Learning *R*

Carl James Schwarz

StatMathComp Consulting by Schwarz
cschwarz.stat.sfu.ca @ gmail.com

Plotting with ggplot2 - Introduction

Table of Contents I

Plotting using *ggplot2*

Plotting with *ggplot2*

Plotting paradigms:

1. Base *R* - AVOID
   - Pen-on-paper; once plotted, cannot be changed;
   - You need to do everything, e.g. legends; grouping data
2. *lattice* graphics - AVOID
   - Extension to Base *R* plotting routines to handle some features (e.g. legends) automatically.
   - Lack of formal graphical models makes it hard to extend
3. *ggplot2* graphics - RECOMMENDED
   - Based on Grammar of Graphs by Cleveland.
   - Build up a graph piece by piece
   - Display the graph only once all parts are added. Axes, etc. automatically adjust once all layers added

`http://ggplot2.org` is the main reference site.
R Graphics Cookbook by Winston Chang.
ggplot2: Elegant Graphics for Data Analysis by Hadley Wickham

Plotting with *ggplot2*

Elements of a graph:

- **data** and **aesthetics**: color, size, title, etc.
- **geometry**: what you see: points, lines, polygons, etc.
- **statistics**: transforming (summarizing) data: bins, fitting lines
- **scales**: map data space to aesthetic space: which points get which color; mpg to distance from origin, etc.
- **co-ordinate system**: cartesian co-ordinates or polar co-ordinates, etc.
- **faceting**: multiple plots on same graph

Only suitable for static graphs - no interactive or dynamic graphs. Visit `http://r-statistics.co/Top50-Ggplot2-Visualizations-MasterList-R-Code.html` for some nice graphic templates.

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat; draw a smoother; fit a line; add confidence bands to the fit

```
1
2  library(ggplot2)
3
4  cereal <- read.csv('cereal.csv',
5                     header=TRUE, as.is=TRUE,
6                     strip.white=TRUE)
7  cereal[1:5,]
8  cereal$shelfF <- factor(cereal$shelf)  # categorical variab
```

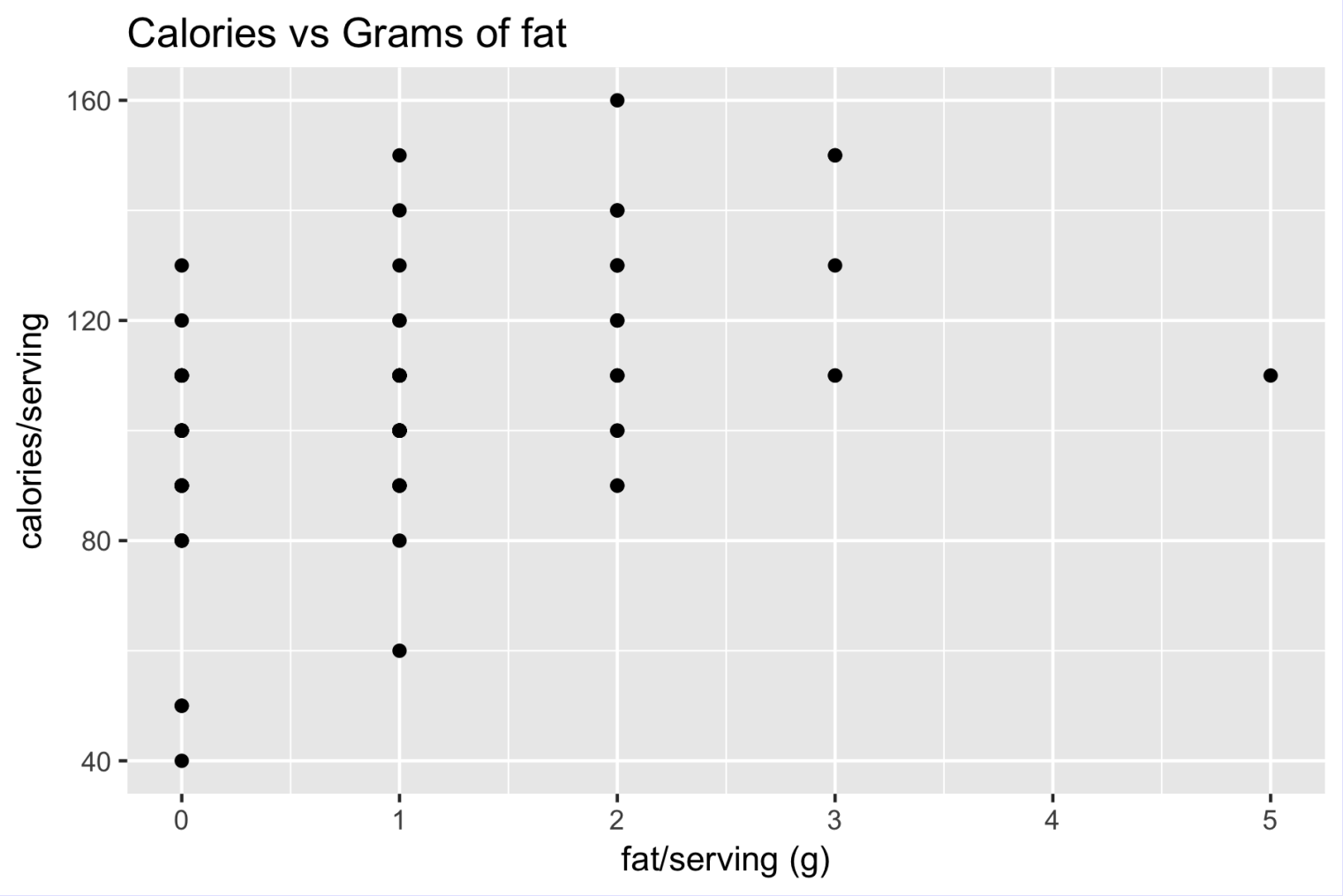
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat

```
1  library(ggplot2)
2
3  # Start with basic plotting
4  newplot <- ggplot(cereal, aes(x=fat, y=calories))+
5     ggtitle("Calories vs Grams of Fat")+
6     xlab("Fat/serving (g)")+ylab("Calories/serving")+
7     geom_point()
8  newplot
9  ggsave(plot=newplot, file='cereal-calories-fat-base.png',
10        h=4, w=6, units="in", dpi=300)
```

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat



Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat; jittering

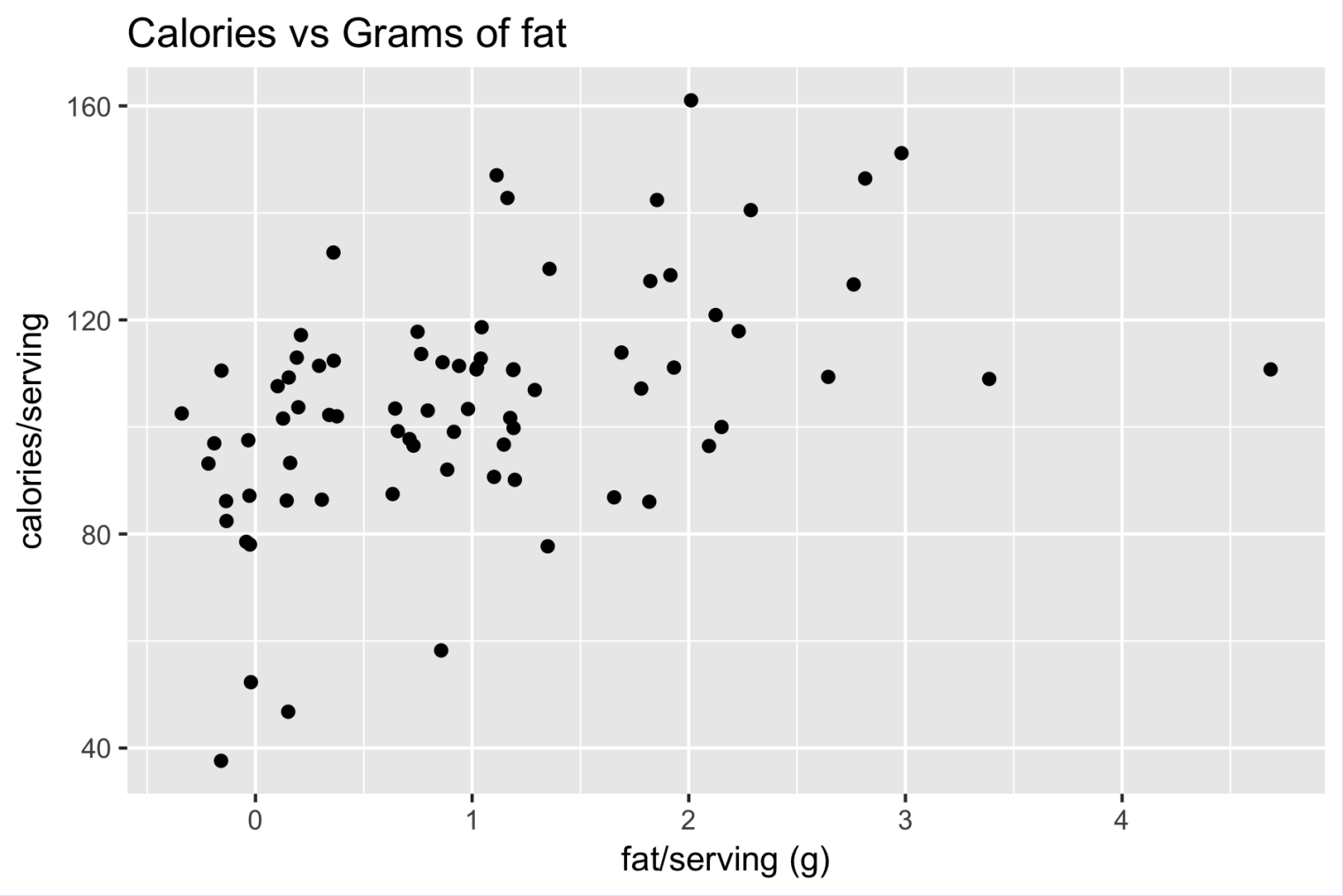
```
1
2  # jittering
3  newplot <- ggplot(cereal, aes(x=fat, y=calories))+
4     ggtitle("Calories vs Grams of Fat")+
5     xlab("Fat/serving (g)")+ylab("Calories/serving")+
6     geom_jitter() #too much
```

You can control the amount of jittering, color, size of points

```
1       ... geom_jitter(position=position_jitter(w=.1, h=.1))
2       ... geom_jitter(position=position_jitter(w=.1, h=.1),
3                       color='red', size=4))
```

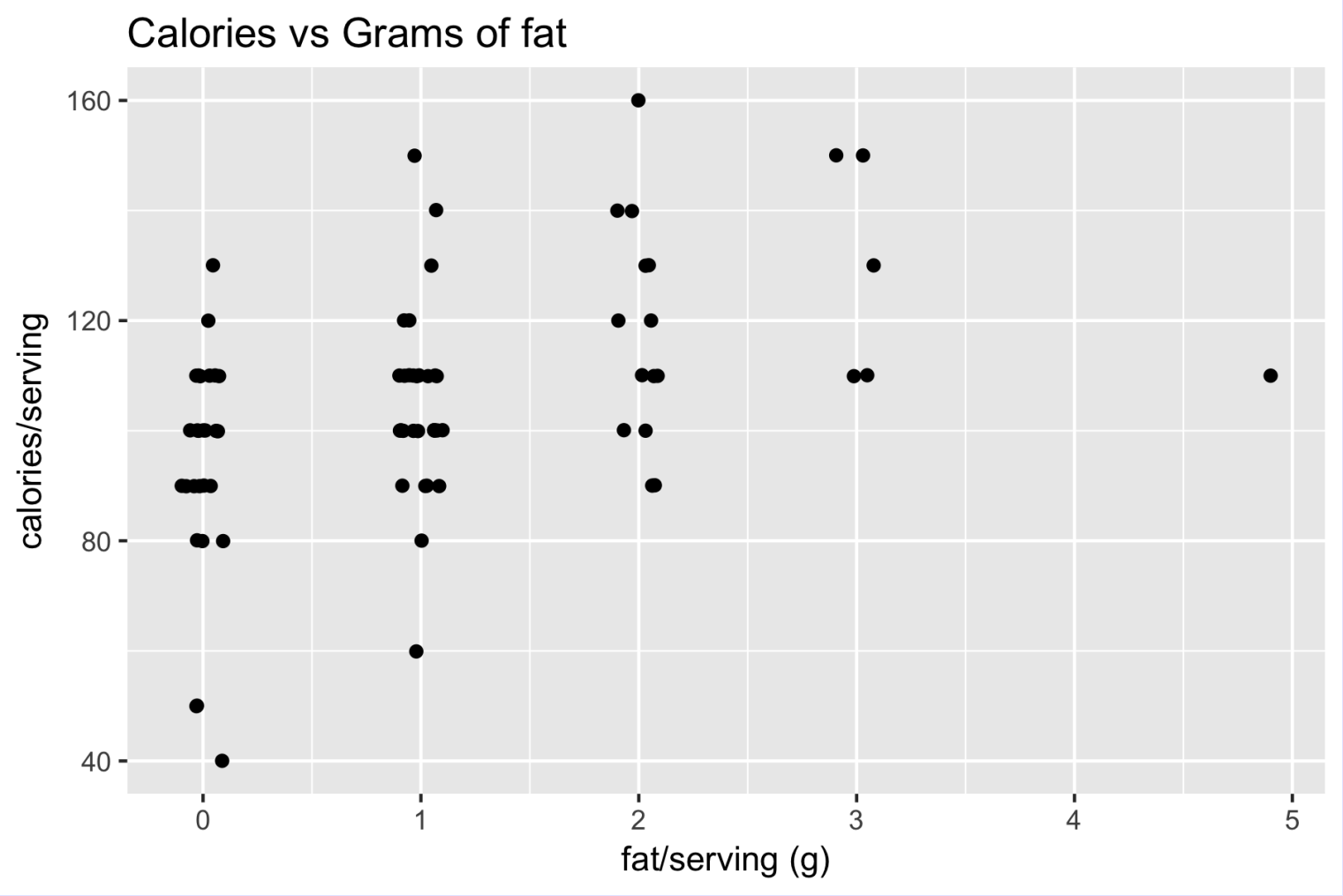
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat



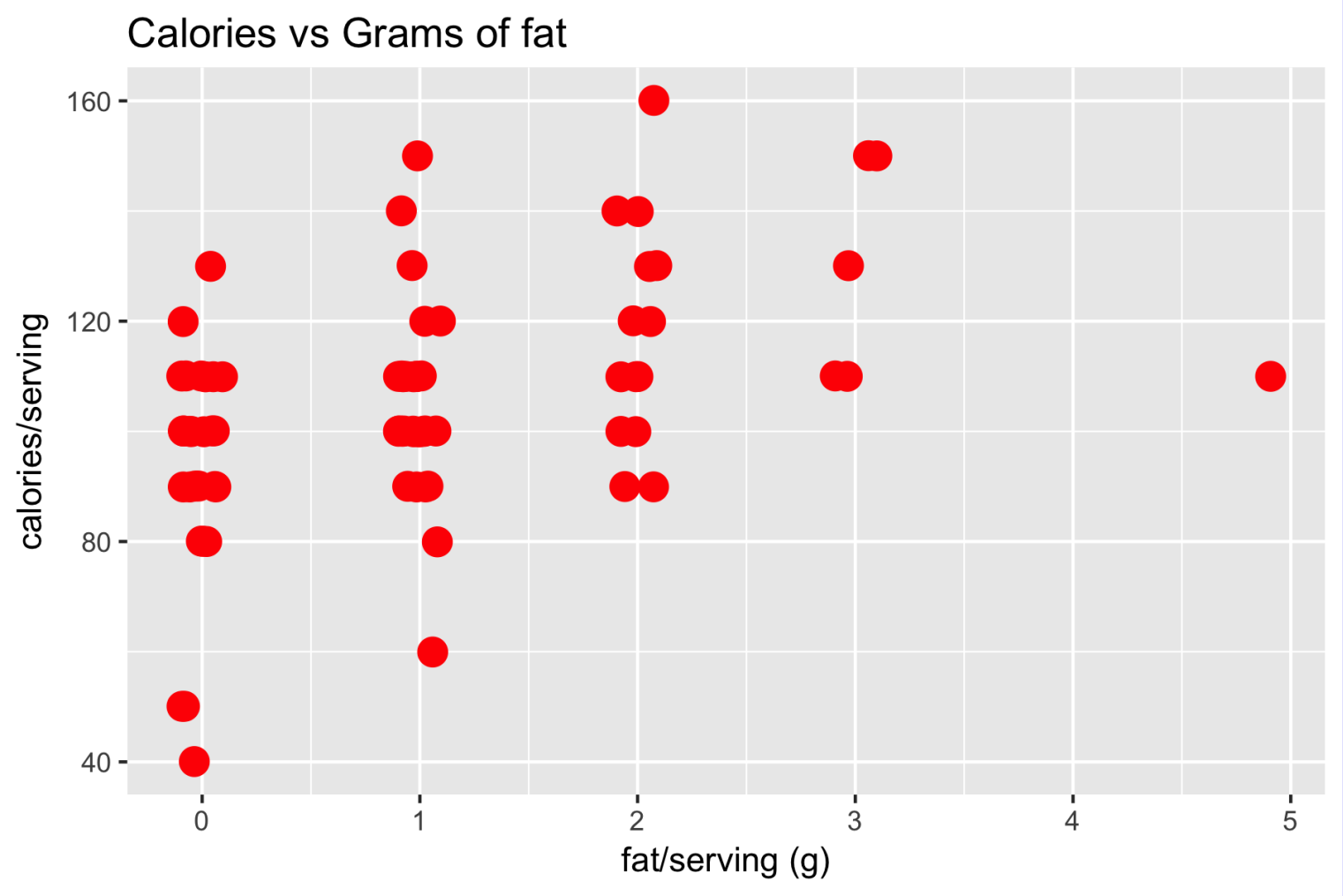
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - w=.1, h=.1



Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - w=.1, h=.1, color="red", size=4



Plotting with *ggplot2* - Dodgine

Apply a consistent shift across groups

position=position_dodge(w=.., h=...)

```
1 report <- plyr::ddply(cereal, c("shelf","fiber.grp"),
2     sf.simple.summary, variable="calories", crd=TRUE)
3 report
```

	shelf	fiber.grp	n	nmiss	mean	sd	se	lcl	ucl
1	1	high	2	0	85.00	7.07	5.00	21.47	148.53
2	1	low	18	0	102.22	10.60	2.50	96.95	107.49
3	2	high	1	0	120.00	NA	NaN	NaN	NaN
4	2	low	20	0	107.00	12.18	2.72	101.30	112.70
5	3	high	9	0	102.22	34.20	11.40	75.94	128.51
6	3	low	27	0	107.41	27.68	5.33	96.46	118.36

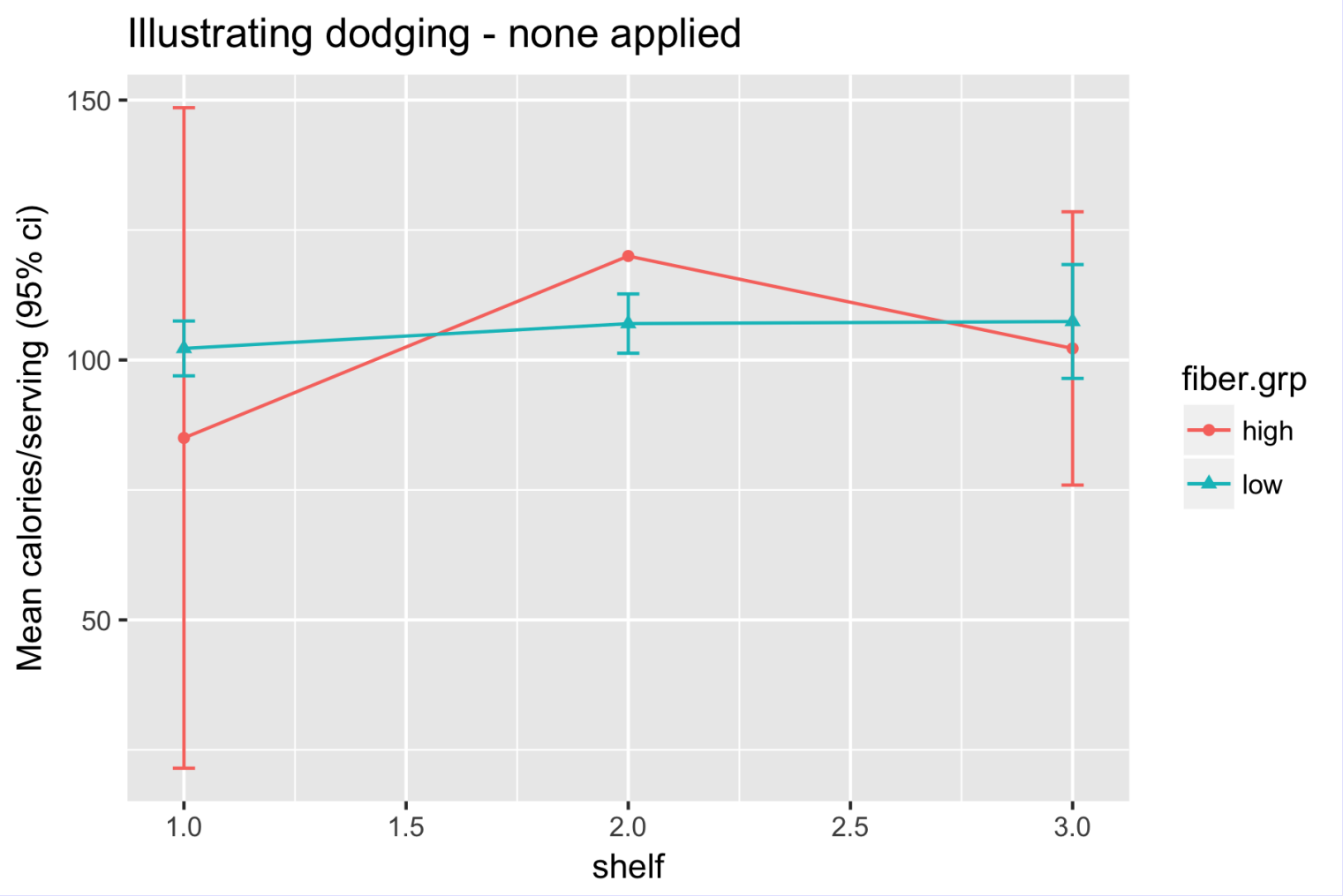
Plotting with *ggplot2* - Dodging

Apply a consistent shift across groups

position=position_dodge(w=.., h=...)

```
1 newplot <- ggplot2::ggplot(report, aes(x=shelf, y=mean,
2             color=fiber.grp, shape=fiber.grp))+
3    ggtitle("Illustrating dodging - none applied")+
4    geom_point()+
5    geom_line()+
6    geom_errorbar(aes(ymin=lcl, ymax=ucl), width=.05)+
7    ylab("Mean calories/serving (95% ci)")
8 newplot
```

Plot of results ignoring dodging.



Plotting with *ggplot2* - Dodging

Apply a consistent shift across groups

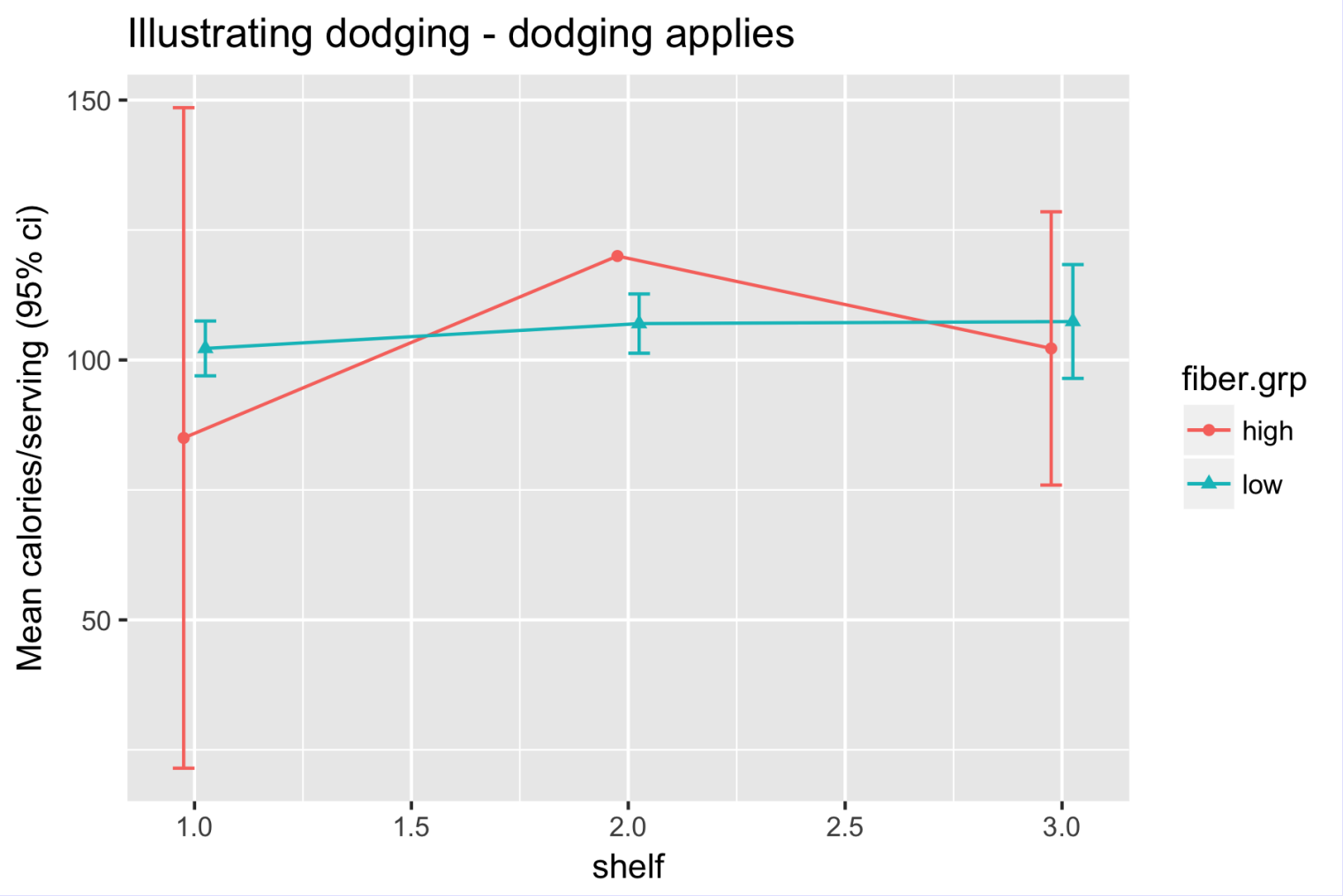
position=position_dodge(w=.., h=...)

```
1 newplot <- ggplot2::ggplot(report, aes(x=shelf, y=mean,
2             color=fiber.grp, shape=fiber.grp))+
3    ggtitle("Illustrating dodging - dodging applies")+
4    geom_point( position=position_dodge(w=0.1))+
5    geom_line(position=position_dodge(w=0.1))+
6    geom_errorbar(aes(ymin=lcl, ymax=ucl), width=.1, positio
7    ylab("Mean calories/serving (95% ci)")
8 newplot
```

Keep the dodging consistent among layers.

It is possible to have different amounts of dodging but unusual.

Dodging applied.



Plotting with *ggplot2* - Dodging

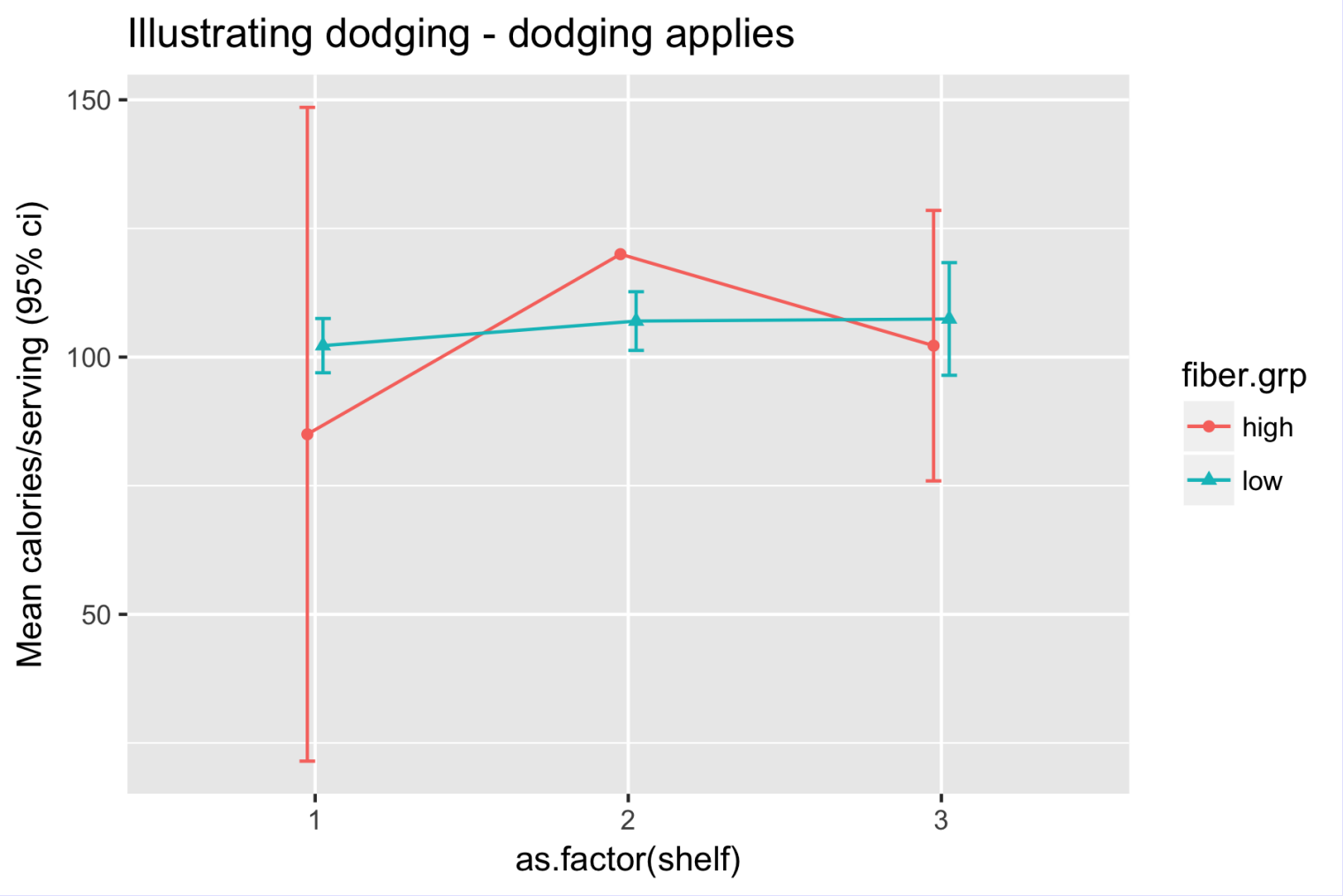
Apply a consistent shift across groups
Force *shelf* to be character and not numeric. Note use of *group*
position=position_dodge(w=.., h=...)

```
newplot <- ggplot2::ggplot(report, aes(x=as.factor(shelf),
      y=mean,
      group=fiber.grp, color=fiber.grp, shape=fiber.grp))+
   ggtitle("Illustrating dodging - dodging applies")+
   geom_point( position=position_dodge(w=0.1))+
   geom_line(position=position_dodge(w=0.1))+
   geom_errorbar(aes(ymin=lcl, ymax=ucl), width=.1,
       position=position_dodge(w=0.1))+
   ylab("Mean calories/serving (95% ci)")
newplot
```

Not clear why *group*= is needed??

Plotting with *ggplot2* - Dodging

Dodging applied with X axis as a categorical variable.



Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable

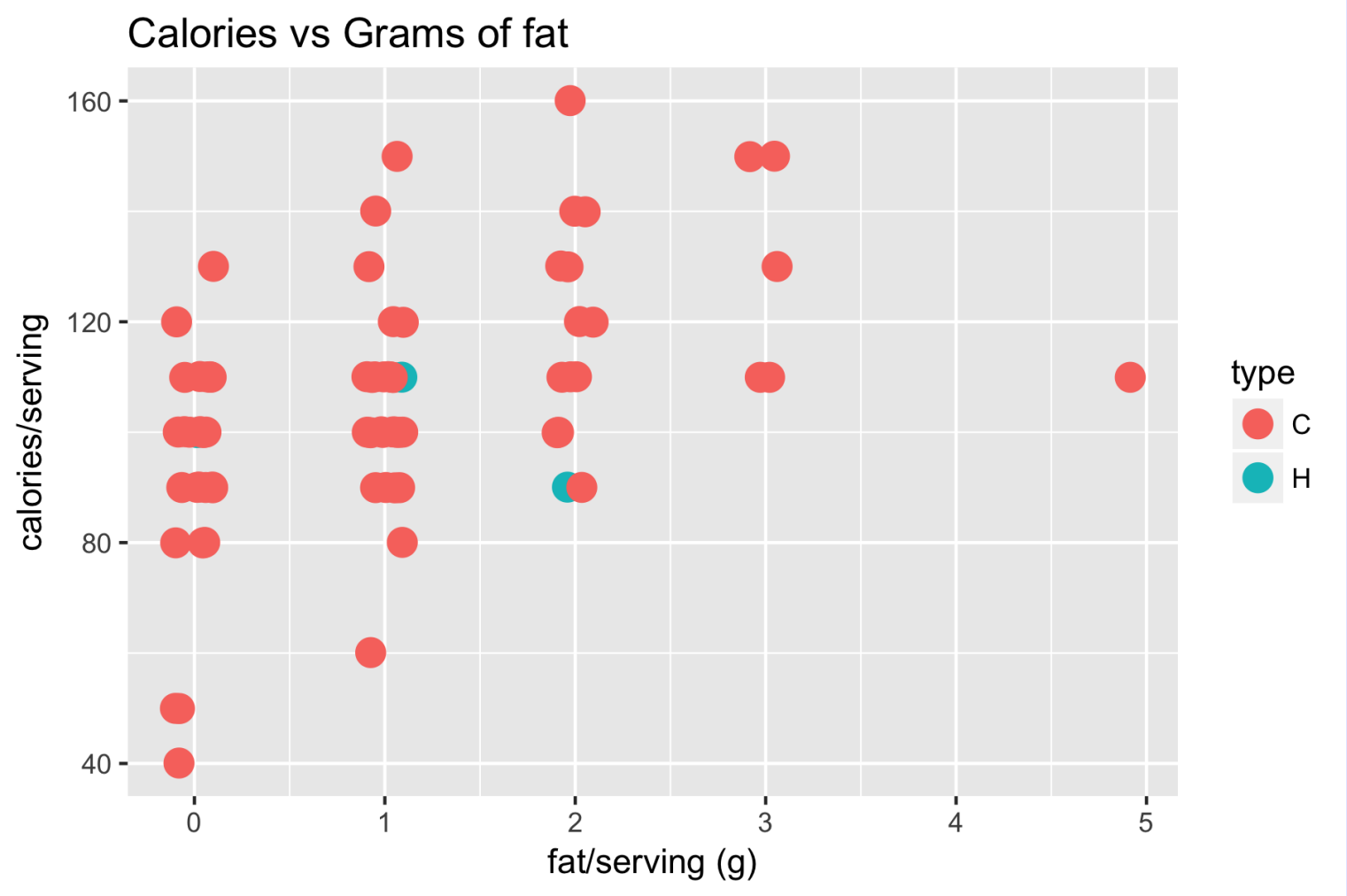
```
newplot <- ggplot(cereal, aes(x=fat, y=calories,
                                    color=type))+
   ggtitle("Calories vs Grams of Fat")+
   xlab("Fat/serving (g)")+ylab("Calories/serving")+
   geom_jitter(position=position_jitter(w=.1, h=.1),size=4)

newplot <- ggplot(cereal, aes(x=fat, y=calories,
                                   color=shelfF))+
   ggtitle("Calories vs Grams of Fat")+
   xlab("Fat/serving (g)")+ylab("Calories/serving")+
   geom_jitter(position=position_jitter(w=.1, h=.1),size=4)

newplot <- ggplot(cereal, aes(x=fat, y=calories,
                              color=shelfF, shape=shelfF))+
   ggtitle("Calories vs Grams of Fat")+
   xlab("Fat/serving (g)")+ylab("Calories/serving")+
   geom_jitter(position=position_jitter(w=.1, h=.1),size=4)
```

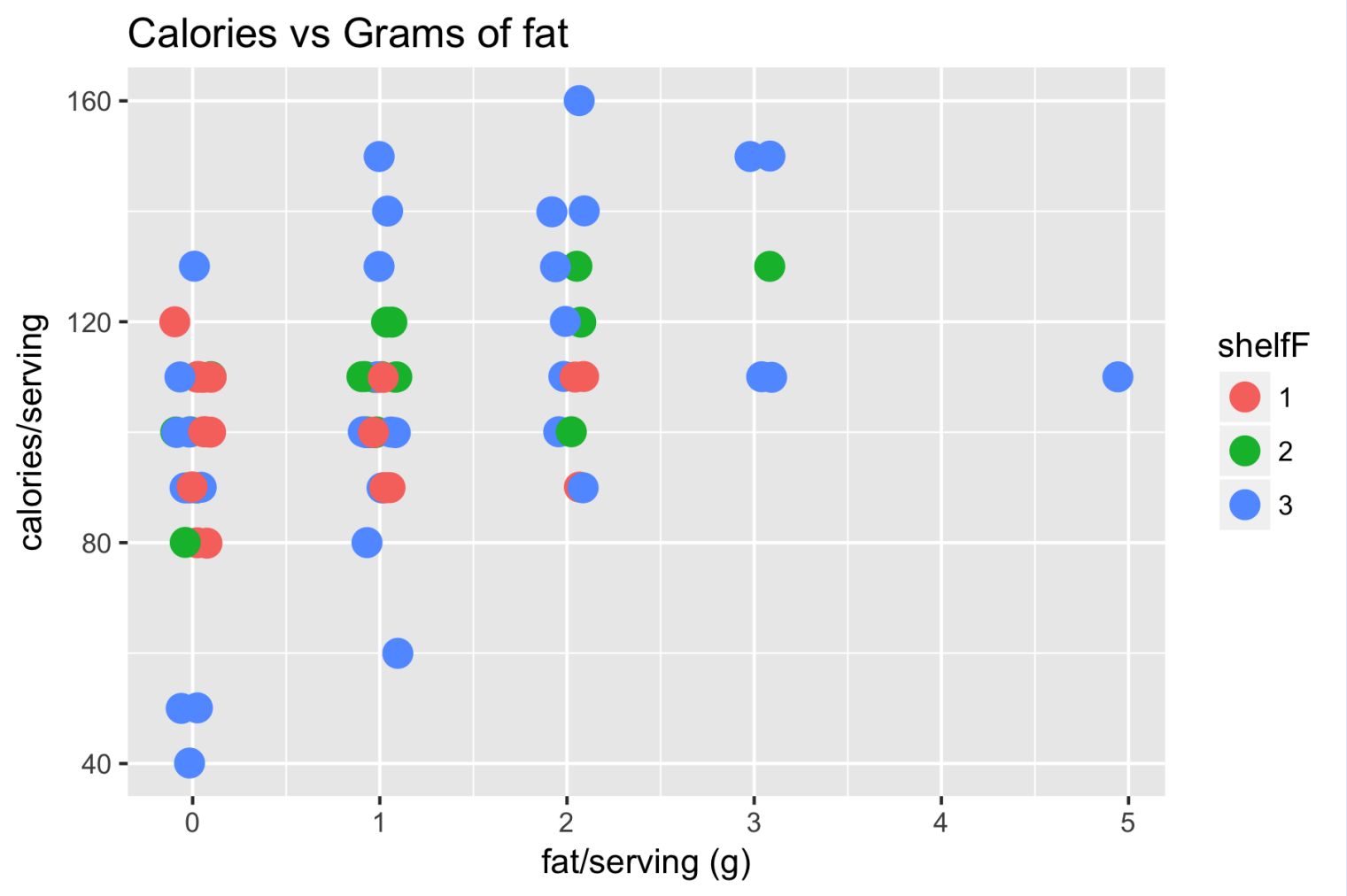
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable



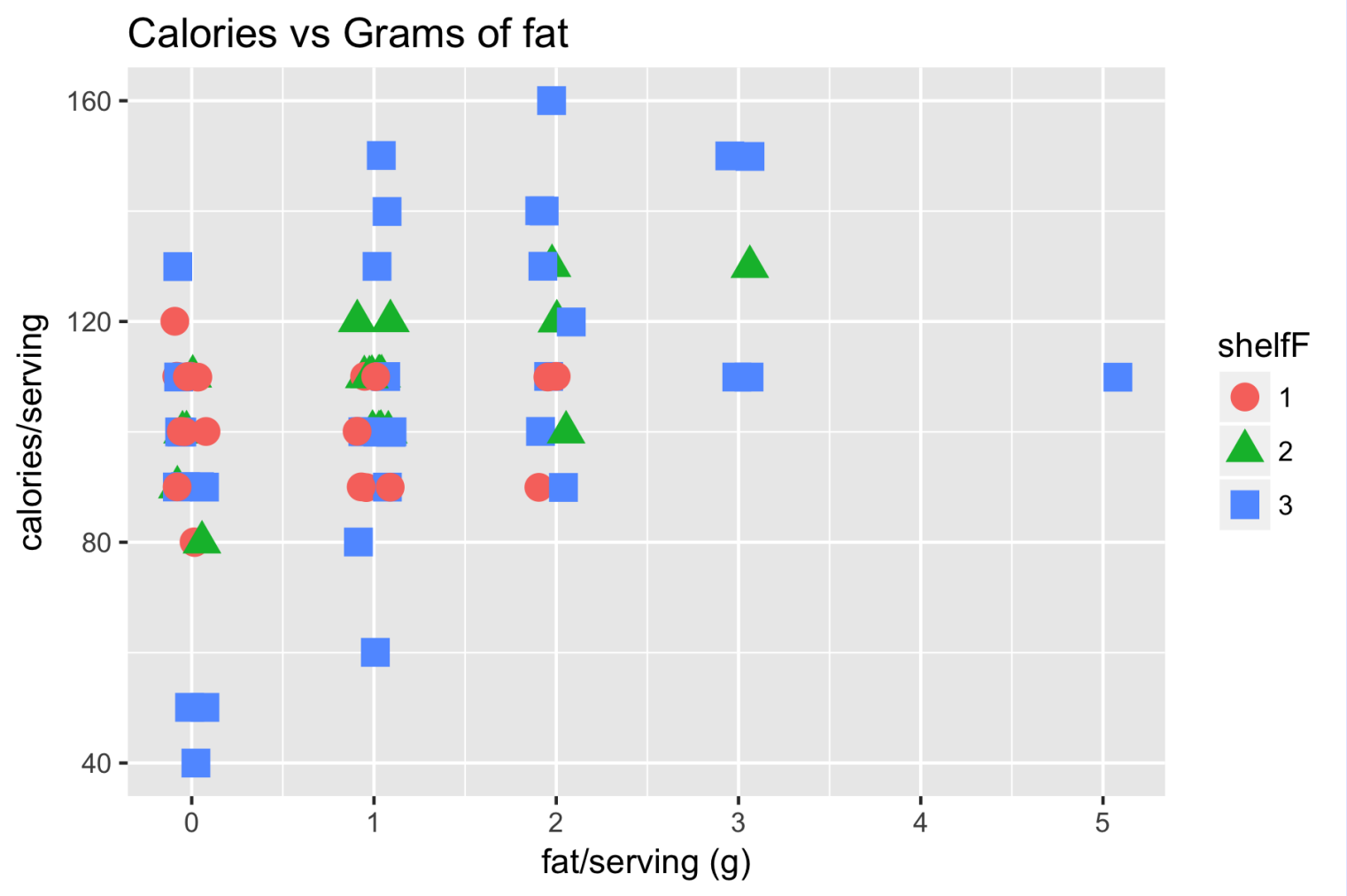
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable



Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable



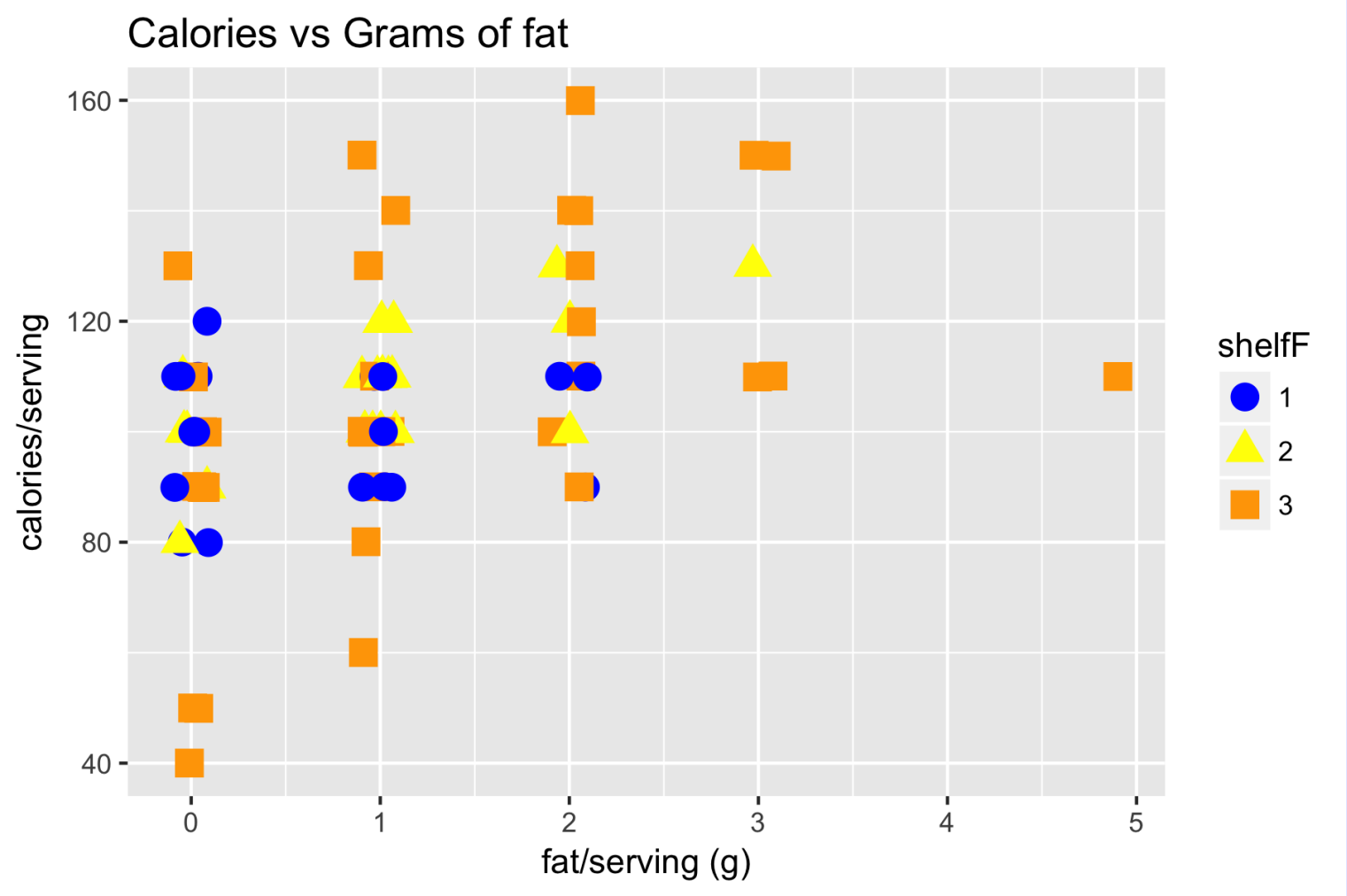
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable.
Specify the colors directly.

```
1 newplot <- ggplot(data=cereal, aes(x=fat, y=calories,
2    color=shelfF, shape=shelfF))+
3    ggtitle("Calories vs Grams of fat")+
4    xlab("fat/serving (g)")+ylab("calories/serving")+
5    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
6    scale_color_manual( values=c("blue","yellow","orange"))
7 newplot
```

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable



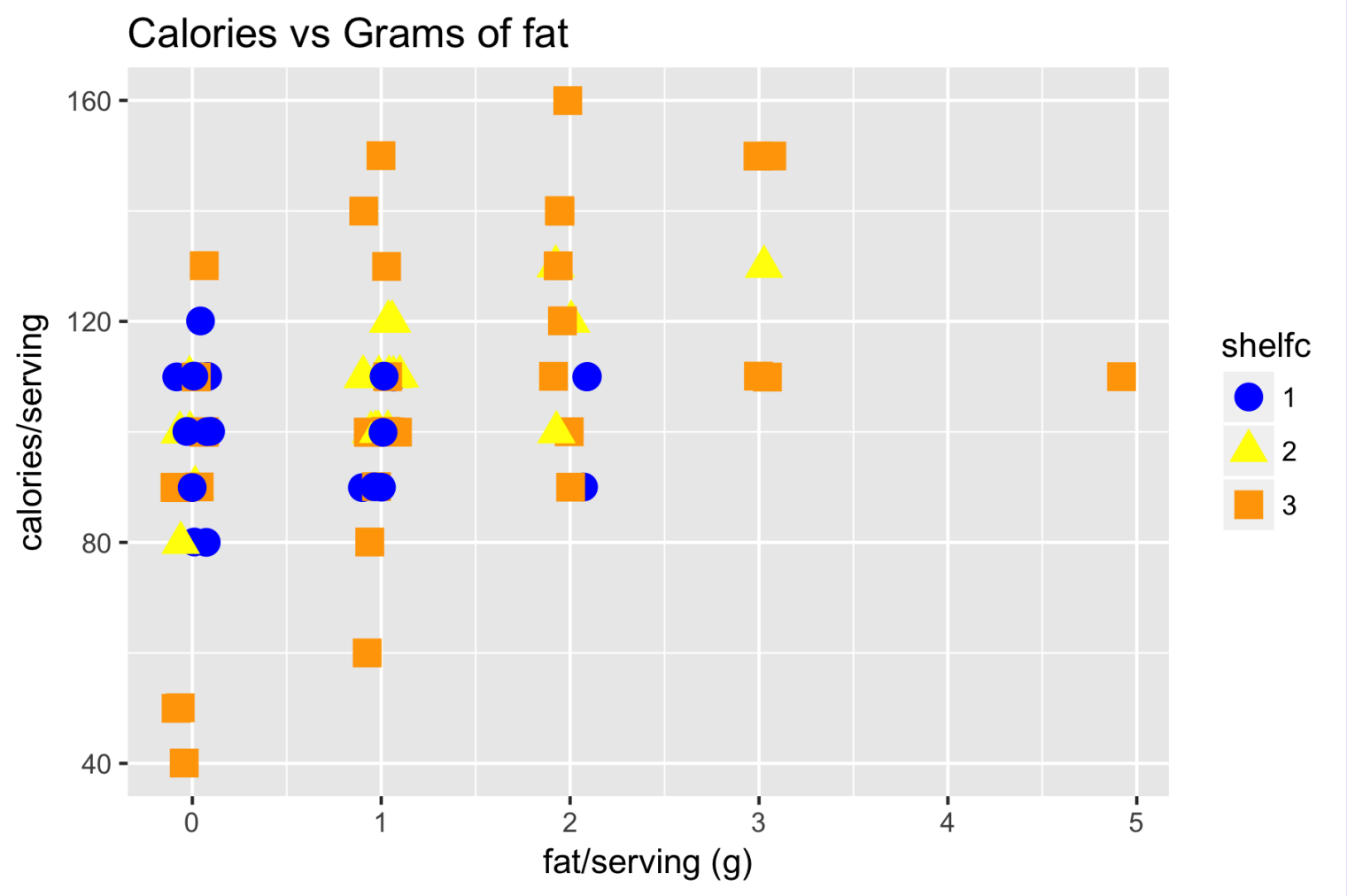
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable.
Specify the colors using a named vector - preferred method rather than relying on alphabetical ordering, but need to create character variable in place of any numeric codes.

```
cereal$shelfc <- as.character(cereal$shelf)
shelf_colors=c("1"="blue", "2"="yellow", "3"="orange")
newplot <- ggplot(data=cereal, aes(x=fat, y=calories,
     color=shelfc, shape=shelfc))+
   ggtitle("Calories vs Grams of fat")+
   xlab("fat/serving (g)")+ylab("calories/serving")+
   geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
   scale_color_manual( values=shelf_colors)
newplot
```

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by categorical variable



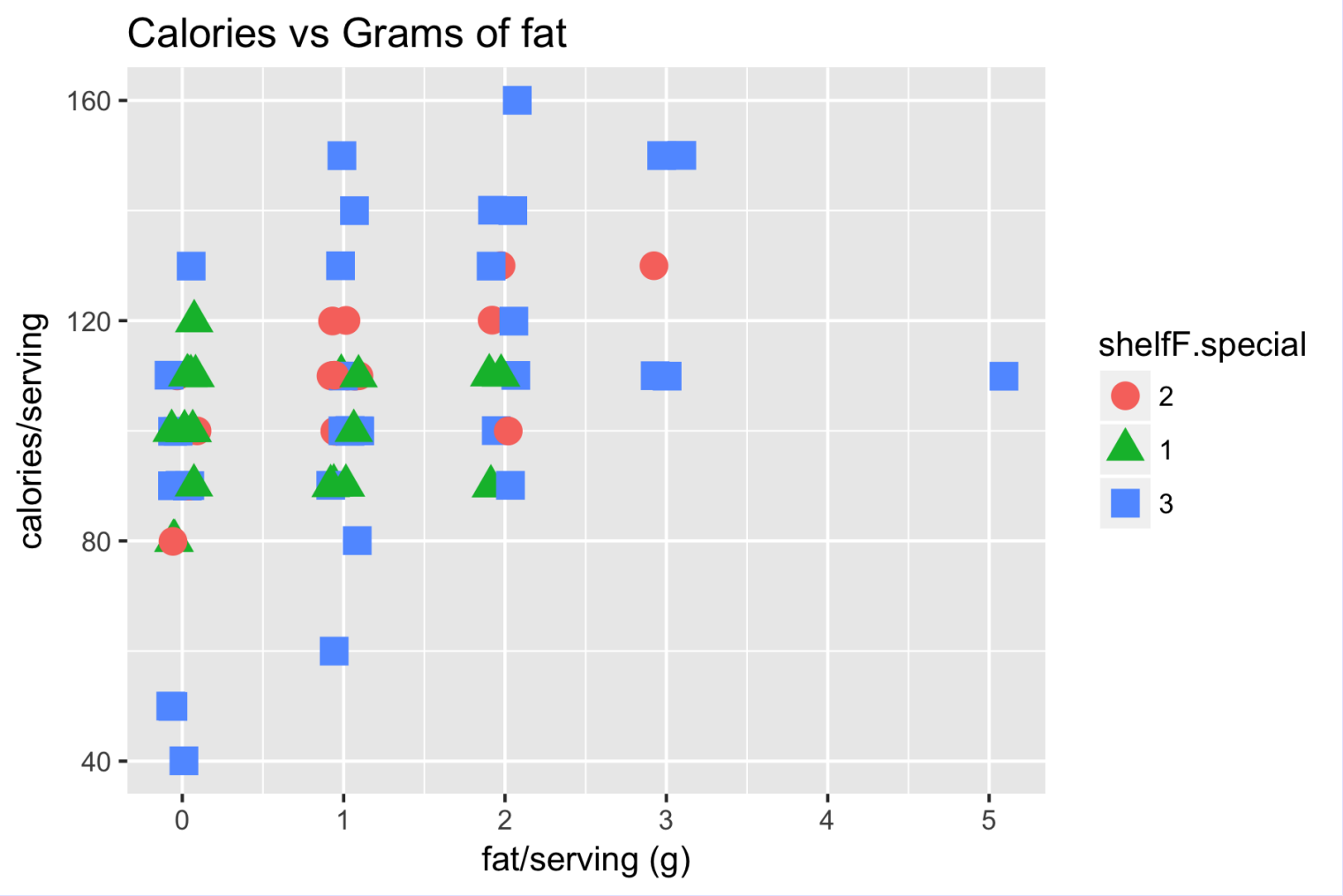
Plotting with *ggplot2* - Scatterplot

Change order of legend.

(1) Define the factor with the ordering you want

```
cereal$shelfF.special <- factor(cereal$shelf,
     levels=c(2,1,3))
newplot <- ggplot(data=cereal, aes(x=fat, y=calories,
          color=shelfF.special,
          shape=shelfF.special))+
   ggtitle("Calories vs Grams of fat")+
   xlab("fat/serving (g)")+ylab("calories/serving")+
   geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
newplot
```

Change order of legend



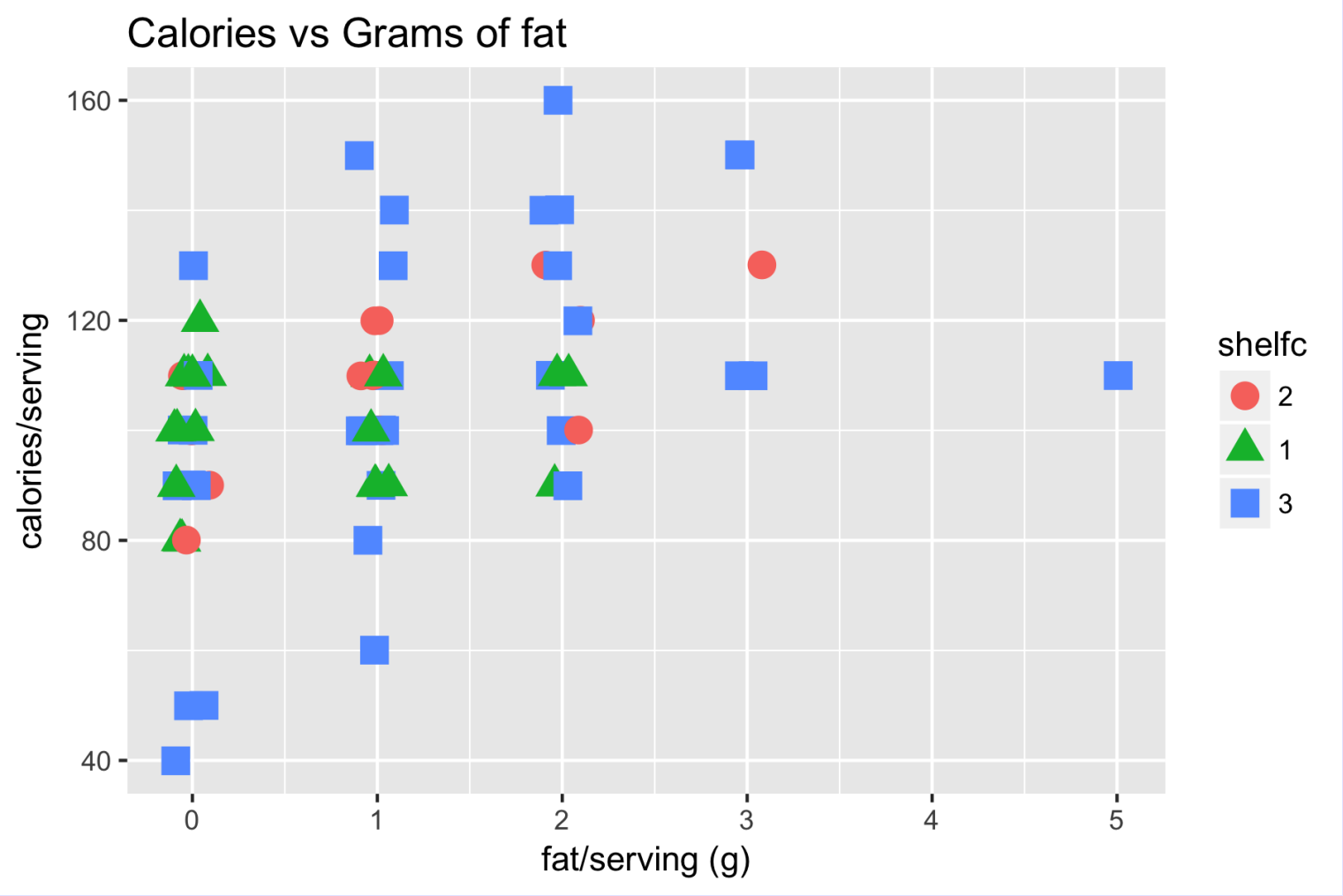
Plotting with *ggplot2* - Scatterplot

Change order of legend.
(2) Define the order in the plot

```
1 cereal$shelfc <- as.character(cereal$shelf)
2 newplot <- ggplot(data=cereal, aes(x=fat, y=calories, color=
3    ggtitle("Calories vs Grams of fat")+
4    xlab("fat/serving (g)")+ylab("calories/serving")+
5    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
6    scale_color_discrete( limits=c("2","1","3"))+
7    scale_shape_discrete( limits=c("2","1","3"))
8 newplot
```

Must be a character variable that defines the groups.

Change order of legend



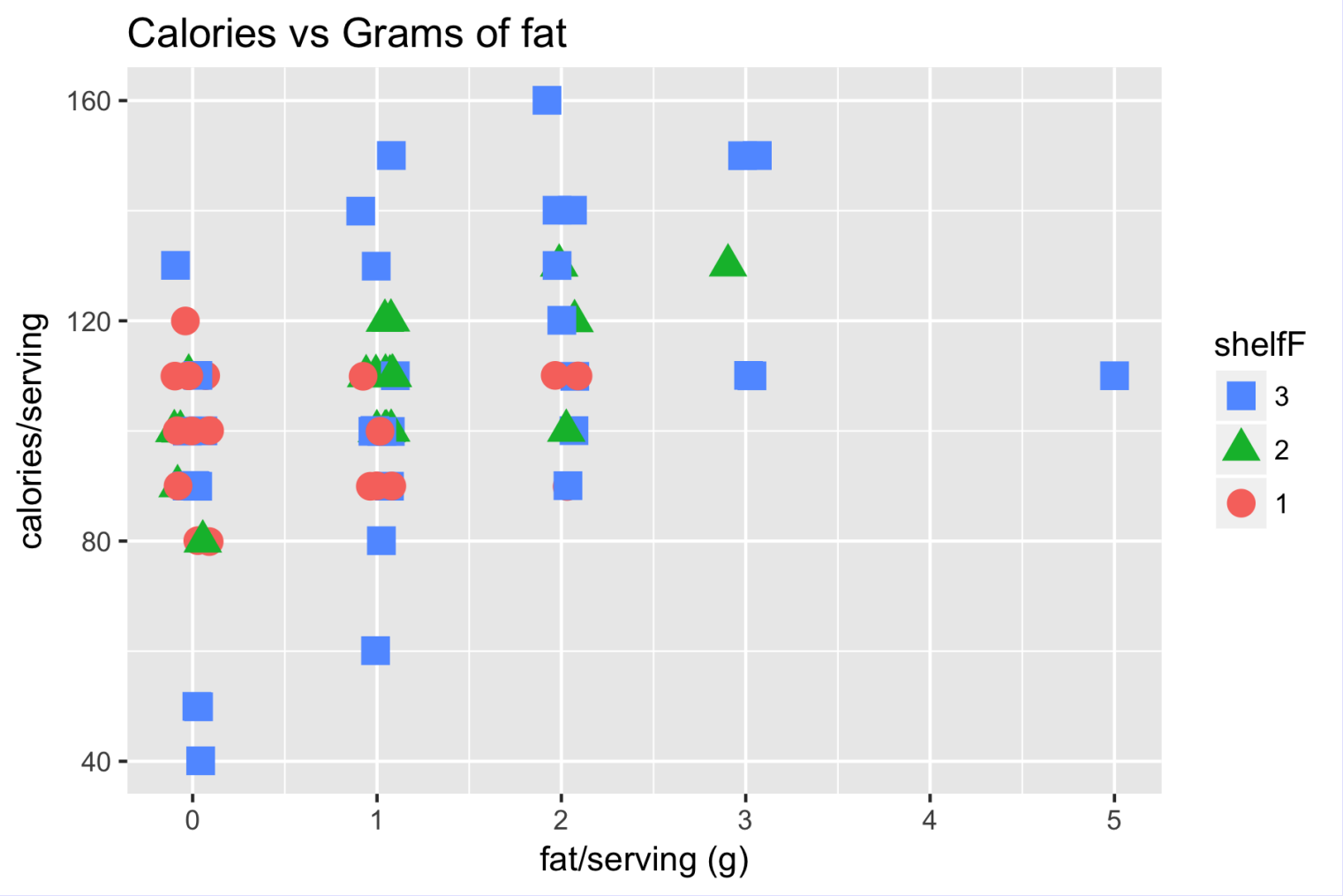
Plotting with *ggplot2* - Scatterplot

Reversing order of the legend

```
1 newplot <- ggplot(data=cereal, aes(x=fat, y=calories, color=
2    ggtitle("Calories vs Grams of fat")+
3    xlab("fat/serving (g)")+ylab("calories/serving")+
4    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
5    scale_color_discrete(guide = guide_legend(reverse=TRUE))+
6    scale_shape_discrete(guide = guide_legend(reverse=TRUE))
7 newplot
```

Need both scale_*_discrete.

Change order of legend

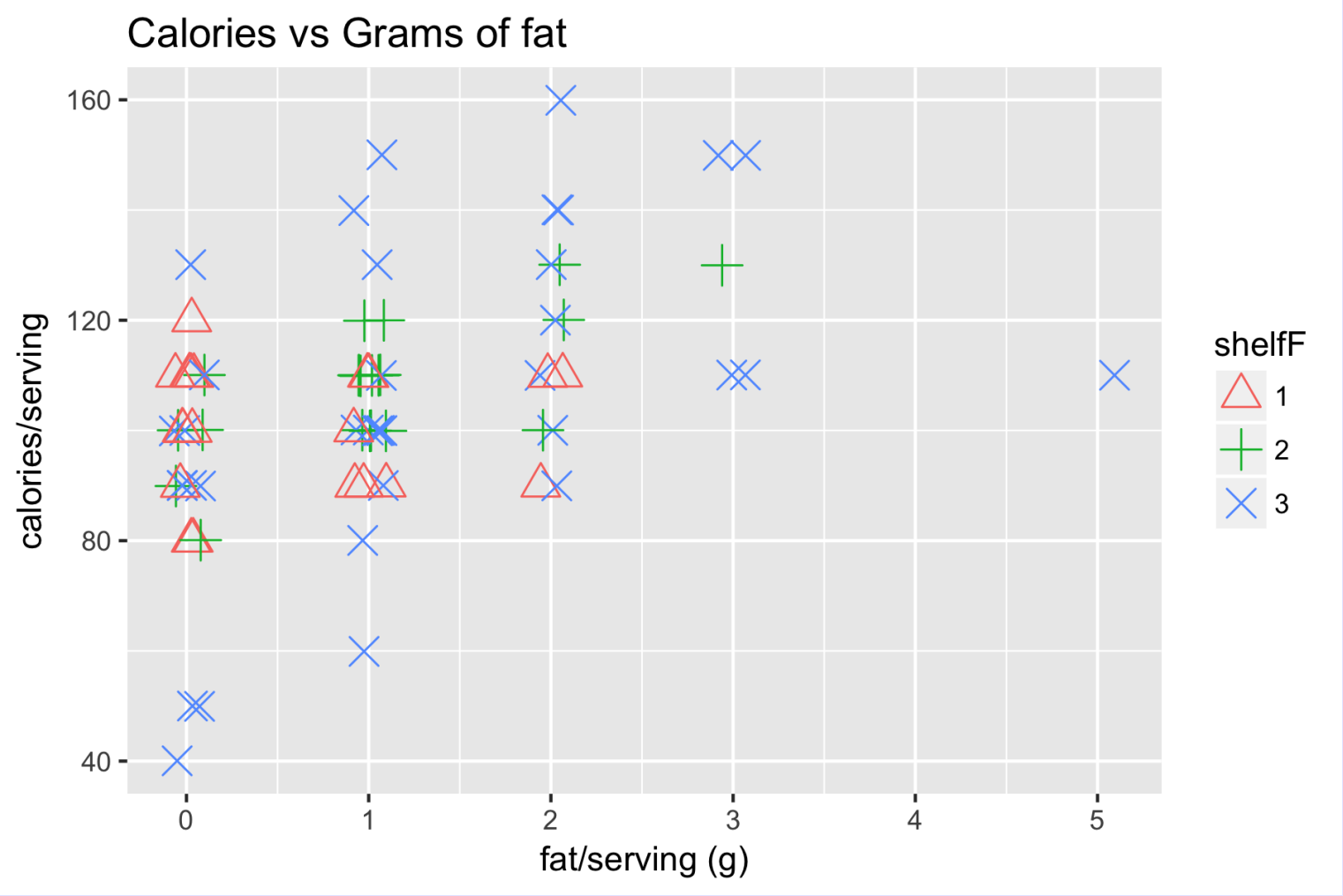


Plotting with *ggplot2* - Scatterplot

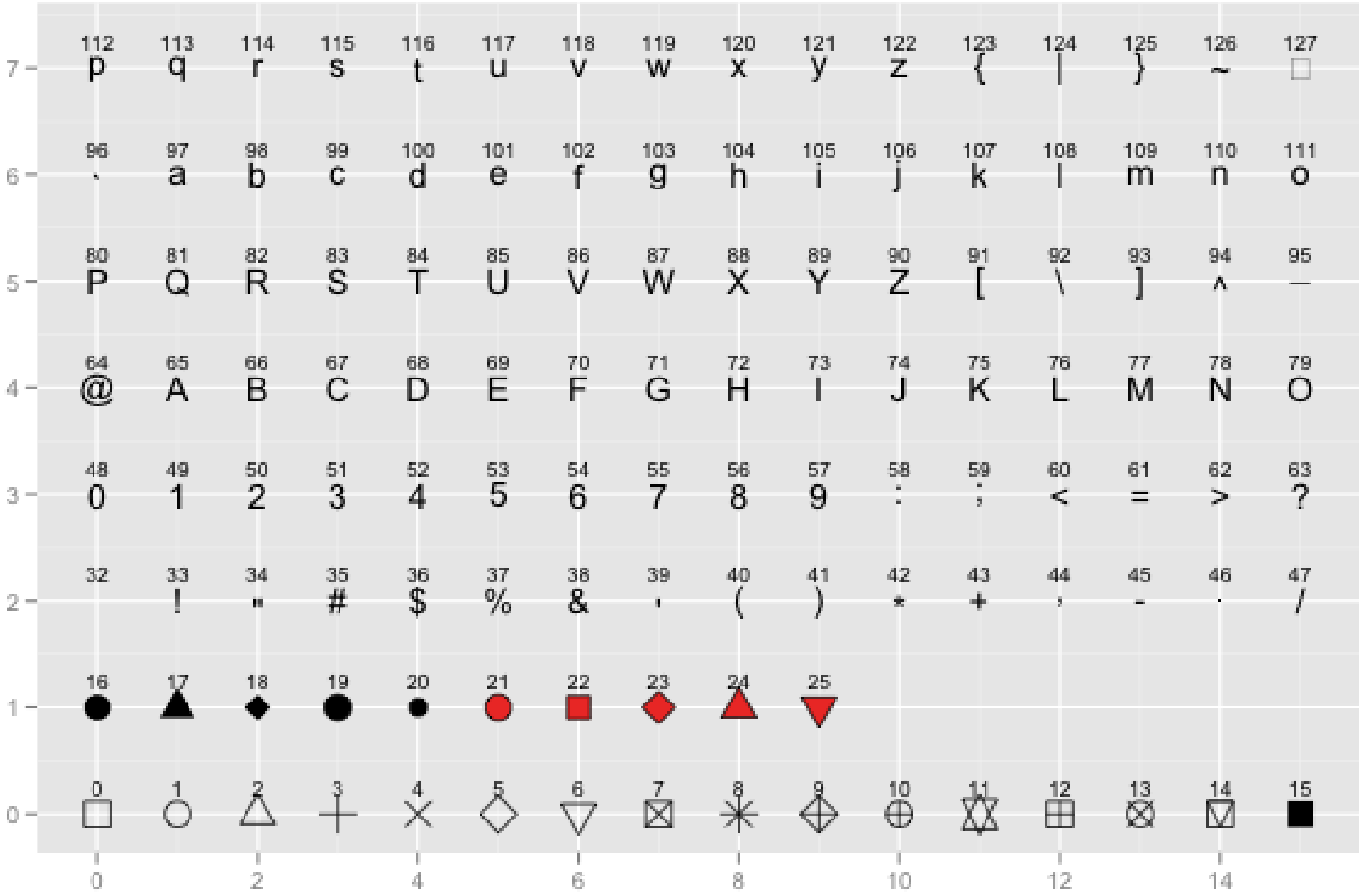
Change the plotting symbol - similar to changing color

```
1 newplot <- ggplot(data=cereal, aes(x=fat, y=calories, color=
2    ggtitle("Calories vs Grams of fat")+
3    xlab("fat/serving (g)")+ylab("calories/serving")+
4    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
5    scale_shape_manual(values=c(2,3,4))
6 newplot
```

Change plotting symbols



Different plotting symbols (see *help(pch)*)

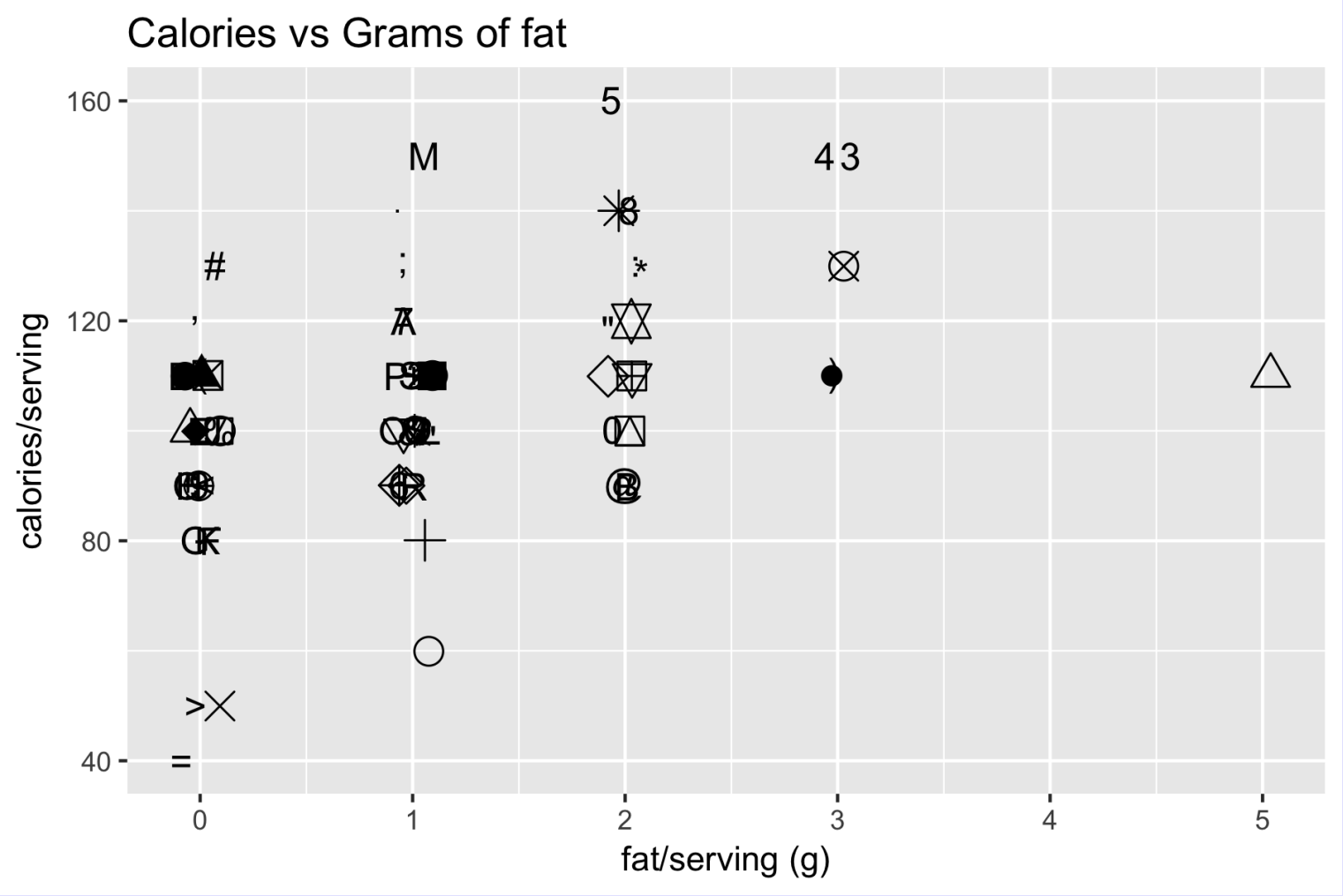


Plotting with *ggplot2* - Scatterplot

```
1  # Different symbol for each point
2  newplot <- ggplot(cereal, aes(x=fat, y=calories))+
3     ggtitle("calories vs Grams of fat")+
4     xlab("fat/serving (g)")+ylab("calories/serving")+
5     geom_jitter(position=position_jitter(w=.1, h=.1),
6              shape=c(1:25,32:83))
7  newplot
```

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - different plotting symbols



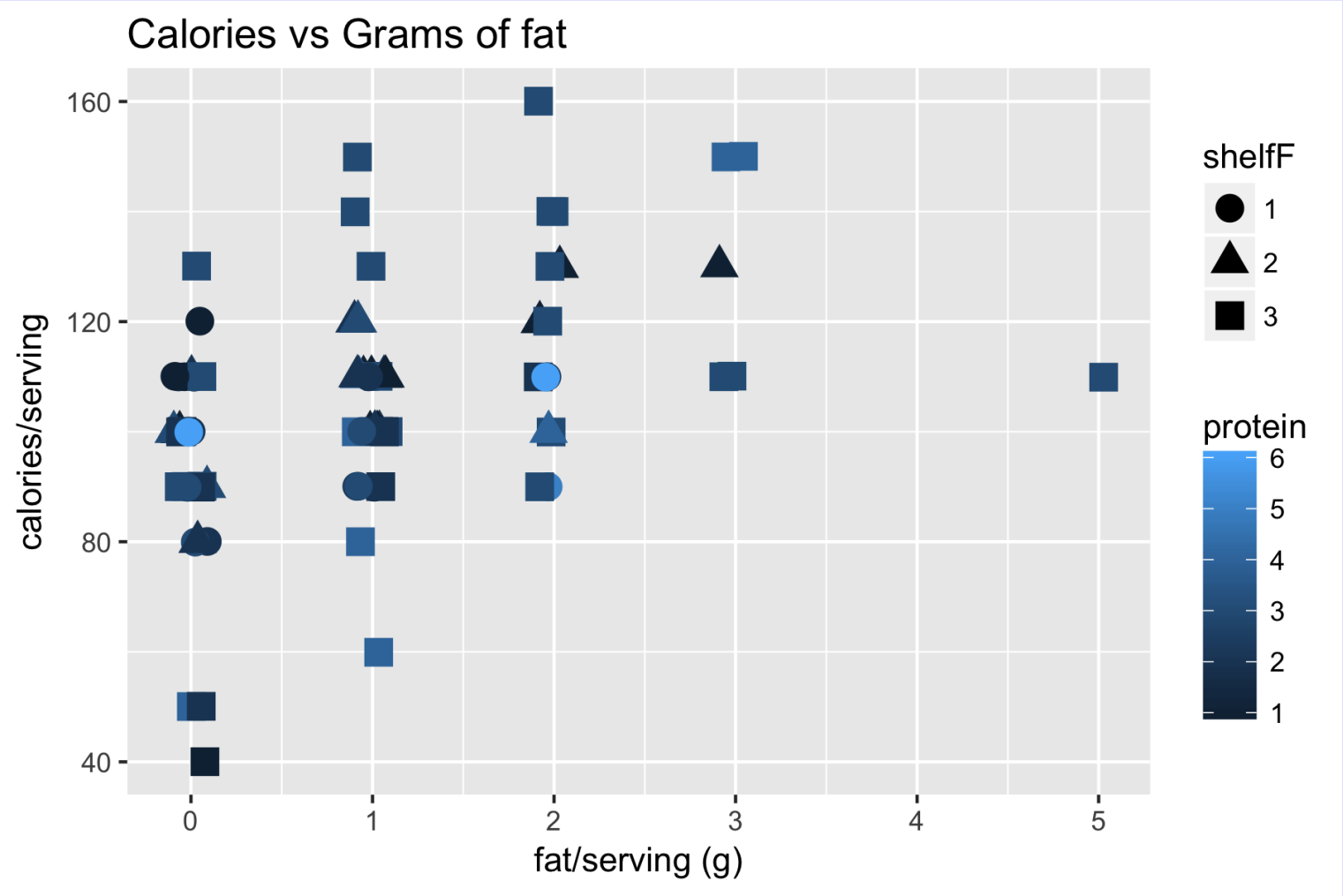
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color by continuous variable

```
1  newplot <- ggplot(cereal, aes(x=fat, y=calories,
2                                color=protein, shape=shelfF))+
3     ggtitle("Calories vs Grams of Fat")+
4     xlab("Fat/serving (g)")+ylab("Calories/serving")+
5     geom_jitter(position=position_jitter(w=.1, h=.1),size=4)
6
7  newplot <- ggplot(cereal, aes(x=fat, y=calories,
8                                color=protein, shape=shelfF))+
9     ggtitle("Calories vs Grams of Fat")+
10    xlab("Fat/serving (g)")+ylab("Calories/serving")+
11    geom_jitter(position=position_jitter(w=.1, h=.1),size=4)+
12    scale_color_gradient(high="black", low="blue")
```

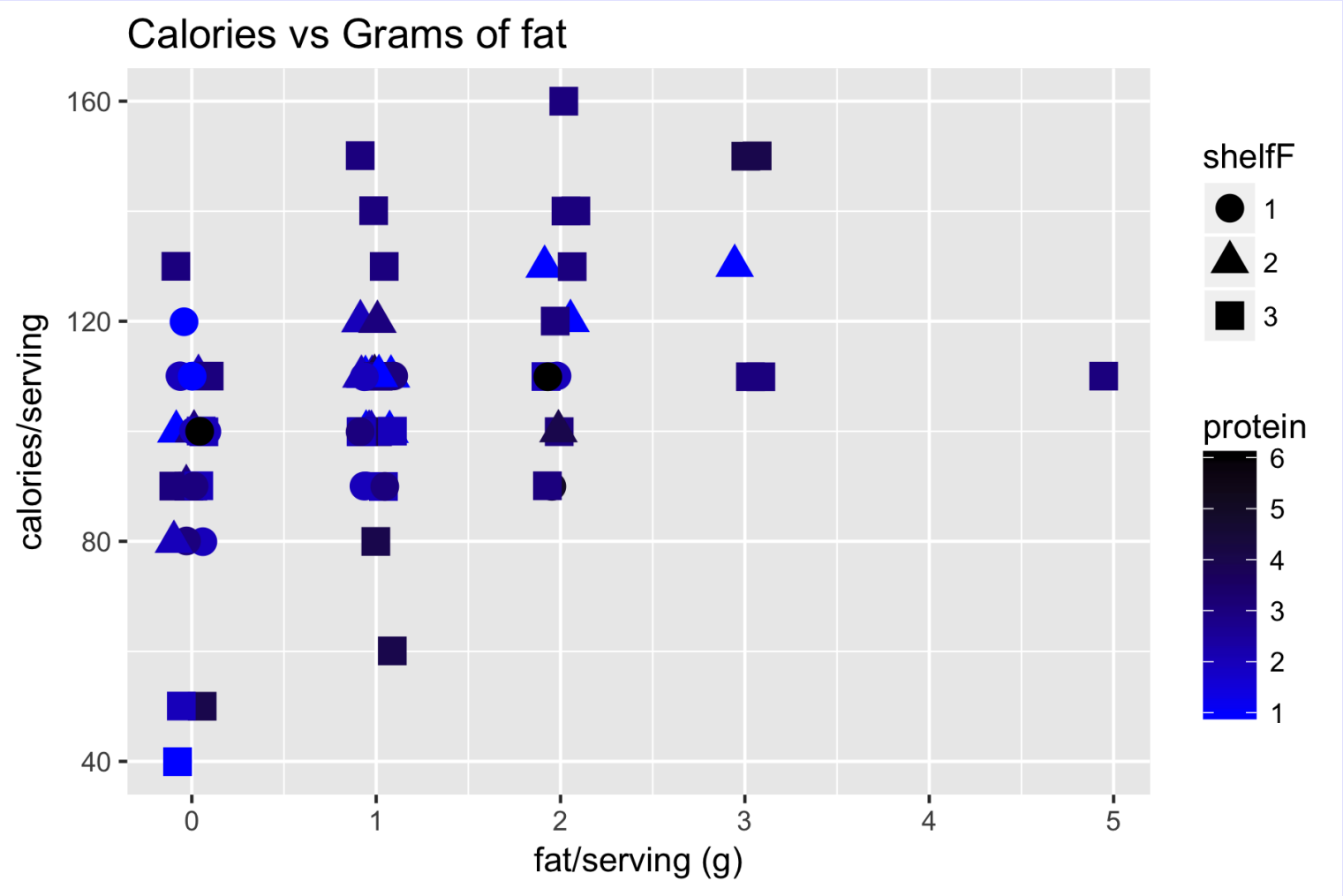
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by continuous variable



Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - color/shape by continuous variable



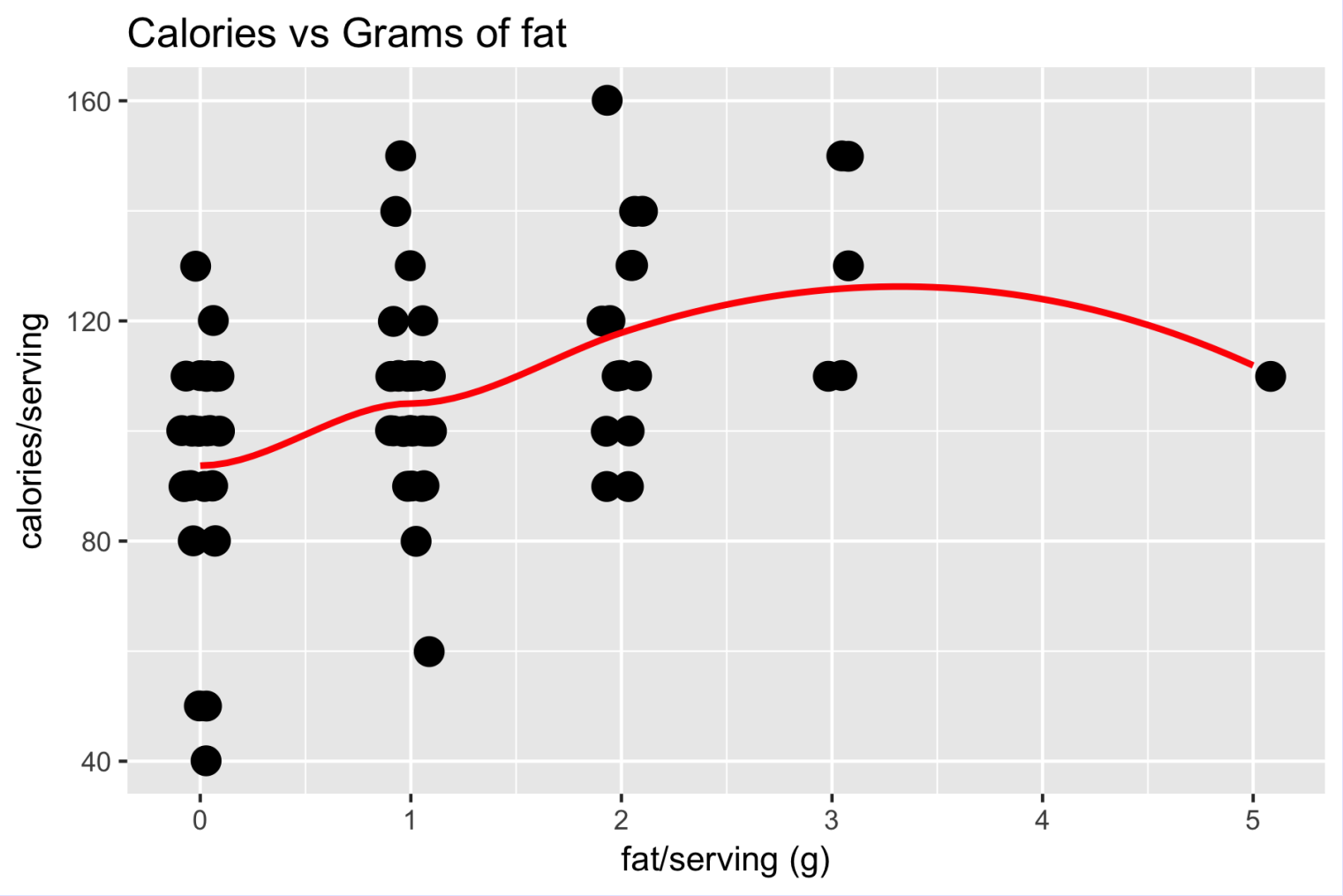
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat; add a smoother

```
1  newplot <- ggplot(cereal, aes(x=fat, y=calories))+
2     ggtitle("calories vs Grams of fat")+
3     xlab("fat/serving (g)")+ylab("calories/serving")+
4     geom_jitter(position=position_jitter(w=.1, h=.1))+
5     geom_smooth(method="loess", se=FALSE, color="red")
6  newplot
7
8  newplot <- ggplot(cereal, aes(x=fat, y=calories))+
9     ggtitle("calories vs Grams of fat")+
10    xlab("fat/serving (g)")+ylab("calories/serving")+
11    geom_jitter(position=position_jitter(w=.1, h=.1))+
12    geom_smooth(method="loess", se=FALSE, color="red")+
13    geom_smooth(method="lm", color="black")
14 newplot
```

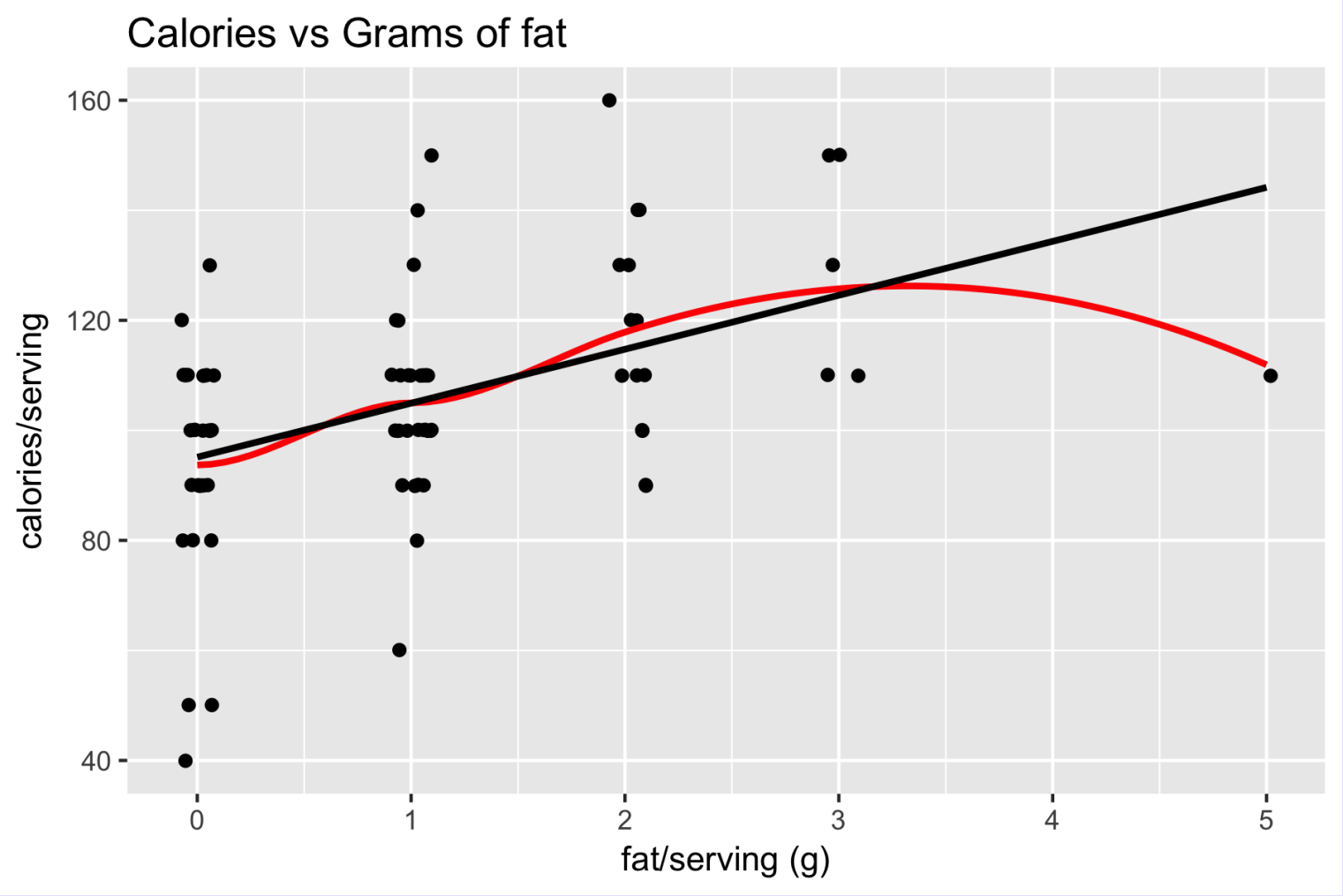
Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - smoothers



Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - smoothers

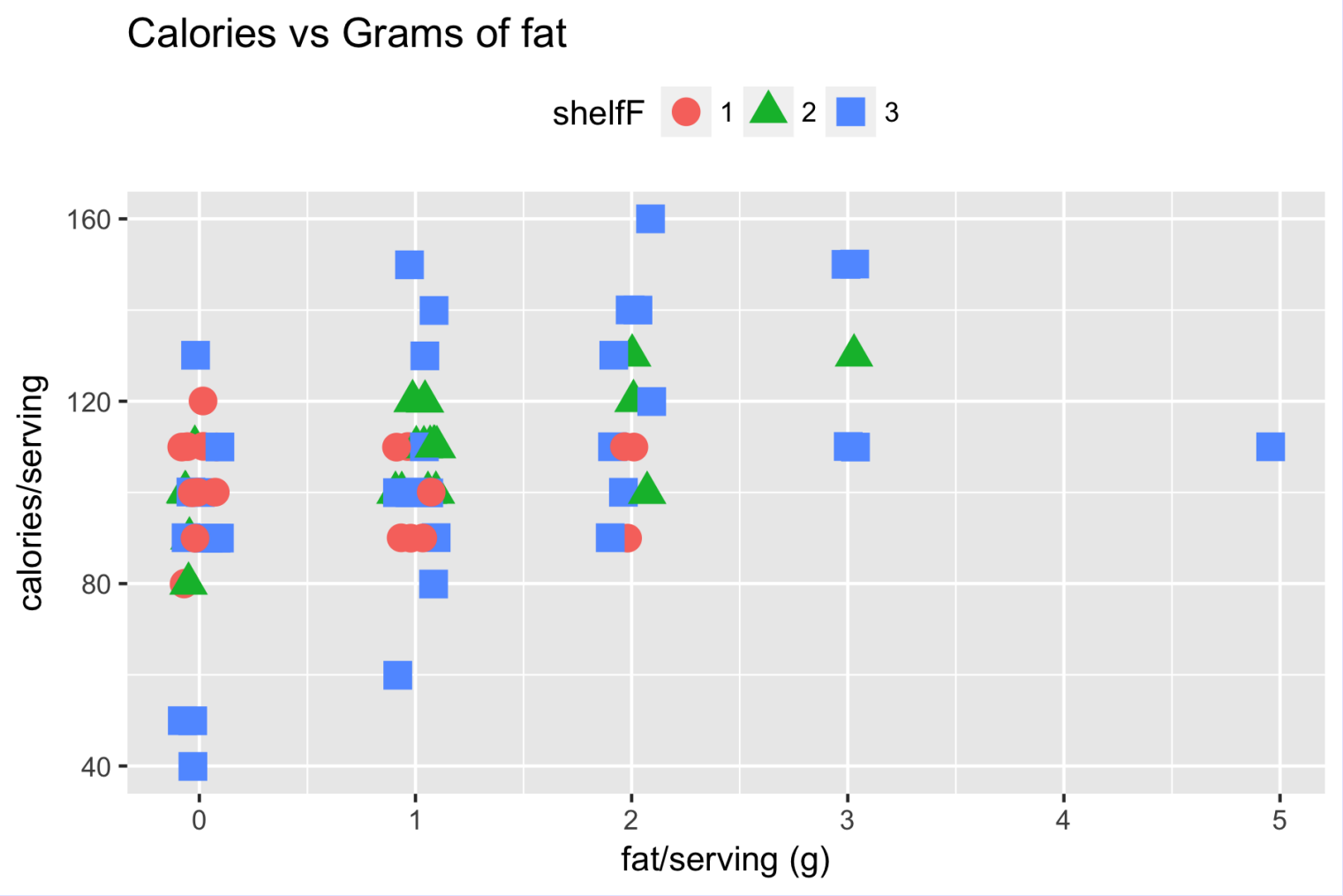


Plotting with *ggplot2* - Modifying the legend

http://www.cookbook-r.com/Graphs/Legends_(ggplot2)

```
1 newplot <- ggplot(cereal, aes(x=fat, y=calories,
2                   shape=shelfF, color=shelfF))+
3    ggtitle("Calories vs Grams of fat")+
4    xlab("fat/serving (g)")+ylab("calories/serving")+
5    geom_jitter(position=position_jitter(w=.1, h=.1),size=4)+
6    theme(legend.position="top")
7 newplot
```

Create a plot of calories vs. grams of fat - modify legend



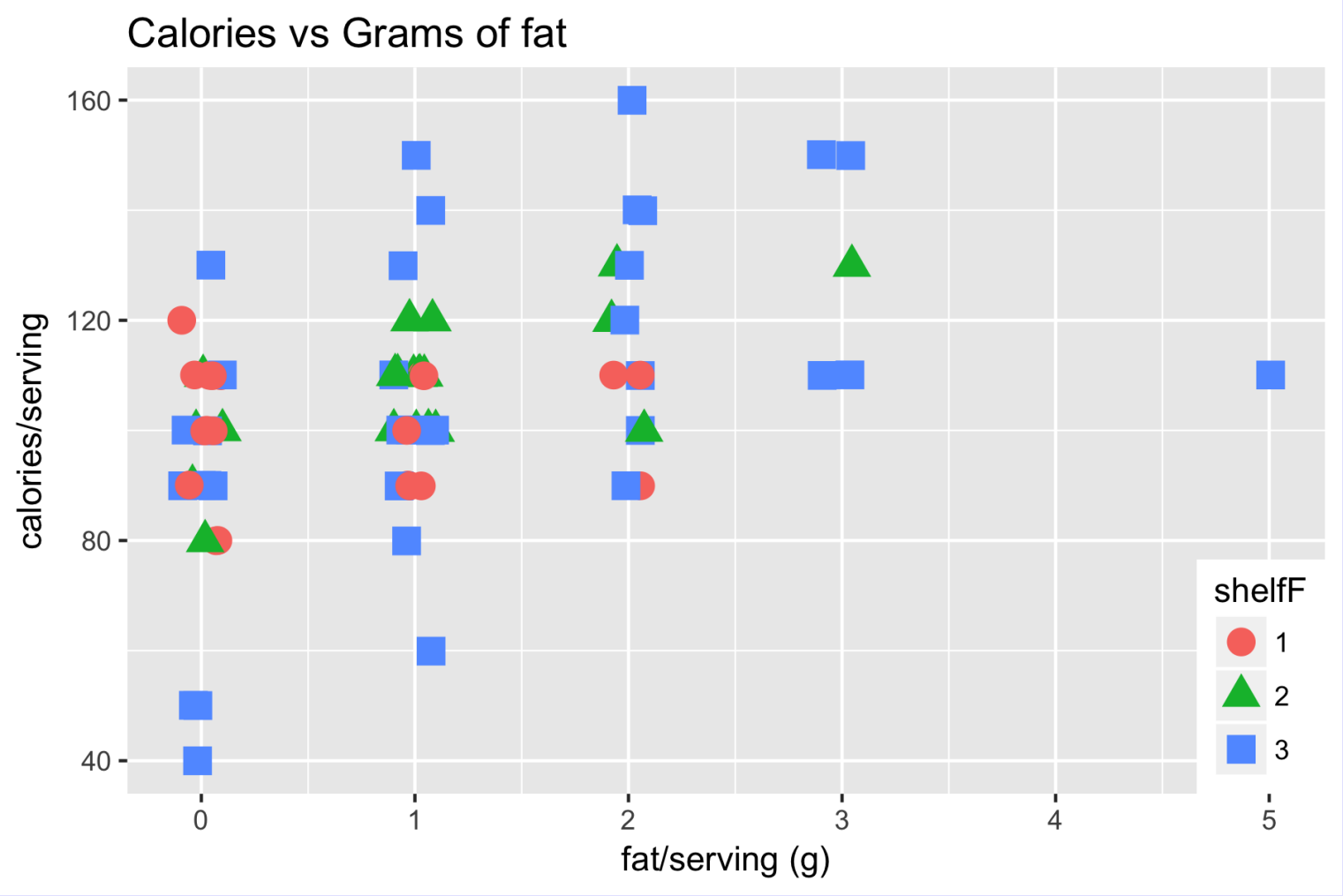
Plotting with *ggplot2* - Modifying the legend

http://www.cookbook-r.com/Graphs/Legends_(ggplot2)

```
1
2  # Set the "anchoring point" of the legend
3  #     (bottom-left is 0,0; top-right is 1,1)
4  # Numeric positions are relative to the entire area,
5  # including titles and labels, not just the plotting area.
6  newplot <- ggplot(cereal, aes(x=fat, y=calories,
7                    shape=shelfF, color=shelfF))+
8     ggtitle("Calories vs Grams of fat")+
9     xlab("fat/serving (g)")+ylab("calories/serving")+
10    geom_jitter(position=position_jitter(w=.1, h=.1),size=4)+
11    theme(legend.justification=c(1,0),
12               legend.position=c(1,0))
```

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - modify legend



Plotting with *ggplot2* - Modifying the legend

http://www.cookbook-r.com/Graphs/Legends_(ggplot2)

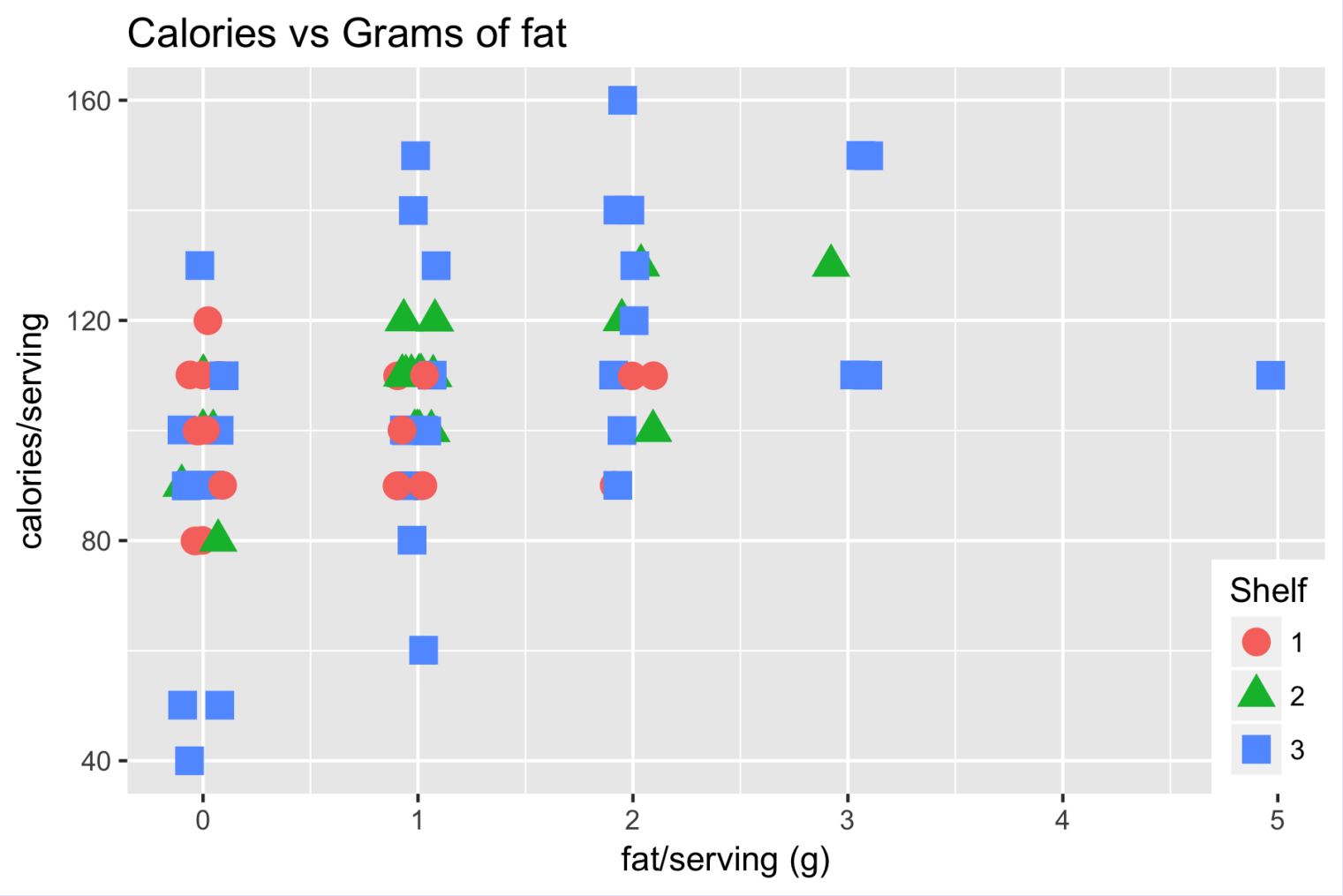
Change legend title.

```
1 newplot <- ggplot(data=cereal, aes(x=fat, y=calories,
2                     shape=shelfF, color=shelfF))+
3    ggtitle("Calories vs Grams of fat")+
4    xlab("fat/serving (g)")+ylab("calories/serving")+
5    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
6    theme(legend.justification=c(1,0), legend.position=c(1,0)
7    scale_color_discrete(name="Shelf")+
8    scale_shape_discrete(name="Shelf")
9 newplot
```

Notice you need BOTH scale commands.

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - set legend title



Plotting with *ggplot2* - Modifying the legend

http://www.cookbook-r.com/Graphs/Legends_(ggplot2)

Use long legend title.

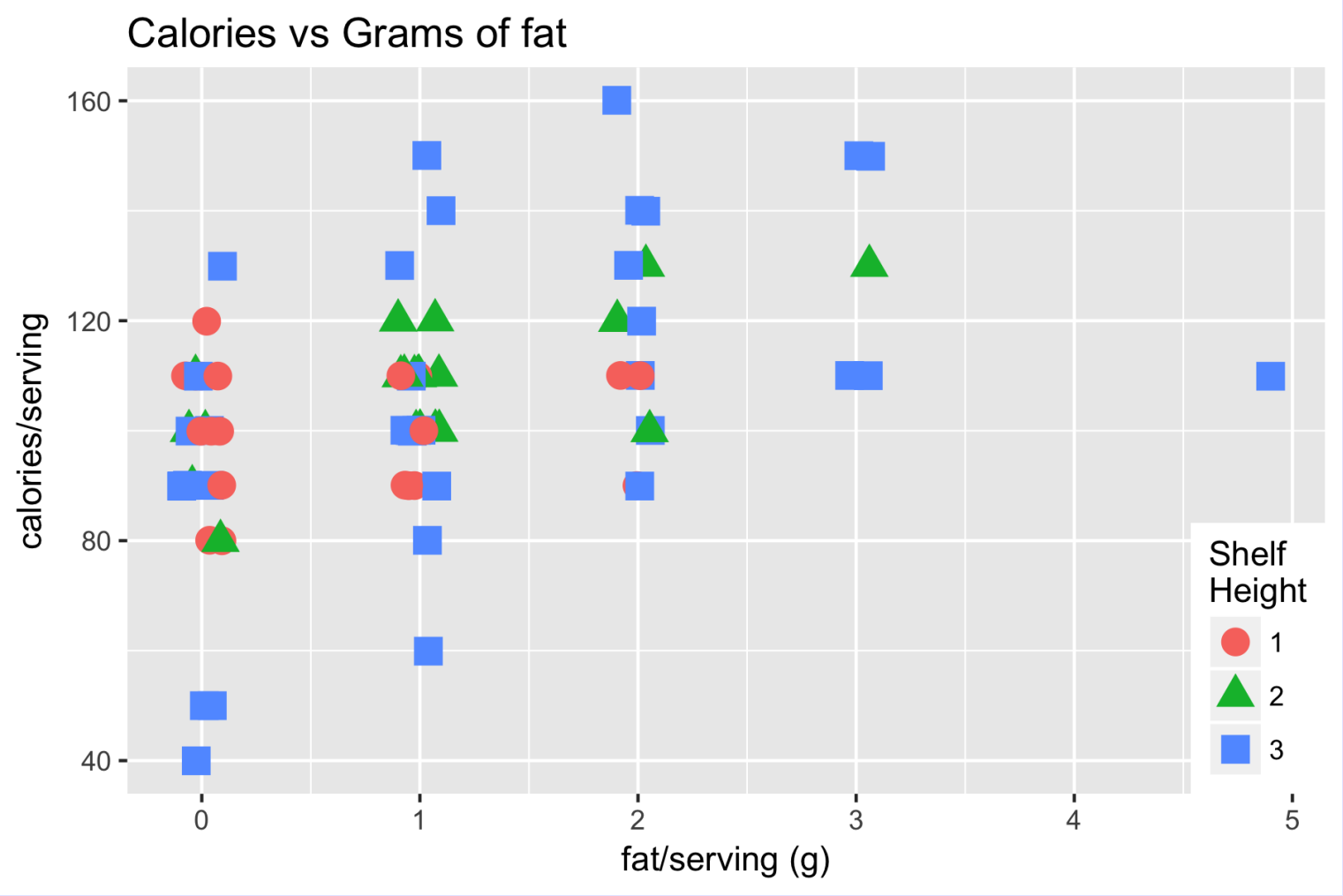
```
1 newplot <- ggplot(data=cereal, aes(x=fat, y=calories,
2                   shape=shelfF, color=shelfF))+
3    ggtitle("Calories vs Grams of fat")+
4    xlab("fat/serving (g)")+ylab("calories/serving")+
5    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
6    theme(legend.justification=c(1,0), legend.position=c(1,0)
7    scale_color_discrete(name="Shelf\nHeight")+
8    scale_shape_discrete(name="Shelf\nHeight")
9 newplot
```

Notice you need BOTH scale commands.

Same method used for long axis labels, title, etc.

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - set legend title



Plotting with *ggplot2* - Modifying the legend

http://www.cookbook-r.com/Graphs/Legends_(ggplot2)

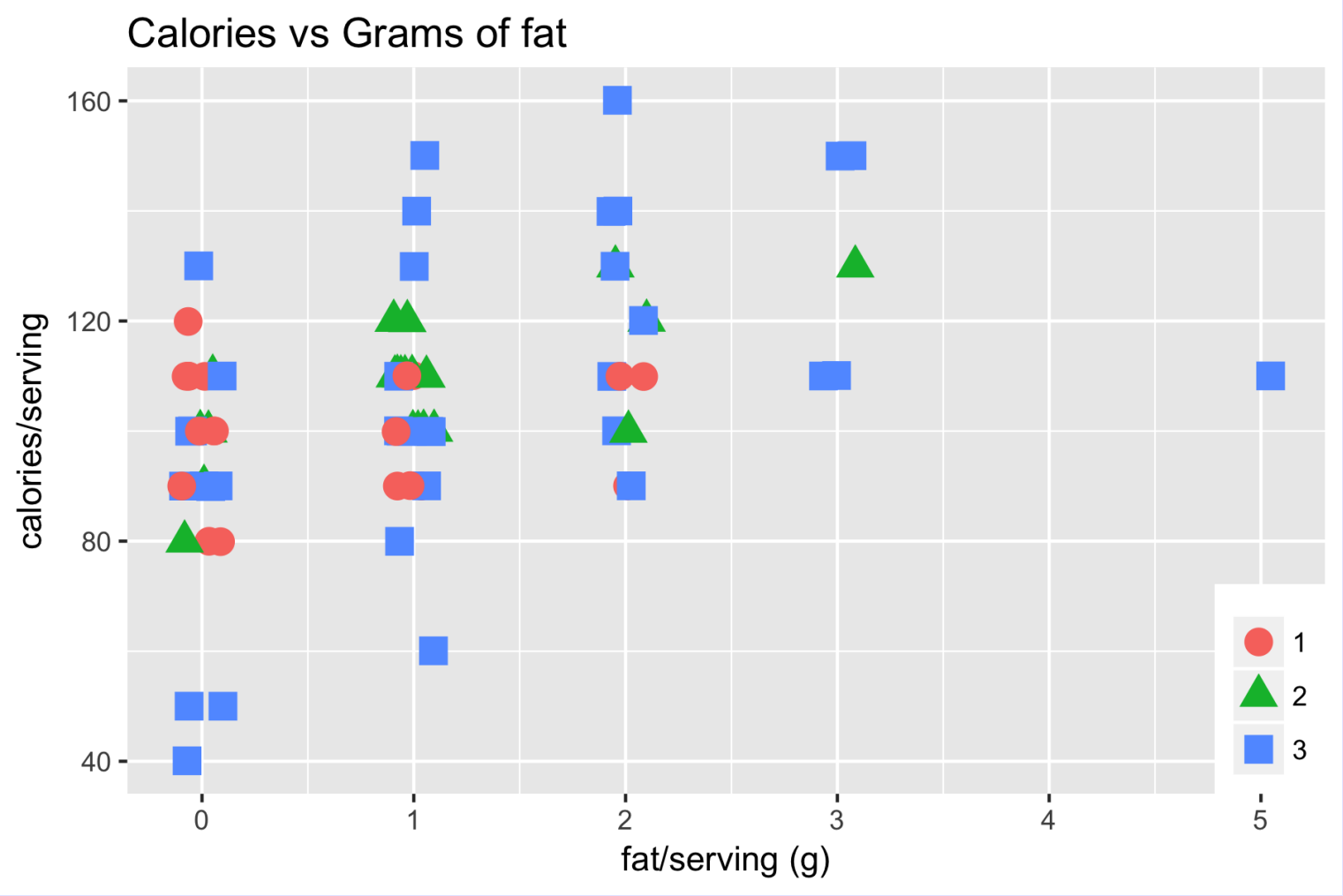
Remove legend title.

```
1 newplot <- ggplot(data=cereal, aes(x=fat, y=calories,
2                     shape=shelfF, color=shelfF))+
3    ggtitle("Calories vs Grams of fat")+
4    xlab("fat/serving (g)")+ylab("calories/serving")+
5    geom_jitter(position=position_jitter(w=.1, h=.1), size=4)
6    theme(legend.justification=c(1,0), legend.position=c(1,0)
7    scale_color_discrete(name=NULL)+
8    scale_shape_discrete(name=NULL)
9 newplot
```

Notice you need BOTH scale commands.

Plotting with *ggplot2* - Scatterplot

Create a plot of calories vs. grams of fat - remove legend title

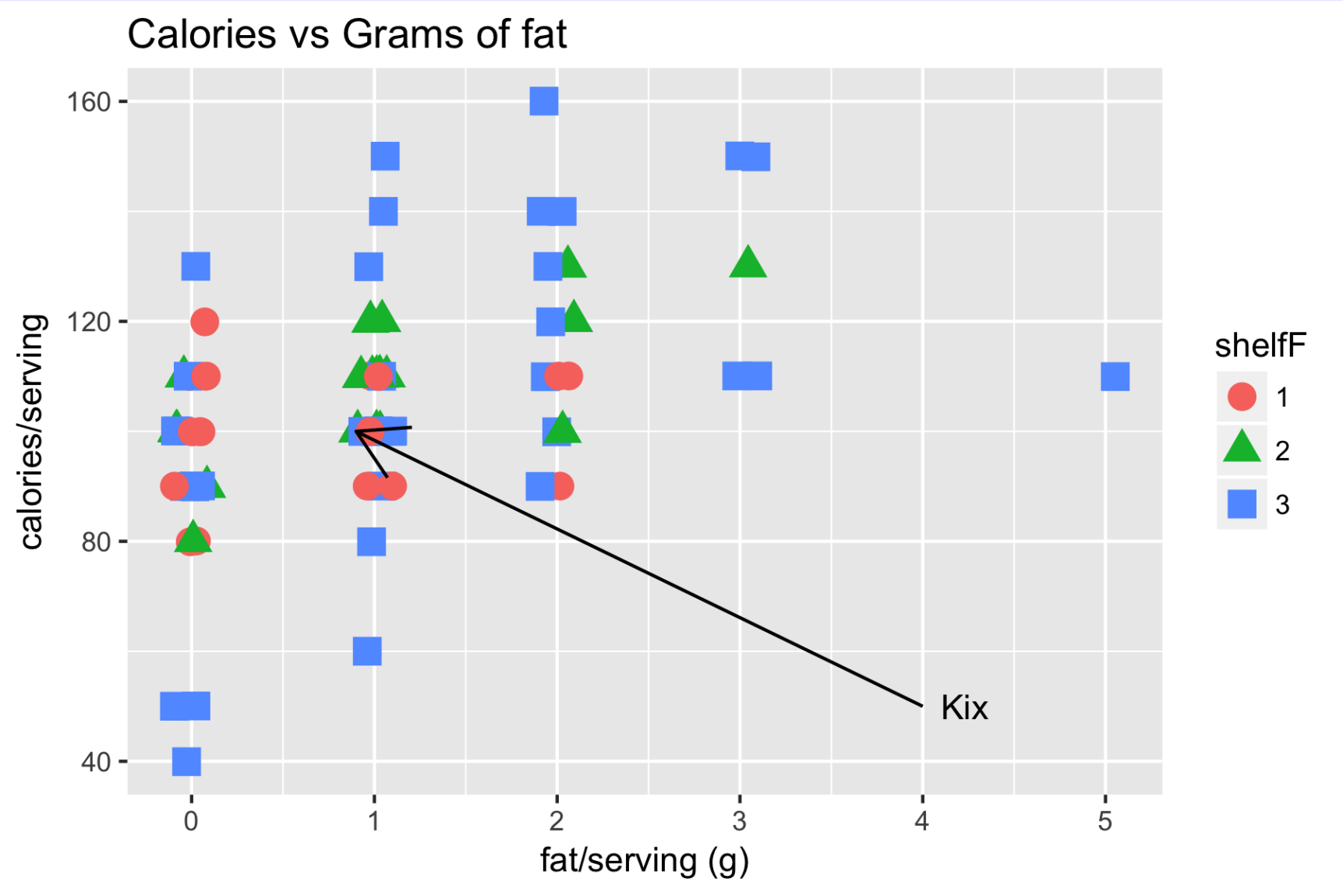


Plotting with *ggplot2* - Scatterplot

Add a call out

```
1  fc <- cereal[ cereal$name == "Kix",]
2  fc
3
4  newplot <- ggplot(cereal, aes(x=fat, y=calories,
5                    shape=shelfF, color=shelfF))+
6     ggtitle("Calories vs Grams of fat")+
7     xlab("fat/serving (g)")+ylab("calories/serving")+
8     geom_jitter(position=position_jitter(w=.1, h=.1),size=4)+
9     annotate("segment",
10              x=4, y=50,
11              xend=fc$fat-.1, yend=fc$calories,
12              arrow=arrow(ends="last")
13              )+
14    annotate("text",
15              x=4.1, y=50,
16              label="Kix",
17              hjust=0)
18 newplot
```

Plotting with *ggplot2* - Scatterplot



Plotting with *ggplot2* - Cheat sheets

- `http://www.cookbook-r.com/Graphs/Legends_(ggplot2)`
- `http://zevross.com/blog/2014/08/04/beautiful-plotting-in-r-a-ggplot2-cheatsheet-3/`
- *Rstudio* → Help → Cheetsheets

Plotting with *ggplot2* - Exercise I

Return to the Birds 'n Butts dataset.

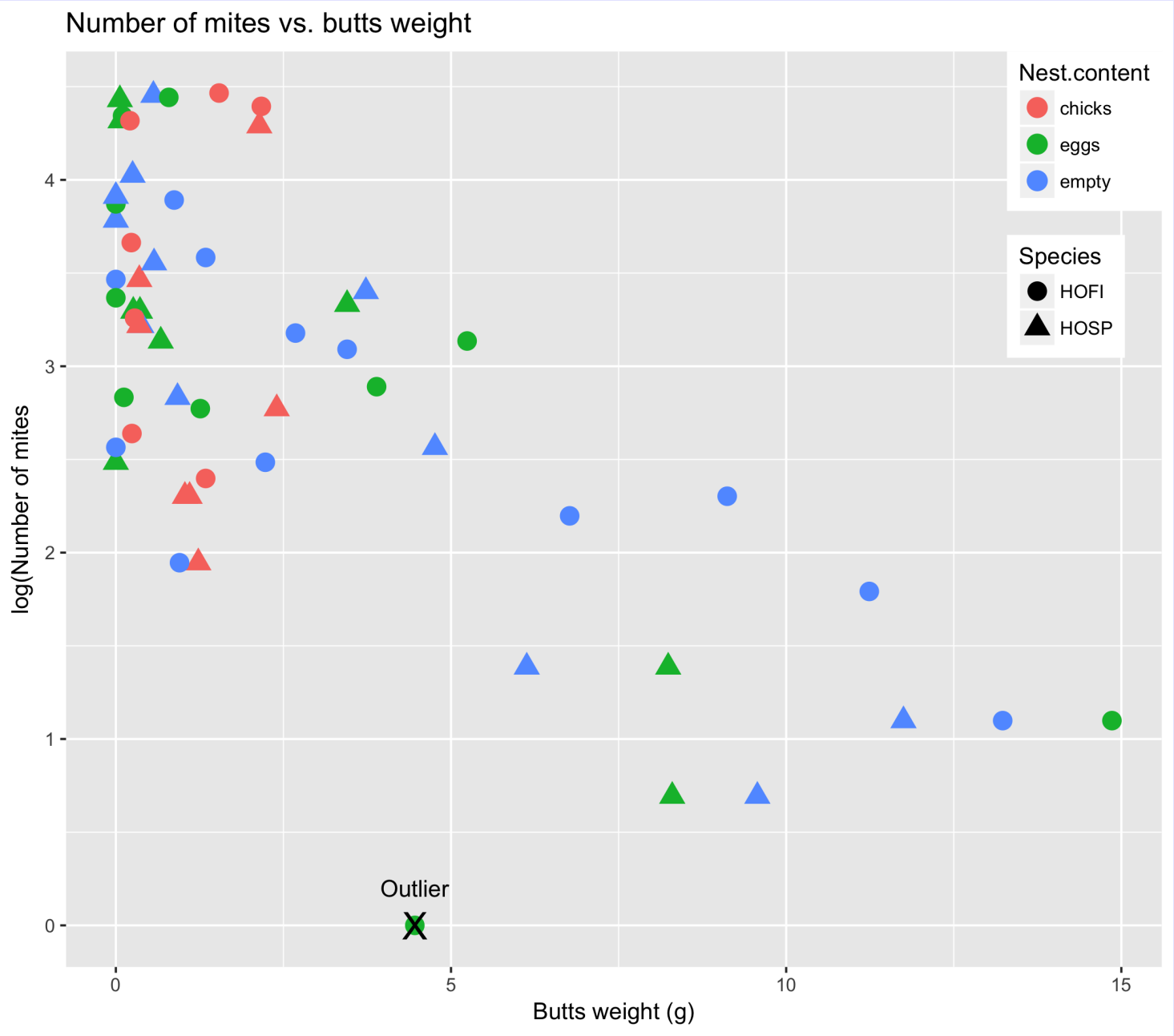
Plot the log(number of parasites) vs. butts weight in nest with a different symbol/colour for a point depending on the species and nest content.

- Use different symbol for each species
- Use different color for each nest content type
- Don't forget the legend.
- Add a callout for the outlier.

Plotting with *ggplot2* - Exercise

```
1  prelimplotlog <- ggplot(data=butts,
2                      aes(x=Butts.weight, y=log.mites,
3                          shape=Species, color=Nest.content))+
4       ggtitle("Number of mites vs. butts weight")+
5       xlab("Butts weight (g)")+ylab("log(Number of mites)')+
6       geom_point(size=4)+
7       annotate("point",
8                x=outlier$Butts.weight,
9                y=outlier$log.mites,
10               shape="X",size=6)+
11      annotate("text",
12               x=outlier$Butts.weight,
13               y=outlier$log.mites+.2,
14               label="Outlier")+
15      theme(legend.justification=c(1,1), legend.position=c(1
16 prelimplotlog
```

Plotting with *ggplot2* - Exercise



Plotting with *ggplot2* - Exercise II

Return to the Birds 'n Butts dataset.

Compute the mean number of mites by combination of species and egg contents along with the 95% confidence intervals (use the examples previously)

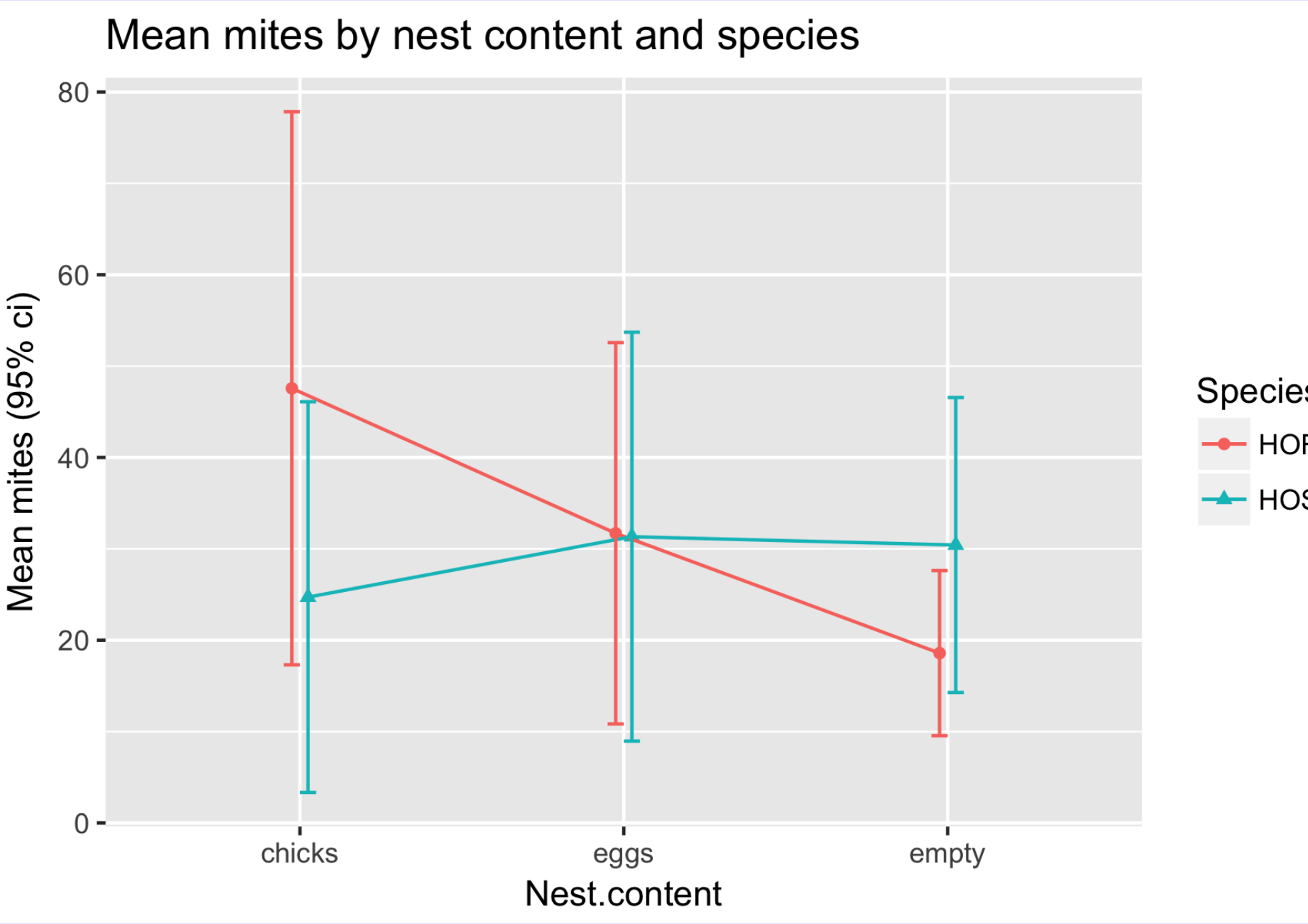
- Create a profile plot (mean by nest contents by species) with and without dodging.

Plotting with *ggplot2* - Exercise

```
report <- plyr::ddply(butts, c("Species","Nest.content"),
    sf.simple.summary, variable="Number.of.mites", crd=TRUE)
report

newplot <- ggplot2::ggplot(report, aes(x=Nest.content, y=mea
                                       group=Species, color=
   ggtitle("Mean mites by nest content and species")+
   geom_point( position=position_dodge(w=0.1))+
   geom_line(position=position_dodge(w=0.1))+
   geom_errorbar(aes(ymin=lcl, ymax=ucl), width=.1, positio
   ylab("Mean mites (95% ci)")
newplot
```

Plotting with *ggplot2* - Exercise

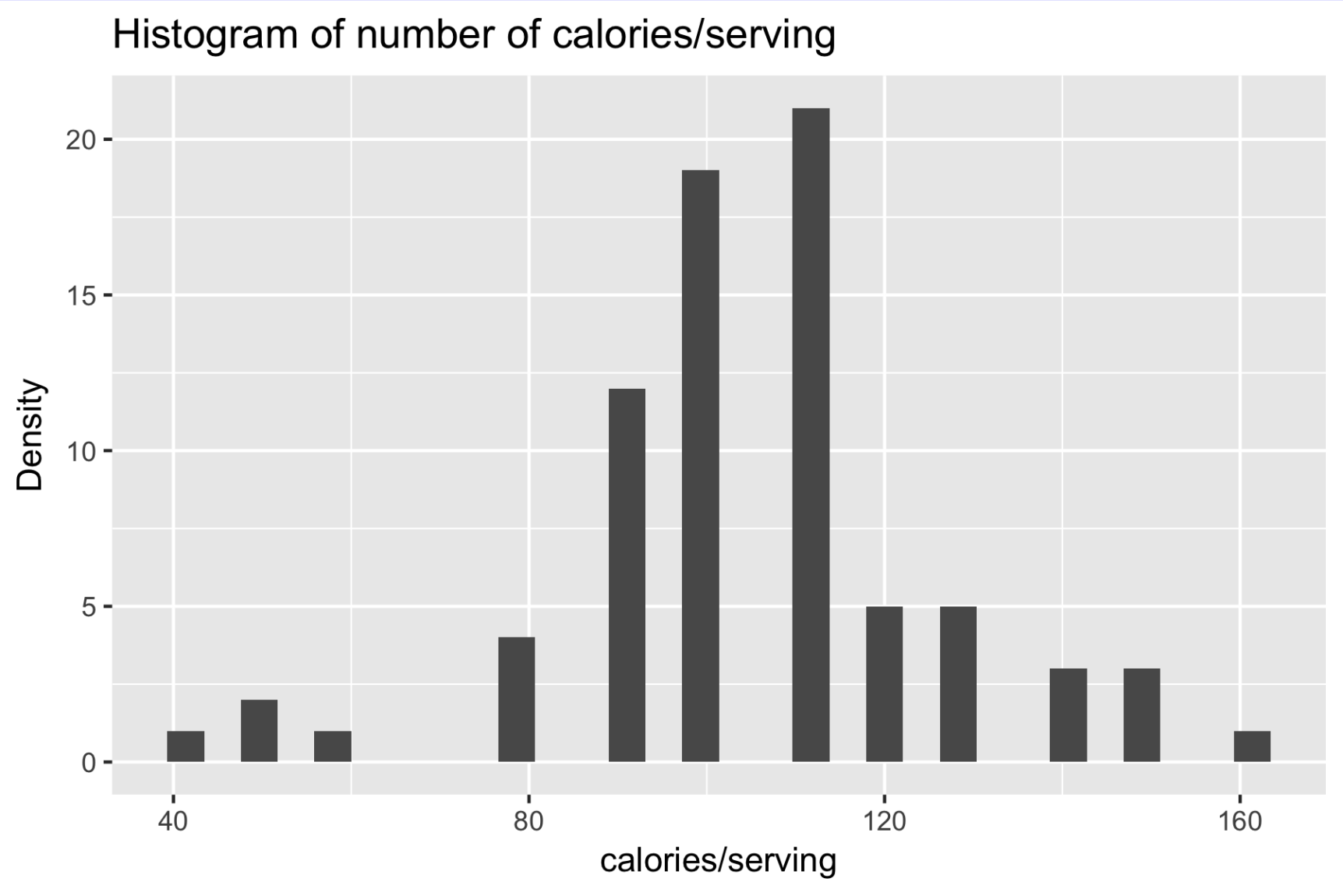


Plotting with *ggplot2* - Histograms

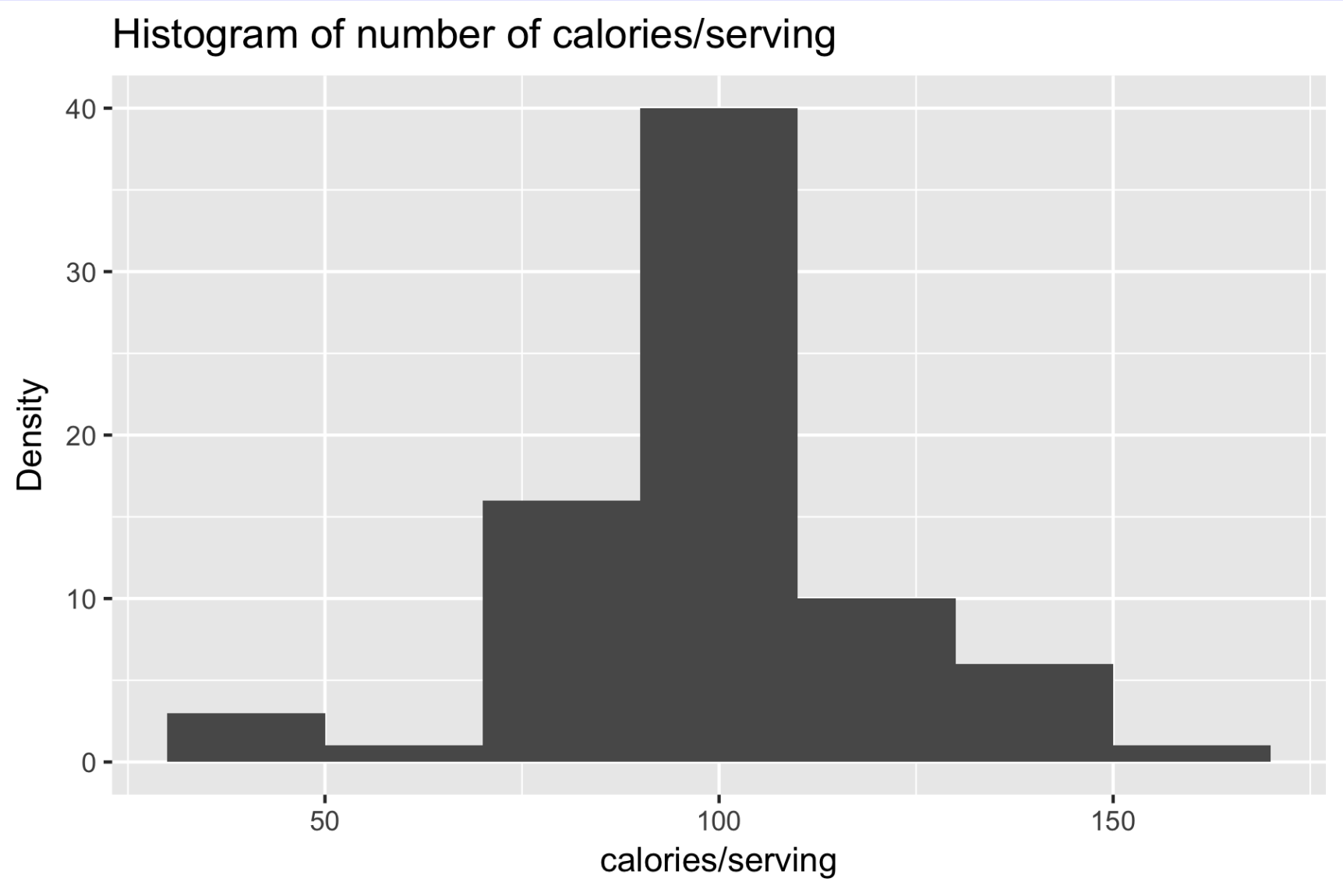
Histograms use the *geom_hist()* geometry. Note only *x* aesthetics used.

```
1  histplot <- ggplot(data=cereal, aes(x=calories))+
2    ggtitle("Histogram of number of calories/serving")+
3    xlab("calories/serving")+ylab("Density")+
4    geom_histogram()
5  histplot
6
7  histplot <- ggplot(data=cereal, aes(x=calories))+
8    ggtitle("Histogram of number of calories/serving")+
9    xlab("calories/serving")+ylab("Density")+
10   geom_histogram(binwidth=20 )
11 histplot
```

Plotting with *ggplot2* - Histograms



Plotting with *ggplot2* - Histograms



Plotting with *ggplot2* - Histograms

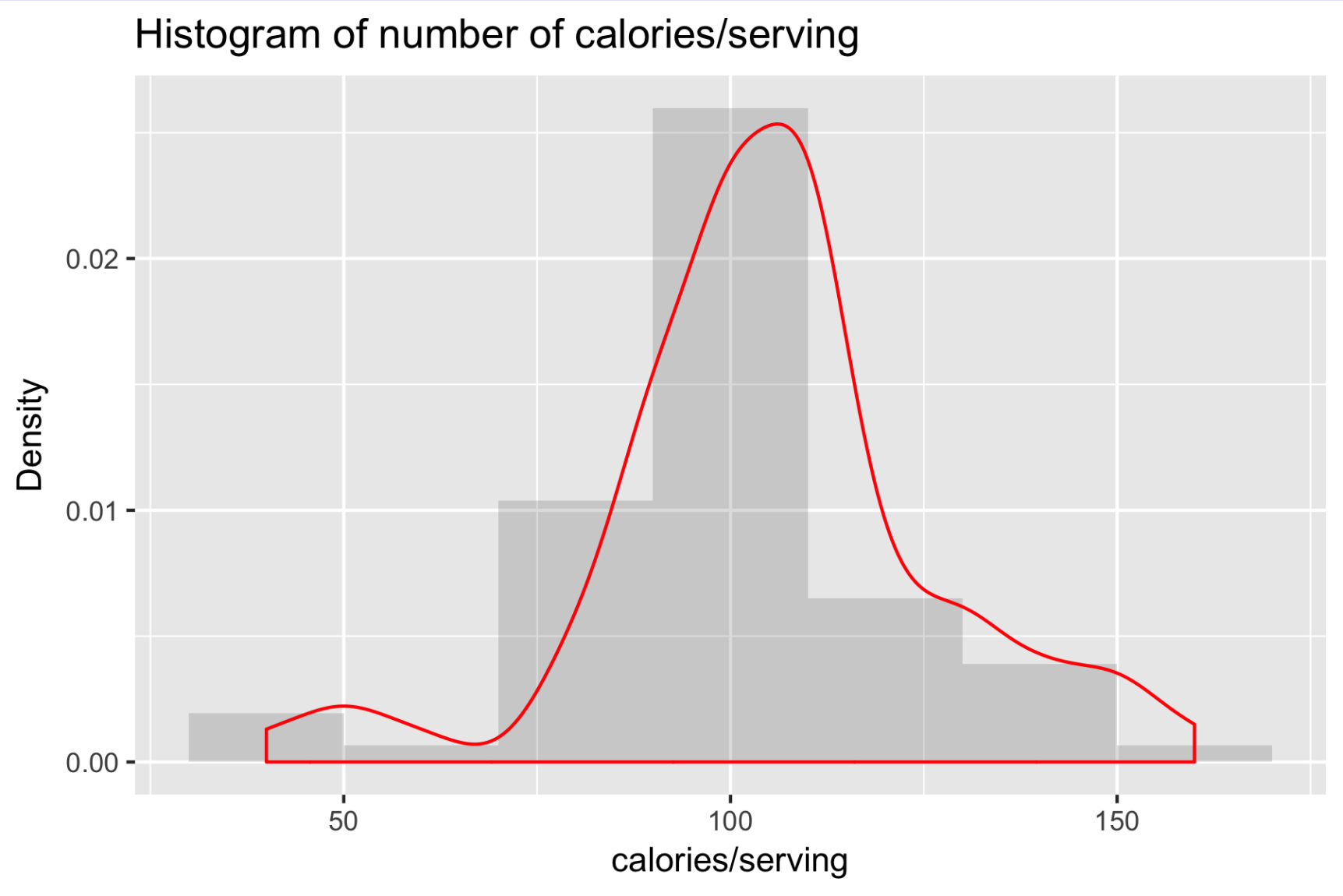
Histograms use the *geom_hist()* geometry. Note only *x* aesthetics used. Adding a non-parametric density estimate.

```
1 histplot <- ggplot(data=cereal, aes(x=calories))+
2   ggtitle("Histogram of number of calories/serving")+
3   xlab("calories/serving")+ylab("Density")+
4   geom_histogram(aes(y=..density..),binwidth=20,alpha=0.2)+
5   geom_density(color='red')
6 histplot
```

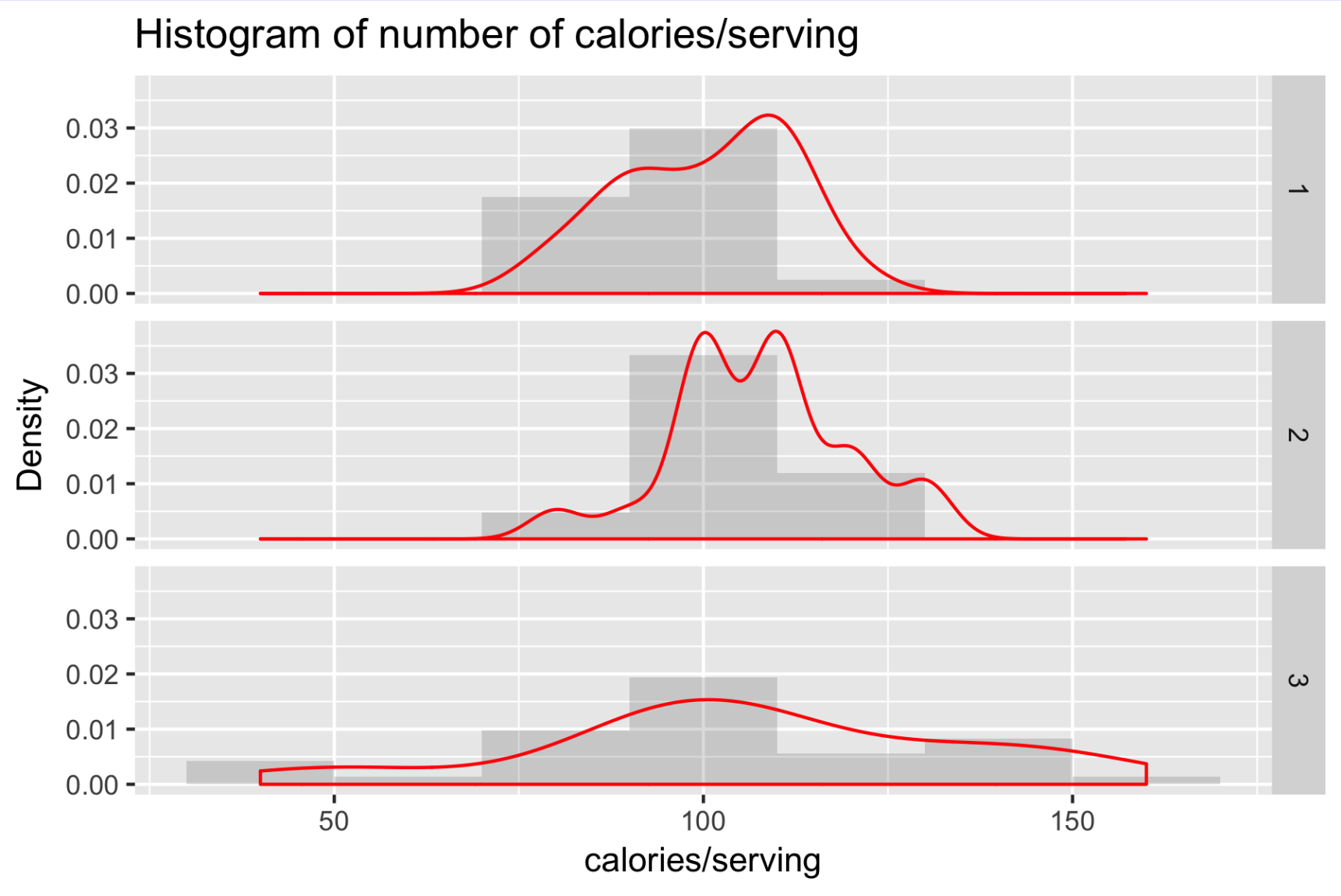
Example of using *facets* - more will be covered later

```
1 histplot <- ggplot(data=cereal, aes(x=calories))+
2   ggtitle("Histogram of number of calories/serving")+
3   xlab("calories/serving")+ylab("Density")+
4   geom_histogram(aes(y=..density..),binwidth=20,alpha=0.2)+
5   geom_density(color='red')+
6   facet_grid(shelfF~ .)
7 histplot
```

Plotting with *ggplot2* - Histograms



Plotting with *ggplot2* - Histograms

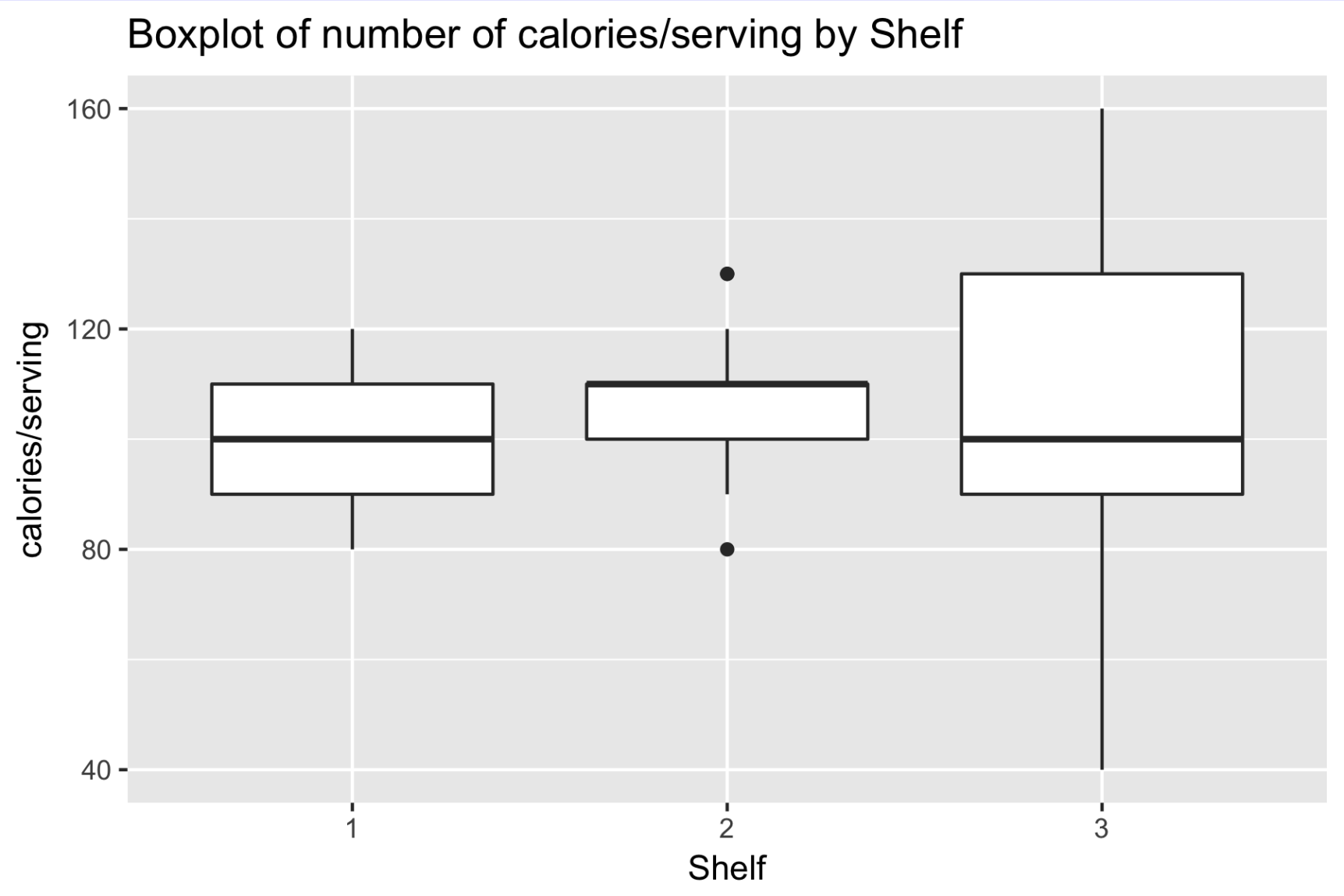


Plotting with *ggplot2* - Boxplots

Side-by-side box plots

```
1 boxplot <- ggplot(data=cereal, aes(x=shelfF, y=calories))+
2   ggtitle("Boxplot of number of calories/serving by Shelf")+
3   ylab("calories/serving")+xlab("Shelf")+
4   geom_boxplot()
5 boxplot
```

Plotting with *ggplot2* - Boxplots



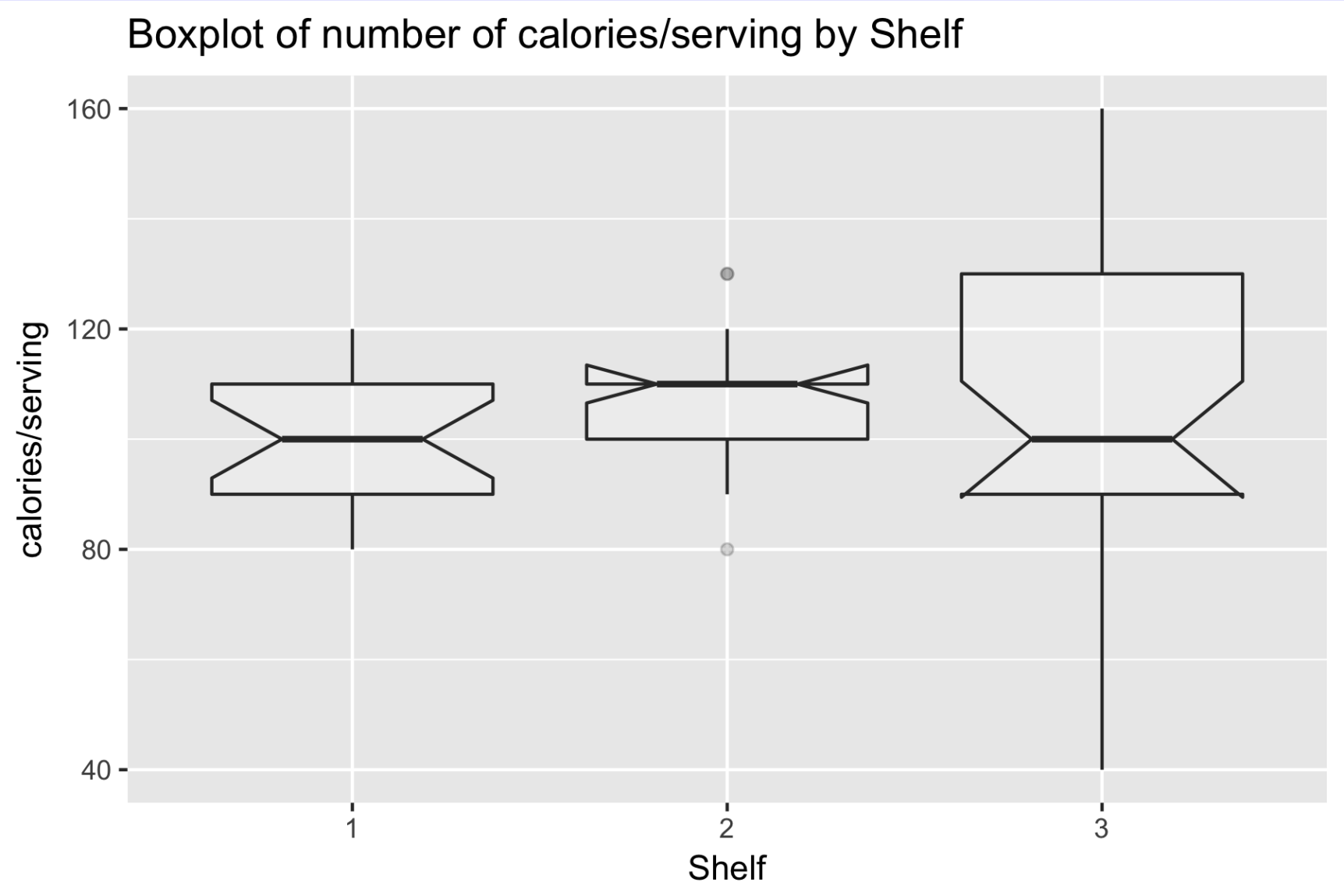
Plotting with *ggplot2* - Boxplots

Side-by-side box plots. Change transparency and add notches.

```
1 boxplot <- ggplot(data=cereal, aes(x=shelfF, y=calories))+
2   ggtitle("Boxplot of number of calories/serving by Shelf")+
3   ylab("calories/serving")+xlab("Shelf")+
4   geom_boxplot(alpha=0.2, notch=TRUE)
5 boxplot
```

Notched box-plots provide a way to compare if population MEDIANS are approximately equal across multiple groups.

Plotting with *ggplot2* - Boxplots



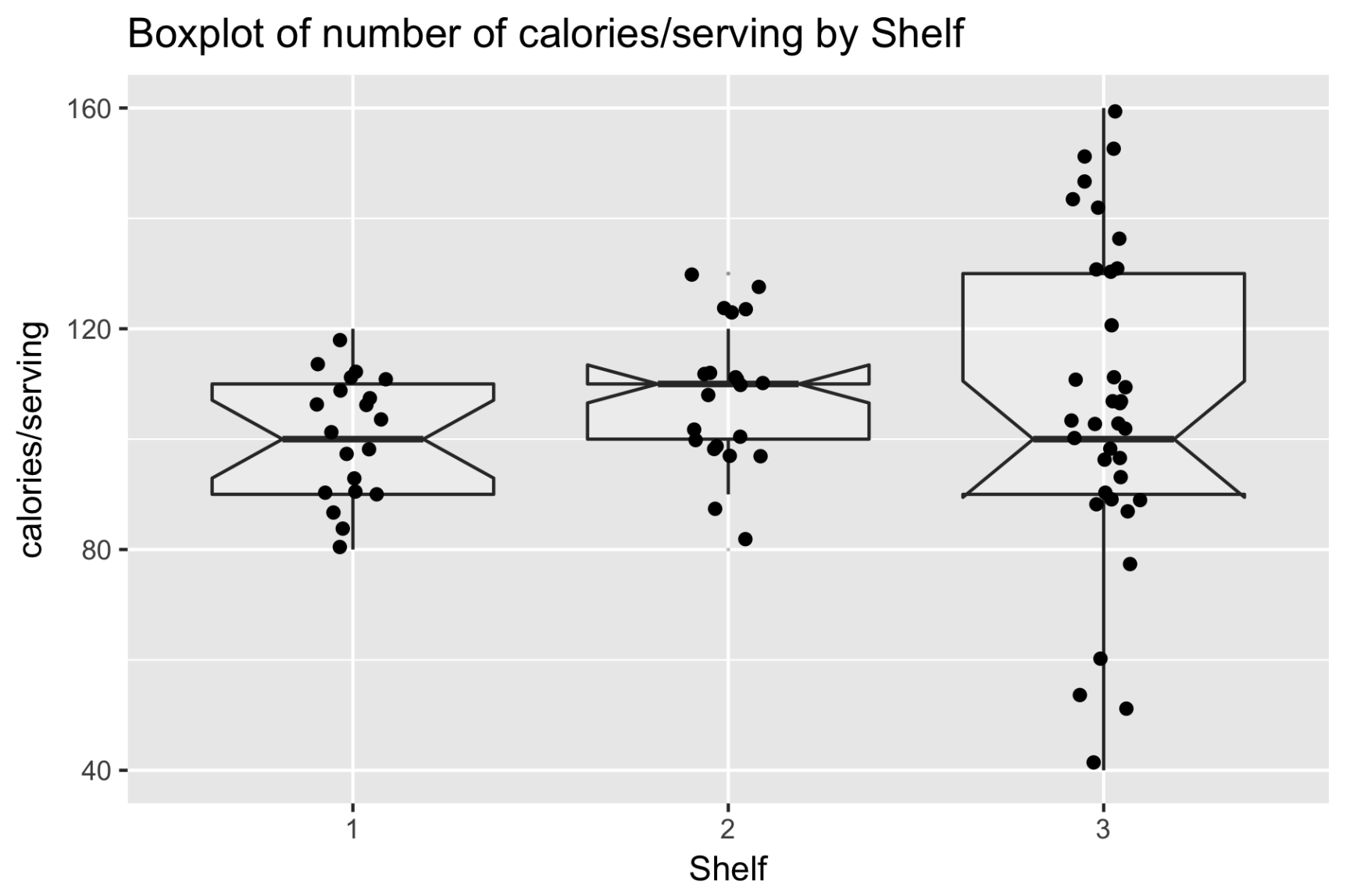
Plotting with *ggplot2* - Boxplots

Side-by-side box plots with data points overlaid

```
1 boxplot <- ggplot(data=cereal, aes(x=shelfF, y=calories))+
2   ggtitle("Boxplot of number of calories/serving by Shelf")+
3   ylab("calories/serving")+xlab("Shelf")+
4   geom_boxplot(alpha=0.2, notch=TRUE, outlier.size=0)+
5   geom_point(position=position_jitter(width=0.1))
6 boxplot
```

'CAUTION: If there are outliers, you might get double plotting. Set *outlier.size=0* in *geom_boxplot()*)

Plotting with *ggplot2* - Boxplots



Usually only used for screening data for outliers.

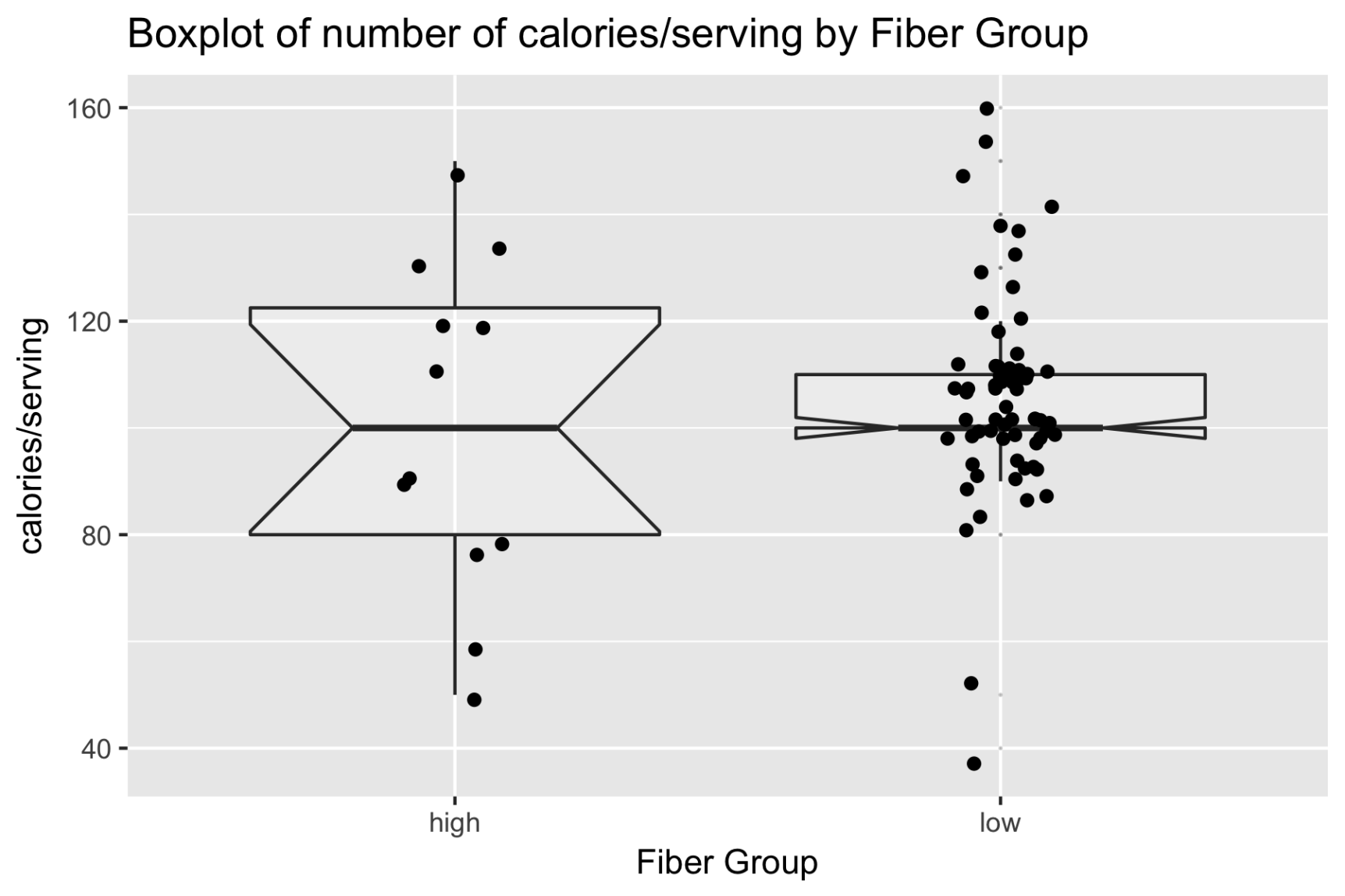
Plotting with *ggplot2* - Boxplots

Side-by-side box plots with data points overlaid and order of boxes changed.
Usual to set up a **FACTOR** (more on this later) with the correct ordering

```
1  cereal$fiber.grp <- recode(cereal$fiber,
2                  "lo:3='low'; 3:hi='high'")
3  xtabs(~fiber.grp, data=cereal)
4  cereal$fiber.grpF <- factor(cereal$fiber.grp)
5
6  boxplot <- ggplot(data=cereal,
7             aes(x=fiber.grpF, y=calories))+
8    ggtitle("Boxplot of number of calories/serving by Fiber Gr
9    xlab("calories/serving")+ylab("Fiber Group")+
10   geom_boxplot(alpha=0.2, notch=TRUE, outlier.size=0)+
11   geom_point(position=position_jitter(width=0.1))
12 boxplot
```

Plotting with *ggplot2* - Boxplots

Notice that default order is ALPHABETICAL.



Box plots usually only used for screening data for outliers.

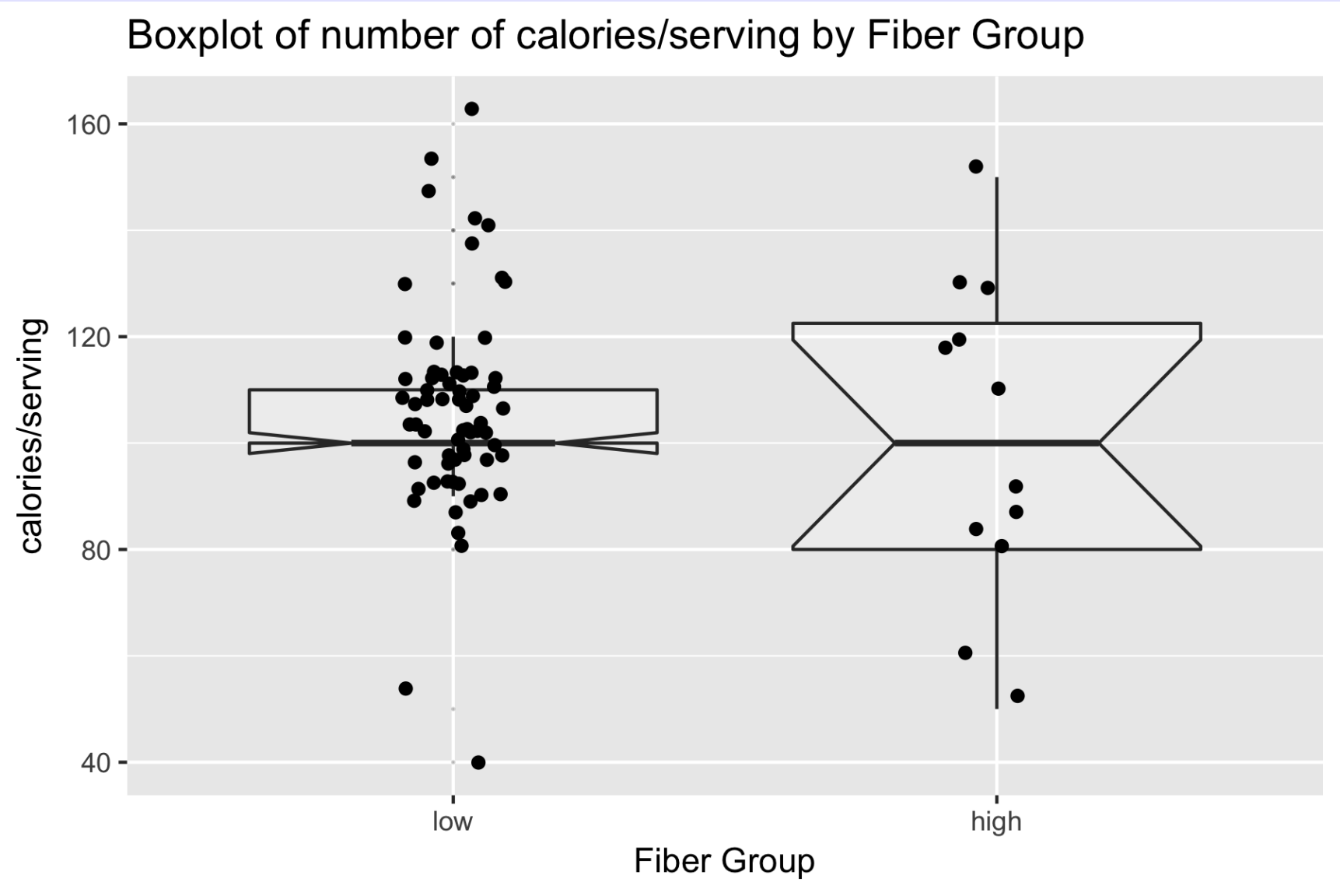
Plotting with *ggplot2* - Boxplots

Side-by-side box plots with data points overlaid and order of boxes changed.
Usual to set up a **FACTOR** (more on this later) with the correct ordering

```
1  cereal$fiber.grpF <- factor(cereal$fiber.grp,
2                              levels=c("low","high"), order=TRUE)
3
4  boxplot <- ggplot(data=cereal, aes(x=fiber.grpF, y=calories)
5    ggtitle("Boxplot of number of calories/serving by Fiber Gr
6    xlab("calories/serving")+ylab("Fiber Group")+
7    geom_boxplot(alpha=0.2, notch=TRUE, outlier.size=0)+
8    geom_point(position=position_jitter(width=0.1))
9  boxplot
```

Plotting with *ggplot2* - Boxplots

Notice that order is specified in previous *factor()* function.



Box plots usually only used for screening data for outliers.

Plotting with *ggplot2* - Exercise

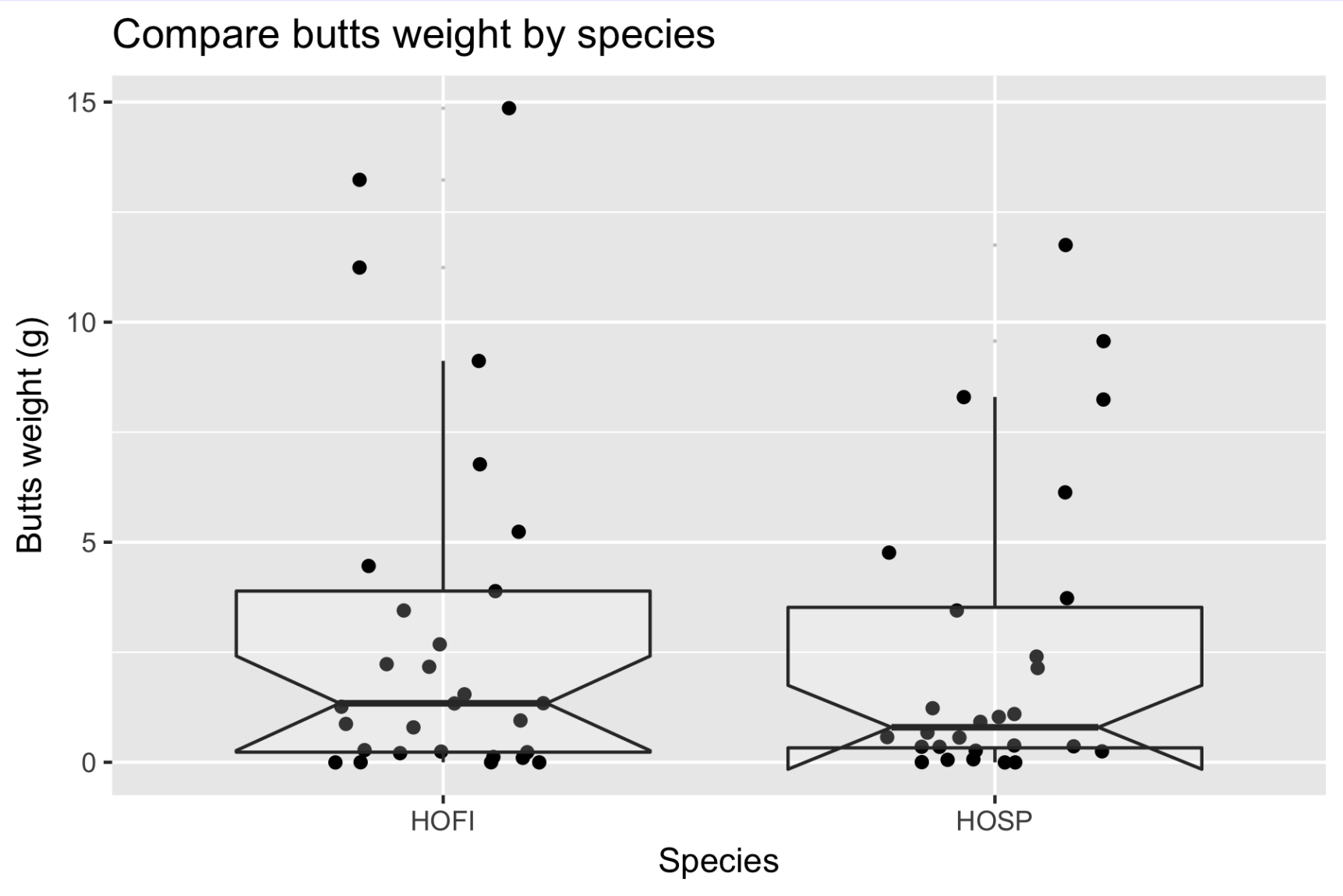
Return to the Birds 'n Butts dataset.

Compare the number of parasites and butts weight by species and/or nest content using box-plots. Try both the regular and log() scales. What do you conclude?

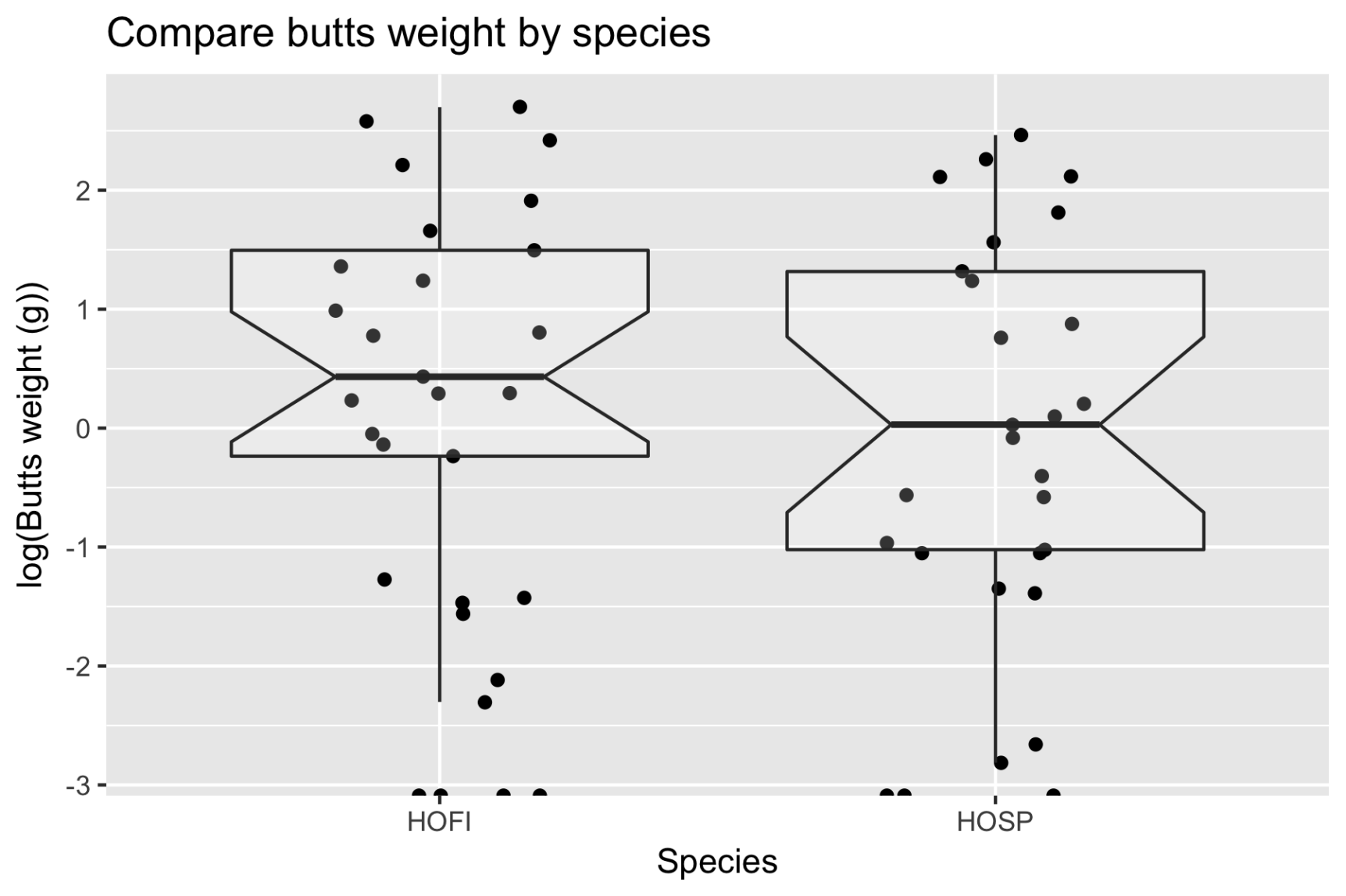
Plotting with *ggplot2* - Exercise

```
1  birdbox <- ggplot(data=butts,
2                    aes(x=Species, y=Butts.weight))+
3               ggtitle("Compare butts weight by species")+
4               xlab("Species")+ylab("Butts weight (g)")+
5               geom_point(position=position_jitter(w=0.2))+
6               geom_boxplot(notch=TRUE, alpha=0.2,outlier.size=
7  birdbox
```

Plotting with *ggplot2* - Exercise



Plotting with *ggplot2* - Exercise



Perhaps on the log-scale is better comparison (except for log(0)?)

Plotting with *ggplot2* - Bar/line charts with error bars

```
source("schwarz.functions.r")  # load some helper functions

# Compute the mean calories/serving by fiber group with se
sumstat.all <- sf.simple.summary(cereal, "calories",
                        crd=TRUE)
sumstat.all

library(plyr)
sumstat.shelf <- ddply(cereal, "shelfF", sf.simple.summary,
                    variable="calories", crd=TRUE)
sumstat.shelf
```

Plotting with *ggplot2* - Bar/line charts with error bars

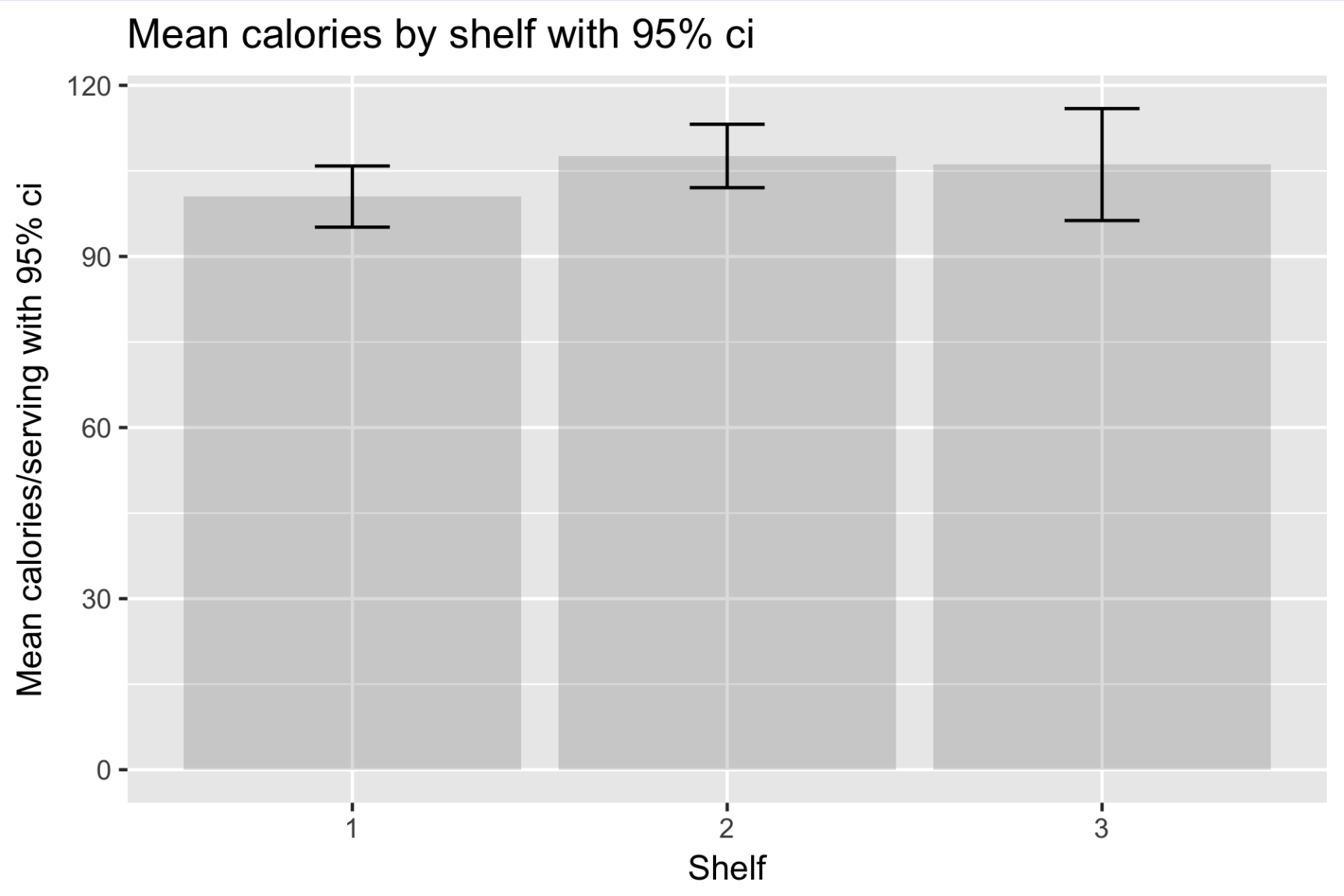
```
> sumstat.all
         n      nmiss       mean         sd         se
 77.000000   0.000000 105.064935  21.620128   2.463842 100.1

> sumstat.shelf
  shelfF  n nmiss     mean       sd       se       lcl
1      1 20     0 100.5000 11.45931 2.562380  95.13688 105.8
2      2 21     0 107.6190 12.20851 2.664114 102.06180 113.1
3      3 36     0 106.1111 29.01012 4.835021  96.29550 115.9
```

Plotting with *ggplot2* - Bar/line charts with error bars

```
# Side-by-side bar charts
cerealbar <- ggplot(data=sumstat.shelf,
                    aes(x=shelfF, y=mean))+
   ggtitle("Mean calories by shelf with 95% ci")+
   xlab("Shelf")+ylab("Mean calories/serving with 95% ci")+
   geom_bar(stat="identity", alpha=0.2)+
   geom_errorbar(aes(ymin=lcl, ymax=ucl), width=0.2)
cerealbar
```

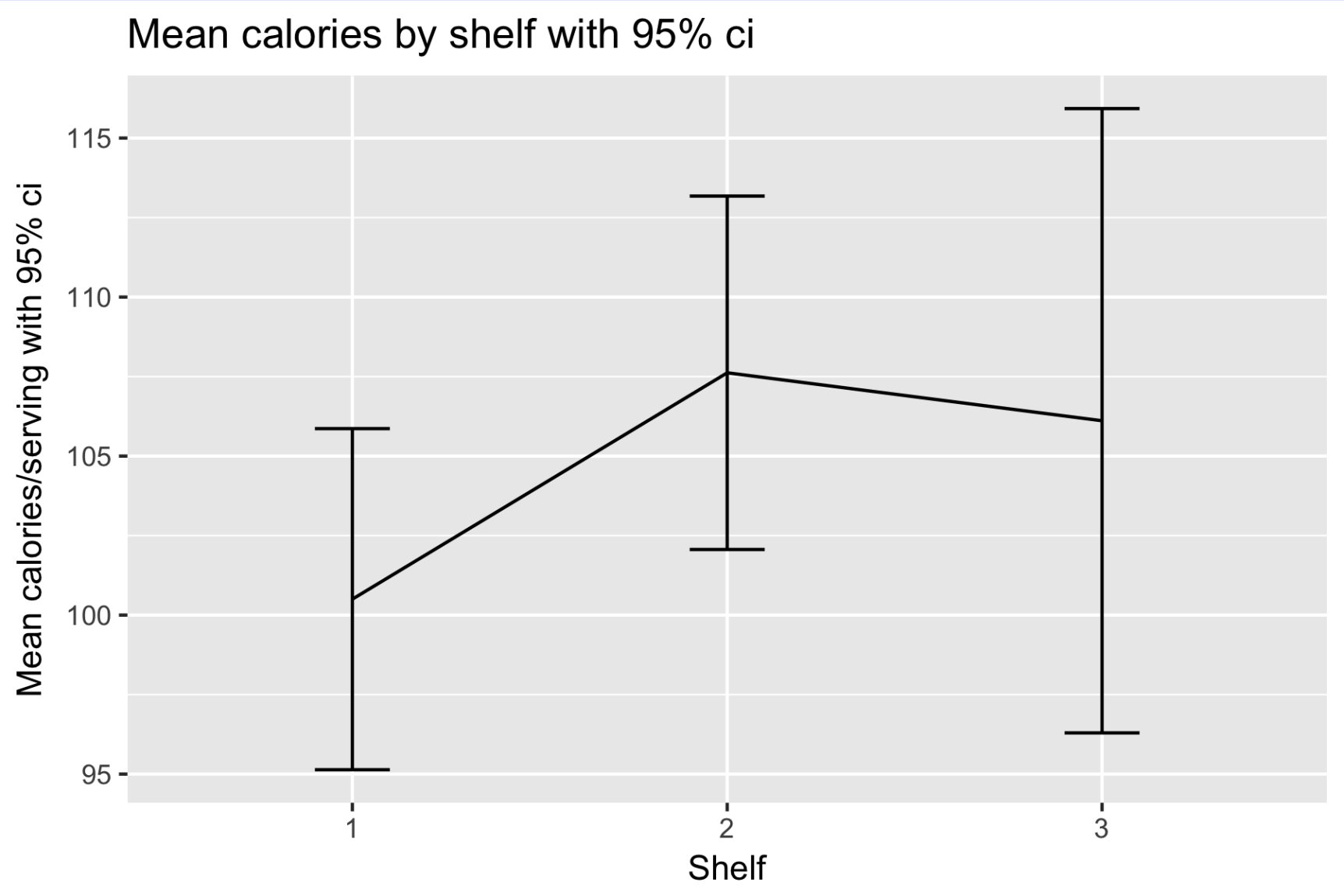
Plotting with *ggplot2* - Bar/line charts with error bars



Plotting with *ggplot2* - Bar/line charts with error bars

```
# A line chart is similar, except you don't need to start
# the bars at zero
cerealline <- ggplot(data=sumstat.shelf,
                             aes(x=shelfF, y=mean))+
   ggtitle("Mean calories by shelf with 95% ci")+
   xlab("Shelf")+ylab("Mean calories/serving wth 95% ci")+
   geom_line(aes(group=1))+
   geom_errorbar(aes(ymin=lcl, ymax=ucl), width=0.2)
cerealline
```

Plotting with *ggplot2* - Bar/line charts with error bars



Plotting with *ggplot2* - Exercise

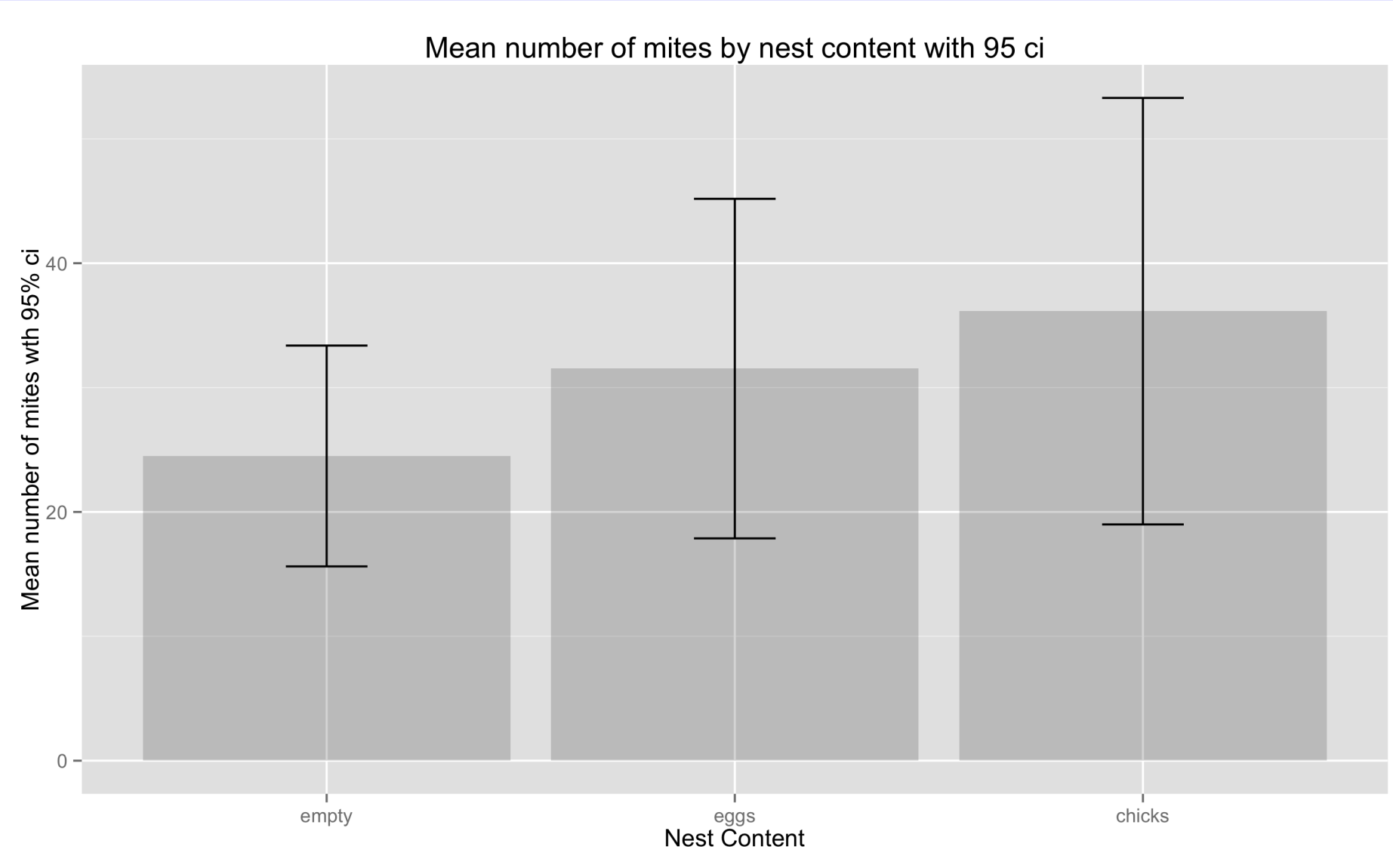
Return to the Birds 'n Butts dataset.

Summaryize the number of parasites and butts weight by nest content using the code given in the notes.
Compare the means using bar/line plots with confidence intervals.
Try both the regular and log() scales. What do you conclude?

Plotting with *ggplot2* - Exercise

```
1  library(plyr)
2  sumstat <- ddply(butts, "Nest.content", sf.simple.summary,
3                   variable="Number.of.mites", crd=TRUE)
4  sumstat$Nest.contentF <- factor(sumstat$Nest.content,
5                           levels=c("empty","eggs","chicks"),
6                           order=TRUE)
7  sumstat
8
9  nestbar <- ggplot(data=sumstat,
10                   aes(x=Nest.contentF, y=mean))+
11    ggtitle("Mean number of mites by nest content with 95% c
12    xlab("Nest Content")+ylab("Mean number of mites with 95%
13    geom_bar(stat="identity", alpha=0.2)+
14    geom_errorbar(aes(ymin=lcl, ymax=ucl), width=0.2)
15 nestbar
```

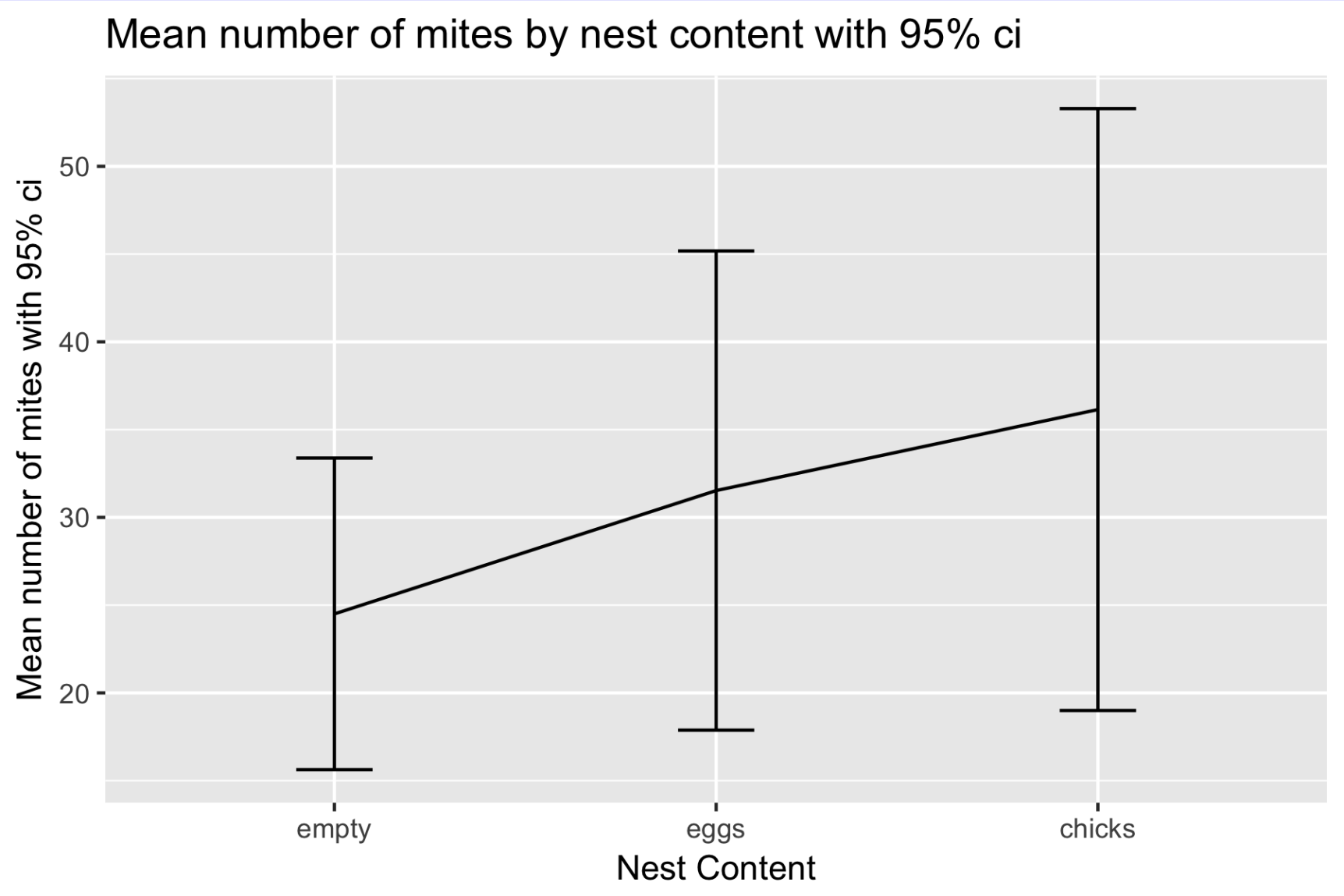
Plotting with *ggplot2* - Exercise



Plotting with *ggplot2* - Exercise

```
1  nestline <- ggplot(data=sumstat,
2                     aes(x=Nest.contentF, y=mean))+
3     ggtitle("Mean number of mites by nest content with 95% ci
4     xlab("Nest Content")+ylab("Mean number of mites with 95%
5     geom_line(aes(group=1))+
6     geom_errorbar(aes(ymin=lcl, ymax=ucl), width=0.2)
7  nestline
```

Plotting with *ggplot2* - Exercise

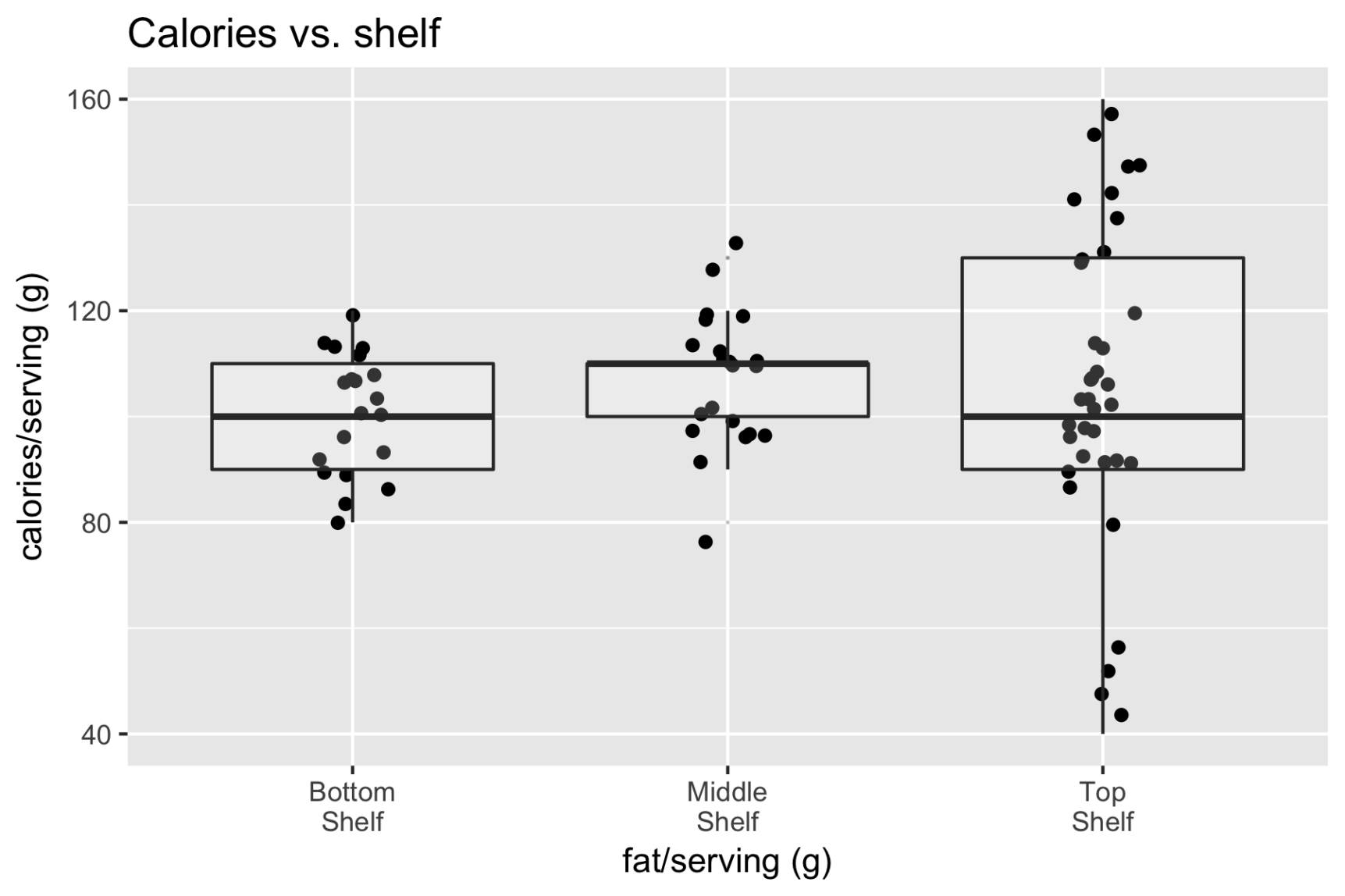


Plotting with *ggplot2* - Tick mark labels

Long tick mark labels

```
cereal$shelf.special <- car::recode(cereal$shelf,
                                    "1='Bottom\nShelf';
                                     2='Middle\nShelf';
                                     3='Top\nShelf'")
cereal$shelf.special <- gsub(" ", "\n", cereal$shelf.special
cerealplot <- ggplot(data=cereal, aes(x=shelf.special, y=cal
     ggtitle("Calories vs fat in each serving")+
     xlab("fat/serving (g)")+ylab("calories/serving (g)")+
     geom_point(position=position_jitter(width=0.1))+
     geom_boxplot( alpha=0.2, outlier.size=0)
cerealplot
```

Long tick mark labels



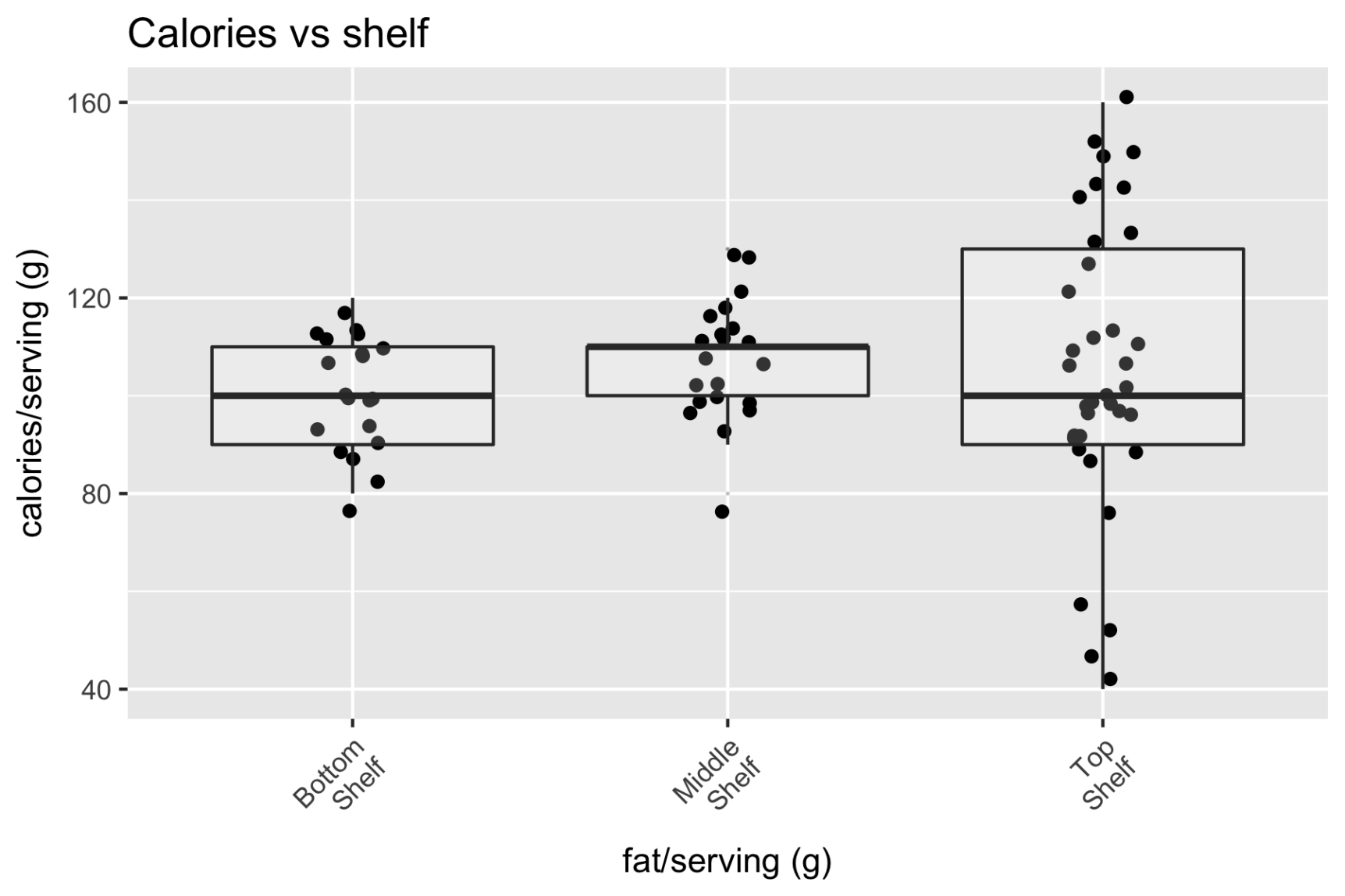
Plotting with *ggplot2* - Tick mark labels

Long tick mark labels that are rotated

```
cereal$shelf.special <- car::recode(cereal$shelf,
                                    "1='Bottom\nShelf';
                                     2='Middle\nShelf';
                                     3='Top\nShelf'")
cereal$shelf.special <- gsub(" ", "\n", cereal$shelf.special
cerealplot <- ggplot(data=cereal, aes(x=shelf.special, y=cal
     ggtitle("Calories vs shelf")+
     xlab("fat/serving (g)")+ylab("calories/serving (g)")+
     geom_point(position=position_jitter(width=0.1))+
     geom_boxplot( alpha=0.2, outlier.size=0)+
     theme(axis.text.x = element_text(angle = 45, hjust = 1)
cerealplot
```

Plotting with *ggplot2* - Tick mark labels

Long tick mark labels that are rotated



Plotting with *ggplot2* - Saving plots

Preferred:

```
ggsave(plot=cerealplot, file="blah.png",
       h=4, w=6, units="in", dpi=300)

ggsave(plot=cerealplot, file="blah.jpg",
       h=4, w=6, units="in", dpi=300)
```

Old fashioned methods:

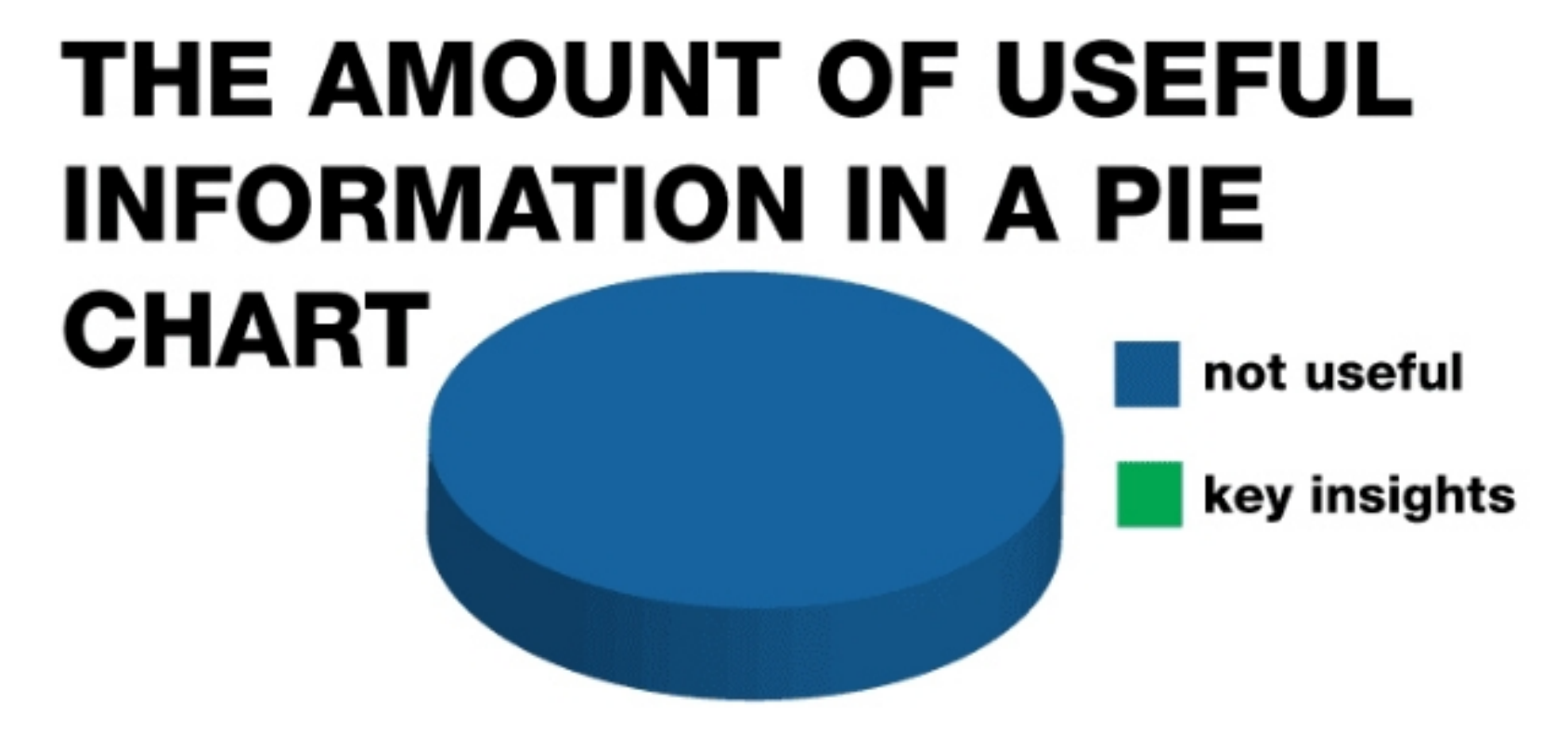
```
png("file.png", h=4, w=6, units="in", res=300)
  ggplot ...
def.off()
```

ggplot() Plotting - Summary

Avoid Base *R* graphics and use *ggplot()*!

- Build up the graph using bits and pieces
  - *ggplot(data=..., aes(x=..., y=...., shape=..., color=..., group=....))+*
  - *ggtitle()+xlab()+ylab()+*
  - *geom_point()* and *geom_jitter()*
  - *geom_histogram()*
  - *geom_boxplot()*
  - *geom_bar()*, *geom_line()*, *geom_errorbar()*
  - *facet_grid()* and *facet_wrap()* - advanced
  - *theme()*
- Once all pieces are together, then create the plot and adjust scales etc.

NEVER, NEVER, NEVER, NEVER, NEVER produce a pie chart.



http://www.perceptualedge.com/articles/08-21-07.pdf