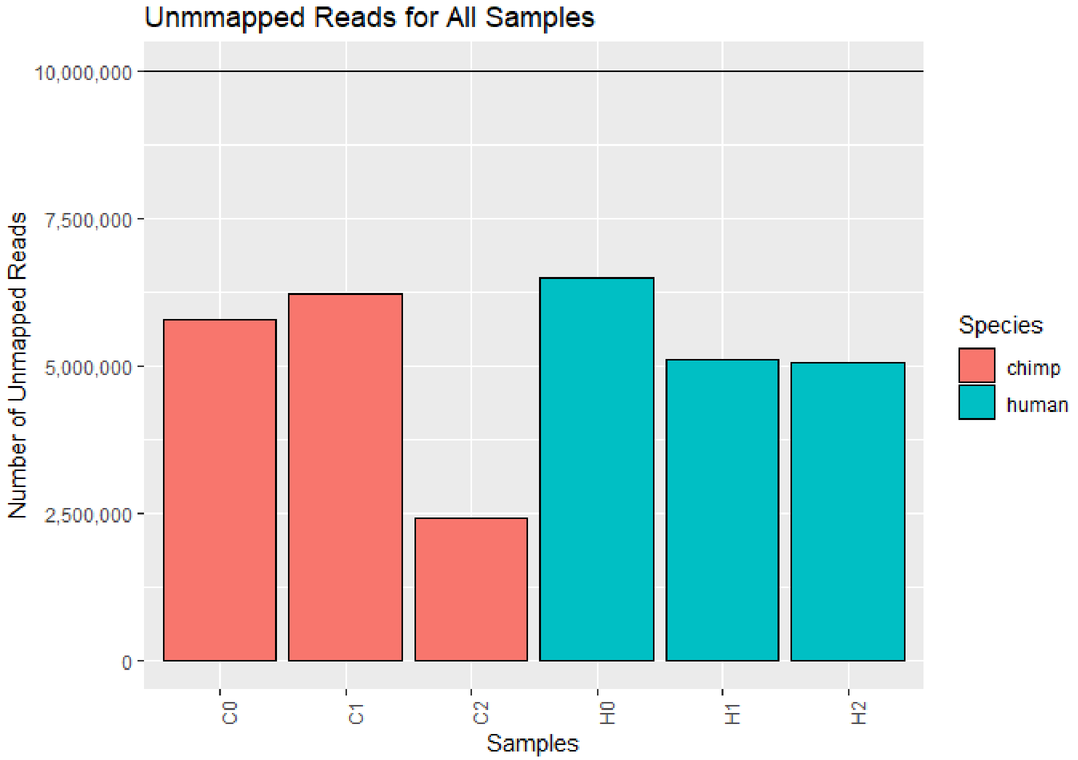


ggplot2: Building a Plot - bar

EXERCISES

Try plotting the number of total reads and unmapped reads per sample.





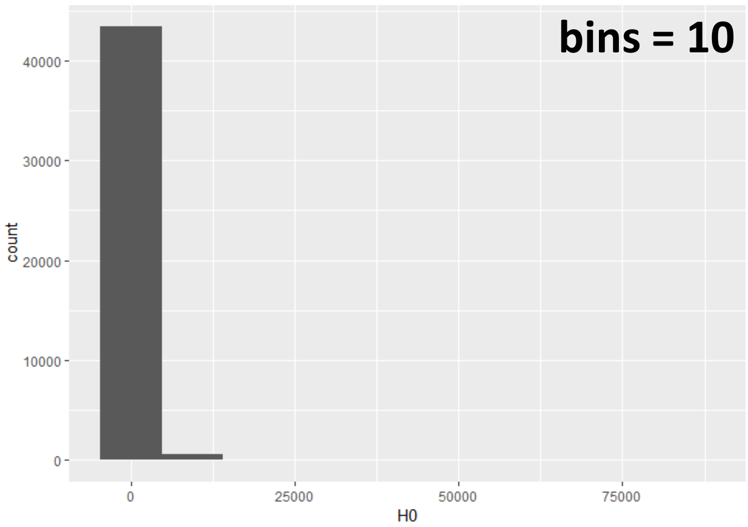
```
ggplot(data = prop.reads) +
  geom_bar(mapping = aes(x = Samples, y = NumTotal, fill = Species), stat = "identity", color= "black") +
  ggtitle("Total Reads for All Samples") +
  ylab("Number of Total Reads") +
  xlab("Samples") +
  geom_hline(yintercept=10000000) +
  theme(axis.text.x=element_text(angle=90, hjust=1)) +
  scale_y_continuous(labels=comma)
```

```
ggplot(data = prop.reads) +
  geom_bar(mapping = aes(x = Samples, y = NumUnmapped, fill = Species), stat = "identity", color= "black") +
  ggtitle("Unmmapped Reads for All Samples") +
  ylab("Number of Unmapped Reads") +
  xlab("Samples") +
  geom_hline(yintercept=10000000) +
  theme(axis.text.x=element_text(angle=90, hjust=1)) +
  scale_y_continuous(labels=comma)
```

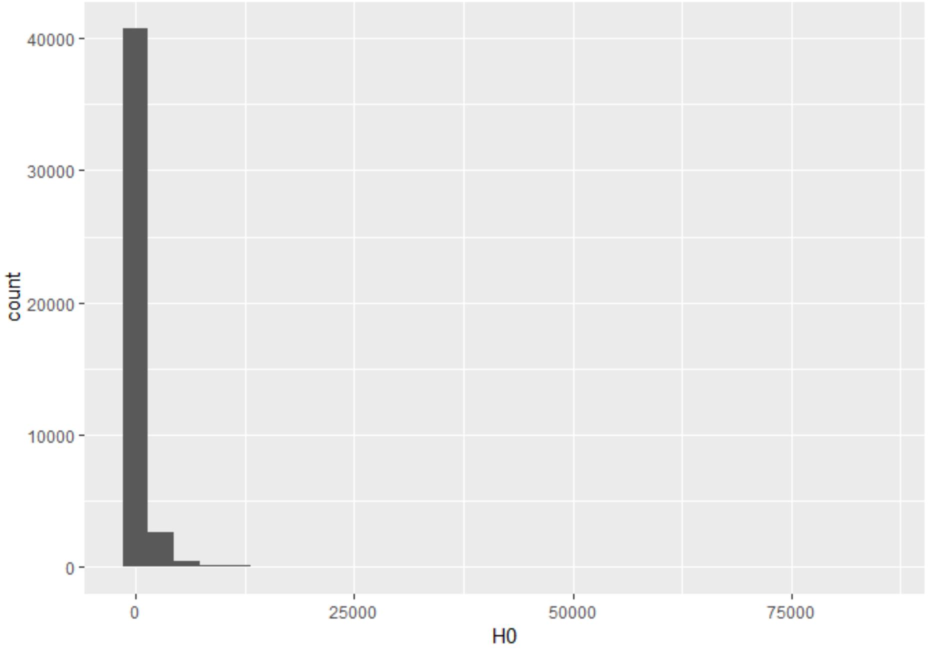
ggplot2: Building a Plot - histogram

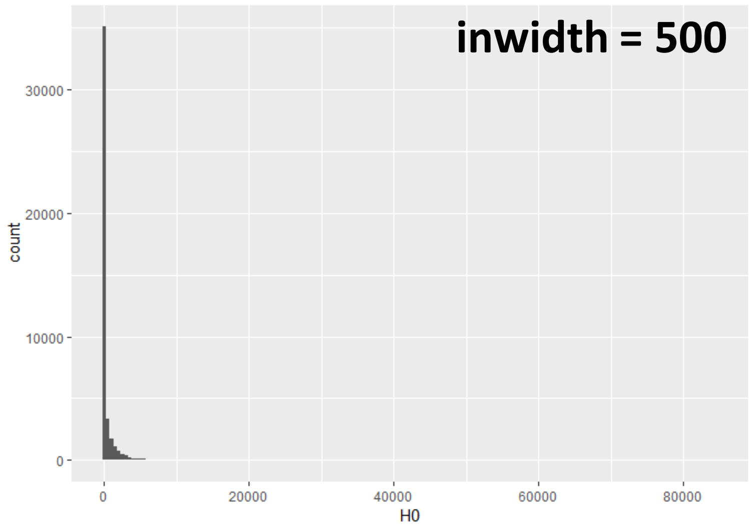
supplying data
mapping aesthetics
defining geometric

```
ggplot(data = gene.counts, mapping = aes(x = H0) +
  geom_histogram()
```



bins = 10



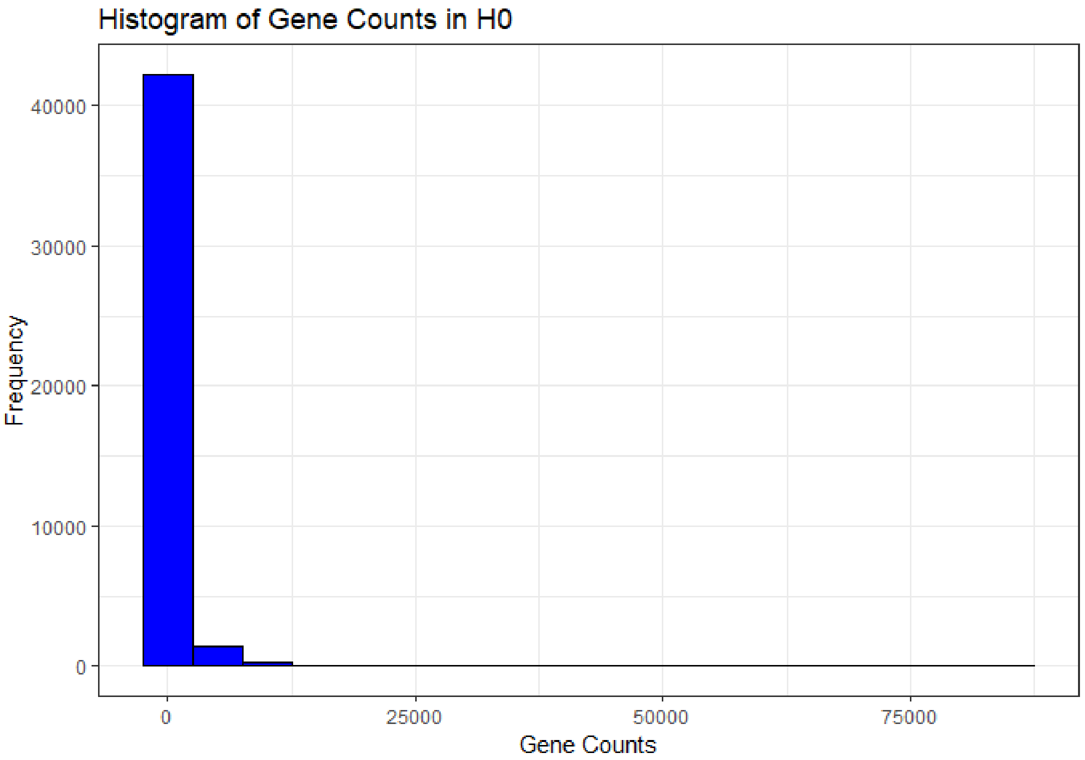


inwidth = 500

ggplot2: Building a Plot - histogram

EXERCISE

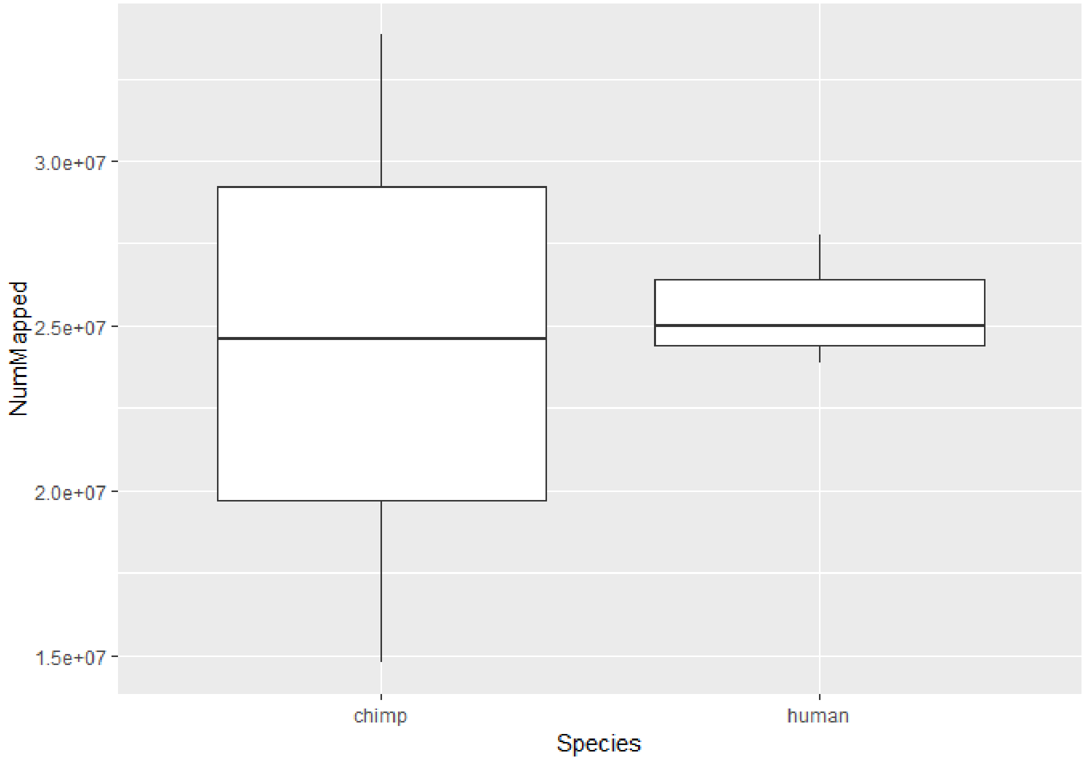
```
ggplot(data = gene.counts, mapping = aes(x = H0)) +
       geom_histogram(binwidth = 5000, fill = "blue", color = "black") +
       labs(title = "Histogram of Gene Counts in H0", y = "Frequency", x = "Gene Counts") +
       theme_bw()
```



ggplot2: Building a Plot - boxplot

supplying data
mapping aesthetics
defining geometric

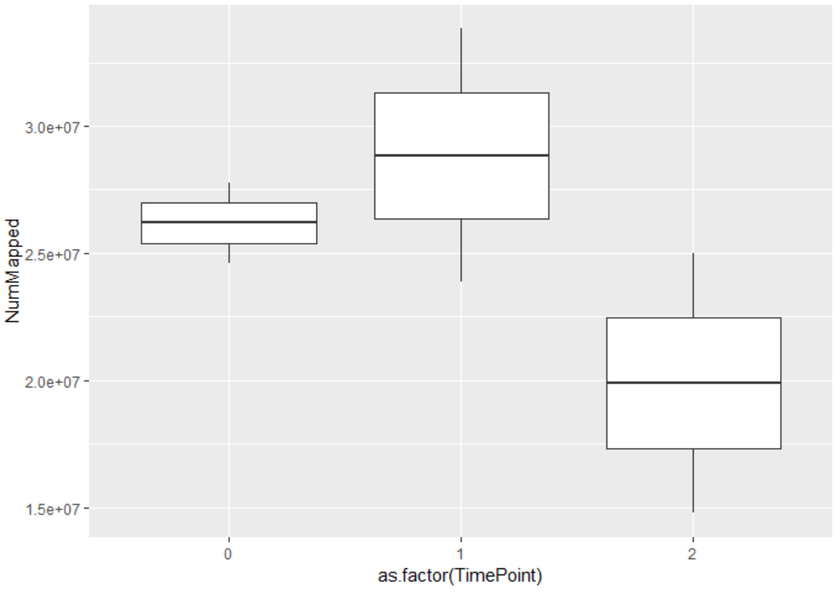
```
ggplot(data = prop.reads) +
  geom_boxplot(
    mapping = aes(x = Species, y = NumMapped)
)
```

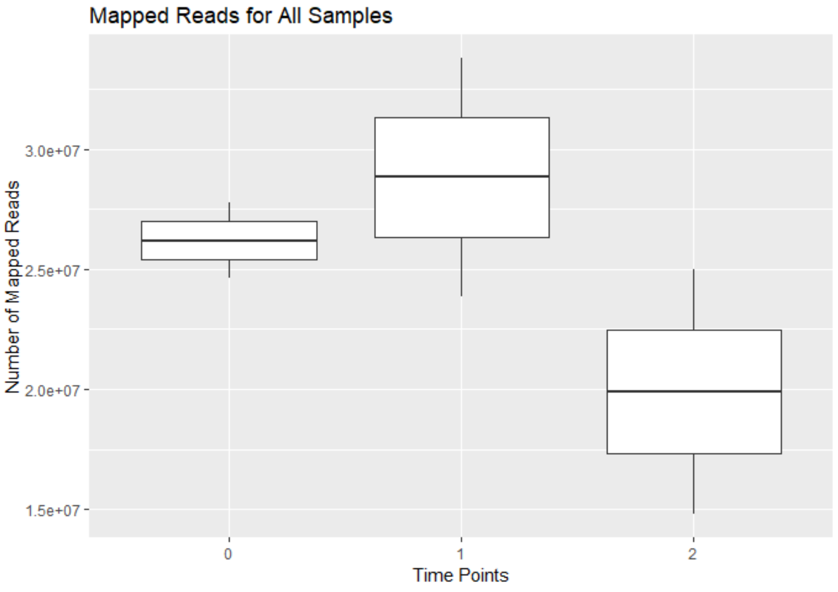


ggplot2: Building a Plot - boxplot

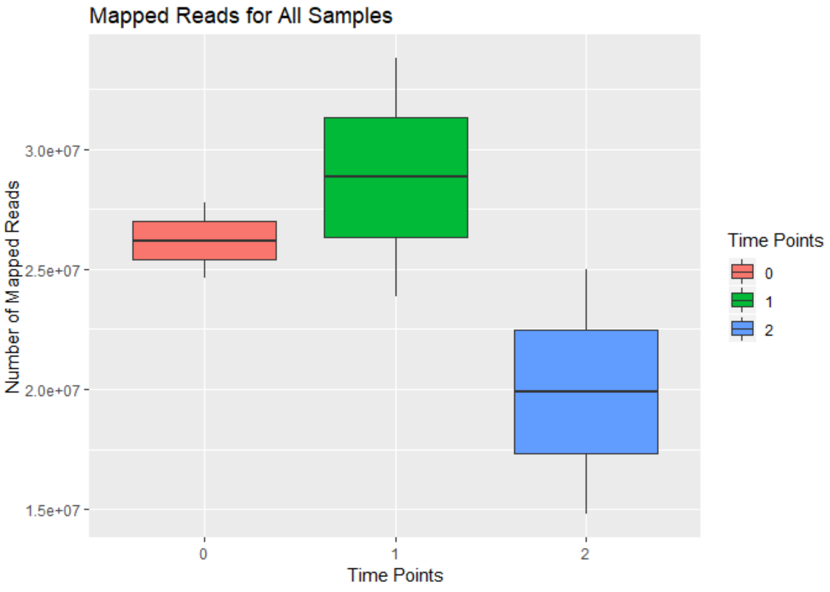
supplying data
mapping aesthetics
defining geometric

```
ggplot(data = prop.reads) +
  geom_boxplot(
    mapping = aes(x = as.factor(TimePoint), y = NumMapped)
)
```





```
+ labs()
```

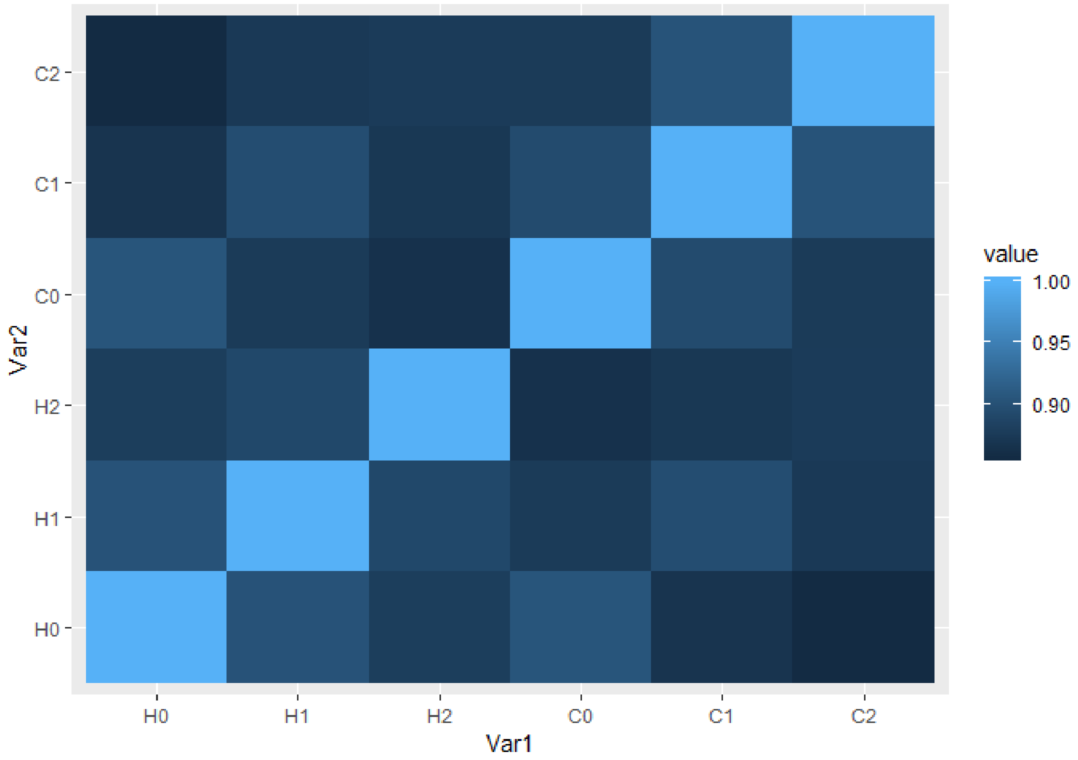


```
+ labs() + guides()
```

ggplot2: Building a Plot - heatmap

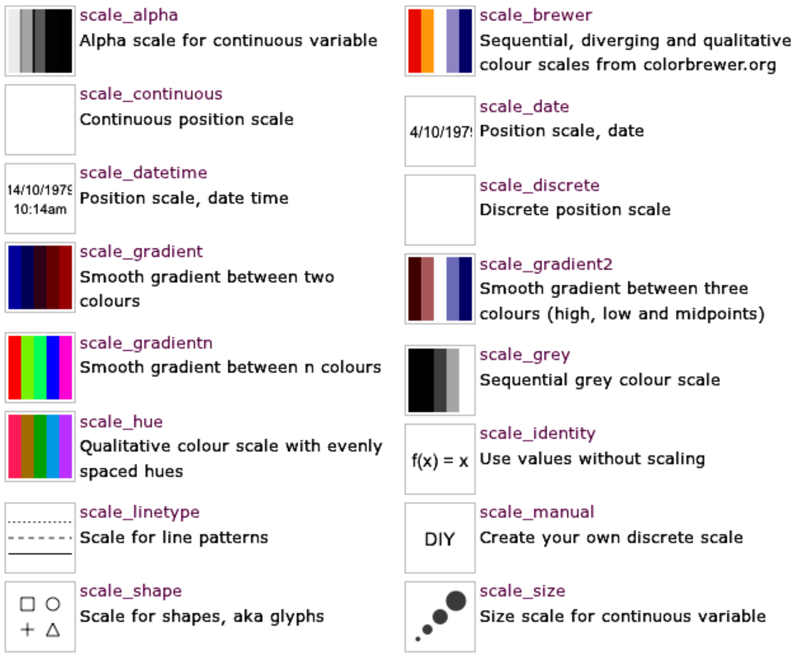
supplying data
mapping aesthetics
defining geometric

```
ggplot(data = cors.melt) +
  geom_tile(mapping = aes(x=Var1, y=Var2, fill=value))
```



ggplot2: Building a Plot - heatmap

```
ggplot(data = cors.melt) +
  geom_tile(mapping = aes(x=Var1, y=Var2, fill=value), color =
  scale_fill_gradient(low = "white", high = "steelblue")
```



scale_alpha
Alpha scale for continuous variable

scale_continuous
Continuous position scale

scale_datetime
Position scale, date time

scale_gradient
Smooth gradient between two colours

scale_gradientn
Smooth gradient between n colours

scale_hue
Qualitative colour scale with evenly spaced hues

scale_linetype
Scale for line patterns

scale_shape
Scale for shapes, aka glyphs

scale_brewer
Sequential, diverging and qualitative colour scales from colorbrewer.org

scale_date
Position scale, date

scale_discrete
Discrete position scale

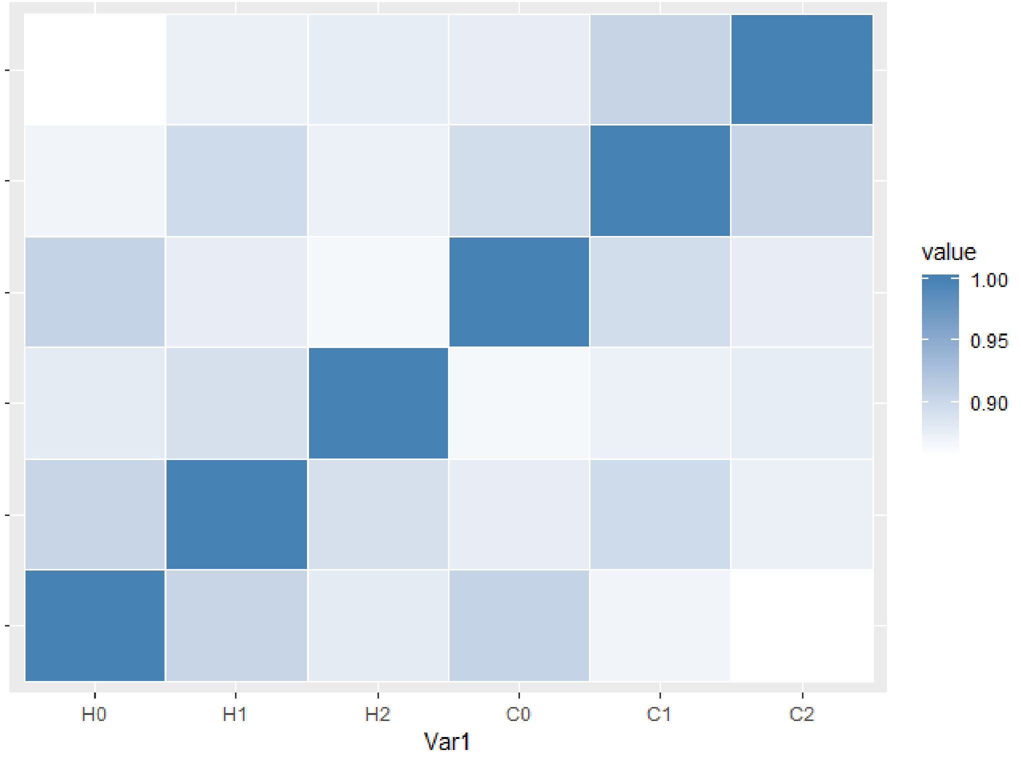
scale_gradient2
Smooth gradient between three colours (high, low and midpoints)

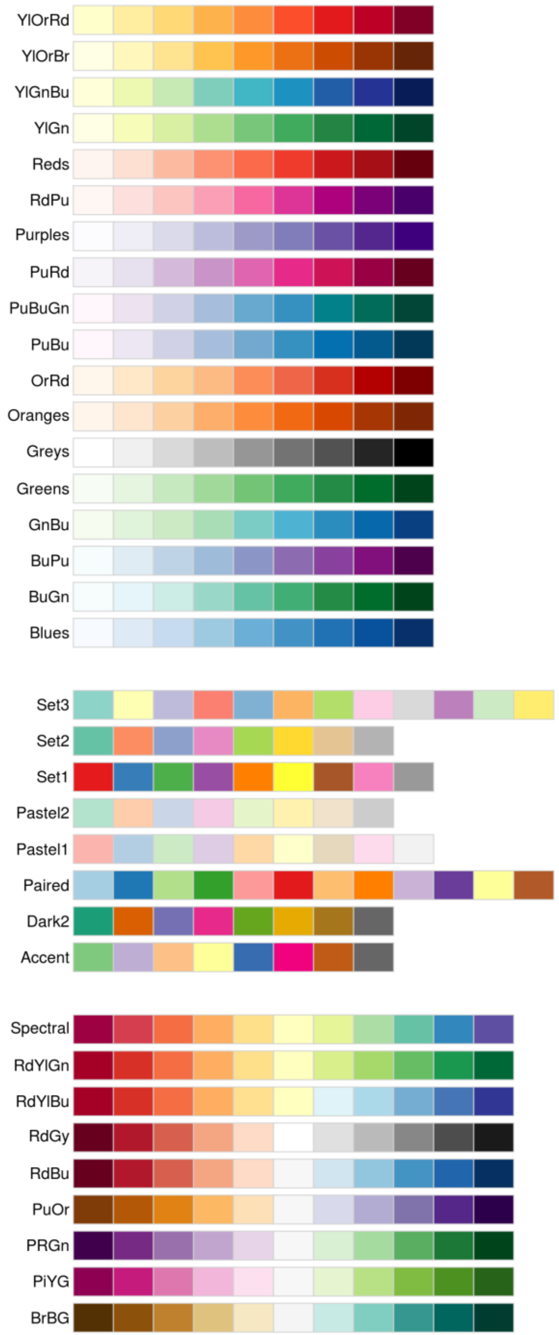
scale_grey
Sequential grey colour scale

scale_identity
Use values without scaling

scale_manual
Create your own discrete scale

scale_size
Size scale for continuous variable

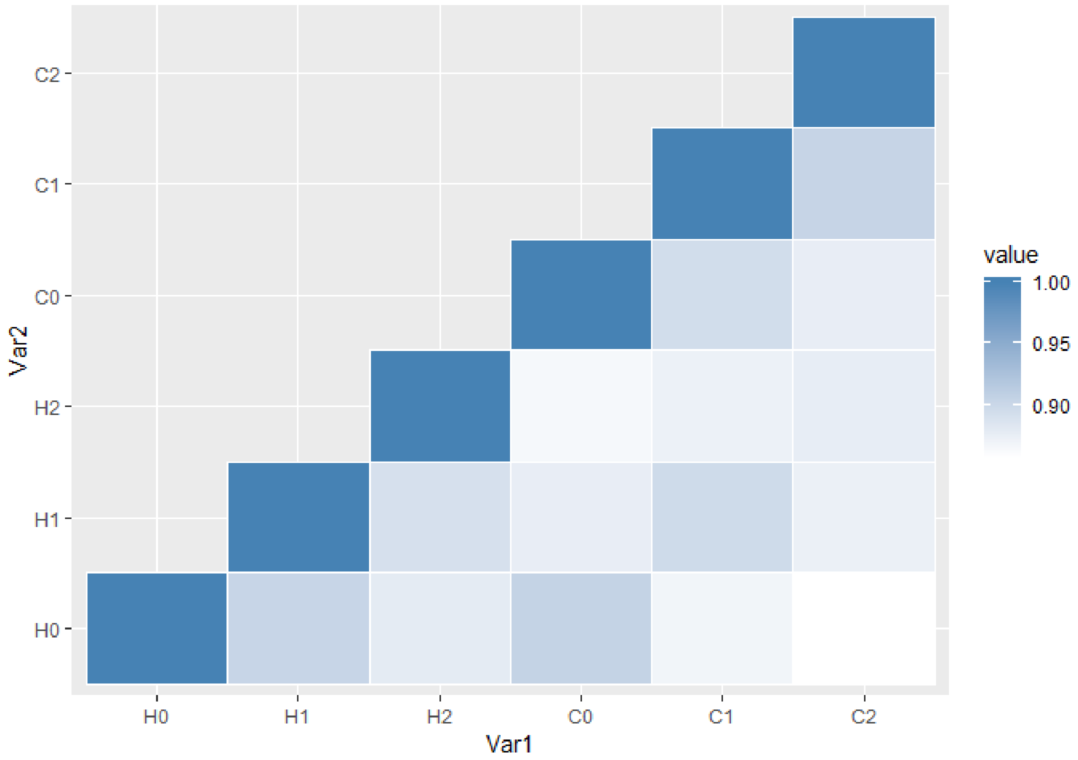




ggplot2: Building a Plot - heatmap

supplying data
mapping aesthetics
defining geometric

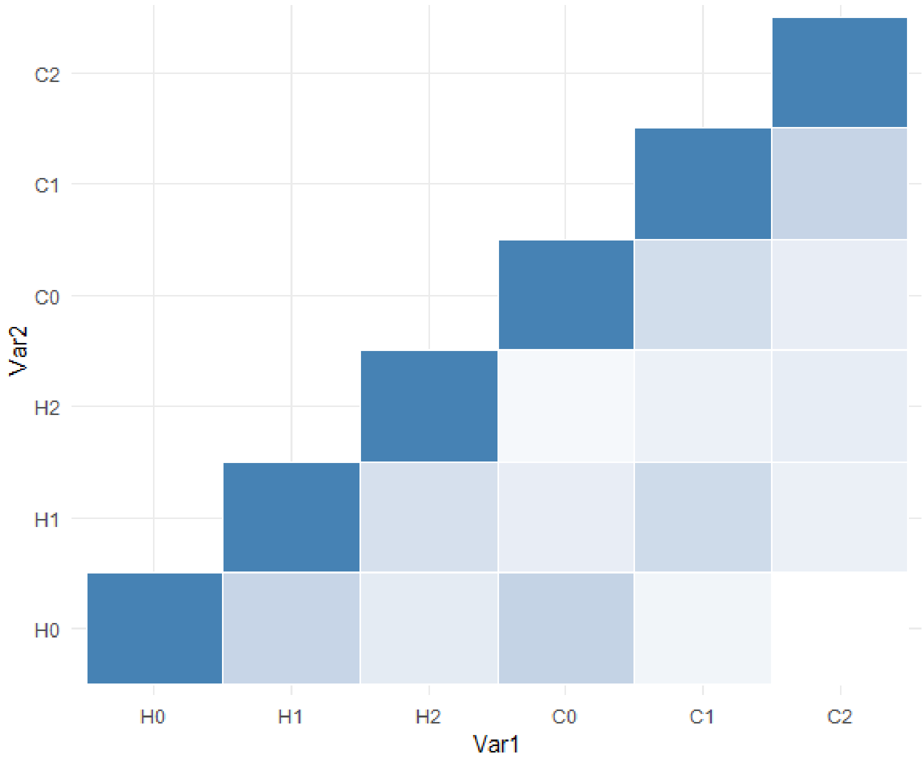
```
ggplot(data = half.cors.melt) +
  geom_tile(mapping = aes(x=Var1, y=Var2, fill=value), colour = "white") +
  scale_fill_gradient(low = "white", high = "steelblue")
```

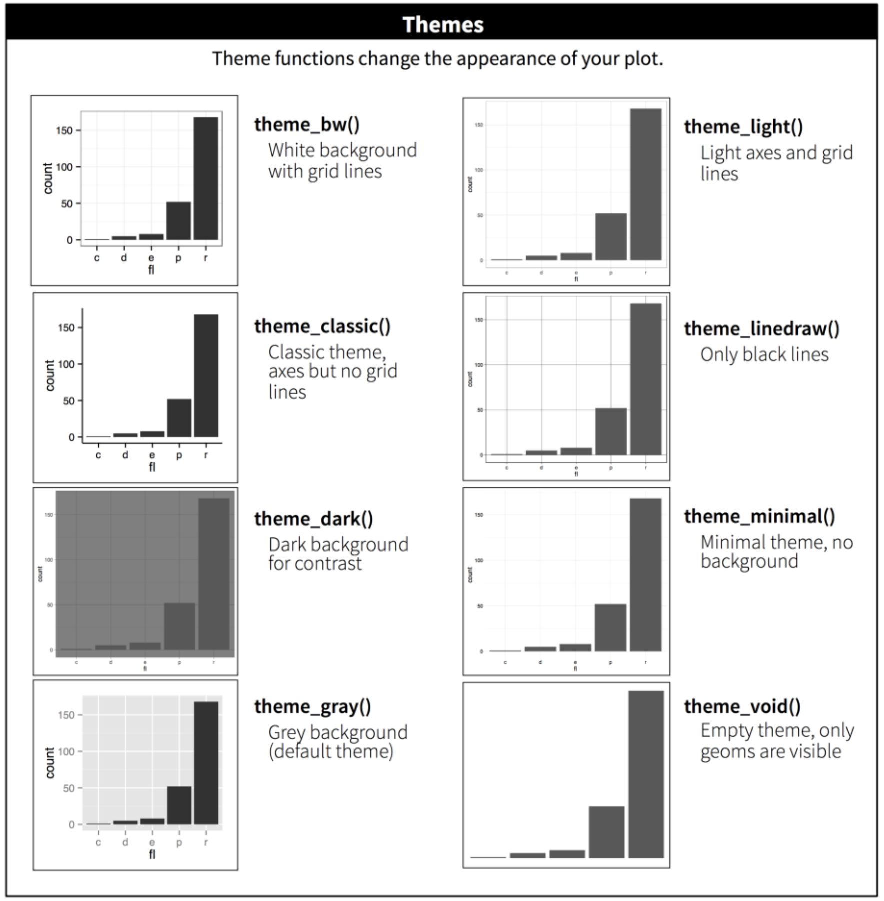


ggplot2: Building a Plot - heatmap

supplying data
mapping aesthetics
defining geometric

```
ggplot(data = half.cors.melt) +
  geom_tile(mapping = aes(x=Var1, y=Var2, fill=value), colour = "white") +
  scale_fill_gradient(low = "white", high = "steelblue", name =
                      "Spearman\nCorrelation", guide = guide_legend()) +
  guides(fill = guide_legend(title.vjust = 0.1)) +
  theme_minimal()
```





Themes

Theme functions change the appearance of your plot.

theme_bw()
White background with grid lines

theme_light()
Light axes and grid lines

theme_classic()
Classic theme, axes but no grid lines

theme_linedraw()
Only black lines

theme_dark()
Dark background for contrast

theme_minimal()
Minimal theme, no background

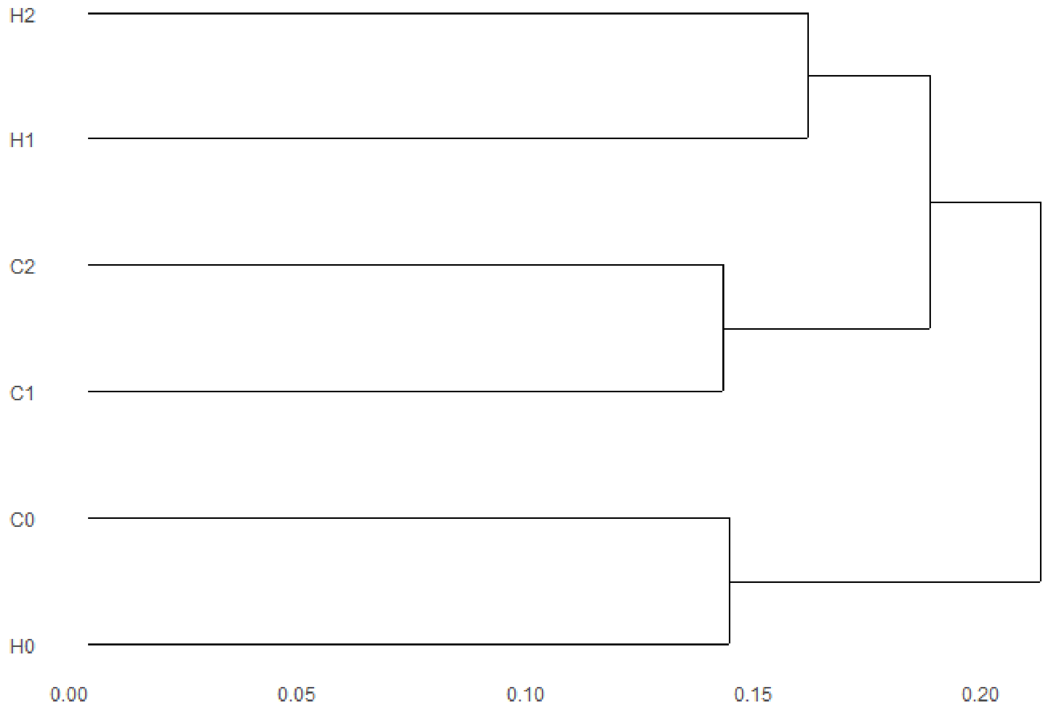
theme_gray()
Grey background (default theme)

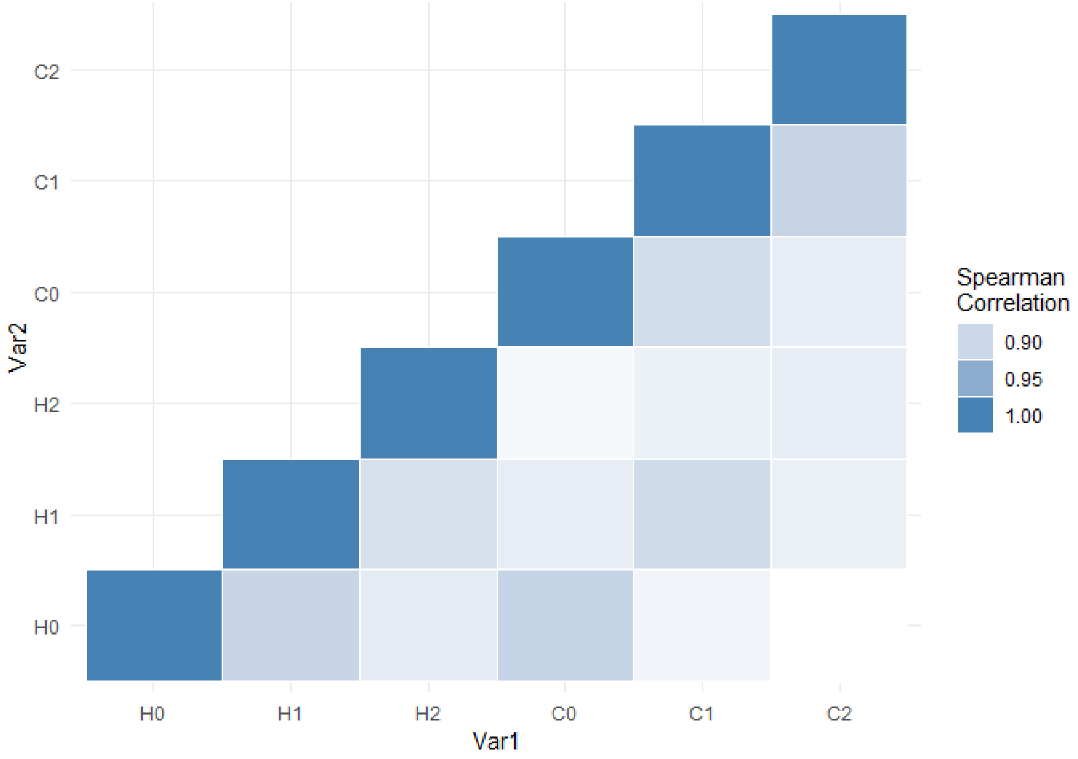
theme_void()
Empty theme, only geoms are visible

ggplot2: Building a Plot - heatmap

supplying data
mapping aesthetics
defining geometric

```
dendro.plot <- ggdendrogram(data = cors.dendro, rotate = TRUE)
heatmap.plot <- ggplot(data = half.cors.melt) +
  geom_tile(mapping = aes(x=Var1, y=Var2, fill=value), color = "white") +
  scale_fill_gradient(low = "white", high = "steelblue", name =
                       "Spearman\nCorrelation", guide = guide_legend()) +
  guides(fill = guide_legend(title.vjust = 0.1)) +
  theme_minimal()
```

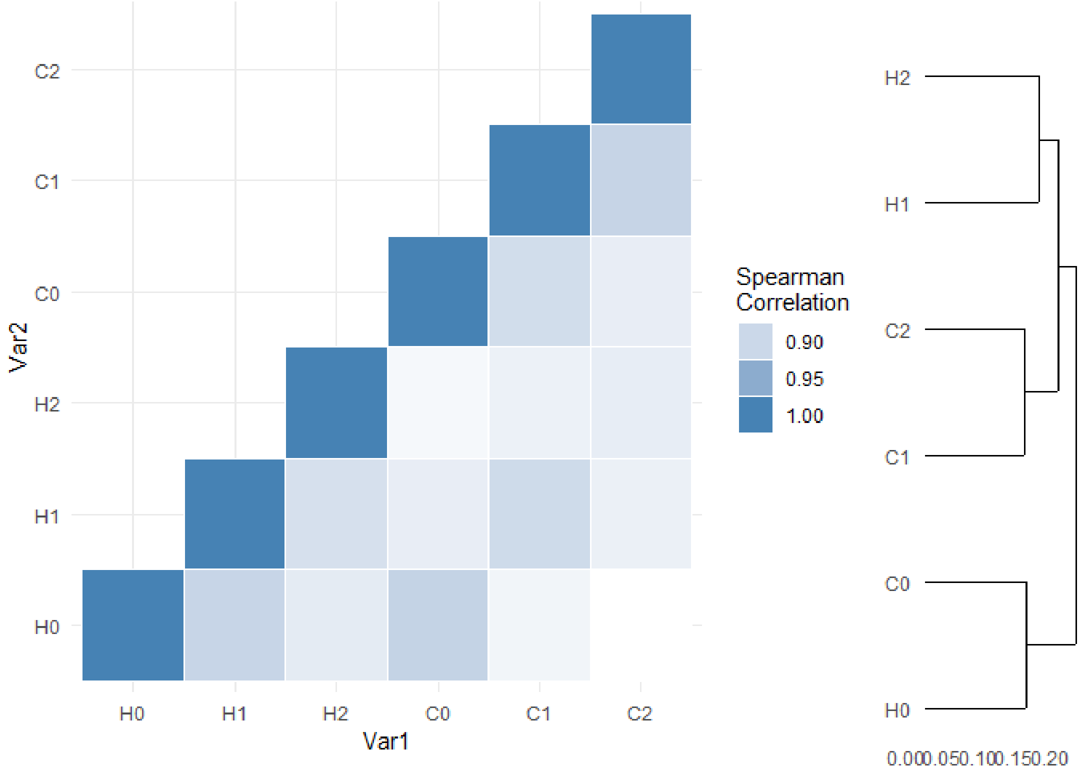




ggplot2: Building a Plot - heatmap

supplying data
mapping aesthetics
defining geometric

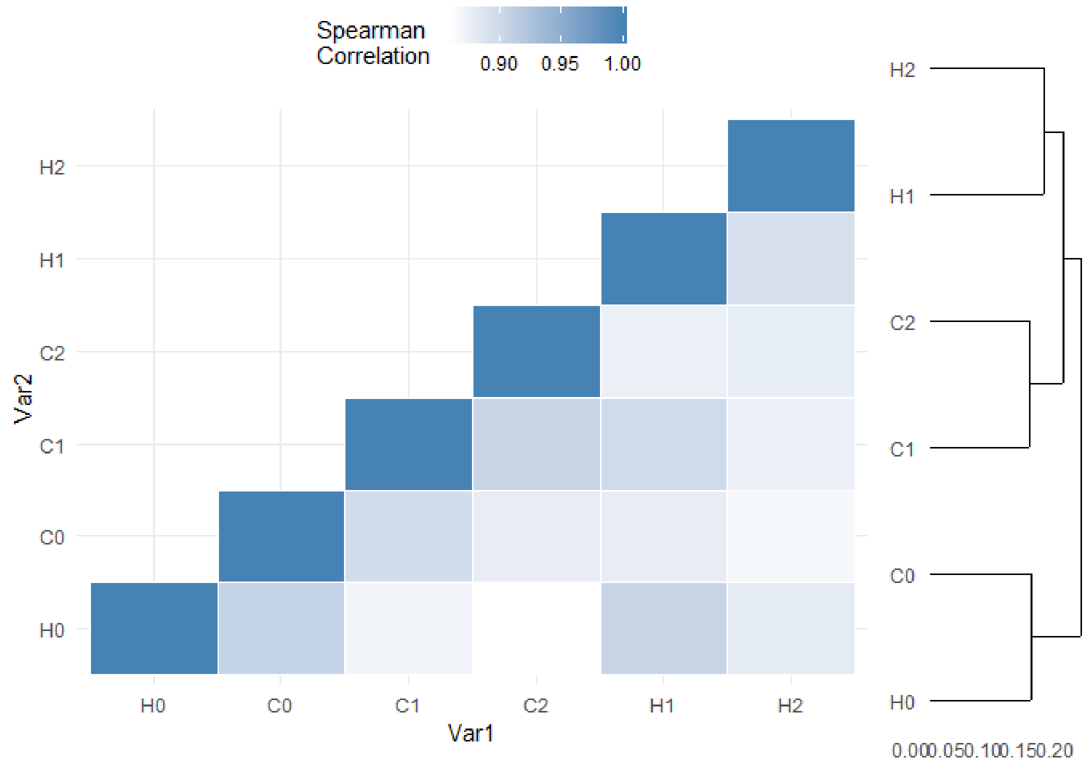
```
grid.newpage()
print(heatmap.plot, vp = viewport(x = 0.4, y = 0.5, width = 0.8, height = 1.0))
print(dendro.plot, vp = viewport(x = 0.90, y = 0.445, width = 0.2, height = 1.0))
```



ggplot2: Building a Plot - heatmap

supplying data
mapping aesthetics
defining geometric

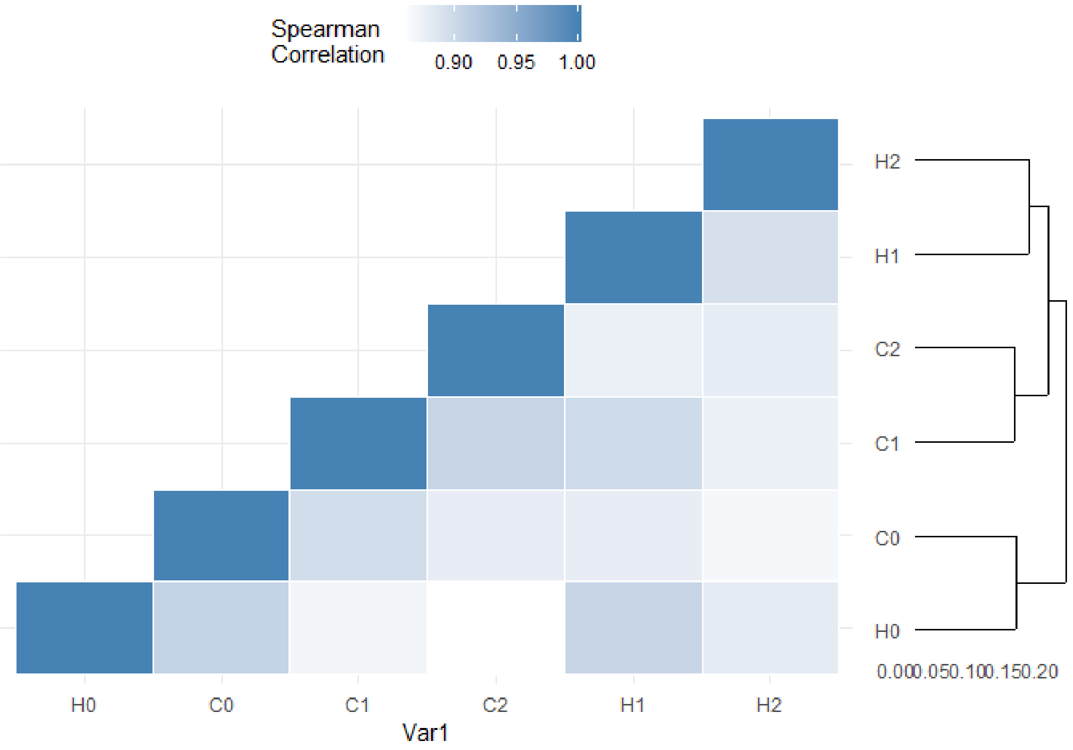
```
grid.newpage()
print(heatmap.plot, vp = viewport(x = 0.4, y = 0.5, width = 0.8, height = 1.0))
print(dendro.plot, vp = viewport(x = 0.90, y = 0.445, width = 0.2, height = 1.0))
```



ggplot2: Building a Plot - heatmap

supplying data
mapping aesthetics
defining geometric

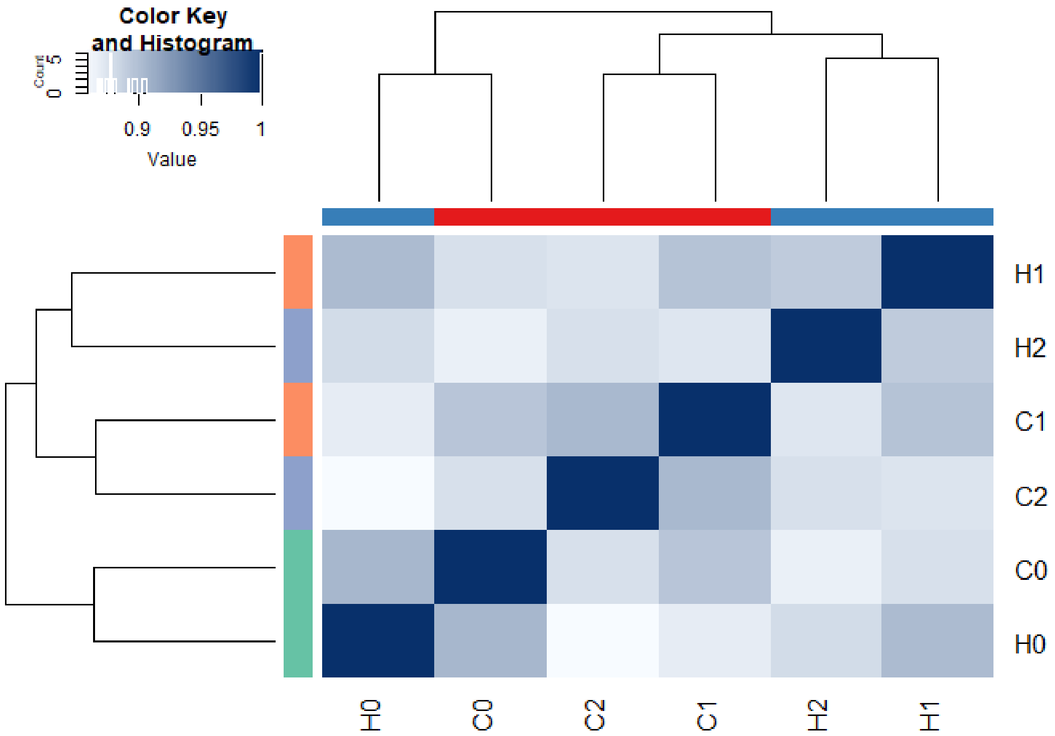
```
grid.newpage()
print(heatmap.plot, vp = viewport(x = 0.4, y = 0.5, width = 0.8, height = 1.0))
print(dendro.plot, vp = viewport(x = 0.90, y = 0.43, width = 0.2, height = 0.77))
```



ggplot2: Building a Plot - heatmap

EXERCISE

```
heatmap.2(cors, distfun=dist, hclustfun=hclust, dendrogram="both",
          scale="none", trace='none', col=colors, densecol="white",
          ColSideColors=pal[as.integer(as.factor(sample.details$Species))],
          RowSideColors=pal[as.integer(as.factor(sample.details$TimePoint))+9])
```



ggplot2: Saving Plots

Option #1: ggsave()

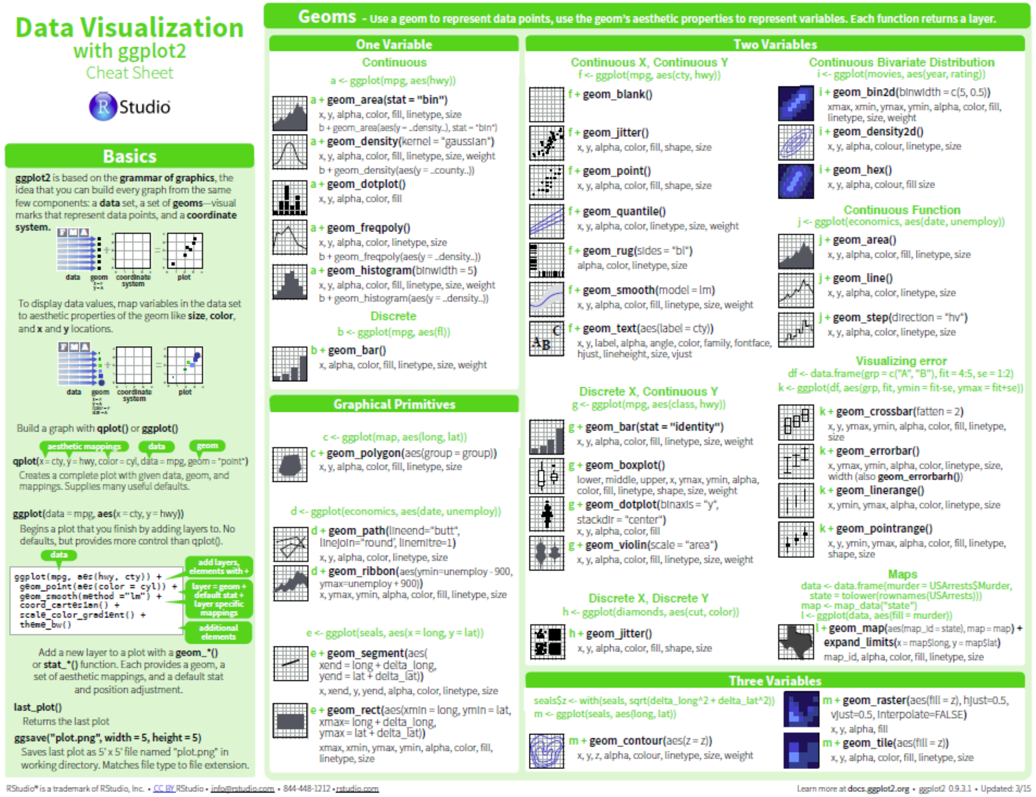
```
ggplot(data = scores) +
  geom_point(mapping = aes(x = PC1, y = PC2, color = Species))
ggsave(filename = "PCA-plot1.png", path = "./", width = 10, height = 7)
```

Option #2: png(), pdf(), jpg()

```
png(filename = "./PCA-plot2.png", width = 10, height = 7, units = "in", res = 300)
ggplot(data = scores) +
  geom_point(mapping = aes(x = PC1, y = PC2, color = Species))
dev.off()
```

ggplot2: Everything Else!

Information about additional functions can be found at:
https://ggplot2.tidyverse.org/reference/



Data Visualization with ggplot2
Cheat Sheet

Basics

ggplot2 is based on the grammar of graphics, the idea that you can build every graph from the same few components: a data set, a set of geoms—visual marks that represent data points, and a coordinate system.

To display data values, map variables in the data set to aesthetic properties of the geom like size, color, and x and y locations.

Build a graph with qplot() or ggplot()

qplot(x = cty, y = hwy, color = cyl, data = mpg, geom = "point")
Creates a complete plot with given data, geom, and mappings. Supplies many useful defaults.

ggplot(data = mpg, aes(x = cty, y = hwy))
Begins a plot that you finish by adding layers to. No defaults, but provides more control than qplot().

```
ggplot(mpg, aes(hwy, cty)) +
geom_point(aes(color = cyl)) +
geom_smooth(method ="lm") +
coord_cartesian() +
scale_color_gradient() +
theme_bw()
```

Add a new layer to a plot with a geom_*() or stat_*() function. Each provides a geom, a set of aesthetic mappings, and a default stat and position adjustment.

last_plot()
Returns the last plot

ggsave("plot.png", width = 5, height = 5)
Saves last plot as 5' x 5' file named "plot.png" in working directory. Matches file type to file extension.

Geoms - Use a geom to represent data points, use the geom's aesthetic properties to represent variables. Each function returns a layer.

One Variable

Continuous

a <- ggplot(mpg, aes(hwy))

a + geom_area(stat = "bin")

a + geom_density(kernel = "gaussian")

a + geom_dotplot()

a + geom_freqpoly()

a + geom_histogram(binwidth = 5)

Discrete

b <- ggplot(mpg, aes(fl))

b + geom_bar()

Graphical Primitives

c <- ggplot(map, aes(long, lat))

c + geom_polygon(aes(group = group))

d <- ggplot(economics, aes(date, unemploy))

d + geom_path(lineend="butt", linejoin="round", linemitre=1)

d + geom_ribbon(aes(ymin=unemploy - 900, ymax=unemploy + 900))

e <- ggplot(seals, aes(x = long, y = lat))

e + geom_segment(aes(xend = long + delta_long, yend = lat + delta_lat))

e + geom_rect(aes(xmin = long, ymin = lat, xmax= long + delta_long, ymax = lat + delta_lat))

Two Variables

Continuous X, Continuous Y

f <- ggplot(mpg, aes(cty, hwy))

f + geom_blank()

f + geom_jitter()

f + geom_point()

f + geom_quantile()

f + geom_rug(sides = "bl")

f + geom_smooth(model = lm)

f + geom_text(aes(label = cty))

Discrete X, Continuous Y

g <- ggplot(mpg, aes(class, hwy))

g + geom_bar(stat = "identity")

g + geom_boxplot()

g + geom_dotplot(binaxis = "y", stackdir = "center")

g + geom_violin(scale = "area")

Discrete X, Discrete Y

h <- ggplot(diamonds, aes(cut, color))

h + geom_jitter()

Continuous Bivariate Distribution

i <- ggplot(movies, aes(year, rating))

i + geom_bin2d(binwidth = c(5, 0.5))

i + geom_density2d()

i + geom_hex()

Continuous Function

j <- ggplot(economics, aes(date, unemploy))

j + geom_area()

j + geom_line()

j + geom_step(direction = "hv")

Visualizing error

df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)

k <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))

k + geom_crossbar(fatten = 2)

k + geom_errorbar()

k + geom_linerange()

k + geom_pointrange()

Maps

data <- data.frame(murder = USArrests$Murder, state = tolower(rownames(USArrests)))

map <- map_data("state")

l <- ggplot(data, aes(fill = murder))

l + geom_map(aes(map_id = state), map = map) + expand_limits(x = map$long, y = map$lat)

Three Variables

seals$z <- with(seals, sqrt(delta_long^2 + delta_lat^2))

m <- ggplot(seals, aes(long, lat))

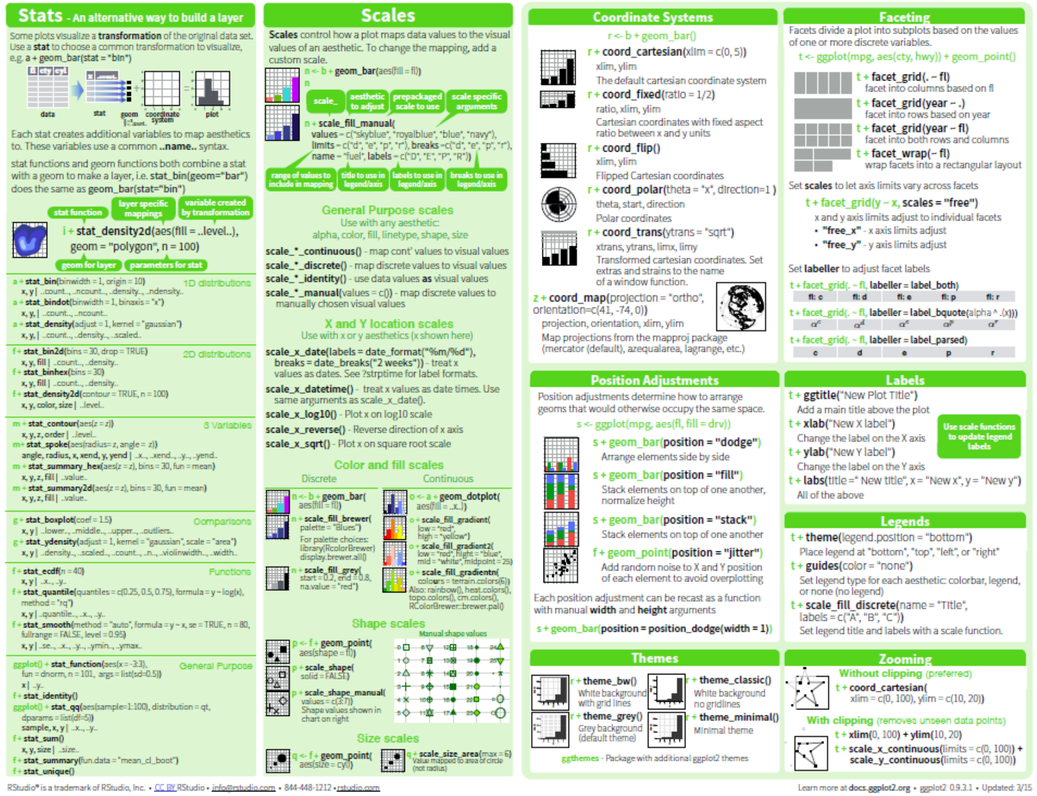
m + geom_contour(aes(z = z))

m + geom_raster(aes(fill = z), hjust=0.5, vjust=0.5, interpolate=FALSE)

m + geom_tile(aes(fill = z))

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com

Learn more at docs.ggplot2.org • ggplot2 0.9.3.1 • Updated: 3/15



Stats - An alternative way to build a layer

Some plots visualize a transformation of the original data set. Use a stat to choose a common transformation to visualize, e.g. a + geom_bar(stat = "bin")

Each stat creates additional variables to map aesthetics to. These variables use a common ..name.. syntax.

stat functions and geom functions both combine a stat with a geom to make a layer, i.e. stat_bin(geom="bar") does the same as geom_bar(stat="bin")

i + stat_density2d(aes(fill = ..level..), geom = "polygon", n = 100)

Scales

Scales control how a plot maps data values to the visual values of an aesthetic. To change the mapping, add a custom scale.

General Purpose scales

scale_*_continuous() - map cont' values to visual values

scale_*_discrete() - map discrete values to visual values

scale_*_identity() - use data values as visual values

scale_*_manual(values = c()) - map discrete values to manually chosen visual values

X and Y location scales

scale_x_date(labels = date_format("%m/%d"), breaks = date_breaks("2 weeks"))

scale_x_datetime()

scale_x_log10() - Plot x on log10 scale

scale_x_reverse() - Reverse direction of x axis

scale_x_sqrt() - Plot x on square root scale

Color and fill scales

Shape scales

Size scales

Coordinate Systems

r <- b + geom_bar()

r + coord_cartesian(xlim = c(0, 5))

r + coord_fixed(ratio = 1/2)

r + coord_flip()

r + coord_polar(theta = "x", direction=1)

r + coord_trans(ytrans = "sqrt")

z + coord_map(projection = "ortho", orientation=c(41, -74, 0))

Position Adjustments

s <- ggplot(mpg, aes(fl, fill = drv))

s + geom_bar(position = "dodge")

s + geom_bar(position = "fill")

s + geom_bar(position = "stack")

f + geom_point(position = "jitter")

Themes

r + theme_bw()

r + theme_classic()

r + theme_grey()

r + theme_minimal()

Faceting

t <- ggplot(mpg, aes(cty, hwy)) + geom_point()

t + facet_grid(. ~ fl)

t + facet_grid(year ~ .)

t + facet_grid(year ~ fl)

t + facet_wrap(~ fl)

Labels

t + ggtitle("New Plot Title")

t + xlab("New X label")

t + ylab("New Y label")

t + labs(title =" New title", x = "New x", y = "New y")

Legends

t + theme(legend.position = "bottom")

t + guides(color = "none")

t + scale_fill_discrete(name = "Title", labels = c("A", "B", "C"))

Zooming

Without clipping (preferred)

t + coord_cartesian(xlim = c(0, 100), ylim = c(10, 20))

With clipping (removes unseen data points)

t + xlim(0, 100) + ylim(10, 20)

t + scale_x_continuous(limits = c(0, 100)) + scale_y_continuous(limits = c(0, 100))

RStudio® is a trademark of RStudio, Inc. • CC BY RStudio • info@rstudio.com • 844-448-1212 • rstudio.com

Learn more at docs.ggplot2.org • ggplot2 0.9.3.1 • Updated: 3/15

Basics of Gviz

(can use ggbio as an alternative)

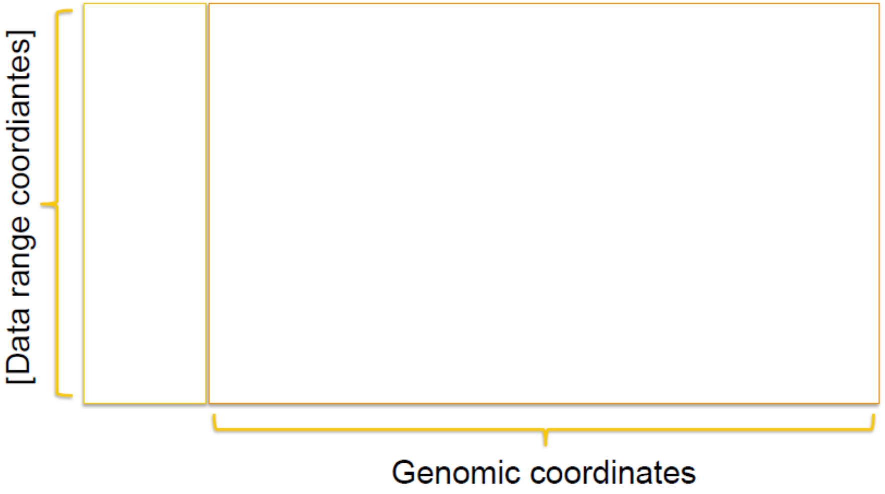
Gviz

- What
  - structured visualization framework to plot any type of data alongof large genomic coordinates
- When
  - integrated into Bioconductor 6.5 years ago
- Why
  - flexible and allows integration of publicly available genomic annotations (UCSC, ENSEMBL, biomaRt)
- How…

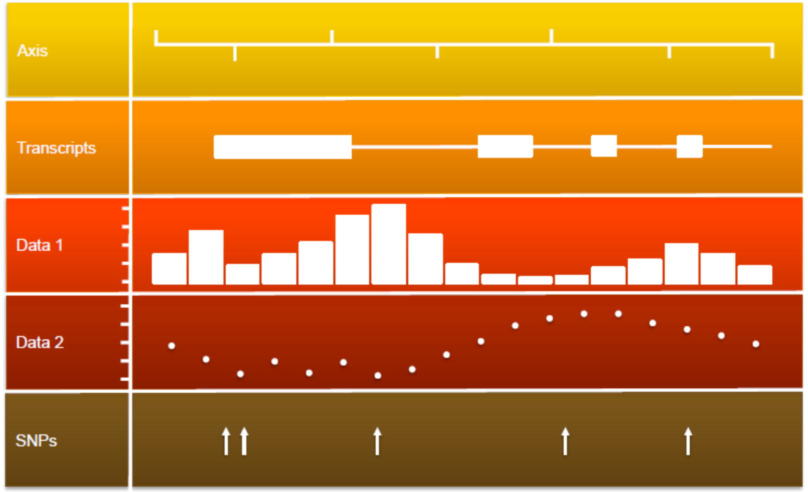
Gviz

- Layout similar to UCSC browser – different data types represented by different track classes

Track Layout



Track Panel Layout



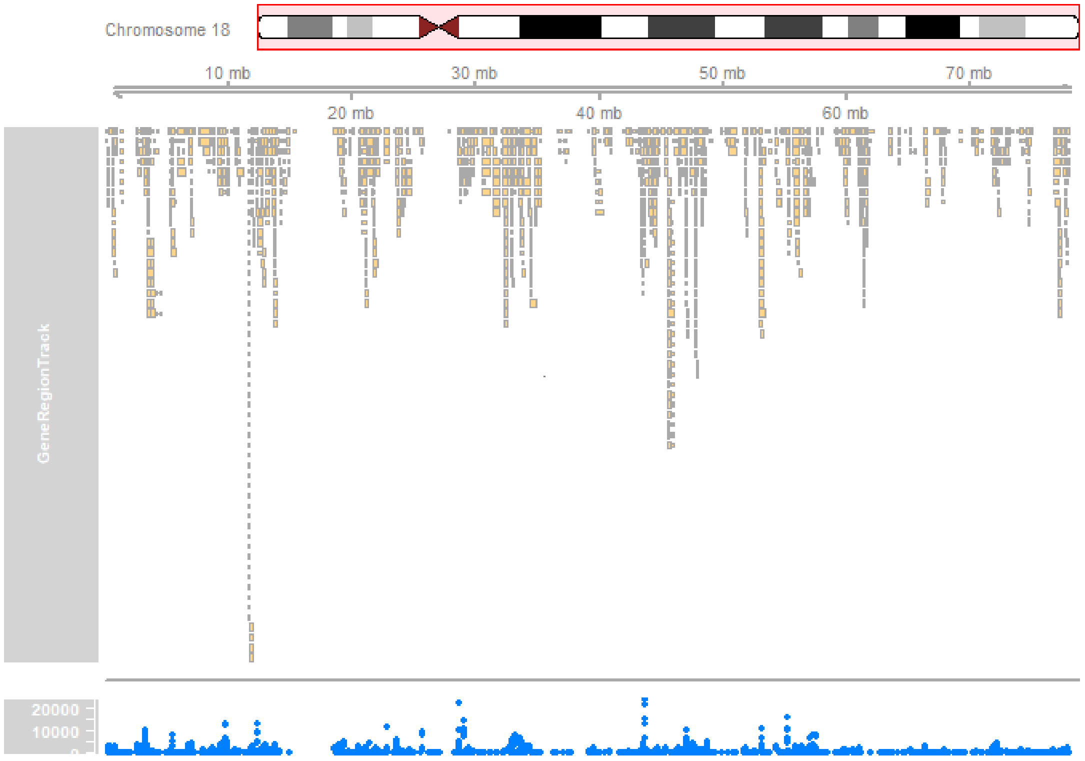
Gviz: A Typical Session

```
GRanges object with 40977 ranges and 10 metadata columns:
                seqnames            ranges strand |      source              type     score     phase          H0          H1          H2          C0          C1          C2
                   <Rle>         <IRanges>  <Rle> |    <factor>          <factor> <numeric> <integer> <character> <character> <character> <character> <character> <character>
      [1]           chr1     887132-887142      * | rtracklayer sequence_feature      <NA>      <NA>          65          30         212           1           1           0
      [2]           chr1     973495-973505      * | rtracklayer sequence_feature      <NA>      <NA>          78         201         177          43         121          70
      [3]           chr1     999920-999930      * | rtracklayer sequence_feature      <NA>      <NA>           2           1           2           2          96          24
      [4]           chr1     998123-998133      * | rtracklayer sequence_feature      <NA>      <NA>           4           1           1           2          63          15
      [5]           chr1   1008012-1008022      * | rtracklayer sequence_feature      <NA>      <NA>           0           2           2           2           6           0
      ...            ...               ...    ... .         ...               ...       ...       ...         ...         ...         ...         ...         ...         ...
  [40973]           chrY 22972174-22972184      * | rtracklayer sequence_feature      <NA>      <NA>           0           0           0           0           0           0
  [40974]           chrY 23023718-23023728      * | rtracklayer sequence_feature      <NA>      <NA>           0           0           0           0           0           0
  [40975]           chrY 23383479-23383489      * | rtracklayer sequence_feature      <NA>      <NA>           0           0           0           0           0           0
  [40976]           chrY 23476963-23476973      * | rtracklayer sequence_feature      <NA>      <NA>           0           0           0           0           0           0
  [40977] chr6_mann_hap4   1821010-1821020      * | rtracklayer sequence_feature      <NA>      <NA>           0           1           0           3           0           2
  -------
  seqinfo: 93 sequences (1 circular) from hg19 genome
```

```
GRanges object with 40977 ranges and 6 metadata columns:
                seqnames            ranges strand |          H0          H1          H2          C0          C1          C2
                   <Rle>         <IRanges>  <Rle> | <character> <character> <character> <character> <character> <character>
      [1]           chr1     887132-887142      * |          65          30         212           1           1           0
      [2]           chr1     973495-973505      * |          78         201         177          43         121          70
      [3]           chr1     999920-999930      * |           2           1           2           2          96          24
      [4]           chr1     998123-998133      * |           4           1           1           2          63          15
      [5]           chr1   1008012-1008022      * |           0           2           2           2           6           0
      ...            ...               ...    ... .         ...         ...         ...         ...         ...         ...
  [40973]           chrY 22972174-22972184      * |           0           0           0           0           0           0
  [40974]           chrY 23023718-23023728      * |           0           0           0           0           0           0
  [40975]           chrY 23383479-23383489      * |           0           0           0           0           0           0
  [40976]           chrY 23476963-23476973      * |           0           0           0           0           0           0
  [40977] chr6_mann_hap4   1821010-1821020      * |           0           1           0           3           0           2
  -------
  seqinfo: 93 sequences (1 circular) from hg19 genome
```

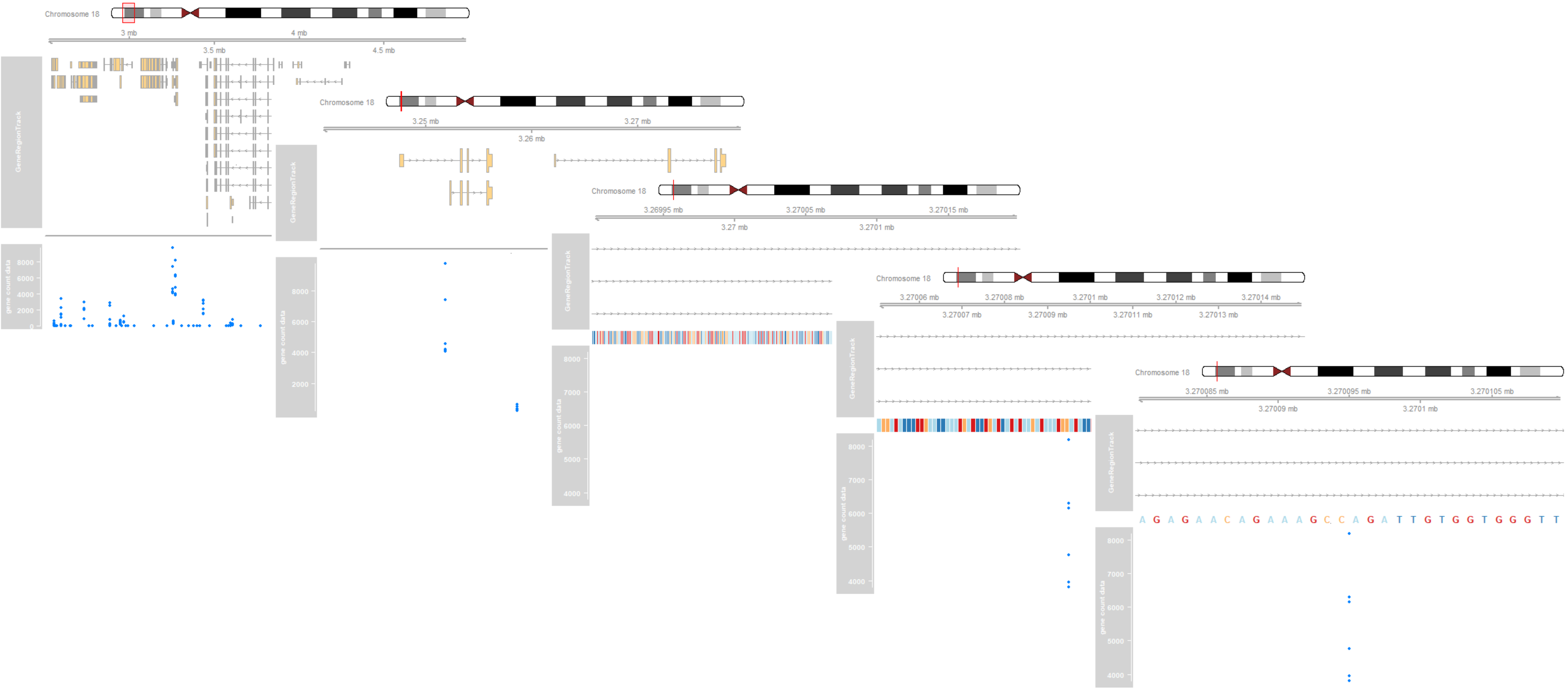
Gviz: A Typical Session

```
iTrack <- IdeogramTrack(genome = "hg19", chromosome = "chr18")
xTrack <- GenomeAxisTrack()
gTrack <- GeneRegionTrack(txdb, chromosome = "chr18")
sTrack <- SequenceTrack(Hsapiens, chromosome = "chr18")
dTrack <- DataTrack(gtf[seqnames(gtf) == "chr18"], name = "gene count data")
plotTracks(list(iTrack, xTrack, gTrack, sTrack, dTrack))
```



Gviz: A Typical Session - adjust view

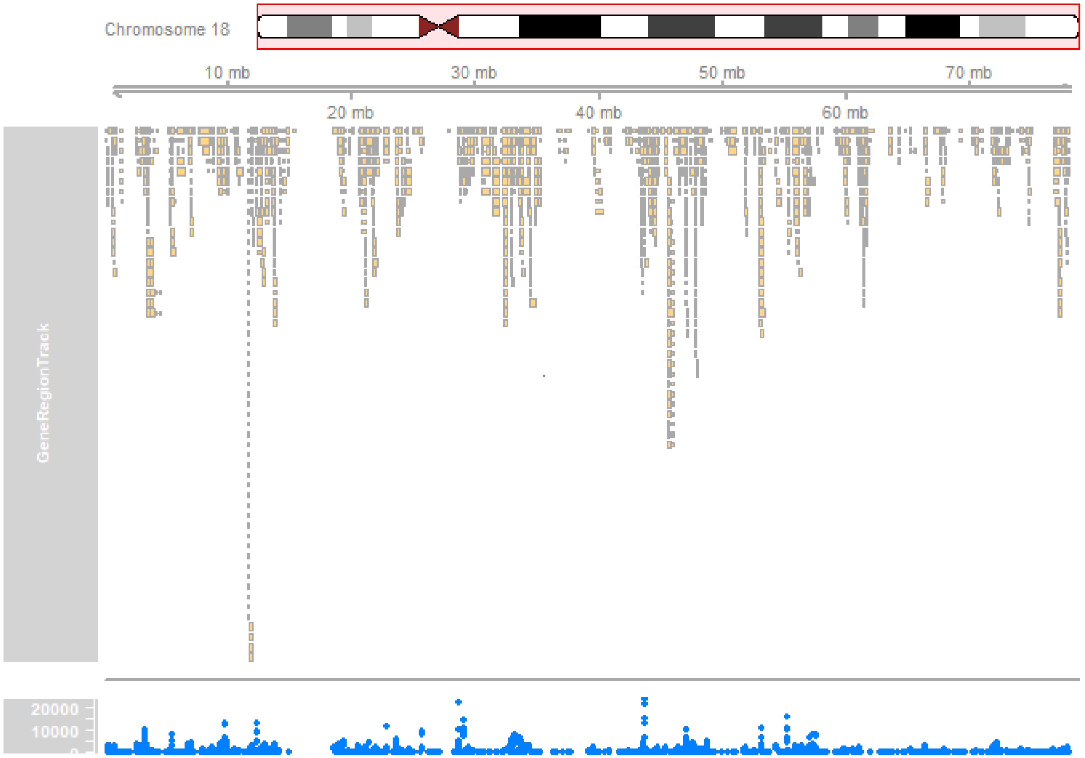
```
plotTracks(list(iTrack, xTrack, gTrack, sTrack, dTrack), from = 3270080, to = 3270110)
```

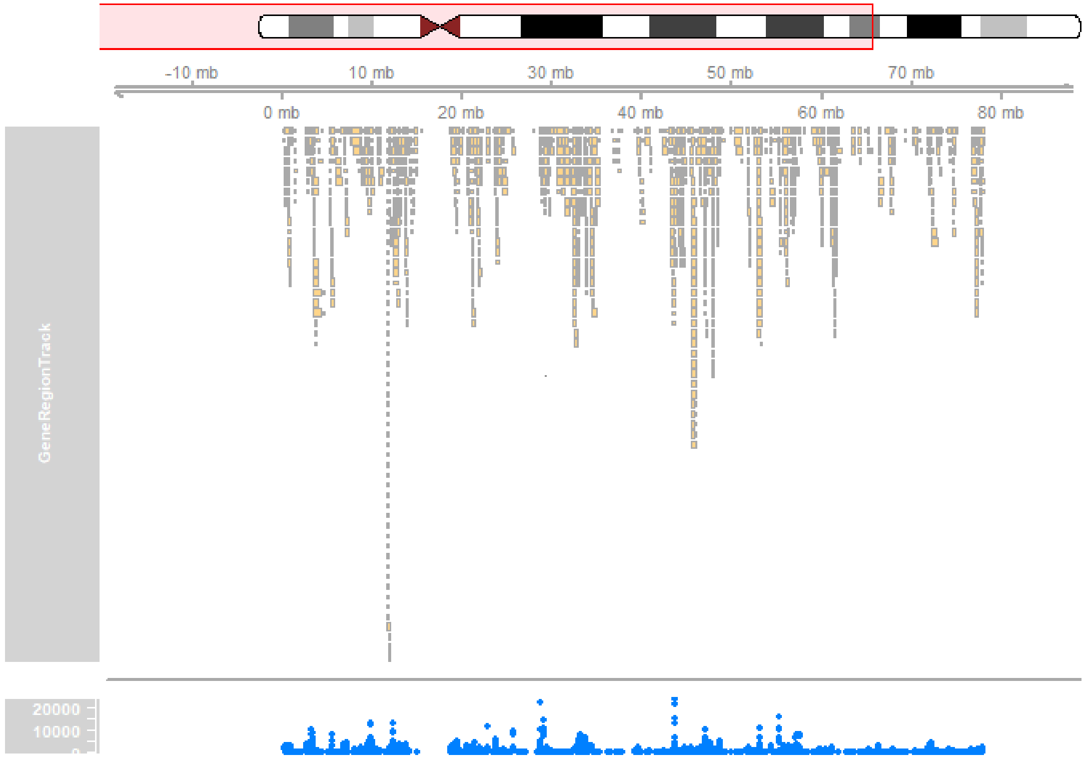


Gviz: A Typical Session - adjust view

```
plotTracks(list(iTrack, xTrack, gTrack, sTrack, dTrack))
```

```
plotTracks(list(iTrack, xTrack, gTrack,
sTrack, dTrack), extend.left = 0.25,
extend.right = 1e+07)
```



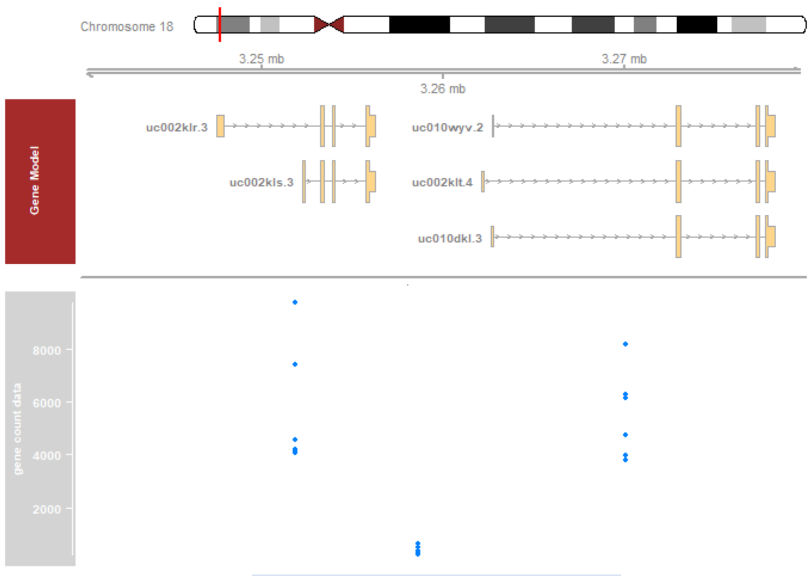


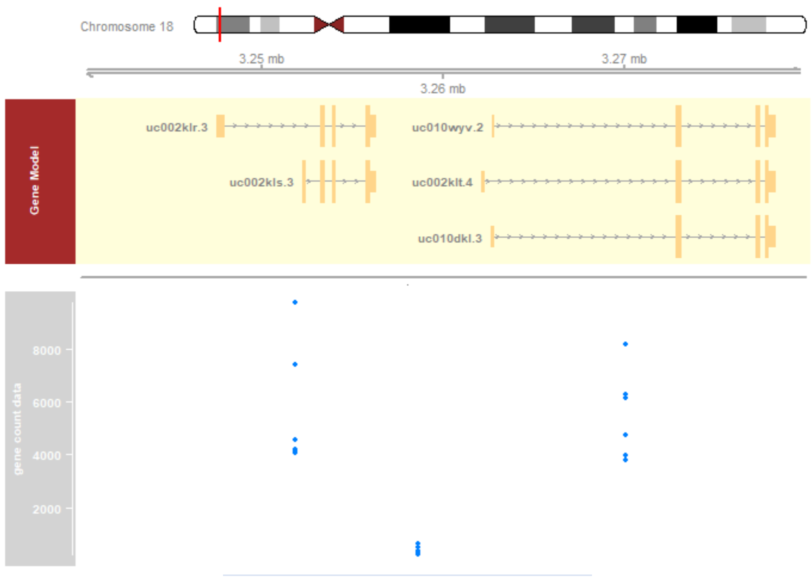
Gviz: A Typical Session – setting parameters

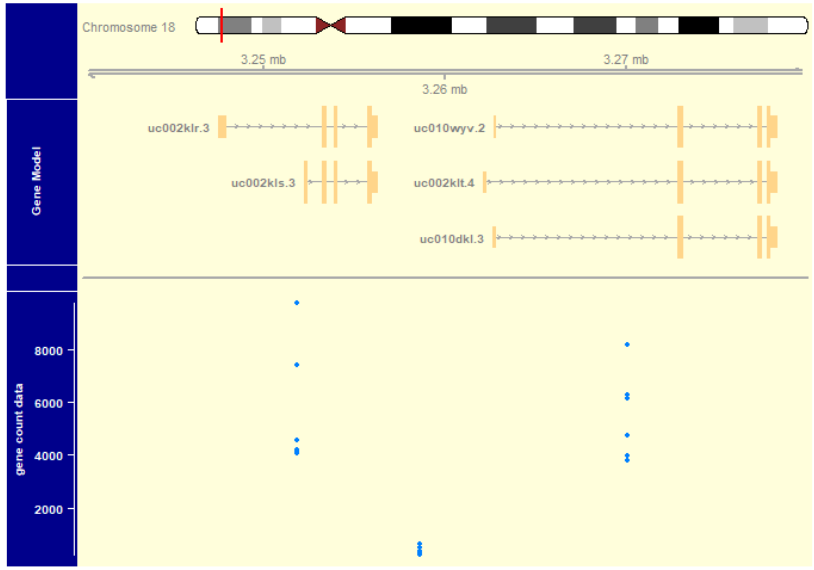
```
gTrack <- GeneRegionTrack(txdb,
          chromosome = "chr18",
          name = "Gene Model",
          transcriptAnnotation =
          "transcript",
          background.title = "brown")

plotTracks(list(iTrack, xTrack,
gTrack, sTrack, dTrack), from =
3240000, to = 3280000)
```

```
displayPars(gTrack) <-
list(background.panel =
"#FFFEDB", col = NULL)

plotTracks(list(iTrack, xTrack,
gTrack, sTrack, dTrack), from =
3240000, to = 3280000)
```

```
plotTracks(list(iTrack, xTrack, gTrack,
sTrack, dTrack), from = 3240000, to =
3280000, background.panel = "#FFFEDB",
background.title = "darkblue")
```

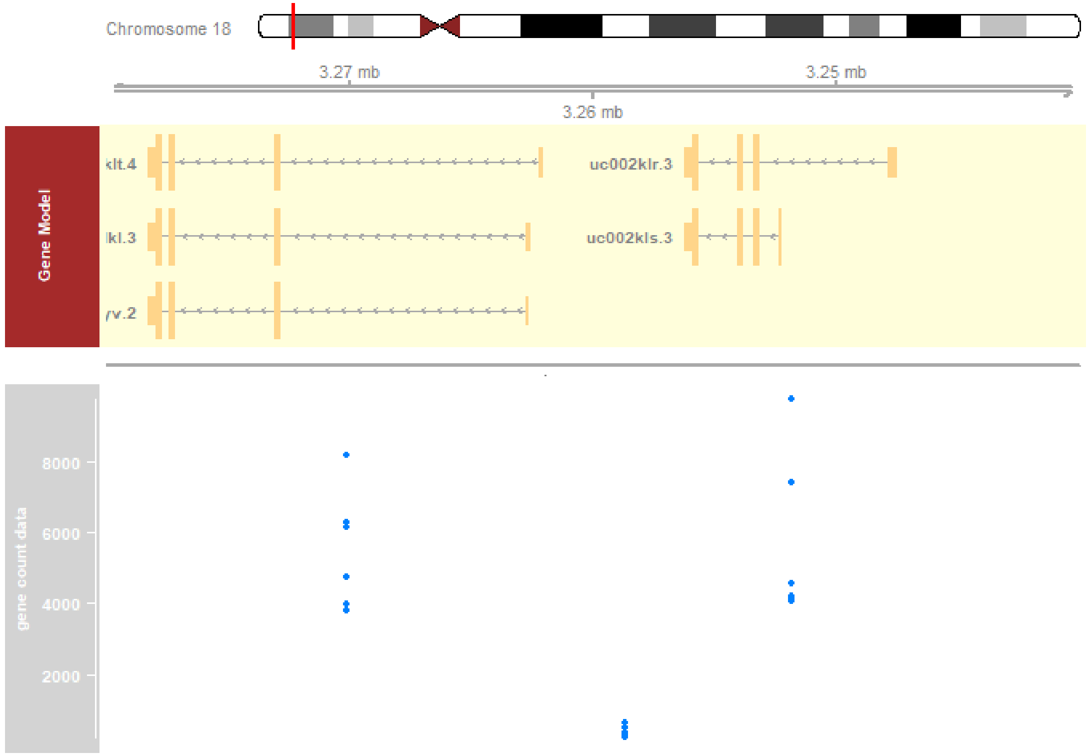






Gviz: A Typical Session - plot reverse strand

```
plotTracks(list(iTrack, xTrack, gTrack, sTrack, dTrack), from = 3240000,
to = 3280000, reverseStrand = TRUE)
```



Gviz: Track Types

IdeogramTrack

- view of the displayed region on a schematic model of a chromosome with chromosome band information from UCSC

GenomeAxisTrack

- genomic axis or scale indicator with optional highlighted regions

SequenceTrack

- genomic sequence in letter or false color representation depending on the zoom level

DataTrack

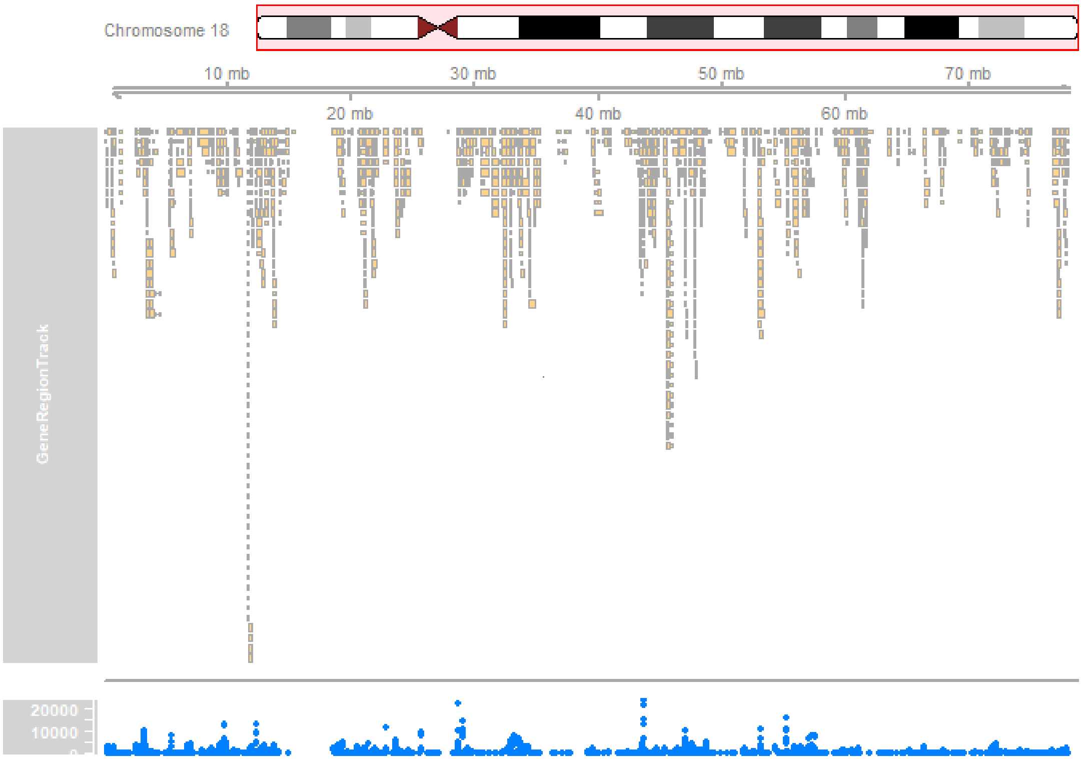
- numeric values (single or grouped) along with genomic coordinates, can be plotted in several ways

AnnotationTrack

- generic annotation features (with at least start, stop, strand, and chromosome information), optional grouping

GeneRegionTrack

- gene or transcript models with grouping on the level of exons and transcripts, can be fetched dynamically from Ensembl as the BiomartGeneRegionTrack child class



Gviz: Ideogram Track

Purpose

- indicate the currently displayed genomic range in the context of the current chromosome

Inputs

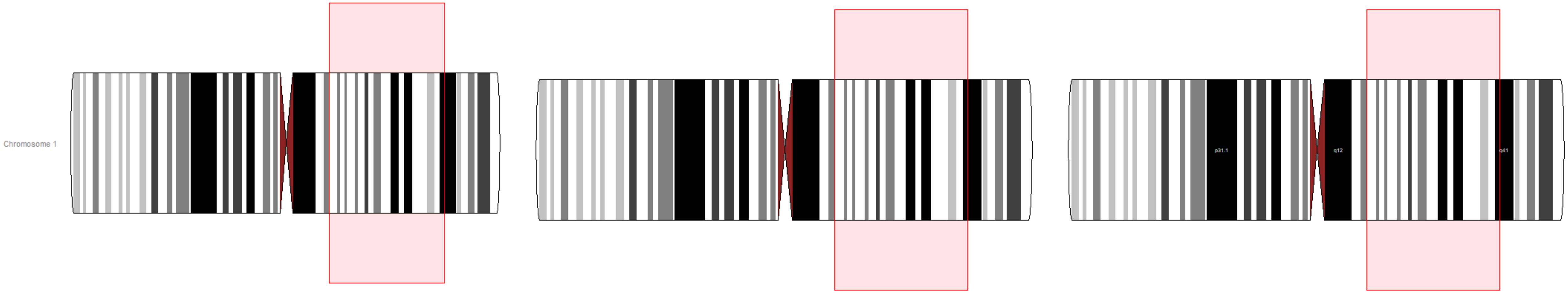
- fetch chromosome band information from UCSC
- data.frame

Details

- after first connection to UCSC, fetched results are cashed for duration of R session

Gviz: Ideogram Track

```
iTrack <- IdeogramTrack(genome = "hg19", chromosome = "chr1")
plotTracks(iTrack, from = 1.5e+08, to = 2.17e+08, showId = FALSE,
showBandId = TRUE, cex.bands = 0.5)
```



Gviz: Genome Axis Track

Purpose

- indicate the currently displayed genomic range either as an x-axis with evently spaced tick marks or as a scaled reference

Inputs

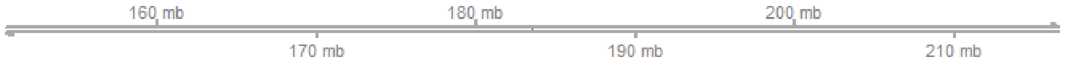
- NA

Details

- ranges on the axis can be highlighted (e.g., to indicate stretches of N nucleotides)

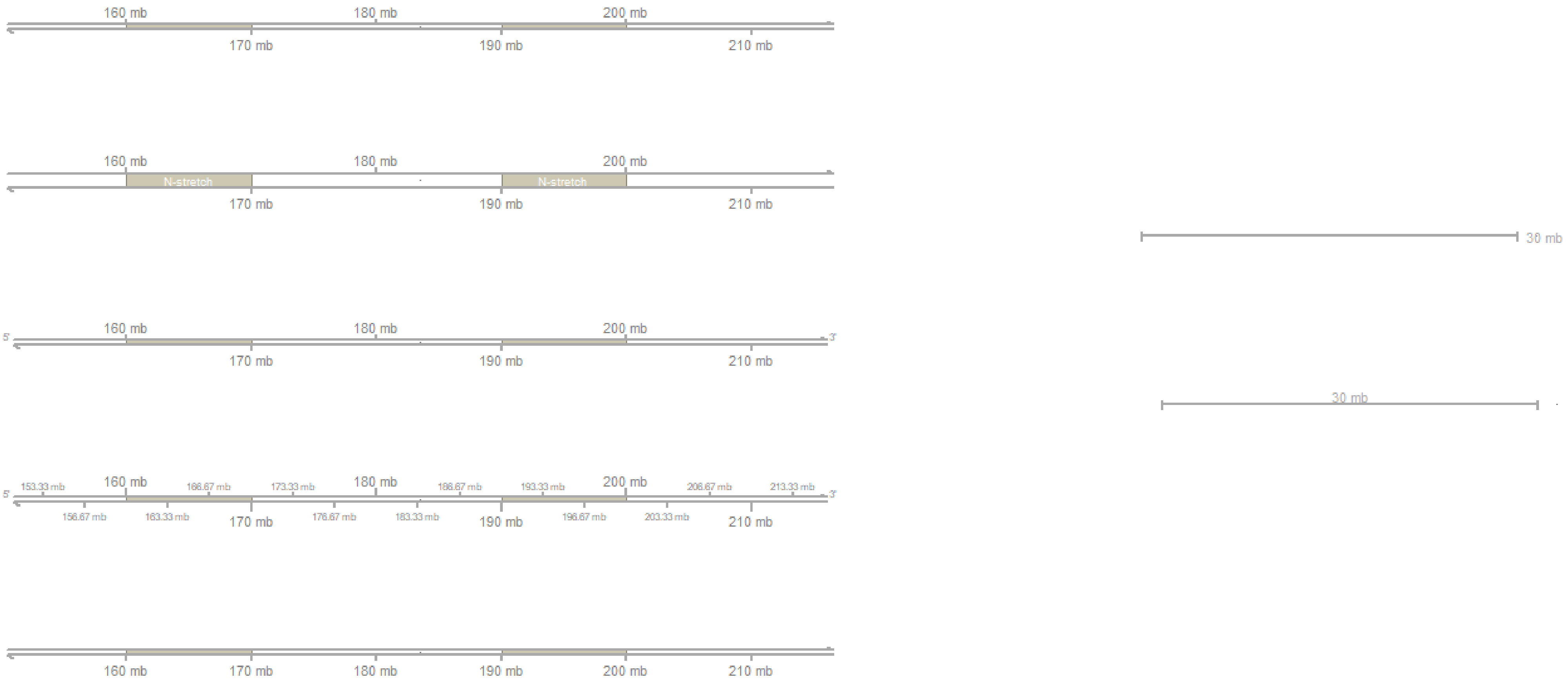
Gviz: Genome Axis Track

```
xTrack <- GenomeAxisTrack()
plotTracks(xTrack, from = 1.5e+08, to = 2.17e+08)
```



Gviz: Genome Axis Track

```
xTrack <- GenomeAxisTrack(range = IRanges(start = c(1.6e+08, 1.9e+08),
          end = c(1.7e+08, 2.0e+08), names = rep("N-stretch", + 2)))
plotTracks(xTrack, from = 1.5e+08, to = 2.17e+08, labelPos = "above")
```



Gviz: Sequence Track

Purpose

- show genomic sequence of the currently displayed region

Inputs

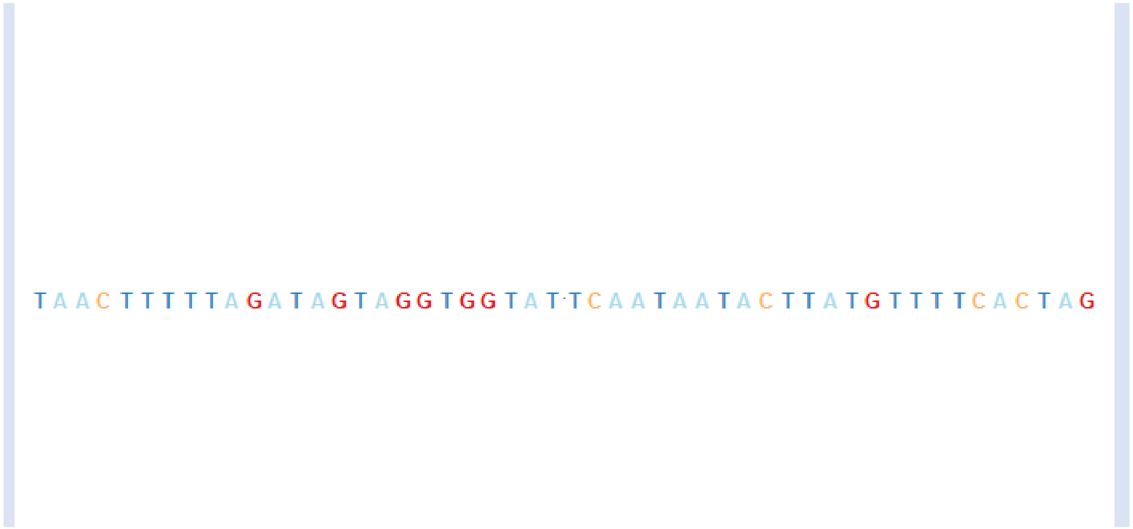
- DNAStringSet
- Bsgenome
- FASTA file (indexed or not indexed)
- 2bit file

Details

- depending on the zoom level, sequences will be shown as individual letters, as color-coded boxes, or as a horizontal line

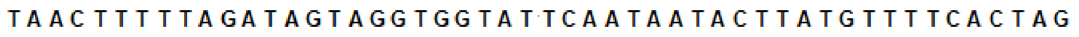
Gviz: Sequence Track

```
sTrack <- SequenceTrack(Hsapiens, chromosome = "chr1")
plotTracks(sTrack, from = 1.5e08, to = 150000050)
```

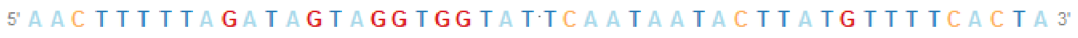


Gviz: Sequence Track

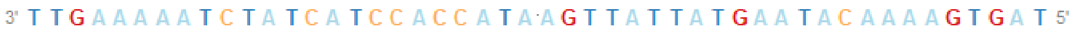
```
sTrack <- SequenceTrack(Hsapiens, chromosome = "chr1")
plotTracks(sTrack, from = 1.5e08, to = 150000050, add53 = TRUE,
complement = TRUE)
```



```
TAACTTTTTAGATAGTAGGTGGTATTCAATAATACTTATGTTTTCACTAG
```



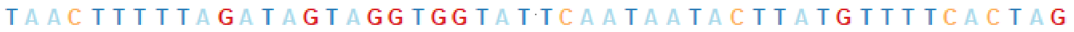
```
5' AACTTTTTAGATAGTAGGTGGTATTCAATAATACTTATGTTTTCACTA 3'
```

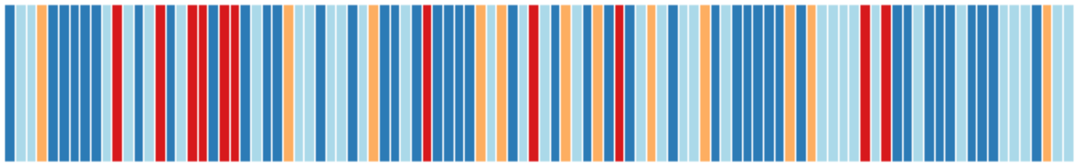


```
3' TTGAAAAATCTATCATCCACCATAAGTTATTATGAATACAAAAGTGAT 5'
```

Gviz: Sequence Track

```
sTrack <- SequenceTrack(Hsapiens, chromosome = "chr1")
plotTracks(sTrack, from = 1.5e08, to = 2.17e+08)
```





TAACTTTTTAGATAGTAGGTGGTATTCAATAATACTTATGTTTTCACTAGATCATCTGTACATAACTATTTTTCTCAAAAGAGTTATTTATTTAAATCAA

Gviz: Data Track

Purpose

- numeric data along genomic coordinates

Inputs

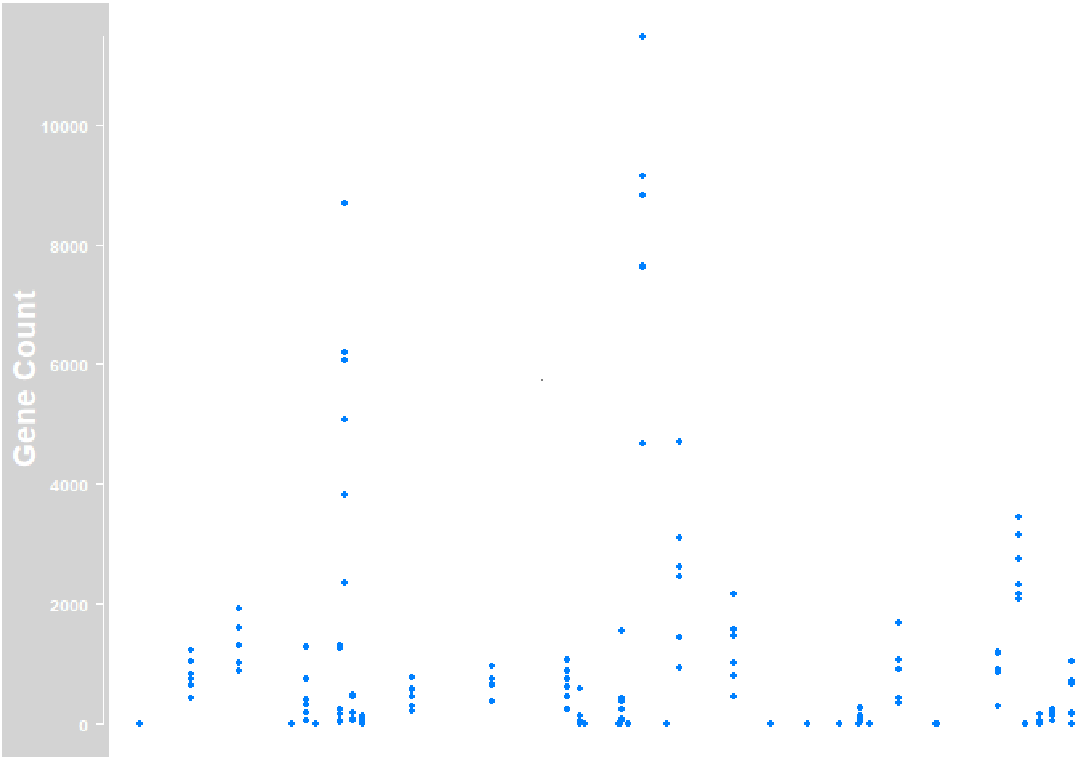
- IRanges (+ chromosome, strand, and data matrix)
- Granges
- various file types (WIG, BedGraph, BigWig, BAM)

Details

- flexible visualization options (e.g., line charts, scatter plots, box plots, bar charts), sample grouping, and data transformations

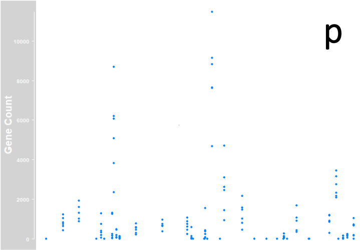
Gviz: Data Track

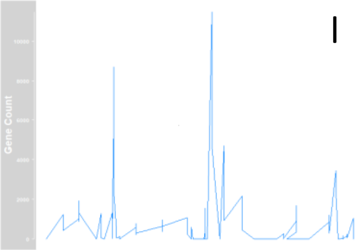
```
dTrack <- DataTrack(gtf, name = "Gene Count")
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08)
```

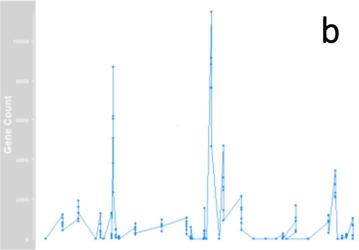


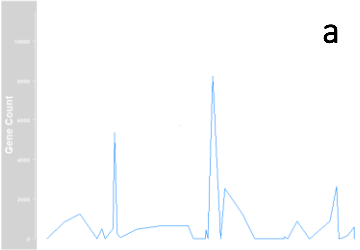
Gviz: Data Track

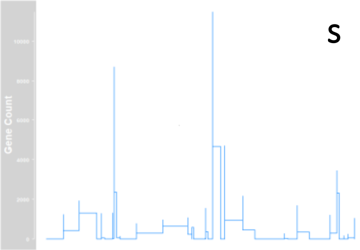
```
dTrack <- DataTrack(gtf)
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08, type = <type>)
```

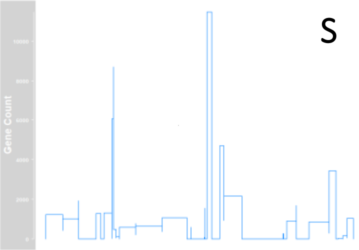


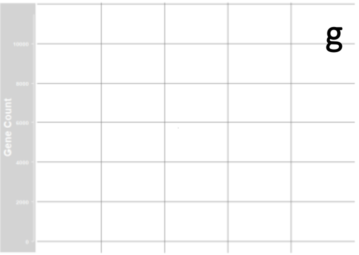


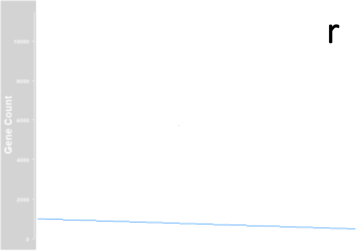


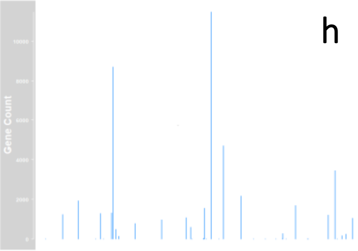


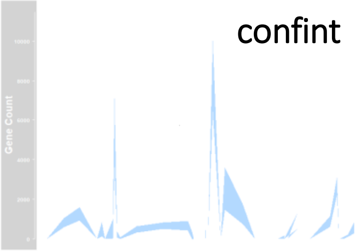


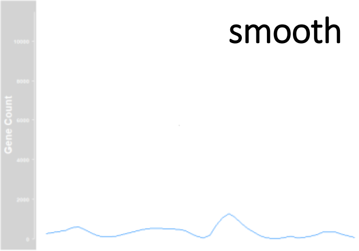


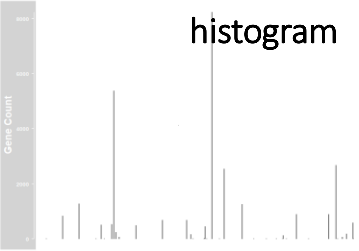


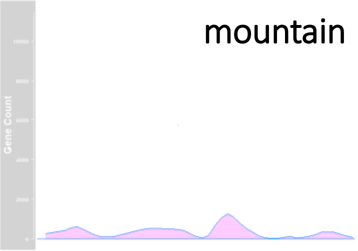


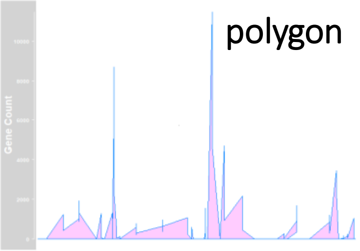


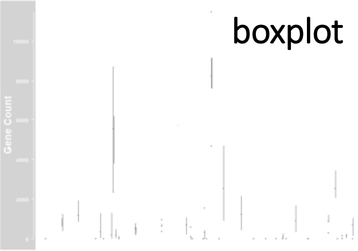


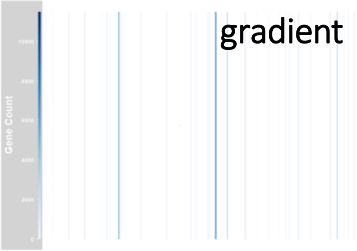


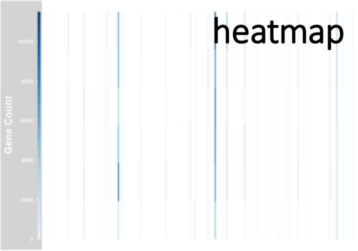


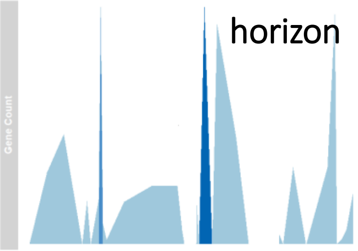






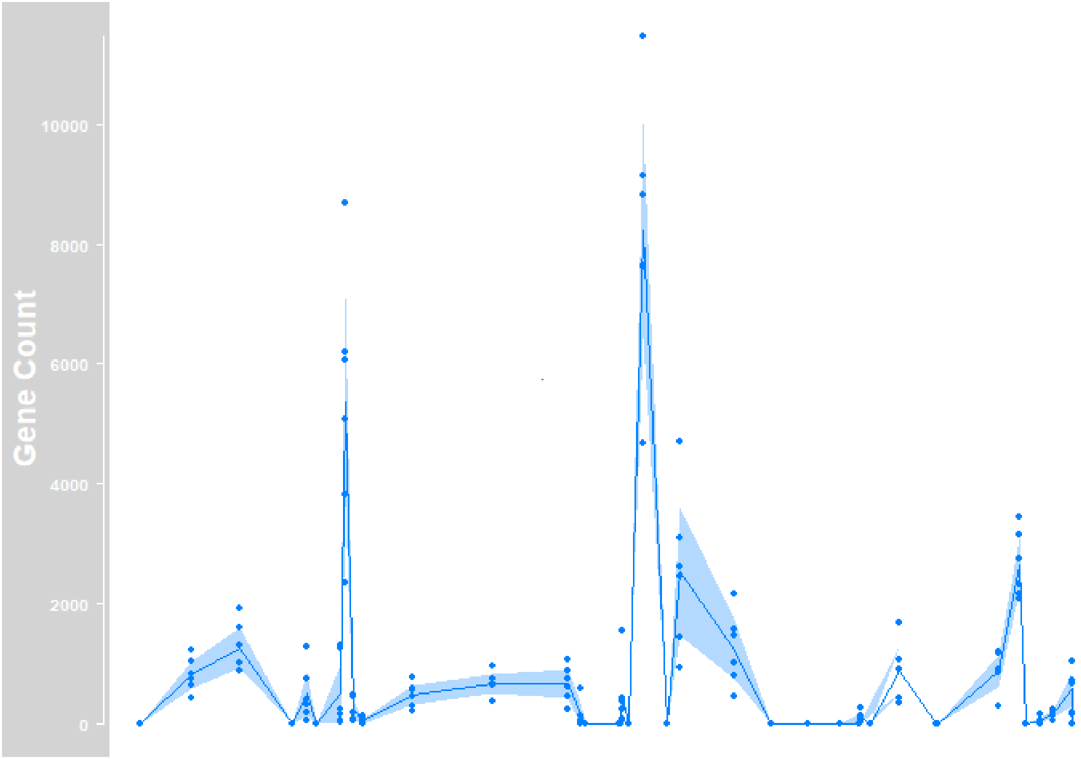






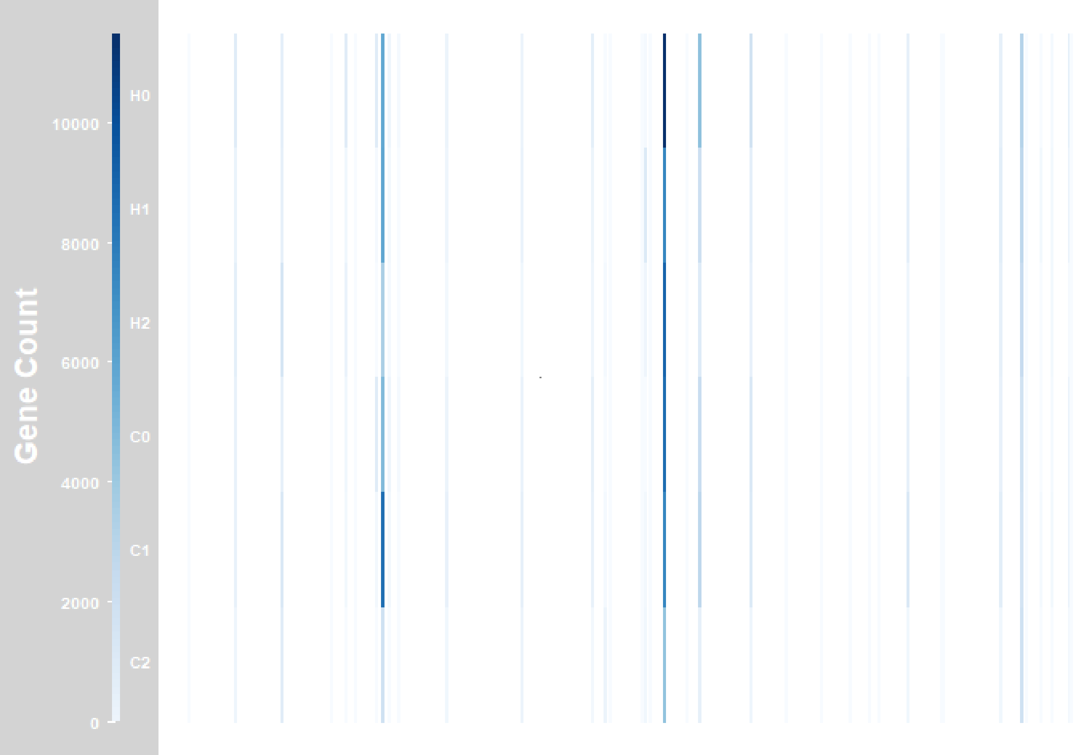
Gviz: Data Track

```
dTrack <- DataTrack(gtf)
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08,
type = c("a", "p", "confint"))
```



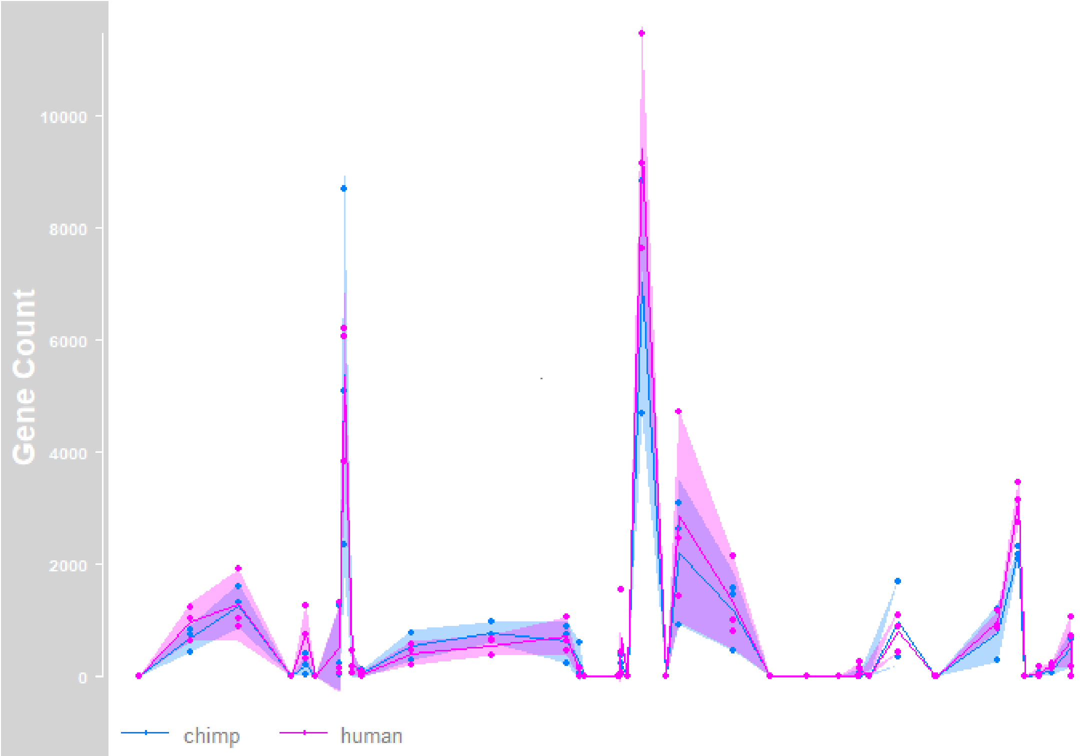
Gviz: Data Track

```
dTrack <- DataTrack(gtf)
colnames(mcols(gtf))
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08,
type = c("heatmap"), showSampleNames = TRUE, cex.sampleNames = 0.6)
```



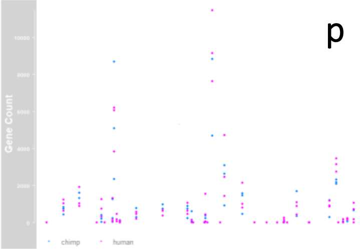
Gviz: Data Track

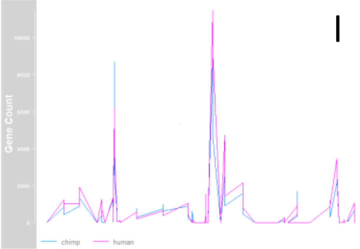
```
dTrack <- DataTrack(gtf, name = "Gene Count",
                    groups = rep(c("human", "chimp"), each = 3))
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08,
type = c("a", "p", "confint"))
```

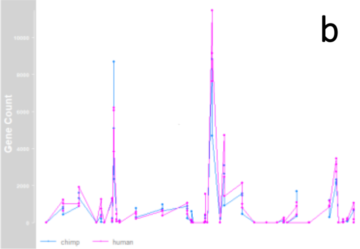


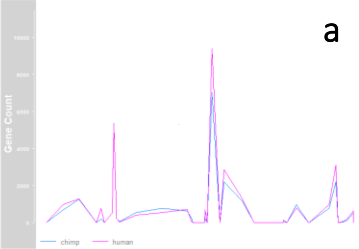
Gviz: Data Track

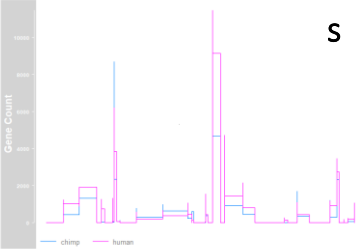
```
dTrack <- DataTrack(gtf, name = "Gene Count",
                    groups = rep(c("human", "chimp"), each = 3))
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08, type = <type>)
```

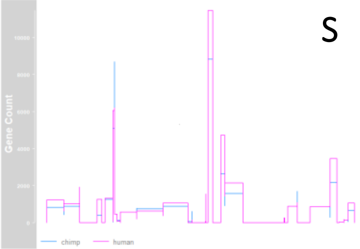

p

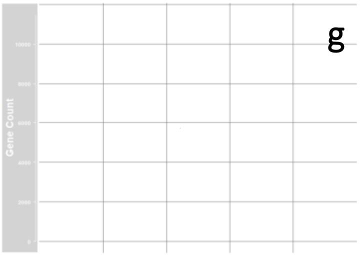

l

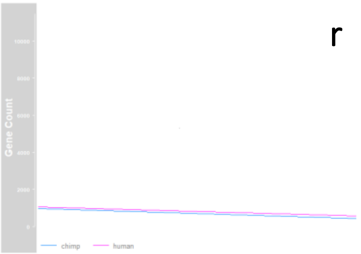

b

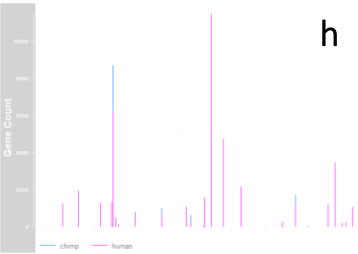

a

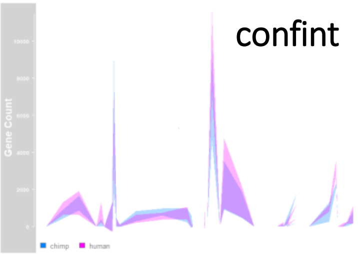

s

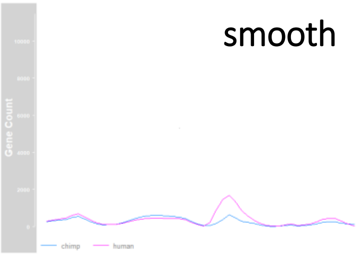

S

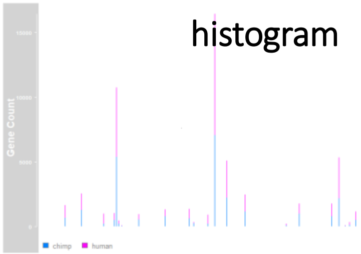

g

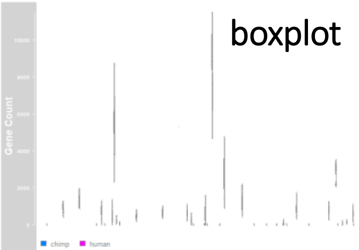

r

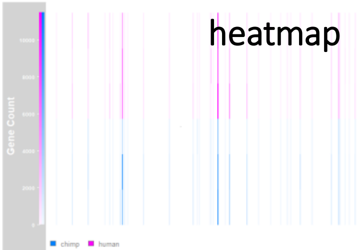

h


confint

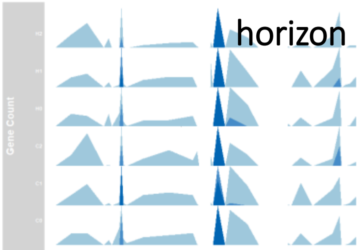

smooth


histogram


boxplot

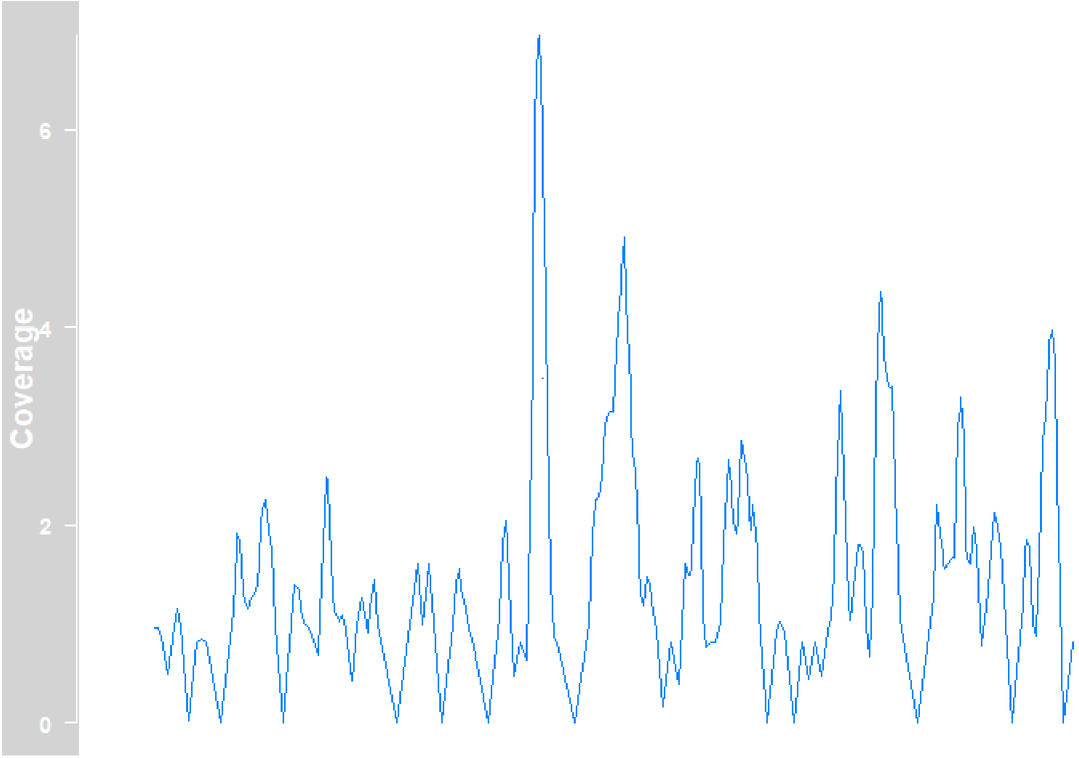

heatmap

```
dTrack <- DataTrack(gtf, name = "Gene Count", groups = colnames(mcols(gtf)))
plotTracks(dTrack, chromosome = "chr1", from = 1.5e08, to = 1.51e+08, type =
"horizon", showSampleNames = TRUE, cex.sampleNames = 0.6)
```


horizon

Gviz: Data Track

```
bamFile <- system.file("extdata/test.bam", package = "Gviz")
bamTrack <- DataTrack(range = bamFile, genome = "hg19", type = "l",
                      name = "Coverage", window = -1, chromosome = "chr1")
plotTracks(bamTrack, from = 189990000, to = 1.9e+08)
```



Gviz: Annotation Track

Purpose

- simple annotation features with at least start, stop, strand, and chromosome information
- items can be grouped and colored according to type

Inputs

- IRanges (+ chromosome and strand as separate arguments)
- GRanges
- GRangesList
- various file types (WIG, BedGraph, BigWig, BAM)

Details

- overlapping items are stacked for optimal utilization of available plotting space
- depending on the available space and resolution some items may be merged
- additional information for each annotation item can be added by means of the DetailsAnnotationTrack child class

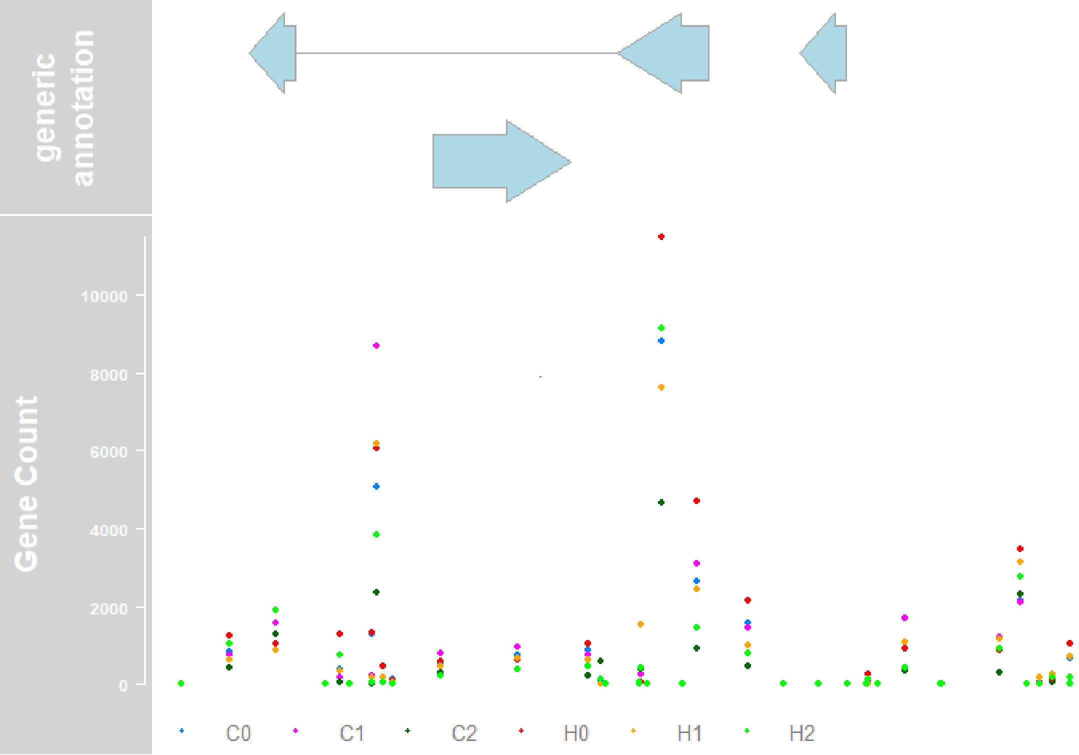
Gviz: Annotation Track

```
annoTrack <- AnnotationTrack()
plotTracks(list(annoTrack, dTrack), from = 1.5e08, to = 1.51e+08)
```

```
annoTrack <- AnnotationTrack(start = st, end =
ed, strand = str, genome = "hg19", chromosome =
"chr1", feature = "test", group = gr, id =
paste("annTrack item", 1:4), name = "generic
annotation", stacking = "squish")
```

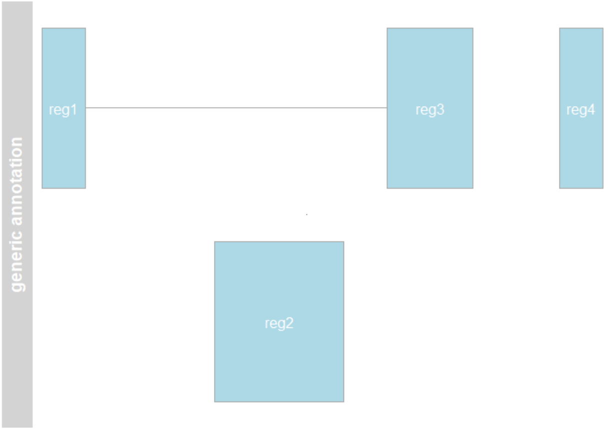
```
annoTrack <- AnnotationTrack(range =
df, genome = "hg19", chromosome =
"chr1", name = "generic annotation",
stacking = "squish")
```

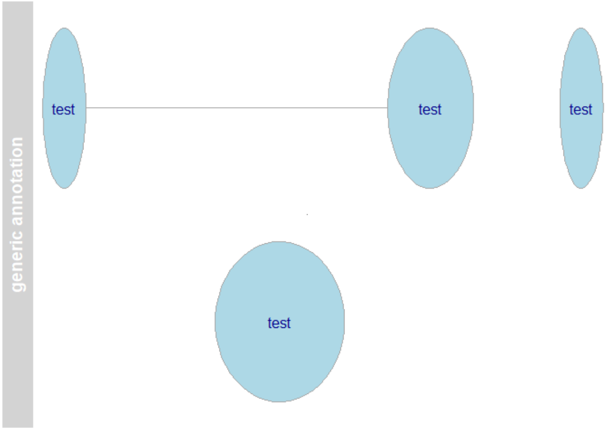
```
annoTrack <- AnnotationTrack(range
= gr, name = "generic annotation",
stacking = "squish")
```

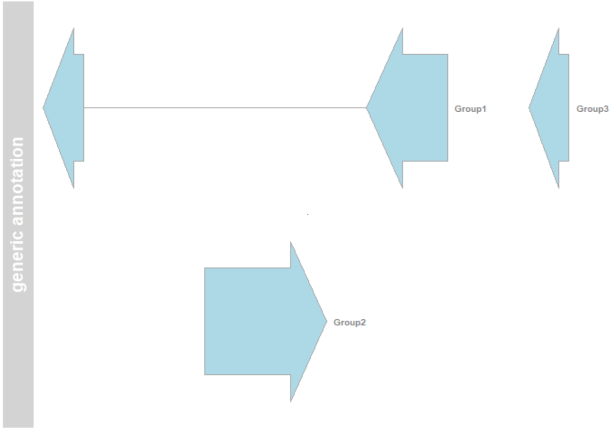


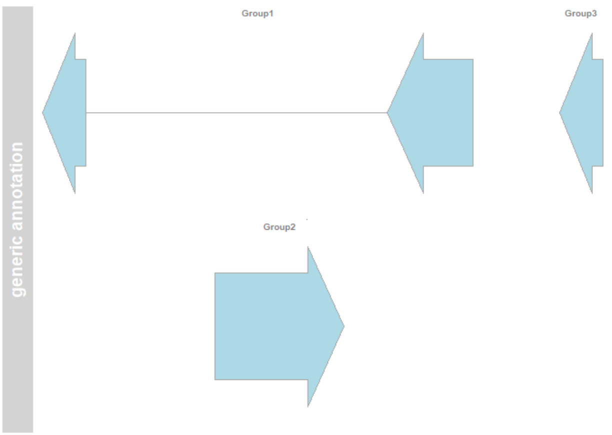
Gviz: Annotation Track

```
plotTracks(annoTrack, shape = <shape>, featureAnnotation = <id>,
           fontcolor.feature = <color>, just.group = <position>)
```









```
shape = "box”
featureAnnotation = "id"
```

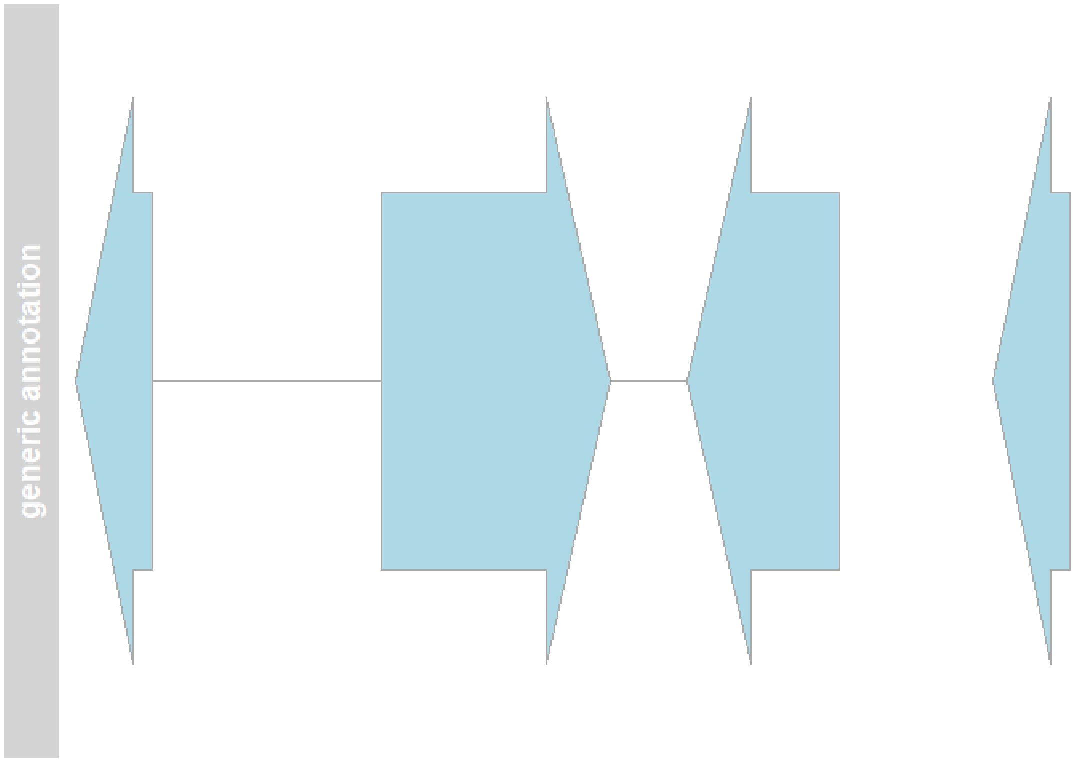
```
shape = "ellipse‶
featureAnnotation = "feature‶
fontcolor.feature = "darkblue"
```

```
groupAnnotation = "group‶
just.group = "right"
```

```
groupAnnotation = "group‶
just.group = ‶above"
```

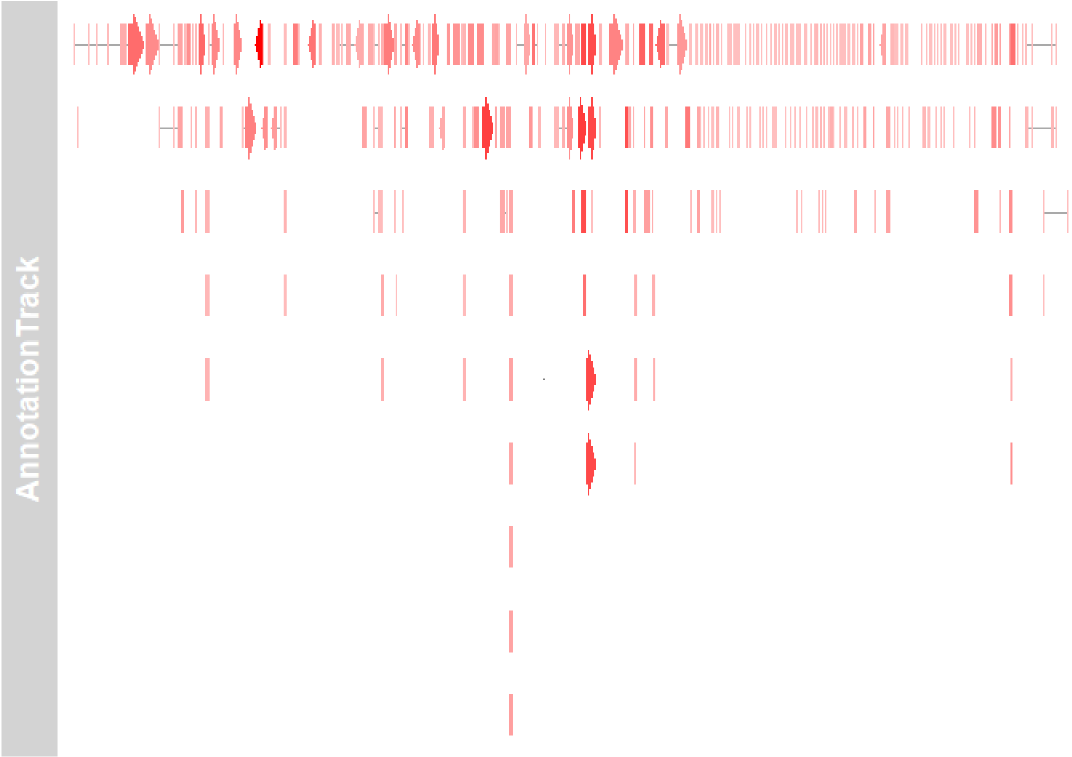
Gviz: Annotation Track

```
plotTracks(annoTrack, stacking = "dense")
```



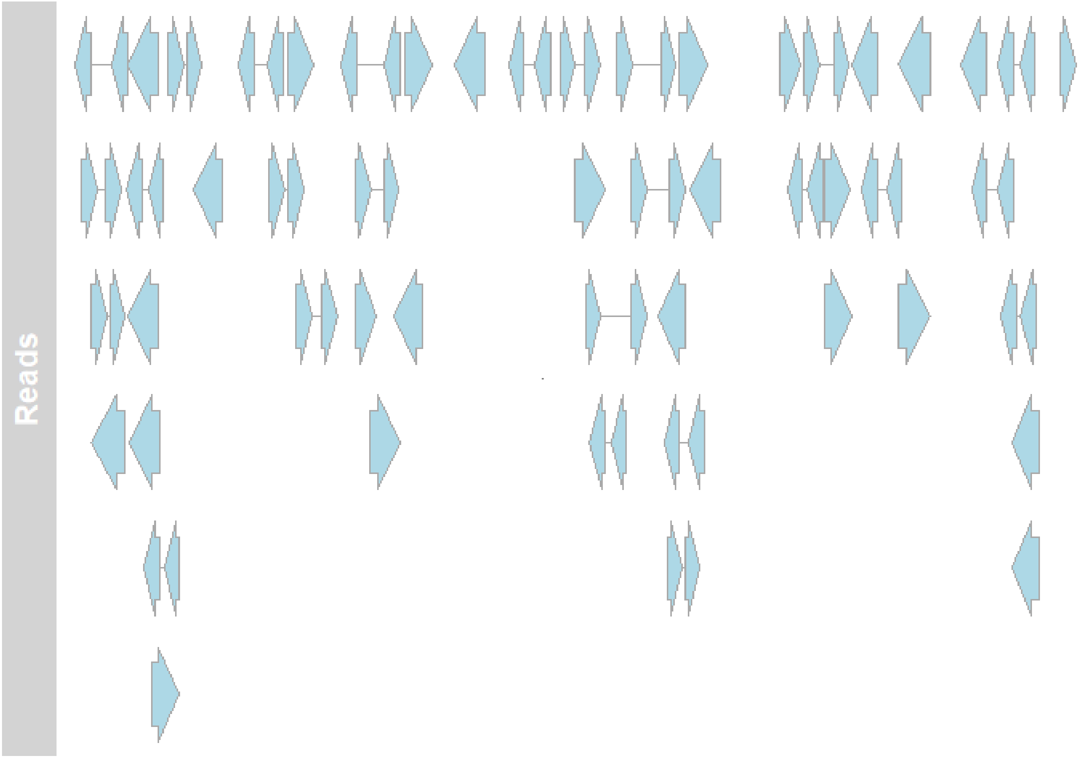
Gviz: Annotation Track

```
plotTracks(denseAnnTrack, showOverplotting = TRUE)
```



Gviz: Annotation Track

```
bamFile <- system.file("extdata/test.bam", package = "Gviz")
bamTrack <- AnnotationTrack(range = bamFile, genome = "hg19", name = "Reads",
            chromosome = "chr1")
plotTracks(bamTrack, from = 189995000, to = 1.9e+08)
```



Gviz: Gene Region Track

Purpose

- gene model annotations

Inputs

- IRanges (+ chromosome and strand as separate arguments)
- GRanges
- GRangesList
- TranscriptDb
- various file types (WIG, BedGraph, BigWig, BAM)
- direct import from Ensembl via biomaRt interface

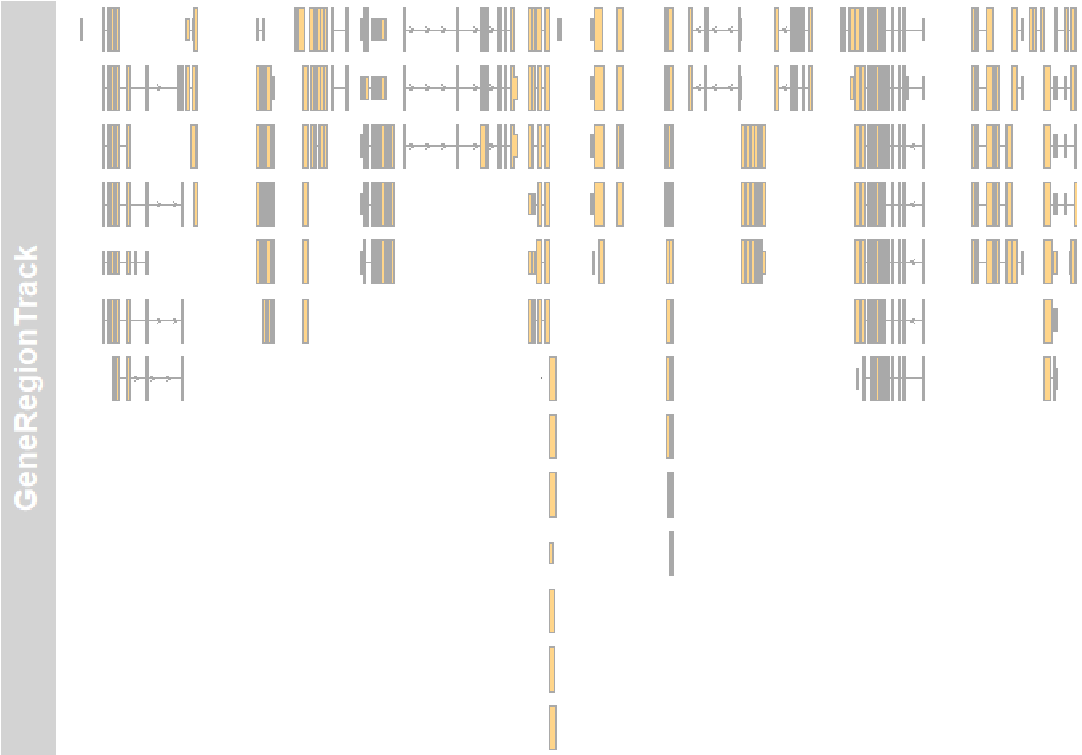
Details

- modeling of exon, transcript, and gene relationships; support for human-readable gene symbols and for coding and non-coding elements

Gviz: Gene Region Track

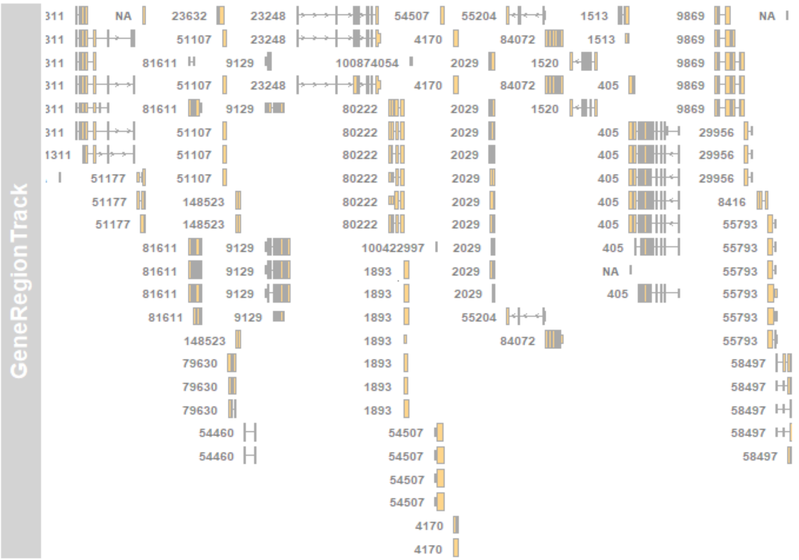
```
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
gTrack <- GeneRegionTrack(txdb, chromosome = "chr1")
plotTracks(gTrack,from = 1.5e08, to = 1.51e+08)
```

```
head(gene(gTrack))
[1] "100287102" "10028                                        02"
> head(transcript(gTrac
[1] "uc010nxq.1" "uc00                                     c001aaa.3"
> head(exon(gTrack))
[1] "uc010nxq.1_1" "uc(                     xq.1_2" "uc001aaa.3_2"
> head(symbol(gTrack))
[1] "uc010nxq.1" "uc00                                     c001aaa.3"
```

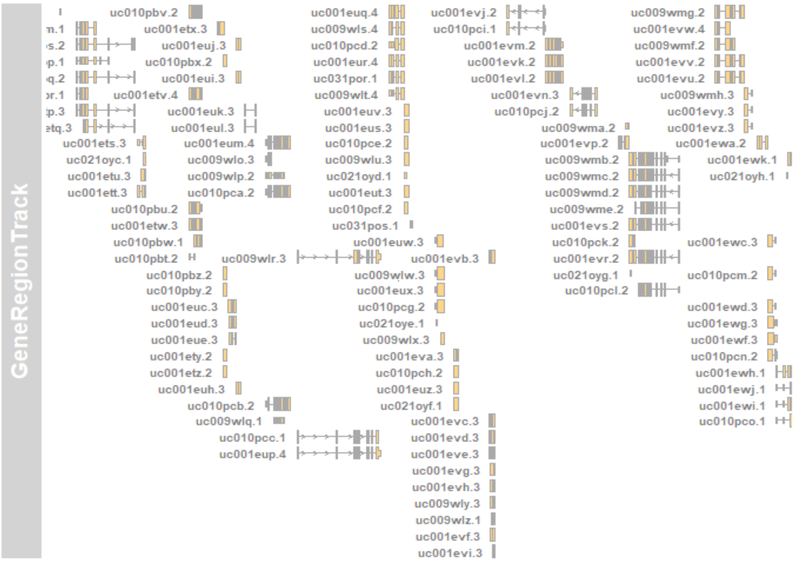


Gviz: Gene Region Track

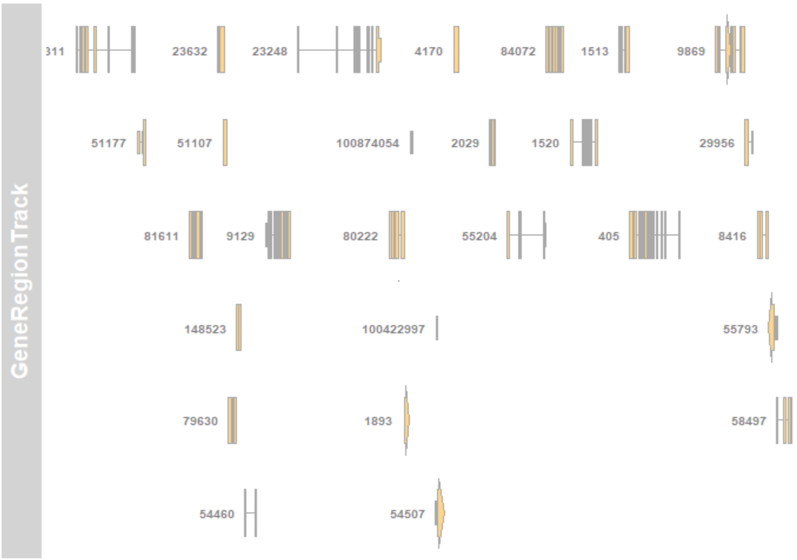
```
txdb <- TxDb.Hsapiens.UCSC.hg19.knownGene
gTrack <- GeneRegionTrack(txdb, chromosome = "chr1")
plotTracks(gTrack,from = 1.5e08, to = 1.51e+08,
           transcriptAnnotation = <type>,
            collapseTranscripts = <length>, shape = <shape>)
```



transcriptAnnotation = "gene"



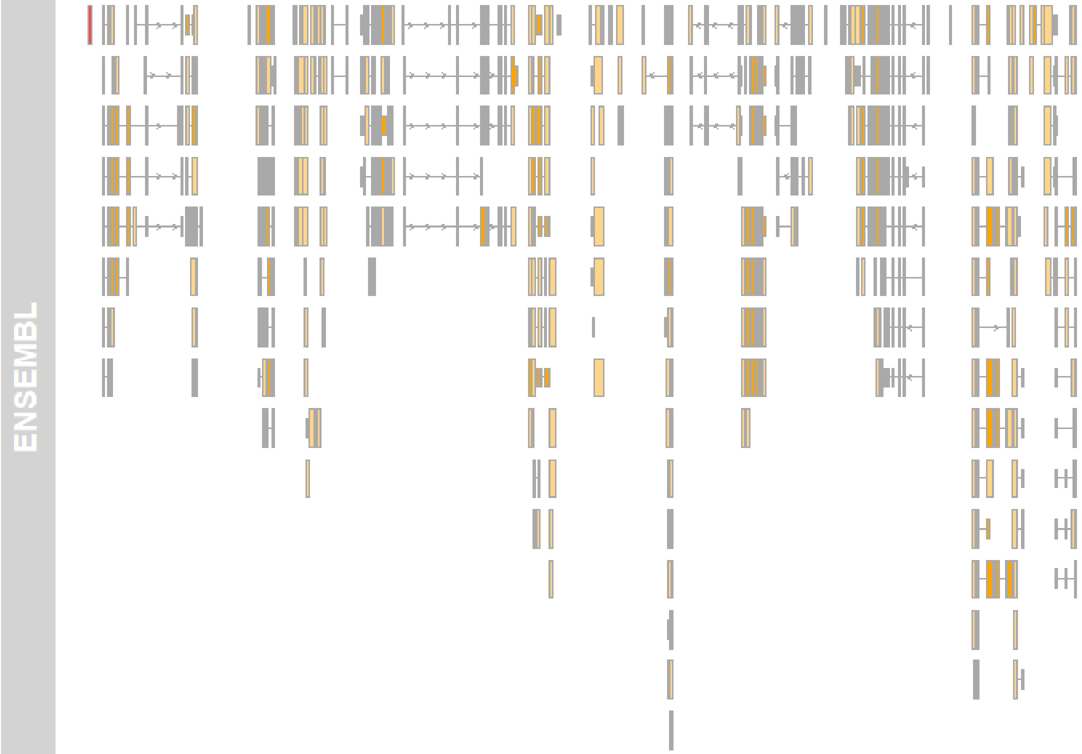
transcriptAnnotation = "transcript"



transcriptAnnotation = "gene‶
collapseTranscripts = "longest‶
shape = "arrow"

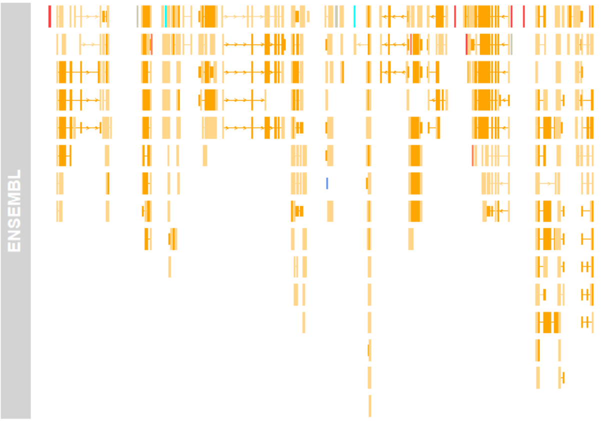
Gviz: Biomart Gene Region Track

```
biomTrack <- BiomartGeneRegionTrack(genome = "hg19",
             chromosome = "chr1", start = 1.5e08, end =
             1.51e+08, name = "ENSEMBL")
plotTracks(biomTrack, from = 1.5e08, to = 1.51e+08)
```

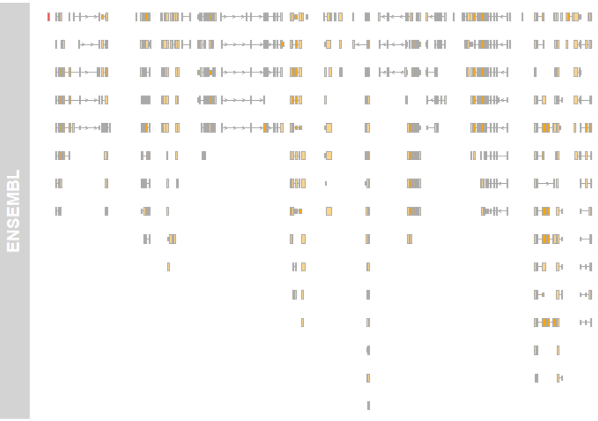


Gviz: Biomart Gene Region Track

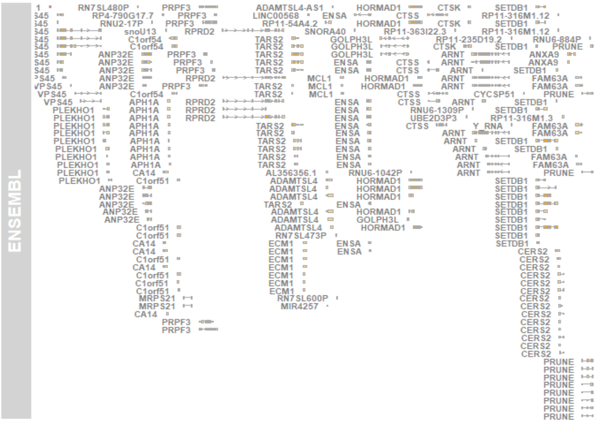
```
plotTracks(biomTrack, from = 1.5e08, to = 1.51e+08,
           col.line = <color>, col = <color>,
           stackHeight = <value>,
           transcriptAnnotation = <type>,
           collapseTranscripts = <T/F>, shape = <shape>)
```



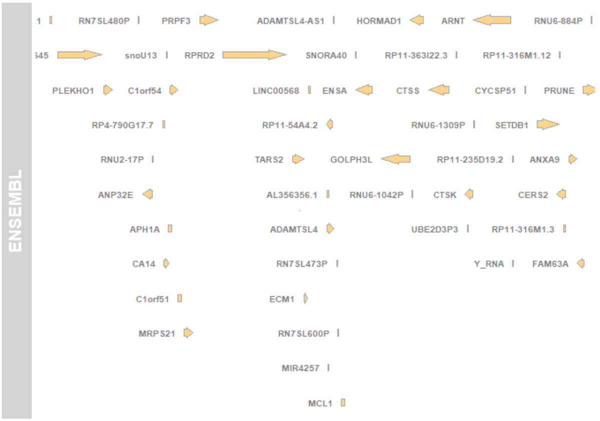
```
col.line = NULL
col = NULL
```



```
stackHeight = 0.3
```



```
stackHeight = 0.3
transcriptAnnotation="symbol"
```

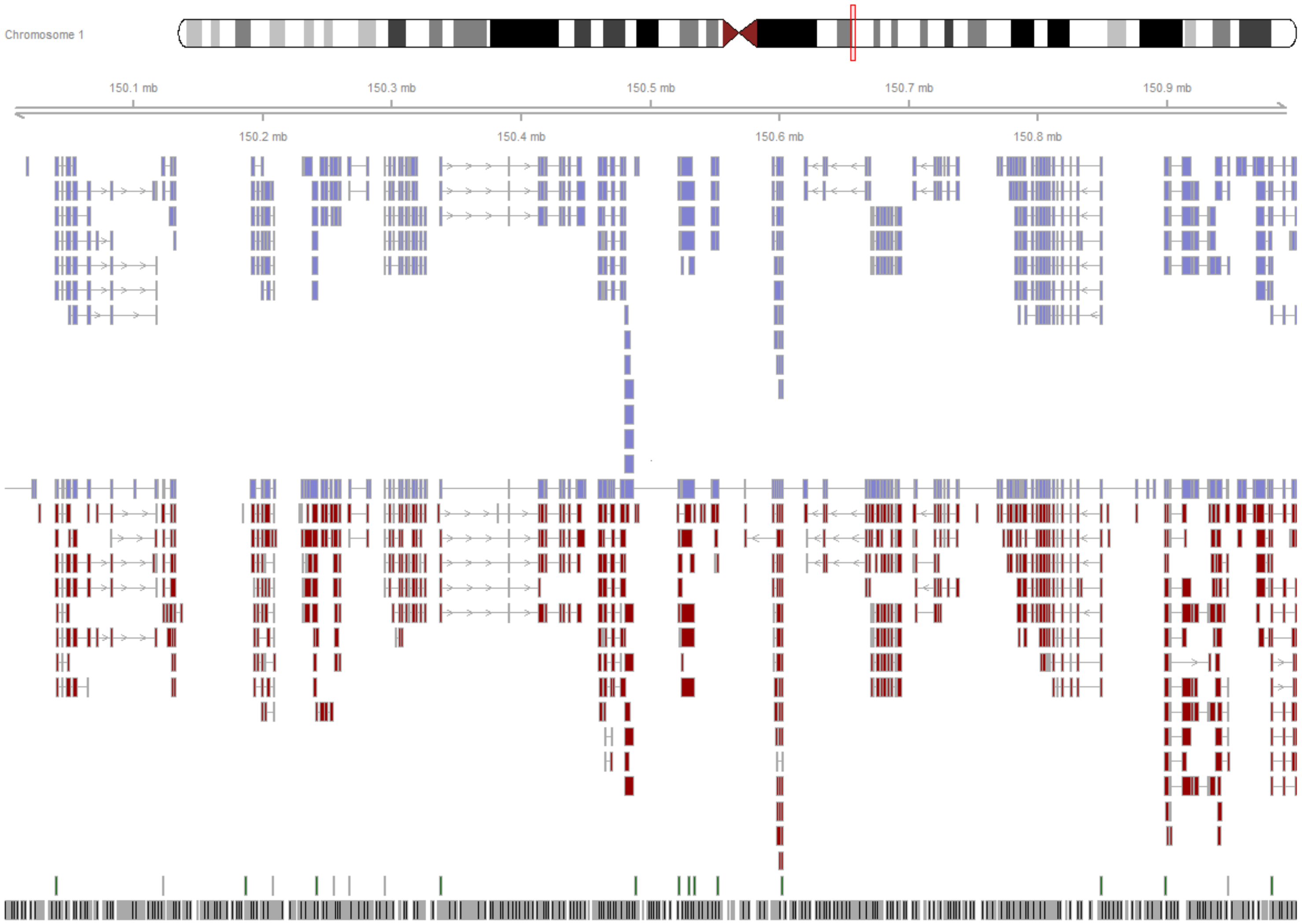


```
stackHeight = 0.3
transcriptAnnotation="symbol"
collapseTranscripts = TRUE
shape = "arrow"
```

Gviz: UCSC Track

```
knownGenes <- UcscTrack()
refGenes <- UcscTrack()
ensGenes <- UcscTrack()
cpgIslands <- UcscTrack()
snpLocations <- UcscTrack()
xTrack <- GenomeAxisTrack()
iTrack <- IdeogramTrack()
```

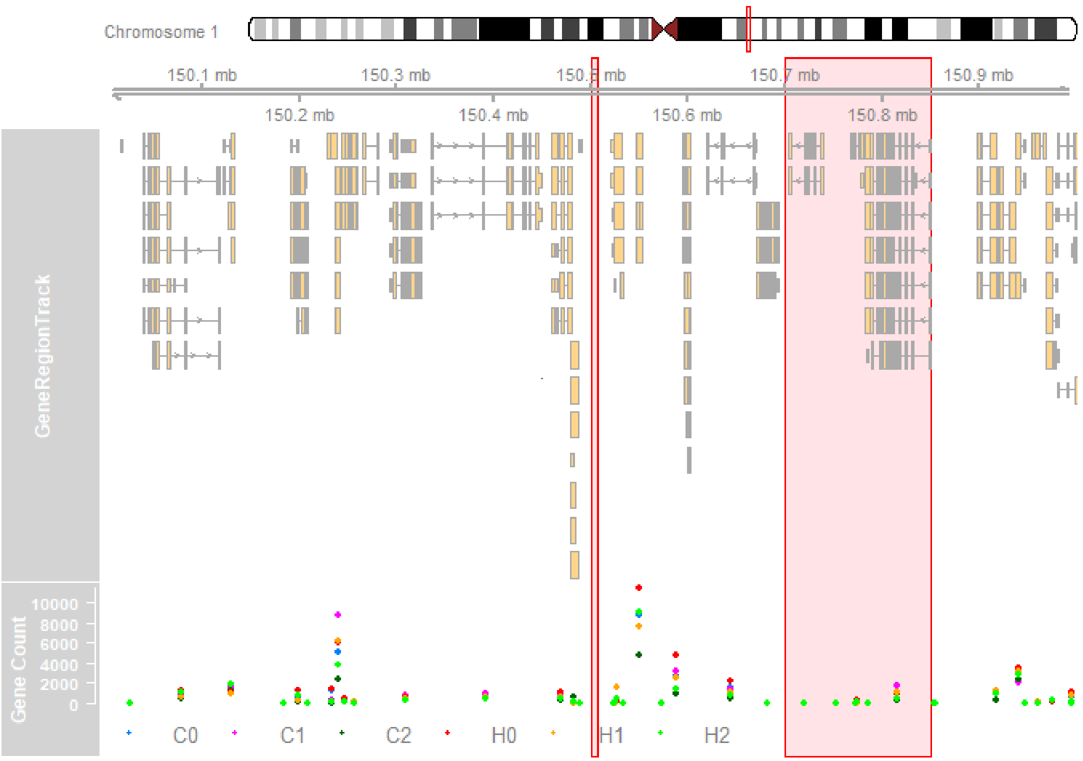
```
plotTracks(list(iTrack, xTrack, knownGenes, refGenes, ensGenes, cpgIslands, snpLocations),
from = 1.5e08, to = 1.51e+08, showTitle = FALSE)
```

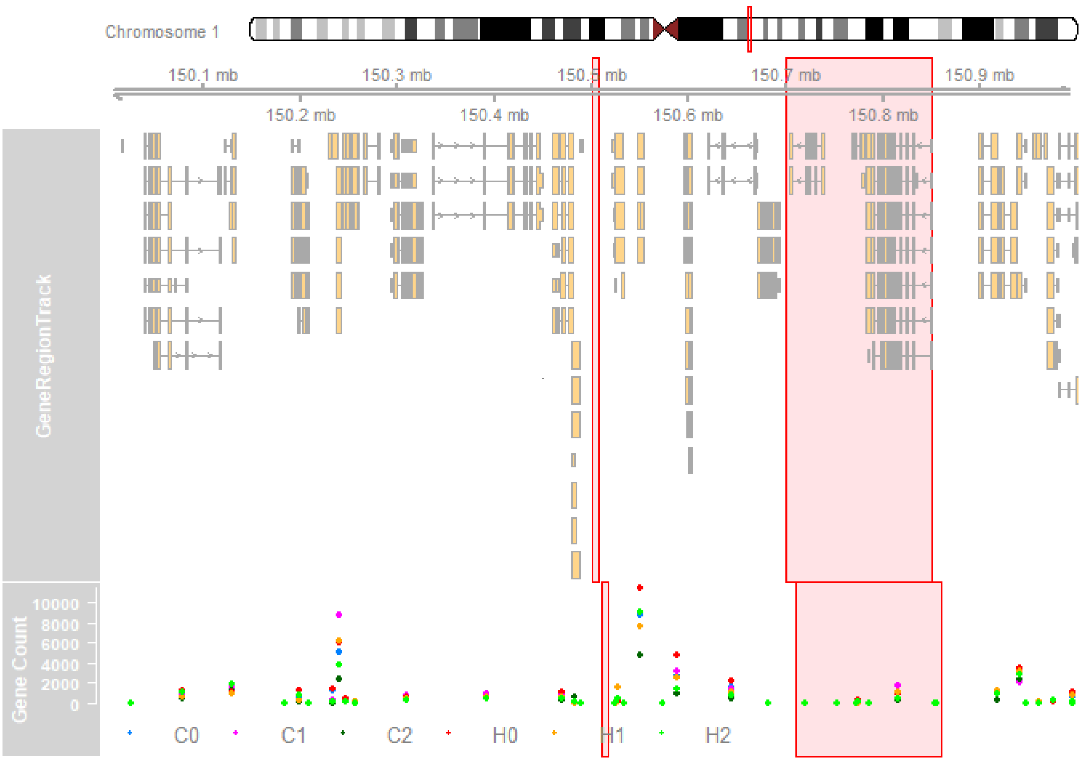


Gviz: Highlighting Regions of Interests

```
ht <- HighlightTrack(trackList = list(xTrack,
      gTrack, dTrack), chromosome = "chr1",
      start = c(150500000, 150700000), width =
      c(7000,150000))
plotTracks(list(iTrack, ht), from = 1.5e08, to =
1.51e08)
```

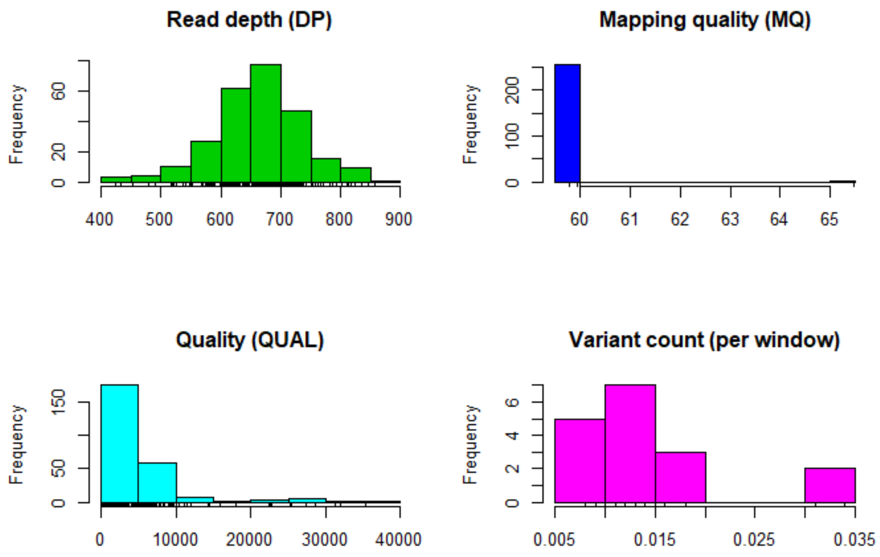
```
ht1 <- HighlightTrack(trackList = list(iTrack, xTrack,
       gTrack), chromosome = "chr1", start =
       c(150500000, 150700000), width = c(7000,150000))
ht2 <- HighlightTrack(trackList = dTrack, chromosome =
      "chr1", start = c(150510000, 150710000), width =
      c(7000,150000))
plotTracks(list(ht1, ht2), from = 1.5e08, to = 1.51e08)
```

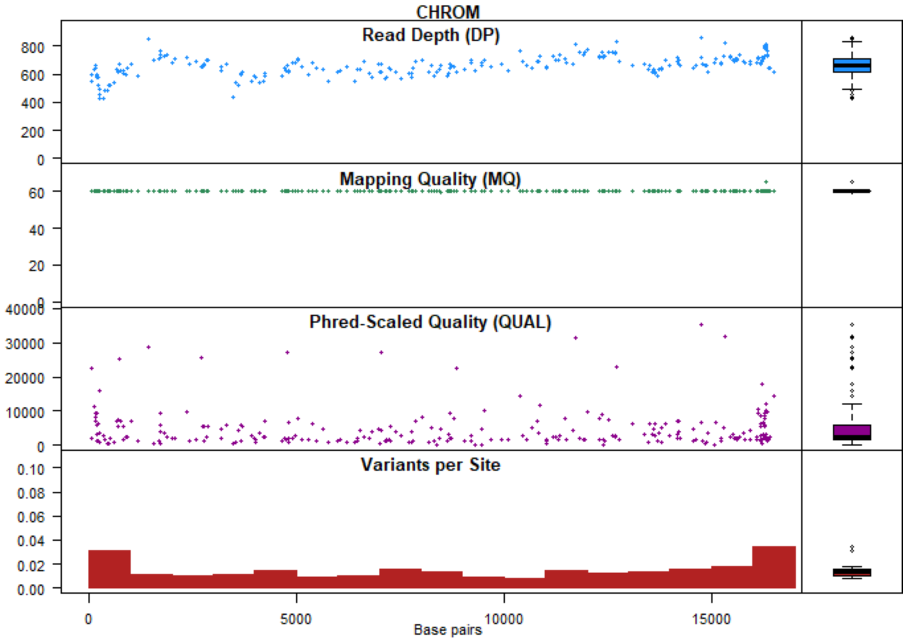


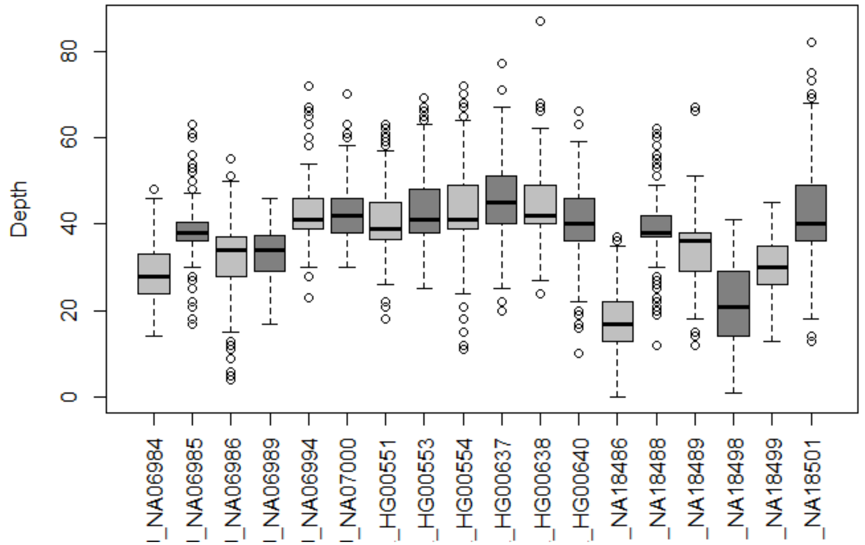


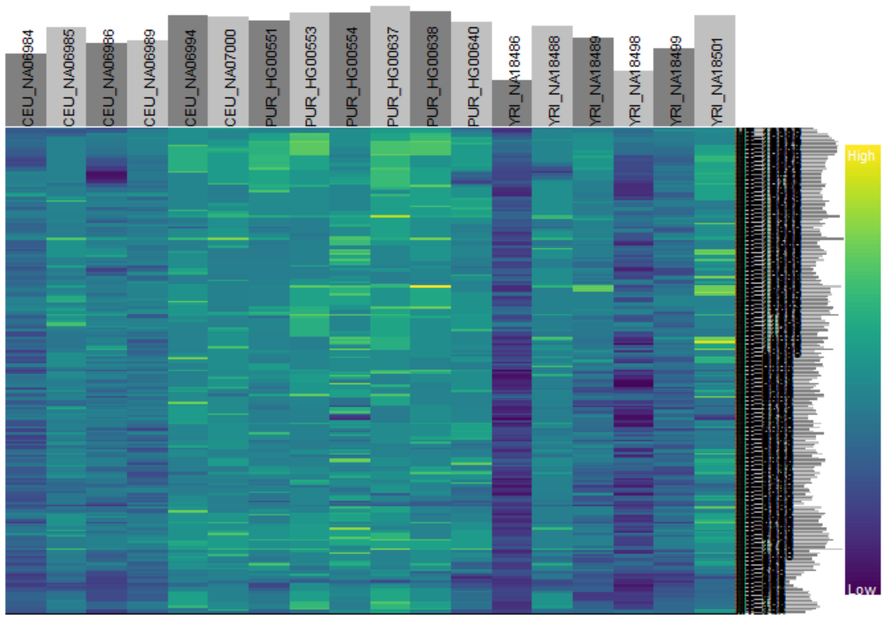
Applying ggplot2 and Gviz to VCF Data

VCF Data: Basic Plots

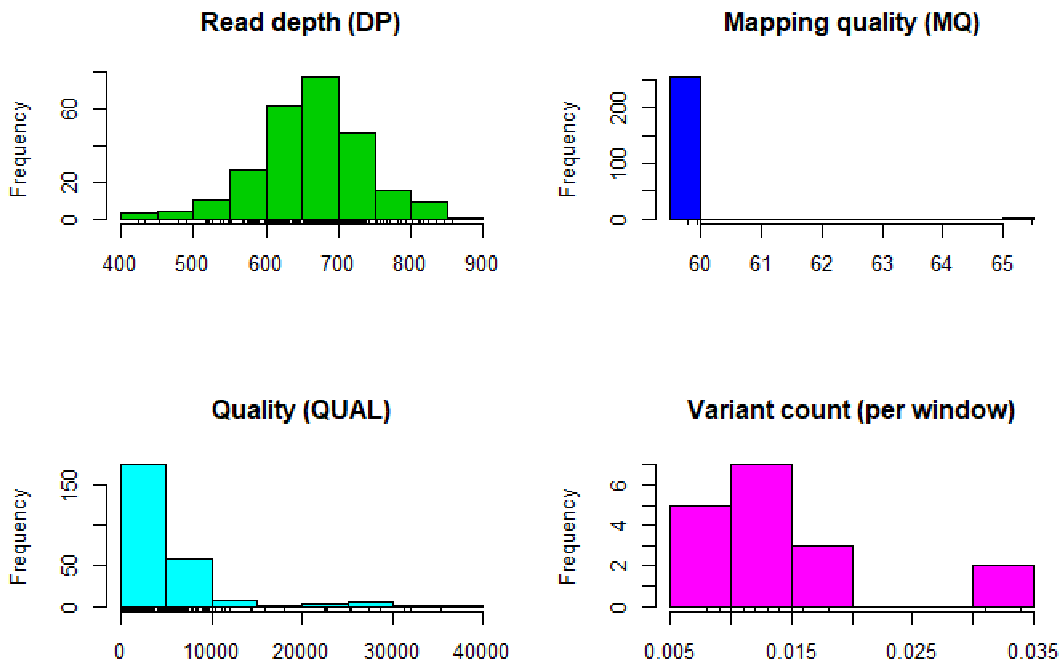


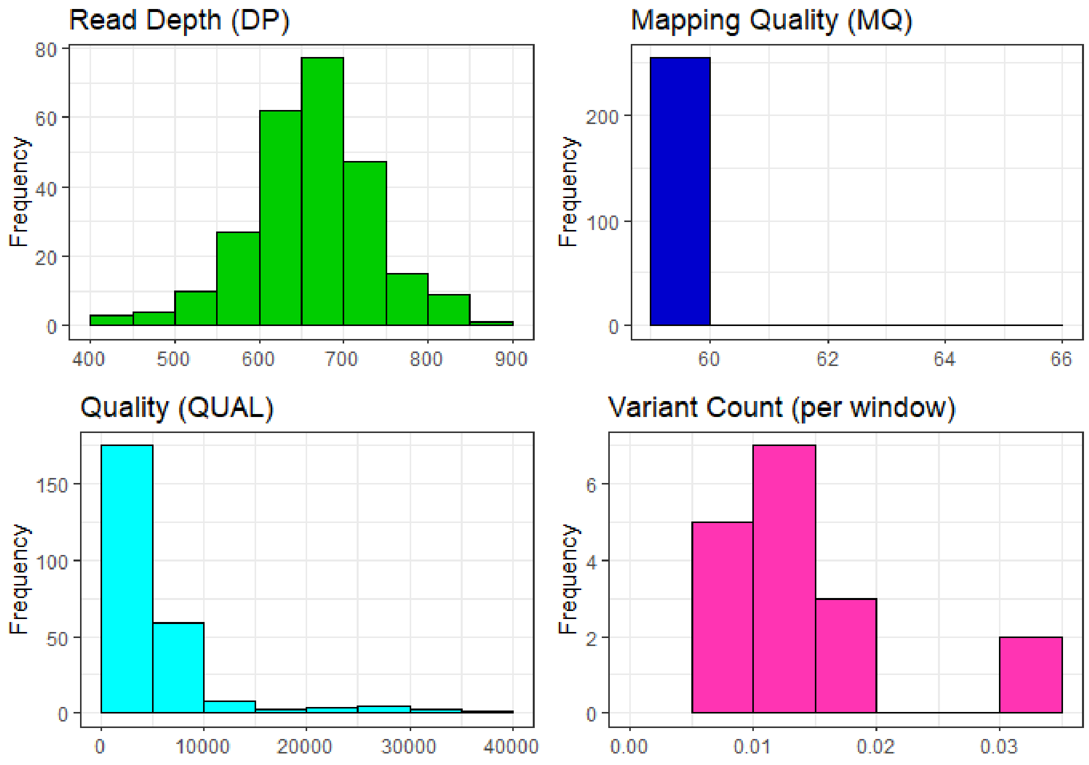




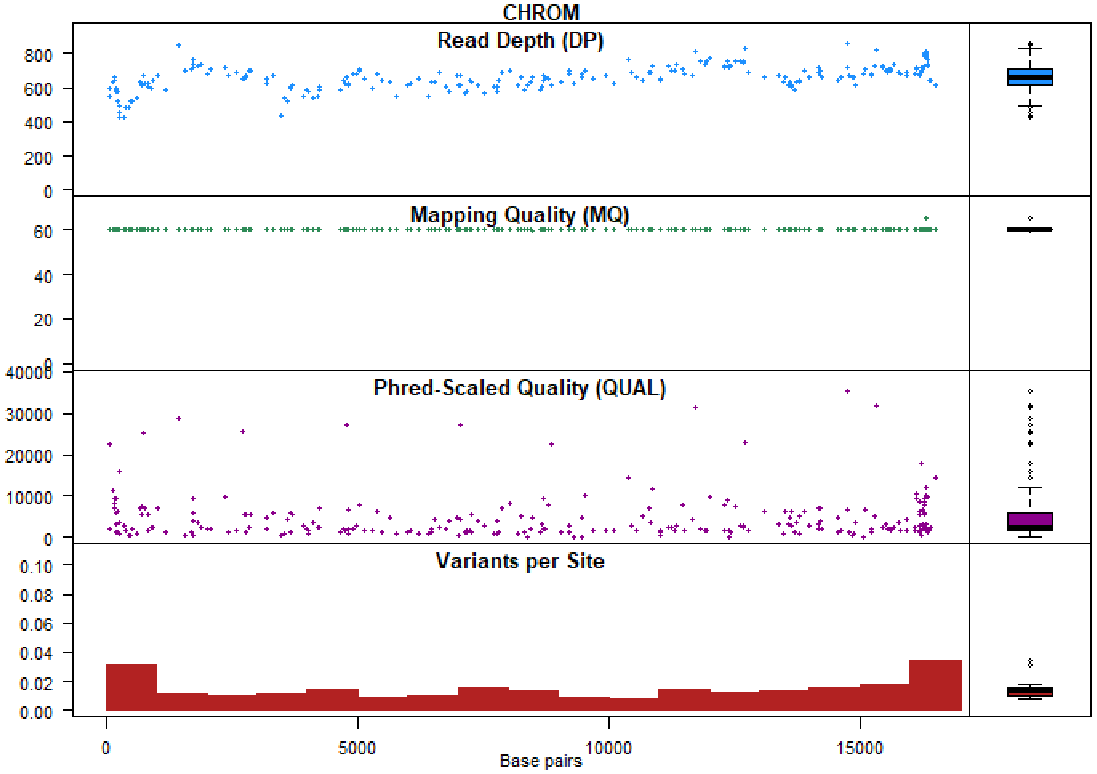


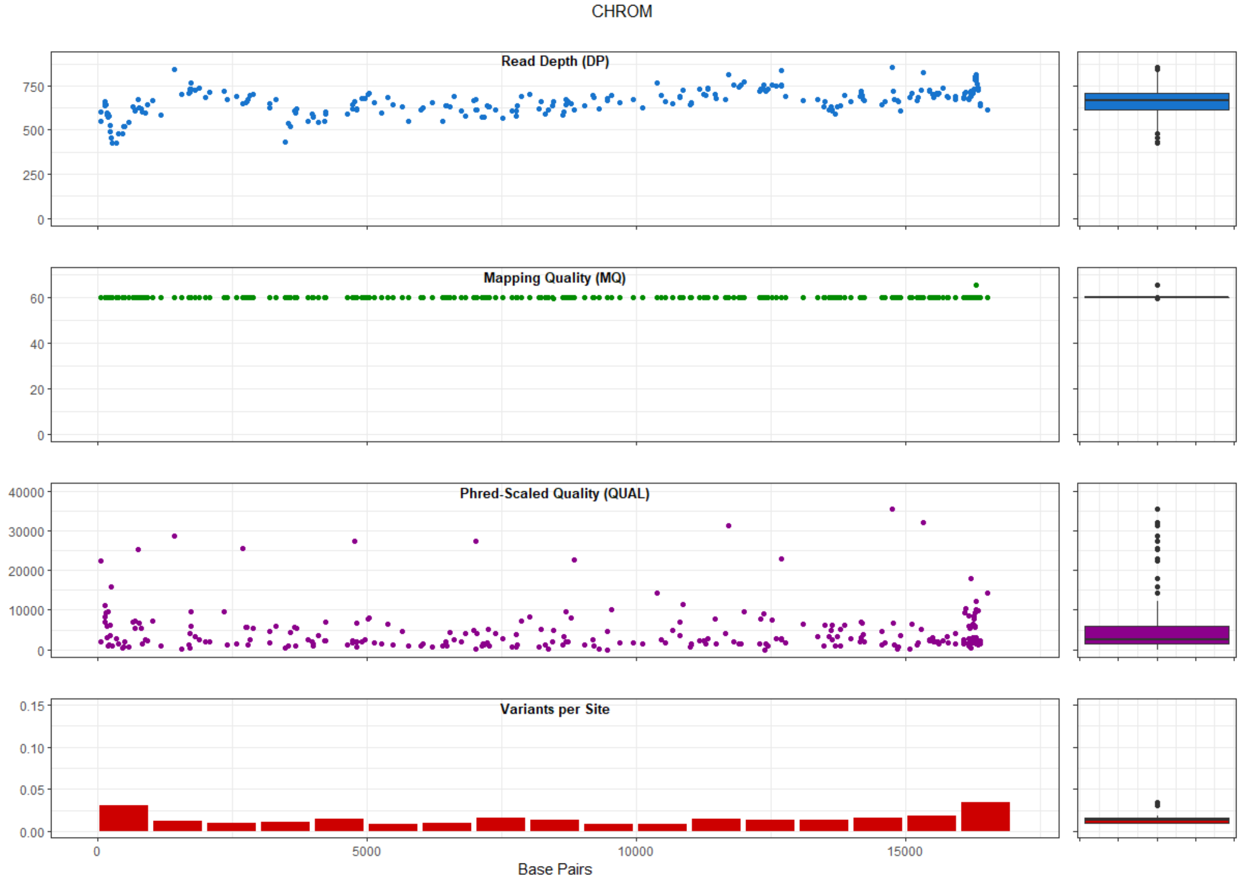
VCF Data: Basic Plots vs. ggplot2



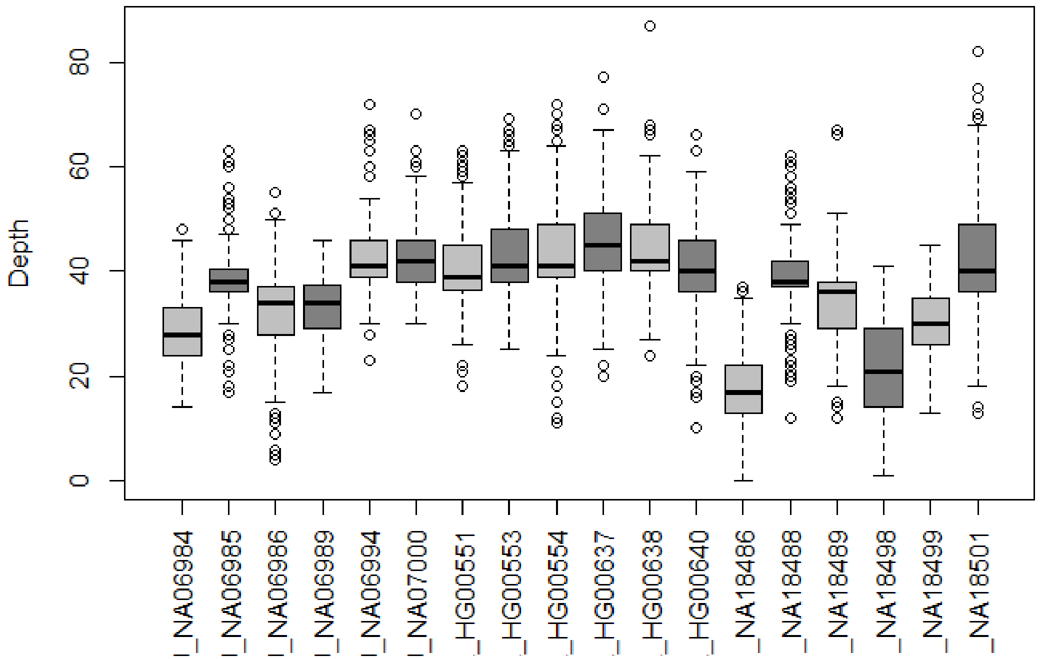


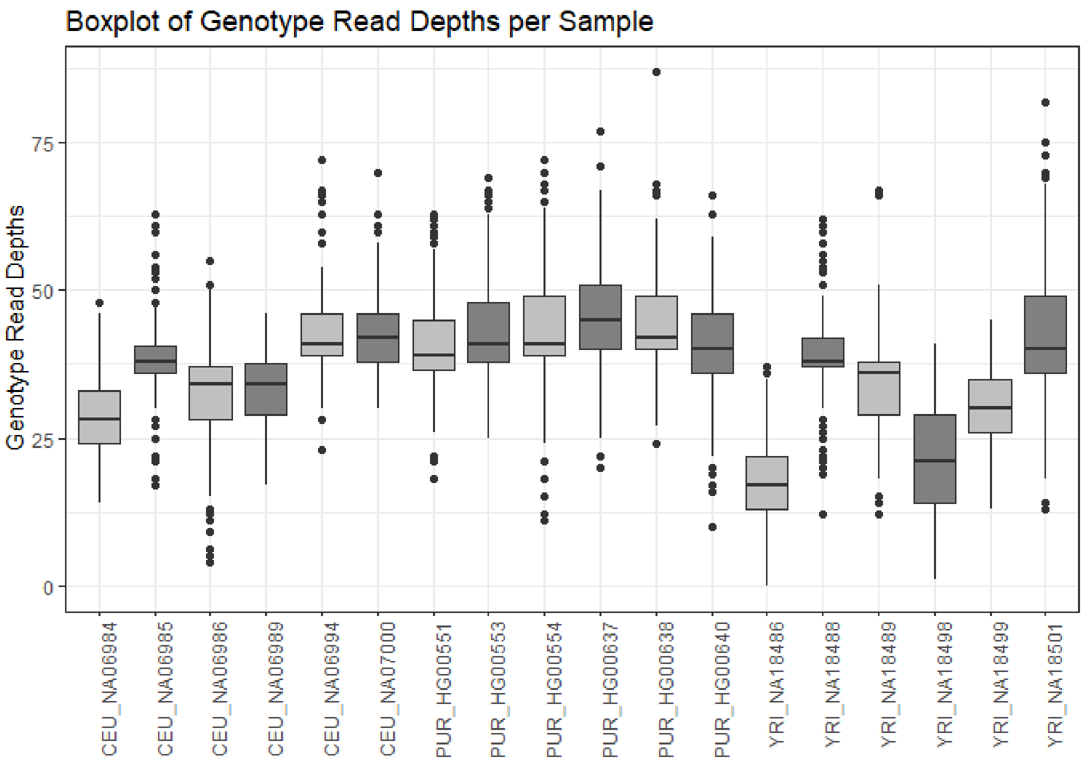
VCF Data: Basic Plots vs. ggplot2



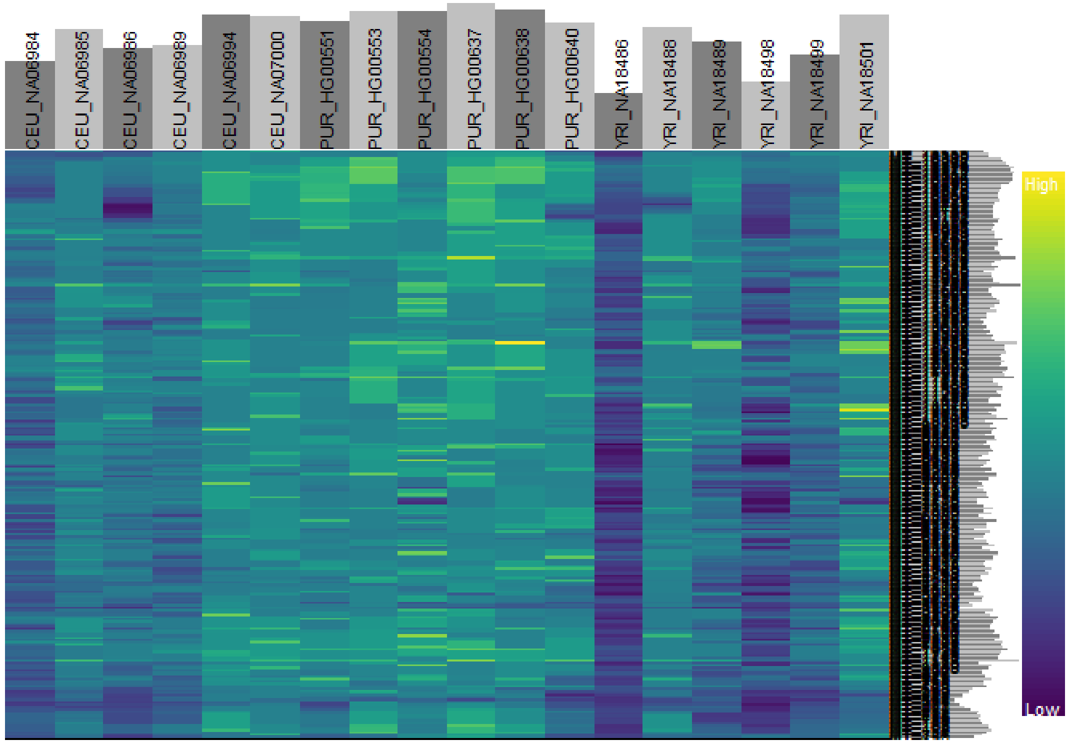


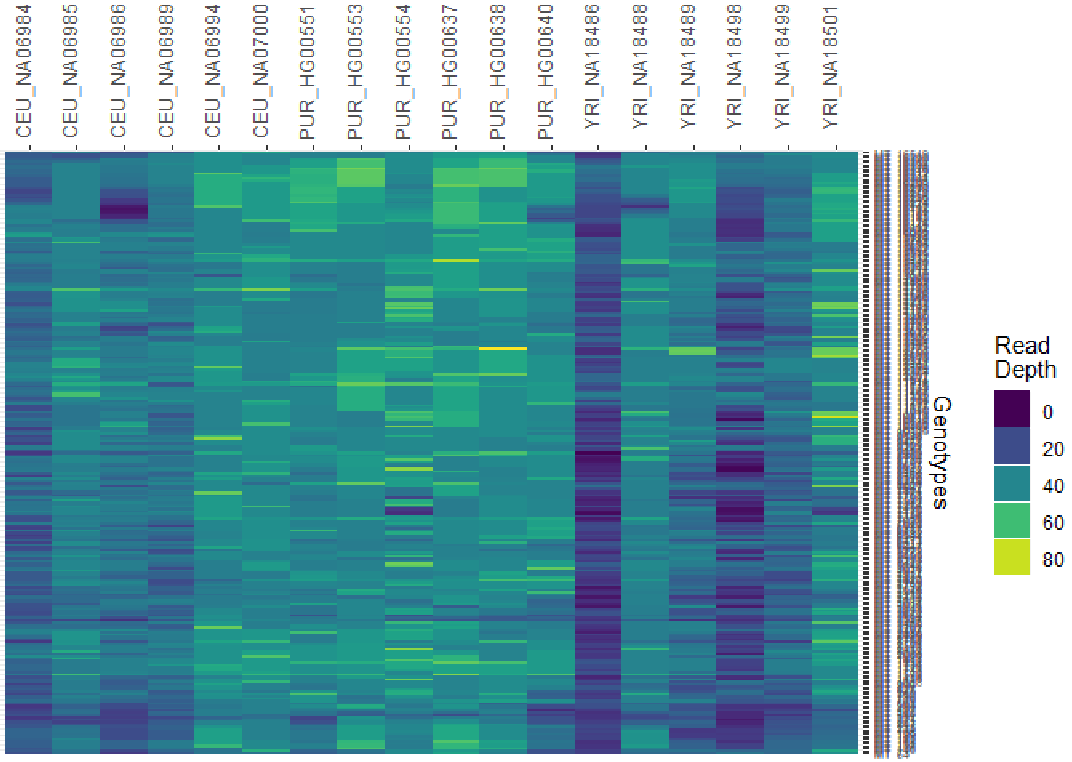
VCF Data: Basic Plots vs. ggplot2



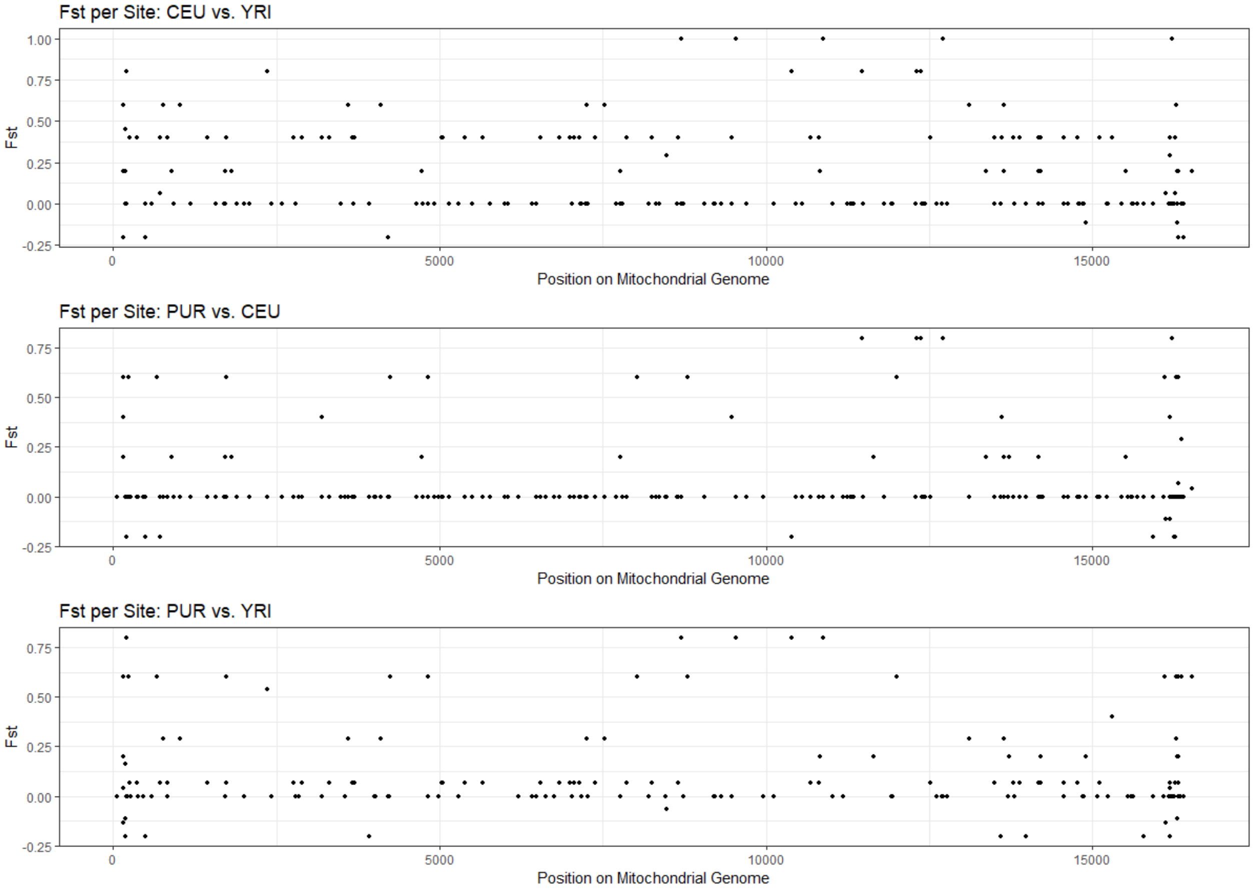


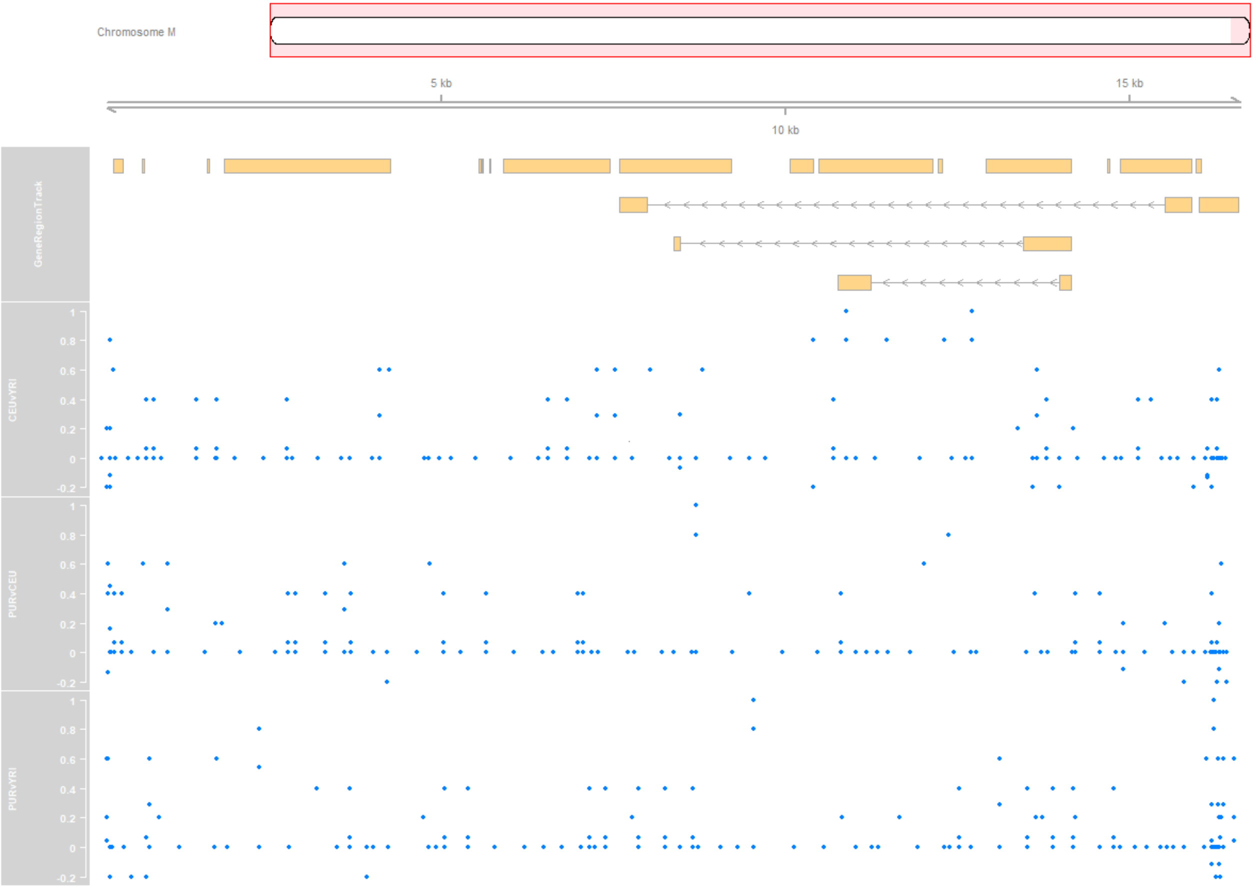
VCF Data: Basic Plots vs. ggplot2





Fst Data: Basic Plots vs. ggplot2





References and Additional Information

The ggplot2 book by Hadley Wickham
The R Graphics Cookbook by Winston Chang (examples in base plots and in ggplot2)
ggplot2 web site (http://ggplot2.org)
ggplot2 mailing list (http://goo.gl/OdW3uB), primarily for developers
https://ggplot2.tidyverse.org/#learning-ggplot2
http://r4ds.had.co.nz/data-visualisation.html
http://r4ds.had.co.nz/graphics-for-communication.html
https://www.datacamp.com/courses/data-visualization-with-ggplot2-1
https://www.rstudio.com/wp-content/uploads/2015/03/ggplot2-cheatsheet.pdf
https://learnr.wordpress.com/2010/01/26/ggplot2-quick-heatmap-plotting/
http://www.sthda.com/english/wiki/ggplot2-quick-correlation-matrix-heatmap-r-software-and-data-visualization
https://jcoliver.github.io/learn-r/008-ggplot-dendrograms-and-heatmaps.html
https://bioconductor.org/packages/devel/bioc/vignettes/Gviz/inst/doc/Gviz.pdf
http://genomicsclass.github.io/book/pages/visualizing_NGS.html
https://www.biostars.org/p/18954/
https://cran.r-project.org/web/packages/egg/vignettes/Ecosystem.html
https://bioconductor.org/packages/release/bioc/vignettes/ggbio/inst/doc/ggbio.pdf
http://www.sthda.com/english/wiki/ggbio-visualize-genomic-data
https://ggvis.rstudio.com/
https://knausb.github.io/vcfR_documentation/
https://cran.r-project.org/web/packages/vcfR/vignettes/intro_to_vcfR.html
Shared scripts from the Gilad lab at UChicago
Florian Hahne's Visualizing genomic features with the Gviz package (December 10, 2012)
Allan Just and Andrew Rundle's EPIC Short Course (June 23, 2011)
Karthik Ram's Data Visualization with R & ggplot2 (September 2, 2013)
Roger Peng's Plotting with ggplot2: Part 1 https://www.youtube.com/watch?v=HeqHMM4ziXA