

Getting Started with R for Mass Spectrometry Data Analysis

1-Day Short Course • 2020 ASMS Annual Meeting

Ryan Benz
Seer, Inc.



Jeffrey Jones
SoCal Bioinformatics



N. Heath Patterson
Vanderbilt School of Medicine



Learn the fundamentals of R and add new data analysis skills to your scientific tool belt!

Course Overview This course will cover the fundamentals of the R programming language and how it can be used to perform data analysis tasks, with a focus on the practical essentials to get you started on your way to becoming an effective R user

Target Audience Anyone new to R (and perhaps coding) who wants to learn the practical fundamentals for data analysis work

Prerequisites

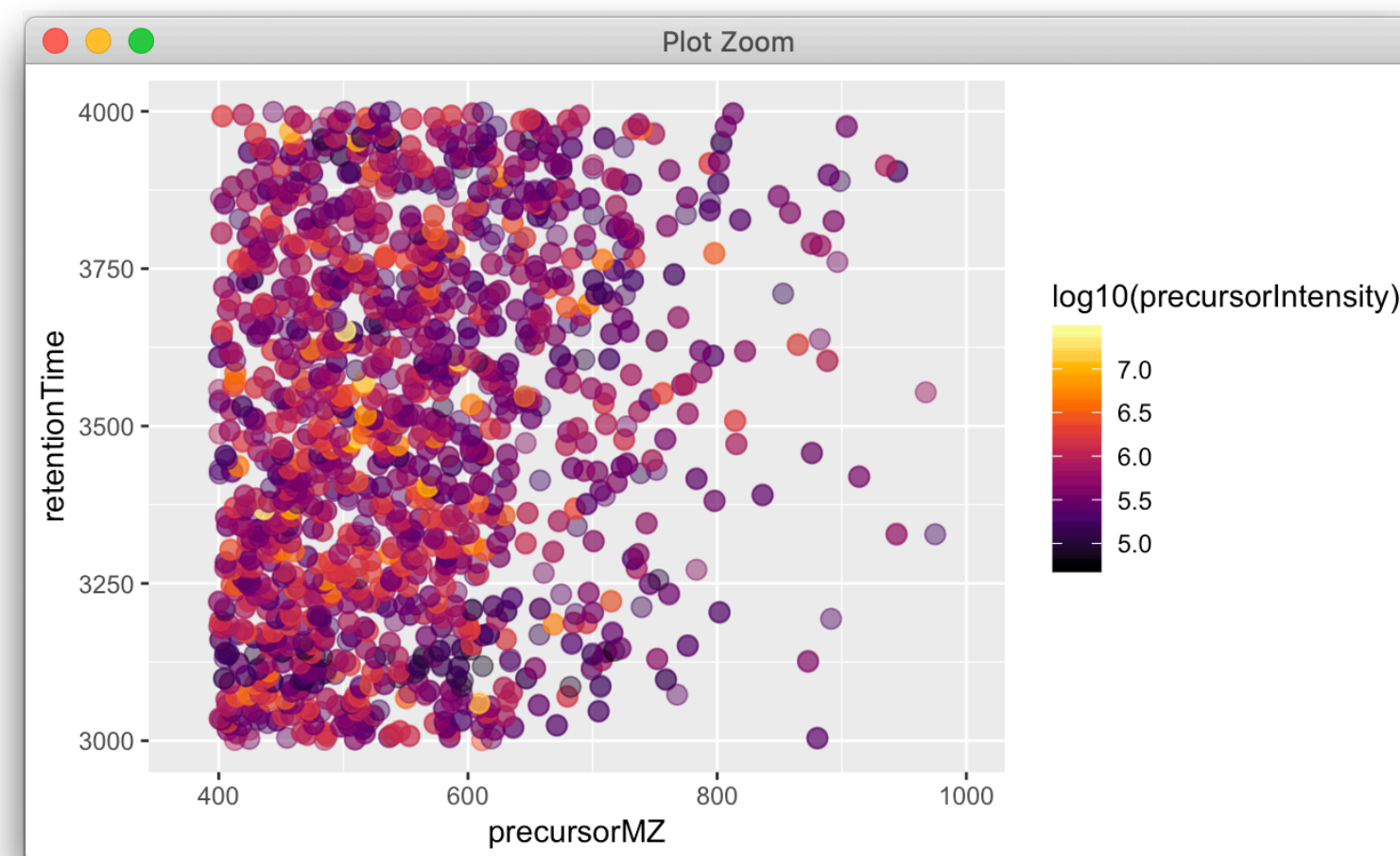
- Desire to learn one of the best data analysis tools around
- Willing to try something new and challenging
- No prior coding experience needed
- Bring a laptop for the example exercises and working sessions

The course will focus on 3 main topics

R Fundamentals

```
> 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
>
> dim(mtcars)
[1] 32 11
>
> mean(mtcars$mpg)
[1] 20.09062
>
> mtcars$hp / mtcars$mpg
 [1] 5.238095  5.238095  4.078947  5.140187  9.358289  5.801105 17.132867
 [8] 2.540984  4.166667  6.406250  6.910112 10.975610 10.404624 11.842105
[15] 19.711538 20.673077 15.646259  2.037037  1.710526  1.917404  4.511628
[22]  9.677419  9.868421 18.421053  9.114583  2.417582  3.500000  3.717105
[29] 16.708861  8.883249 22.333333  5.093458
> |
```

Data Visualization with ggplot2

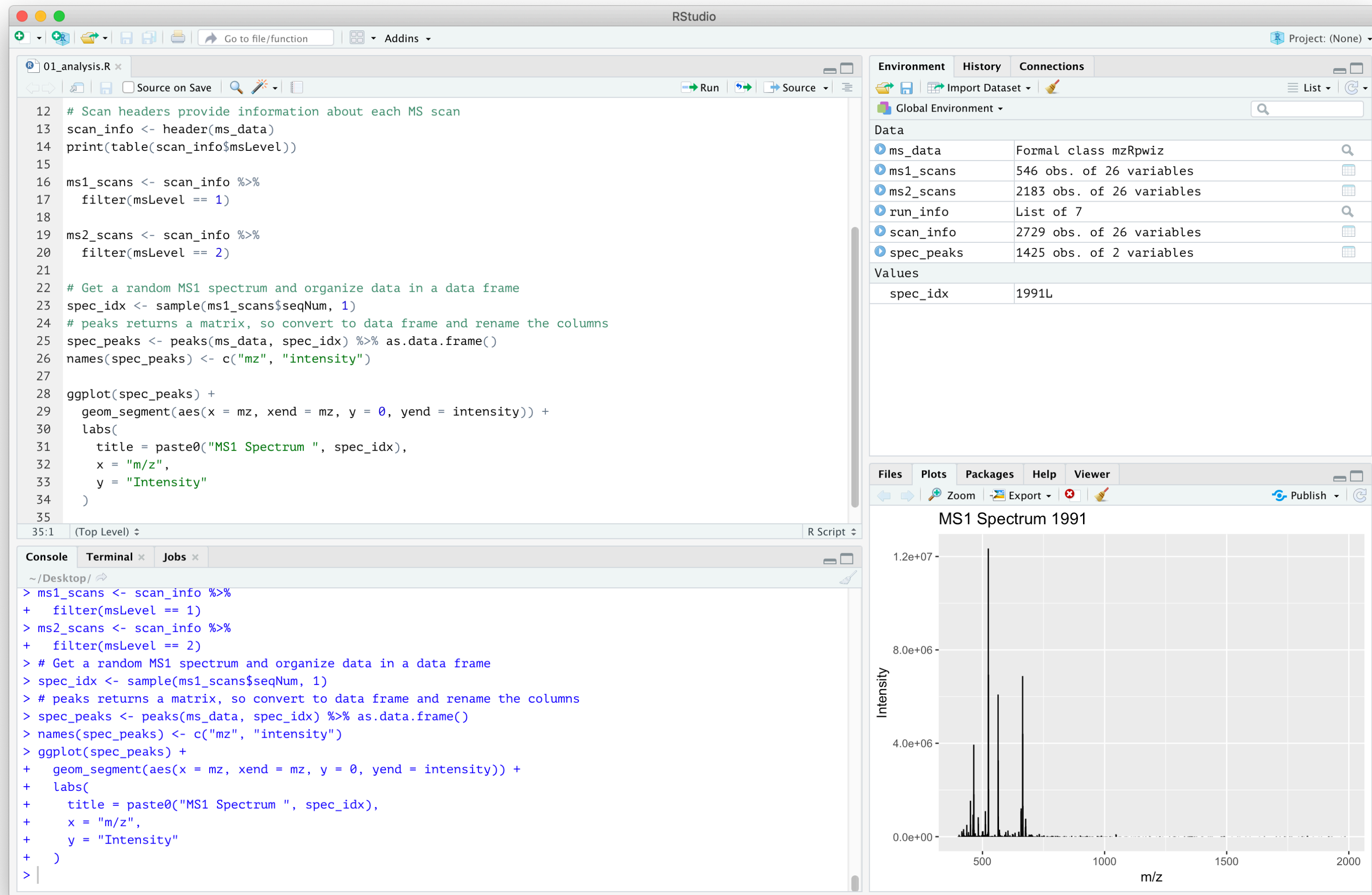


Data Analysis with the tidyverse

```
49 scan_info %>%
50   filter(msLevel == 2) %>%
51   select(precursorCharge, precursorIntensity) %>%
52   group_by(precursorCharge) %>%
53   summarize(mean_intensity = mean(precursorIntensity),
54             sd_intensity = sd(precursorIntensity),
55             median_intensity = median(precursorIntensity)) %>%
56   arrange(precursorCharge)
```

```
# A tibble: 3 x 4
  precursorCharge mean_intensity sd_intensity median_intensity
      <int>          <dbl>         <dbl>          <dbl>
1           2      873799.      1589582.       457223.
2           3      953256.      1619545.       514319.
3           4      784594.      1068287.       413015.
```


Get started with R and RStudio, learn best practices for data analysis with R



- RStudio provides an integrated environment for coding
 - code editor
 - interactive terminal
 - variables, plots, documentation
 - version control
 - *and much more*
- Streamlines all aspects of the data analysis process
- Learn best practices for organizing analyses into projects and make your work reproducible

Start making plots and visualizations from your own data

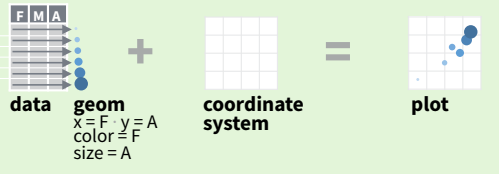
Data Visualization with ggplot2 : : CHEAT SHEET

Basics

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



To display values, map variables in the data to visual properties of the geom (**aesthetics**) like **size**, **color**, and **x** and **y** locations.



Complete the template below to build a graph.

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

required
Not required, sensible defaults supplied

ggplot(data = mpg, aes(x = cty, y = hwy)) Begins a plot that you finish by adding layers to. Add one geom function per layer.

aesthetic mappings data geom

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

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

GRAPHICAL PRIMITIVES

```
a <- ggplot(economics, aes(date, unemployment))  
b <- ggplot(seals, aes(x = long, y = lat))  
  
a + geom_blank()  
(Useful for expanding limits)  
  
b + geom_curve(aes(yend = lat + 1,  
  xend = long + 1, curvature = z)) ~ x, yend, y, yend,  
  alpha, angle, color, curvature, linetype, size  
  
a + geom_path(lineend = "butt", linejoin = "round",  
  linemitre = 1)  
x, y, alpha, color, group, linetype, size  
  
a + geom_polygon(aes(group = group))  
x, y, alpha, color, fill, group, linetype, size  
  
b + geom_rect(aes(xmin = long, ymin = lat, xmax =  
  long + 1, ymax = lat + 1)) ~ x, xmax, xmin, ymax,  
  ymin, alpha, color, fill, linetype, size  
  
a + geom_ribbon(aes(ymin = unemployment - 900,  
  ymax = unemployment + 900)) ~ x, ymax, ymin,  
  alpha, color, fill, group, linetype, size
```

LINE SEGMENTS

```
common aesthetics: x, y, alpha, color, linetype, size  
  
b + geom_abline(aes(intercept = 0, slope = 1))  
b + geom_hline(aes(yintercept = lat))  
b + geom_vline(aes(xintercept = long))  
  
b + geom_segment(aes(yend = lat + 1, xend = long + 1))  
b + geom_spoke(aes(angle = 1:1155, radius = 1))
```

ONE VARIABLE continuous

```
c <- ggplot(mpg, aes(hwy)); c2 <- ggplot(mpg)  
  
c + geom_area(stat = "bin")  
x, y, alpha, color, fill, linetype, size  
  
c + geom_density(kernel = "gaussian")  
x, y, alpha, color, fill, group, linetype, size, weight  
  
c + geom_dotplot()  
x, y, alpha, color, fill  
  
c + geom_freqpoly() x, y, alpha, color, group,  
  linetype, size  
  
c + geom_histogram(binwidth = 5) x, y, alpha,  
  color, fill, linetype, size, weight  
  
c2 + geom_qq(aes(sample = hwy)) x, y, alpha,  
  color, fill, linetype, size, weight
```

discrete

```
d <- ggplot(mpg, aes(f))  
  
d + geom_bar()  
x, alpha, color, fill, linetype, size, weight
```

TWO VARIABLES

```
continuous x, continuous y  
e <- ggplot(mpg, aes(cty, hwy))  
  
e + geom_label(aes(label = cty), nudge_x = 1,  
  nudge_y = 1, check_overlap = TRUE) x, y, label,  
  alpha, angle, color, family, fontface, hjust,  
  lineheight, size, vjust  
  
e + geom_jitter(height = 2, width = 2)  
x, y, alpha, color, fill, shape, size  
  
e + geom_point(), x, y, alpha, color, fill, shape,  
  size, stroke  
  
e + geom_quantile(), x, y, alpha, color, group,  
  linetype, size, weight  
  
e + geom_rug(sides = "bl"), x, y, alpha, color,  
  linetype, size  
  
e + geom_smooth(method = lm), x, y, alpha,  
  color, fill, group, linetype, size, weight  
  
e + geom_text(aes(label = cty), nudge_x = 1,  
  nudge_y = 1, check_overlap = TRUE) x, y, label,  
  alpha, angle, color, family, fontface, hjust,  
  lineheight, size, vjust
```

discrete x, continuous y

```
f <- ggplot(mpg, aes(class, hwy))  
  
f + geom_col(), x, y, alpha, color, fill, group,  
  linetype, size  
  
f + geom_boxplot(), x, y, lower, middle, upper,  
  ymax, ymin, alpha, color, fill, group, linetype,  
  shape, size, weight  
  
f + geom_dotplot(binaxis = "y", stackdir =  
  "center"), x, y, alpha, color, fill, group  
  
f + geom_violin(scale = "area"), x, y, alpha, color,  
  fill, group, linetype, size, weight
```

discrete x, discrete y

```
g <- ggplot(diamonds, aes(cut, color))  
  
g + geom_count(), x, y, alpha, color, fill, shape,  
  size, stroke
```

THREE VARIABLES

```
sealsSz <- with(seals, sqrt(delta_long^2 + delta_lat^2)); l <- ggplot(seals, aes(long, lat))  
  
l + geom_contour(aes(z = z))  
x, y, z, alpha, colour, group, linetype,  
  size, weight
```

continuous bivariate distribution

```
h <- ggplot(diamonds, aes(carat, price))  
  
h + geom_bin2d(binwidth = c(0.25, 500))  
x, y, alpha, color, fill, linetype, size, weight  
  
h + geom_density2d()  
x, y, alpha, colour, group, linetype, size  
  
h + geom_hex()  
x, y, alpha, colour, fill, size
```

continuous function

```
i <- ggplot(economics, aes(date, unemployment))  
  
i + geom_area()  
x, y, alpha, color, fill, linetype, size  
  
i + geom_line()  
x, y, alpha, color, group, linetype, size  
  
i + geom_step(direction = "hv")  
x, y, alpha, color, group, linetype, size
```

visualizing error

```
df <- data.frame(grp = c("A", "B"), fit = 4:5, se = 1:2)  
j <- ggplot(df, aes(grp, fit, ymin = fit-se, ymax = fit+se))  
  
j + geom_crossbar(fatten = 2)  
x, y, ymax, ymin, alpha, color, fill, group, linetype,  
  size  
  
j + geom_errorbar(), x, y, ymax, ymin, alpha, color,  
  group, linetype, size, width (also  
  geom_errorbarh())  
  
j + geom_linerange()  
x, y, ymin, ymax, alpha, color, group, linetype, size  
  
j + geom_pointrange()  
x, y, ymin, ymax, alpha, color, fill, group, linetype,  
  shape, size
```

maps

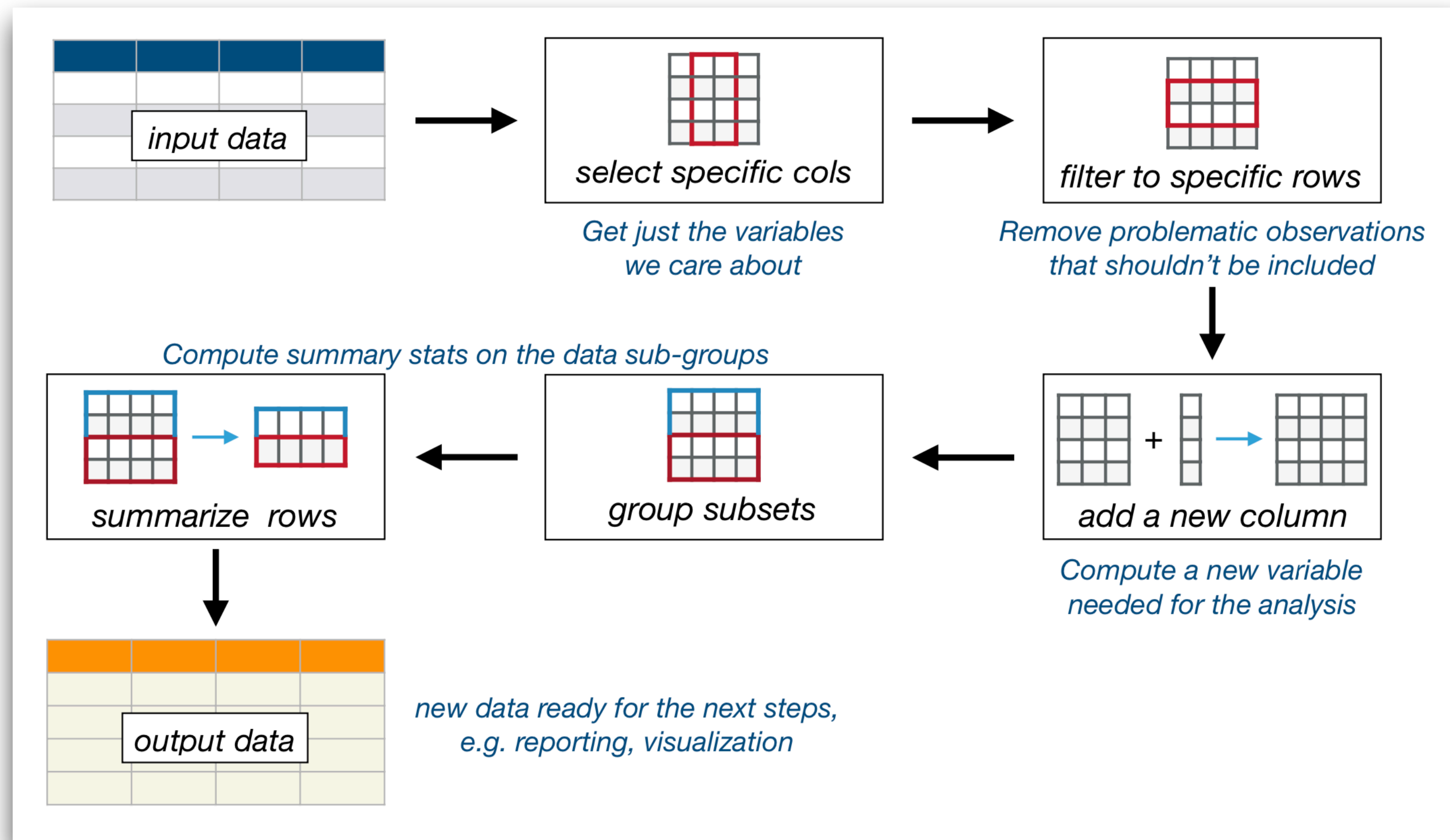
```
data <- data.frame(murder = USArrests$Murder,  
  state = tolower(rownames(USArrests)))  
map <- map_data("state")  
k <- ggplot(data, aes(fill = murder))  
  
k + geom_map(aes(map_id = state), map = map)  
+ expand_limits(x = map$long, y = map$lat),  
  map_id, alpha, color, fill, linetype, size
```



<https://github.com/rstudio/cheatsheets/blob/master/data-visualization-2.1.pdf>

- ggplot2 is a powerful plotting package in R that allows you to easily make plots from your data
- Provides the fundamental pieces to make all kinds of visualizations, from simple to complex
- Quickly make plots for exploratory data analysis
- Fine tune the look and style of your plots, make them ready for presentation and publication

Learn about tidy data and how the tidyverse collection of packages can streamline your analyses



- Learn how tidy data and the tidyverse can make your data analysis code easier to read, write and understand
- Learn about dplyr, an R package that allows you to create streamlined data analysis pipelines
- The tidyverse covers all steps of the analysis process, from data import & cleaning to visualization and final reporting

Additional course info...

- This course is designed to help participants get over the initial hurdles new R users often face
- Mass spectrometry and related data sets / examples will be used during the course to help put the material into context
- Information on any required pre-course set-up (i.e. software to install) will be provided in advance
- If you've been thinking about trying to learn R, this course is a great way to get started