Module 4: Bivariate and multivariate analysis

BIOSTAT 213: Introduction to Computing in the R Software Environment

Yea-Hung Chen, PhD, MS

Monday, October 14, 2019

Continuous versus continuous

Scatter plots

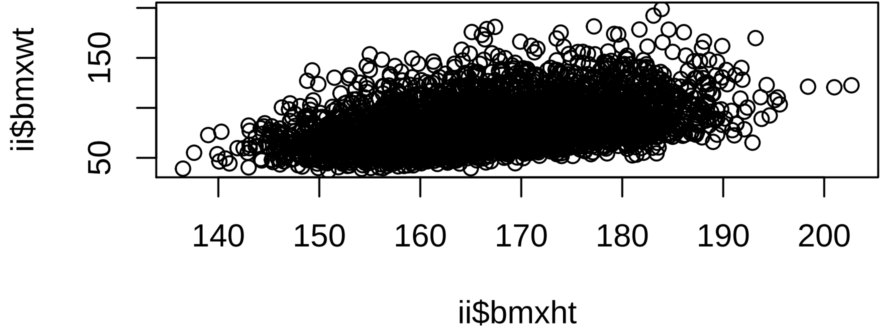
- to create a scatter plot of weight and height, with height on the y axis, use `plot()`:

```
plot(x=ii$bmxht,y=ii$bmxwt)
```

- equivalently, without argument names:

```
plot(ii$bmxht,ii$bmxwt)
```

Scatter plots



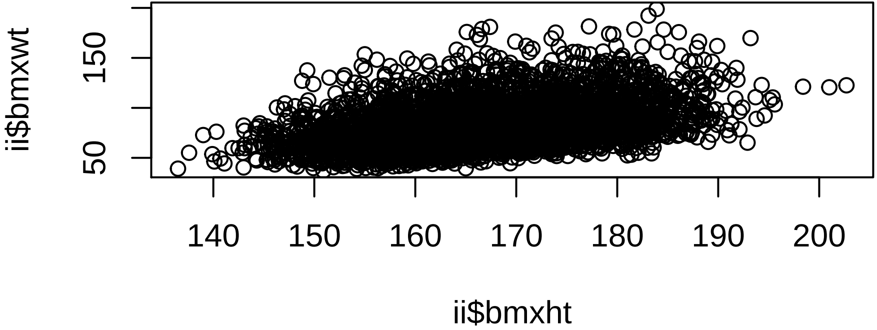
Scatter plots

- equivalently, we can use formula notation:

```
plot(ii$bmxwt~ii$bmxht)
```

- the tilde (~) means versus
- formula notation is used throughout R
- we'll see it throughout the day as we discuss plotting functions, hypothesis tests, and linear regression
- formula notation, in general: `y~x`

Scatter plots

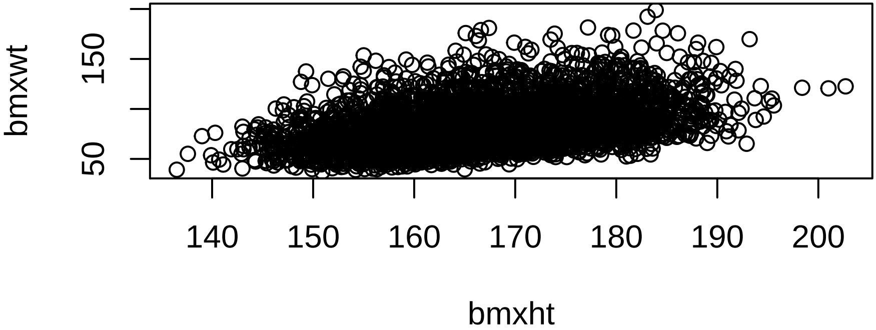


Scatter plots

- equivalently, using the `data` argument:

```
plot(bmxwt~bmxht,data=ii)
```

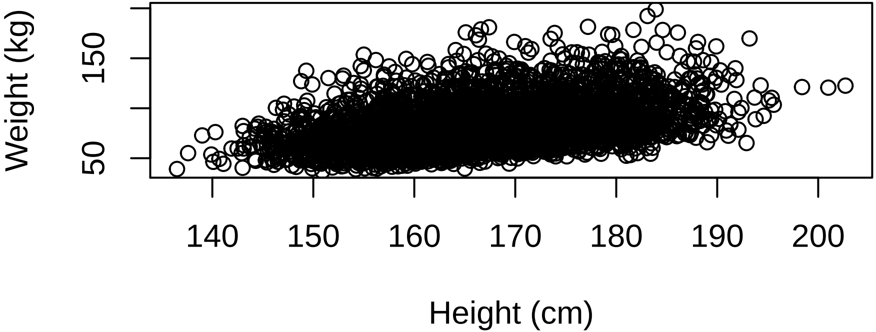
Scatter plots



Scatter plots

```
plot(ii$bmxwt~ii$bmxht,
     xlab='Height (cm)',
     ylab='Weight (kg)')
```

Scatter plots



Scatter plots

- what if we want to create the scatter plot using ggplot2?
- remember we have to load the library once per session:

```
library(ggplot2)
```

- remember also that the basic form of ggplot2 code involves 2 parts:

$$\text{variables or other dimensions} + \text{type of plot}$$

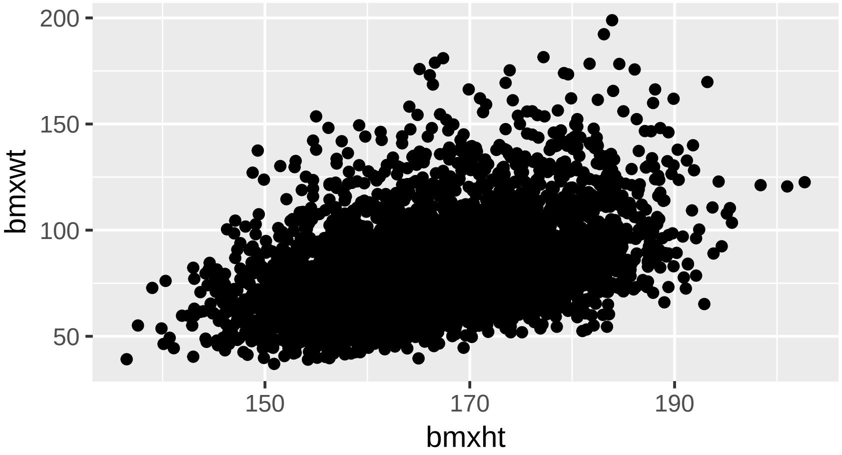
Scatter plots

- to create the scatter plot using ggplot2:

```
ggplot(mapping=aes(x=bmxht,y=bmxwt),data=ii)+
  geom_point()
```

- in the first part, we use the `aes()` function to specify an x variable and a y variable
- in the second part, we use `geom_point()` to indicate that we want to create a scatter plot

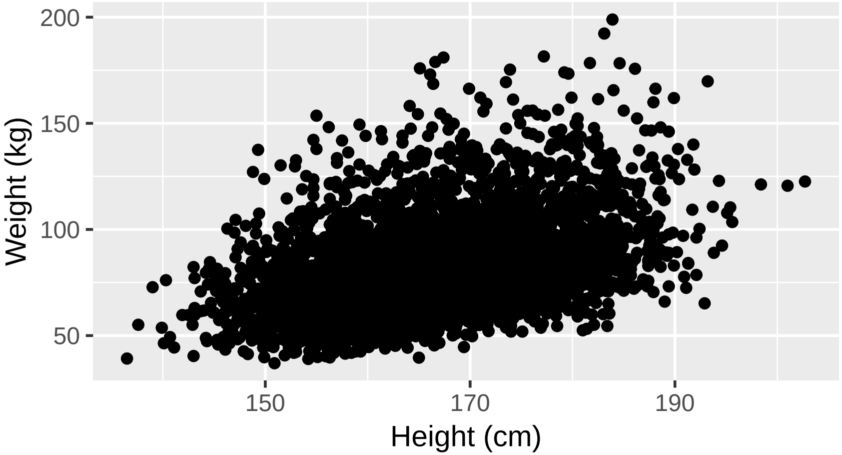
Scatter plots



Scatter plots

```
ggplot(mapping=aes(x=bmxht,y=bmxwt),data=ii)+
  geom_point()+
  xlab('Height (cm)')+
  ylab('Weight (kg)')
```

Scatter plots

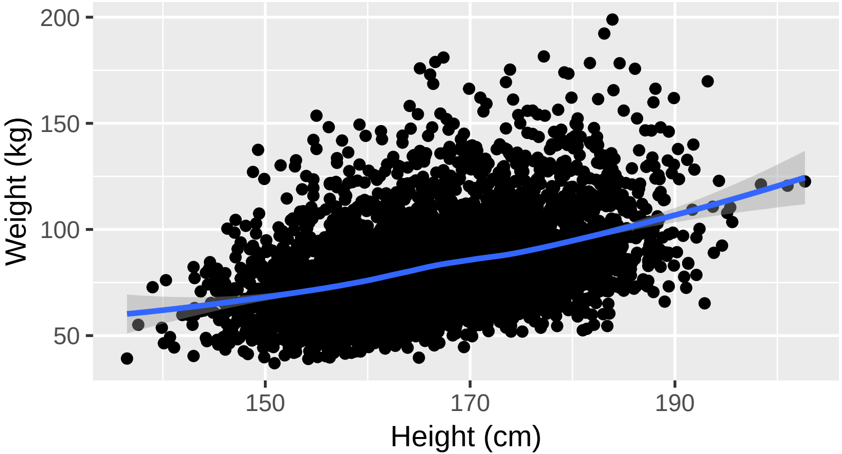


LOESS lines

- to add a LOESS line, add geom_smooth() with the method='loess' argument:

```
ggplot(mapping=aes(x=bmxht,y=bmxwt),data=ii)+
  geom_point()+
  xlab('Height (cm)')+
  ylab('Weight (kg)')+
  geom_smooth(method='loess')
```

LOESS lines

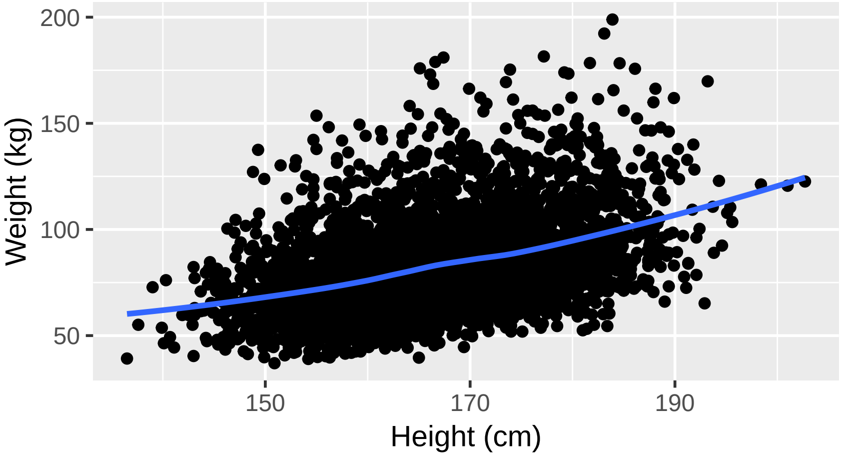


LOESS lines

- to remove the confidence band, add the `se=FALSE` argument to `geom_smooth()`:

```
ggplot(mapping=aes(x=bmxht,y=bmxwt),data=ii)+
  geom_point()+
  xlab('Height (cm)')+
  ylab('Weight (kg)')+
  geom_smooth(method='loess',se=FALSE)
```

LOESS lines

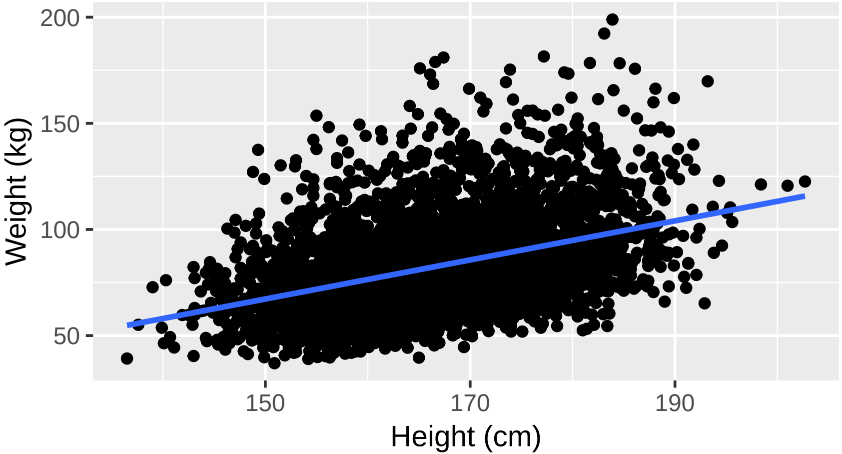


Regression lines

- to add a regression line, use `geom_smooth()` with the `method='lm'` argument and the `formula` argument:

```
ggplot(mapping=aes(x=bmxht,y=bmxwt),data=ii)+
  geom_point()+
  xlab('Height (cm)')+
  ylab('Weight (kg)')+
  geom_smooth(method='lm',formula=y~x,se=FALSE)
```

Regression lines



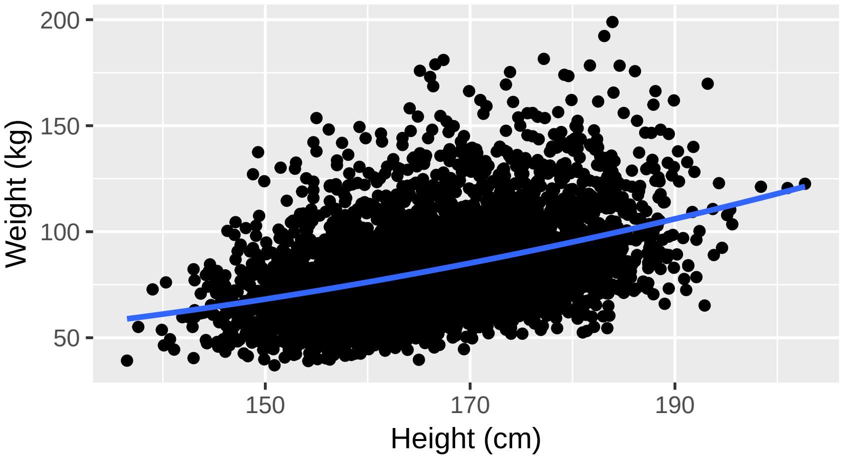
Regression lines

- for a quadratic regression line, use the `formula` argument with the `poly()` function:

```
ggplot(mapping=aes(x=bmxht,y=bmxwt),data=ii)+
  geom_point()+
  xlab('Height (cm)')+
  ylab('Weight (kg)')+
  geom_smooth(method='lm',formula=y~poly(x,degree=2),se=FALSE)
```

- the `degree=2` argument to `poly()` indicates that we want a quadratic formula

Regression lines



Correlation coefficients

- for Pearson's correlation coefficients, use `cor()`:

```
> cor(x=ii$bmxht,y=ii$bmxwt,use='complete.obs')
[1] 0.4073721
```

- the `use` argument specifies how missing values should be treated
- for Spearman's correlation coefficient, add the `method='spearman'` argument:

```
> cor(x=ii$bmxht,y=ii$bmxwt,use='complete.obs',
+     method='spearman')
[1] 0.4214076
```

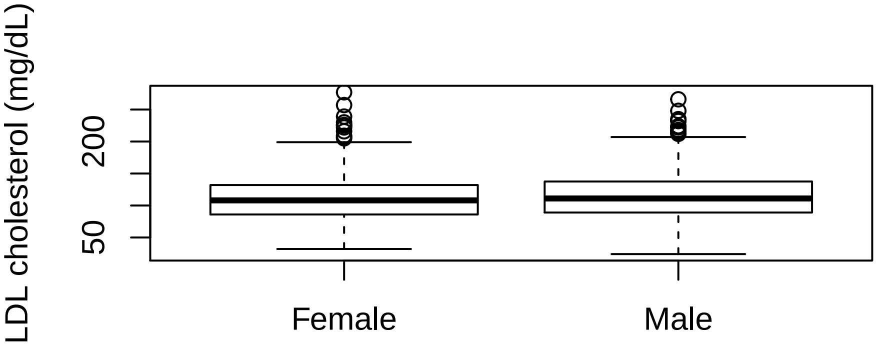
Continuous versus categorical

Box plots

- to create a box plot of LDL (low-density lipoprotein) cholesterol and gender, with LDL cholesterol on the y axis, use `plot()`:

```
plot(ii$lbdldl~ii$gender,xlab='',ylab='LDL cholesterol (mg/dL)')
```

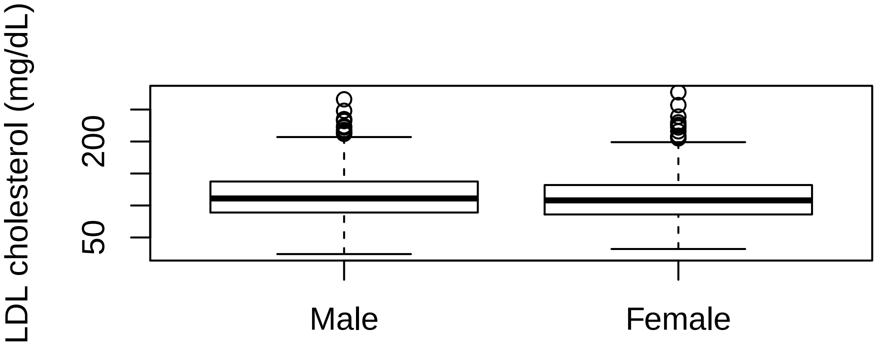
Box plots



Box plots

```
ii$gender<-relevel(ii$gender,'Male')
plot(ii$lbdldl~ii$gender,xlab='',ylab='LDL cholesterol (mg/dL)')
```

Box plots



Box plots

```
ii$gender<-relevel(ii$gender,'Female')
```

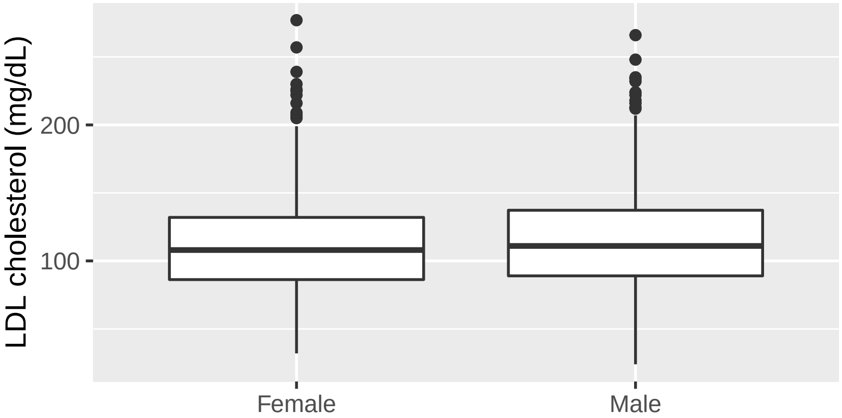
Box plots

- to create the box plot using ggplot2:

```
ggplot(mapping=aes(x=gender,y=lbdldl),data=ii)+
  geom_boxplot()+
  xlab('')+
  ylab('LDL cholesterol (mg/dL)')+
  labs(color='Gender')
```

- in the first part, we use the `aes()` function to specify an x variable and a y variable
- in the second part, we use `geom_boxplot()` to indicate that we want to created scatter plot

Box plots

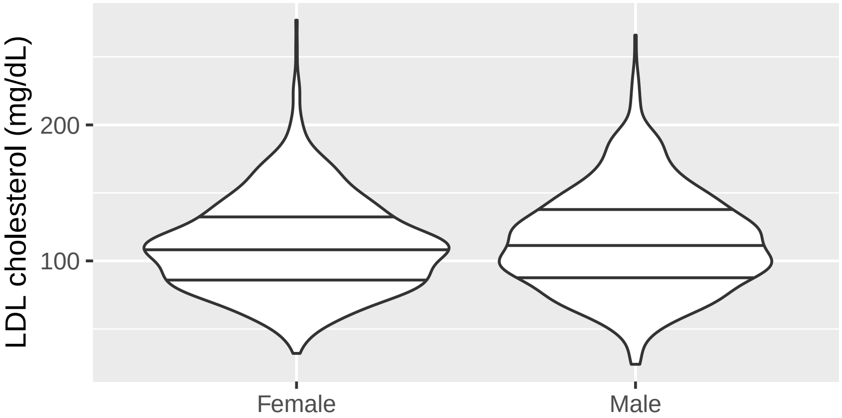


Violin plots

- to create a violin plot, use geom_violin():

```
ggplot(mapping=aes(x=gender,y=lbdldl),data=ii)+
  geom_violin(draw_quantiles=seq(0.25,0.75,0.25))+
  xlab('')+
  ylab('LDL cholesterol (mg/dL)')
```

Violin plots

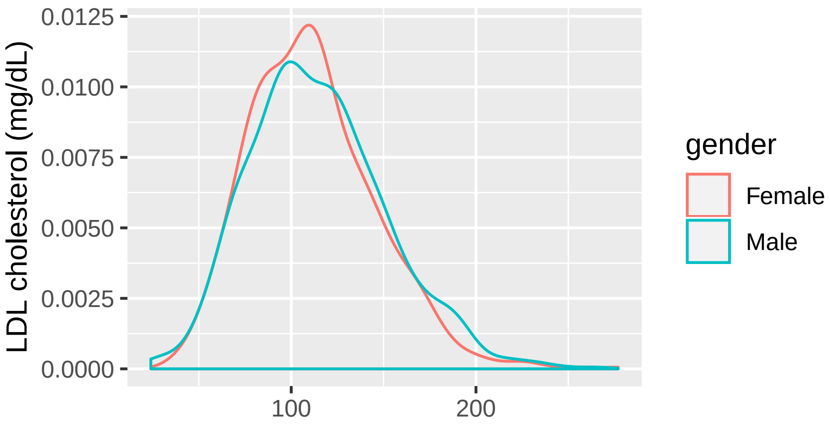


Overlapping density curves

- to create overlapping density curves, use `geom_density()`:

```
ggplot(mapping=aes(x=lbdldl,color=gender),data=ii)+
  geom_density()+
  xlab('')+
  ylab('LDL cholesterol (mg/dL)')
```

Overlapping density curves



t-tests

```
> t.test(ii$lbdldl~ii$gender)

	Welch Two Sample t-test

data:  ii$lbdldl by ii$gender
t = -1.8543, df = 1743.4, p-value = 0.06386
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -6.4798216  0.1817395
sample estimates:
mean in group Female   mean in group Male
            111.3429             114.4919
```

t-tests

```
> tt<-t.test(ii$lbdldl~ii$gender)
> names(tt)
[1] "statistic"   "parameter"   "p.value"     "conf.int"
[5] "estimate"    "null.value"  "alternative" "method"
[9] "data.name"
> tt$statistic
        t
-1.854309
> tt$p.value
[1] 0.0638638
```

t-tests

- to assume equal variance, add the `var.equal=TRUE` argument

t-tests

```
> t.test(ii$lbdldl~ii$gender,var.equal=TRUE)

	Two Sample t-test

data:  ii$lbdldl by ii$gender
t = -1.858, df = 1772, p-value = 0.06334
alternative hypothesis: true difference in means is not equal to 0
95 percent confidence interval:
 -6.4731902  0.1751082
sample estimates:
mean in group Female   mean in group Male
            111.3429             114.4919
```

Wilcoxon rank-sum tests

```
> wilcox.test(ii$lbdldl~ii$gender)

    Wilcoxon rank sum test with continuity correction

data:  ii$lbdldl by ii$gender
W = 373600, p-value = 0.07032
alternative hypothesis: true location shift is not equal to 0
```

Wilcoxon rank-sum tests

```
> tt<-wilcox.test(ii$lbdldl~ii$gender)
> names(tt)
[1] "statistic"   "parameter"   "p.value"     "null.value"
[5] "alternative" "method"      "data.name"
> tt$statistic
     W
373603
> tt$p.value
[1] 0.07032355
```

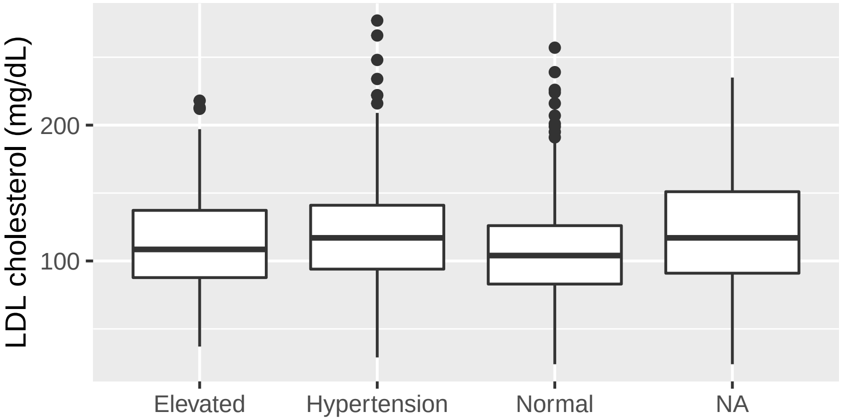
Box plots

- to create a box plot of LDL (low-density lipoprotein) cholesterol and categorized blood pressure, with LDL cholesterol on the *y* axis, using ggplot2:

```
ggplot(mapping=aes(x=bpcat,y=lbdldl),data=ii)+
  geom_boxplot()+
  xlab('')+
  ylab('LDL cholesterol (mg/dL)')
```

- in the first part, we use the `aes()` function to specify an `x` variable and a `y` variable
- in the second part, we use `geom_boxplot()` to indicate that we want to create a scatter plot

Box plots


LDL cholesterol (mg/dL)
200
100
Elevated
Hypertension
Normal
NA

Box plots

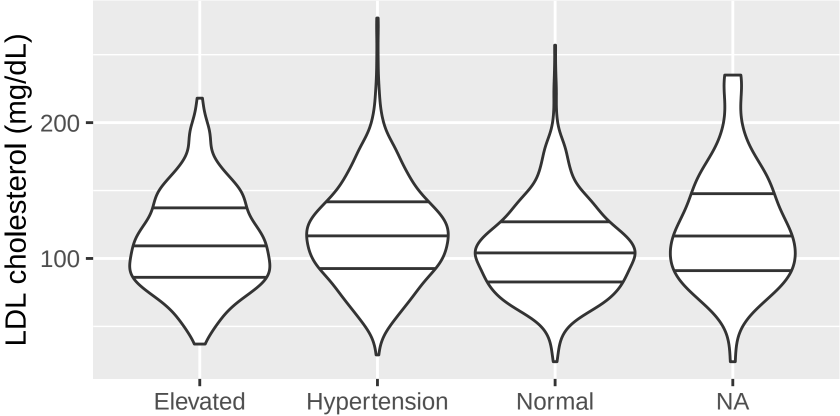
- we'll discuss strategies for omitting the NA group on a later day

Violin plots

- to create a violin plot, use geom_violin():

```
ggplot(mapping=aes(x=bpcat,y=lbdldl),data=ii)+
  geom_violin(draw_quantiles=seq(0.25,0.75,0.25))+
  xlab('')+
  ylab('LDL cholesterol (mg/dL)')
```

Violin plots

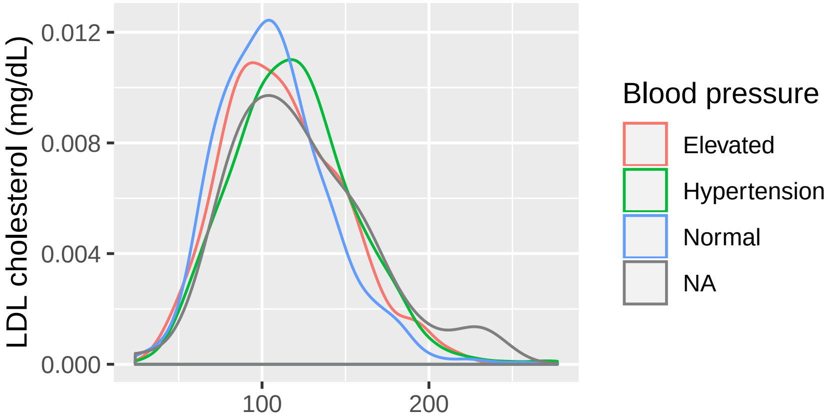


Overlapping density curves

- to create overlapping density curves, use geom_density():

```
ggplot(mapping=aes(x=lbdldl,color=bpcat),data=ii)+
  geom_density()+
  xlab('')+
  ylab('LDL cholesterol (mg/dL)')+
  labs(color='Blood pressure')
```

Overlapping density curves



Analysis of variance (ANOVA)

```
> aov(ii$lbdldl~ii$bpcat)
Call:
   aov(formula = ii$lbdldl ~ ii$bpcat)

Terms:
                ii$bpcat Residuals
Sum of Squares     47456   2067302
Deg. of Freedom        2      1694

Residual standard error: 34.93375
Estimated effects may be unbalanced
2512 observations deleted due to missingness
```

Analysis of variance (ANOVA)

```
> summary(aov(ii$lbdldl~ii$bpcat))
               Df  Sum Sq Mean Sq F value   Pr(>F)
ii$bpcat        2   47456   23728   19.44 4.48e-09 ***
Residuals    1694 2067302    1220
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
2512 observations deleted due to missingness
```

Kruskal-Wallis tests

```
> kruskal.test(ii$lbdldl~ii$bpcat)

    Kruskal-Wallis rank sum test

data:  ii$lbdldl by ii$bpcat
Kruskal-Wallis chi-squared = 39.269, df = 2, p-value =
2.971e-09
```

Kruskal-Wallis tests

```
> tt<-kruskal.test(ii$lbdldl~ii$bpcat)
> names(tt)
[1] "statistic" "parameter" "p.value"   "method"    "data.name"
> tt$statistic
Kruskal-Wallis chi-squared
                  39.26876
> tt$p.value
[1] 2.970961e-09
```

Categorical versus categorical

Tables

- to cross tabulate smoking and asthma, use `table()` with 2 arguments:

```
> table(ii$hs,ii$asthma)

                       History of asthma   No
  History of smoking                 308 1379
  No                                 372 2144
```

Row-wise percents

- for row-wise percents, use `prop.table()` with the `margin=1` argument:

```
> prop.table(table(ii$hs,ii$asthma),margin=1)*100

                      History of asthma       No
  History of smoking           18.25726 81.74274
  No                           14.78537 85.21463
```

Column-wise percents

- for column-wise percents, use prop.table() with the margin=2 argument:

```
> prop.table(table(ii$hs,ii$asthma),margin=2)*100

                  History of asthma       No
  History of smoking         45.29412 39.14278
  No                         54.70588 60.85722
```

χ^2 tests

```
> chisq.test(ii$hs,ii$asthma)

    Pearson's Chi-squared test with Yates' continuity
    correction

data:  ii$hs and ii$asthma
X-squared = 8.7221, df = 1, p-value = 0.003144
```

χ^2 tests

```
> tt<-chisq.test(ii$hs,ii$asthma)
> names(tt)
[1] "statistic" "parameter" "p.value"   "method"    "data.name"
[6] "observed"  "expected"  "residuals" "stdres"
> tt$statistic
X-squared
 8.722061
> tt$p.value
[1] 0.003143824
```

Fisher's exact test

```
> fisher.test(ii$hs,ii$asthma)

    Fisher's Exact Test for Count Data

data:  ii$hs and ii$asthma
p-value = 0.003176
alternative hypothesis: true odds ratio is not equal to 1
95 percent confidence interval:
 1.086828 1.523854
sample estimates:
odds ratio
  1.287168
```

Fisher's exact test

```
> tt<-fisher.test(ii$hs,ii$asthma)
> names(tt)
[1] "p.value"     "conf.int"    "estimate"    "null.value"
[5] "alternative" "method"      "data.name"
> tt$p.value
[1] 0.003176426
```

Linear regression

With a continuous x

Simple linear regression

- to fit a linear regression model, use `lm()`:

```
> lm(ii$bpxdi1~ii$bmxbmi)

Call:
lm(formula = ii$bpxdi1 ~ ii$bmxbmi)

Coefficients:
(Intercept)    ii$bmxbmi
    64.4573       0.2033
```

Simple linear regression

- equivalently, using the data argument:

```
> lm(bpxdi1~bmxbmi,data=ii)

Call:
lm(formula = bpxdi1 ~ bmxbmi, data = ii)

Coefficients:
(Intercept)       bmxbmi
    64.4573       0.2033
```

Simple linear regression

```
> mm<-lm(bpxdi1~bmxbmi,data=ii)
> names(mm)
 [1] "coefficients"  "residuals"     "effects"
 [4] "rank"          "fitted.values" "assign"
 [7] "qr"            "df.residual"   "na.action"
[10] "xlevels"       "call"          "terms"
[13] "model"
> mm$coefficients
(Intercept)      bmxbmi
 64.4572684   0.2033233
```

Simple linear regression

- for a confidence interval for the coefficients, use `confint()`:

```
> confint(mm)
                2.5 %     97.5 %
(Intercept) 62.8807722 66.0337646
bmxbmi       0.1516063  0.2550403
> confint(mm,level=0.9)
                  5 %       95 %
(Intercept) 63.1343246 65.7802123
bmxbmi       0.1599241  0.2467225
```

Simple linear regression

- for more information, use `summary()`:

```
> summary(mm)
...
Coefficients:
            Estimate Std. Error t value Pr(>|t|)
(Intercept) 64.45727    0.80411  80.160  < 2e-16 ***
bmxbmi       0.20332    0.02638   7.708  1.6e-14 ***
...
```

Simple linear regression


```
> names(summary(mm))
 [1] "call"          "terms"         "residuals"
 [4] "coefficients"  "aliased"       "sigma"
 [7] "df"            "r.squared"     "adj.r.squared"
[10] "fstatistic"    "cov.unscaled"  "na.action"
> summary(mm)$coefficients
              Estimate Std. Error   t value     Pr(>|t|)
(Intercept) 64.4572684 0.80410745 80.160019 0.000000e+00
bmxbmi       0.2033233 0.02637879  7.707834 1.601957e-14
```

Simple linear regression

```
> summary(mm)$r.squared
[1] 0.01455998
> summary(mm)$adj.r.squared
[1] 0.01431491
```

Working with `lm()` objects, in summary

- if `mm` is a `lm()` object:

```
> mm<-lm(bpxdi1~bmxbmi,data=ii)
```

- `mm$coefficients` provides just the coefficient estimates
- `confint(mm)` provides confidence intervals for the coefficients
- `summary(mm)$coefficients` provides the coefficient estimates, plus standard errors, test statistics, and p-values (for the null hypothesis that the coefficient is equal to 0)
- `summary(mm)$r.squared` and `summary(mm)$adj.r.squared` provide R^2
- for more information on output, see `help(lm)` and `help(summary.lm)`

Quadratic adjustment

- for quadratic adjustment, use `poly()`:

```
> mm<-lm(bpxdi1~poly(ridageyr,2),data=ii)
> mm$coefficients
        (Intercept) poly(ridageyr, 2)1 poly(ridageyr, 2)2
            70.4920           138.8831          -211.7725
```

Natural cubic splines

- for spline adjustment, install and load the splines package:

```
install.packages('splines')
library(splines)
```

Natural cubic splines

- for natural cubic splines, use `ns()`, in the splines package:

```
> mm<-lm(bpxdi1~ns(ridageyr,df=3),data=ii)
> mm$coefficients
         (Intercept) ns(ridageyr, df = 3)1
           61.614337             13.031173
ns(ridageyr, df = 3)2 ns(ridageyr, df = 3)3
           17.631901             -1.035596
```

With a categorical x

With a categorical x

```
> class(ii$hs)
[1] "factor"
```

- hs is a factor so R will automatically use dummy-variable adjustment, with a coefficient for each non-reference group
- however, we might want to change the reference:

```
> levels(ii$hs)
[1] "History of smoking" "No"
> ii$hs<-relevel(ii$hs,'No')
```

With a categorical x

```
> mm<-lm(bpxdi1~hs,data=ii)
> mm$coefficients
        (Intercept) hsHistory of smoking
          69.904960             1.423849
```

With a categorical x

```
> confint(mm)
                          2.5 %    97.5 %
(Intercept)          69.4146724 70.395248
hsHistory of smoking  0.6517504  2.195948
```

With a categorical x

```
> summary(mm)$coefficients
                     Estimate Std. Error    t value
(Intercept)         69.904960  0.2500762 279.534665
hsHistory of smoking  1.423849  0.3938164   3.615514
                         Pr(>|t|)
(Intercept)          0.0000000000
hsHistory of smoking 0.0003034298
```

With a categorical x

```
> class(ii$bpcat)
[1] "factor"
> levels(ii$bpcat)
[1] "Elevated"     "Hypertension" "Normal"
> ii$bpcat<-relevel(ii$bpcat,'Normal')
> levels(ii$bpcat)
[1] "Normal"       "Elevated"     "Hypertension"
```

With a categorical x

```
> mm<-lm(bmxbmi~bpcat,data=ii)
> mm$coefficients
      (Intercept)     bpcatElevated bpcatHypertension
        27.878717          2.635415          3.273457
```

With a categorical x

```
> confint(mm)
                       2.5 %    97.5 %
(Intercept)        27.546150 28.211285
bpcatElevated       2.024769  3.246060
bpcatHypertension   2.784601  3.762312
```

With a categorical x

```
> summary(mm)$coefficients
                    Estimate Std. Error    t value     Pr(>|t|)
(Intercept)        27.878717  0.1696293 164.350792 0.000000e+00
bpcatElevated       2.635415  0.3114659   8.461327 3.654312e-17
bpcatHypertension   3.273457  0.2493458  13.128181 1.397161e-38
```

With a categorical x

```
> summary(mm)$r.squared
[1] 0.04452897
> summary(mm)$adj.r.squared
[1] 0.04405361
```

With several x

Linear regression with several x

- to fit a linear regression model with several x variables, separate the x variables with +:

```
> lm(bmxwt~bmxht+gender,data=ii)

Call:
lm(formula = bmxwt ~ bmxht + gender, data = ii)

Coefficients:
(Intercept)        bmxht   genderMale
    -93.044        1.066       -4.319
```

Linear regression with several x

```
> mm<-lm(bmxwt~bmxht+gender,data=ii)
> mm$coefficients
(Intercept)       bmxht  genderMale
 -93.044004    1.066081   -4.319399
```

Linear regression with several x

```
> confint(mm)
                  2.5 %     97.5 %
(Intercept) -106.522747 -79.565260
bmxht          0.982252   1.149911
genderMale    -5.985612  -2.653186
```

Linear regression with several x

```
> summary(mm)$coefficients
              Estimate Std. Error   t value      Pr(>|t|)
(Intercept) -93.044004 6.87505741 -13.533560  7.029941e-41
bmxht         1.066081 0.04275853  24.932598 4.345747e-128
genderMale   -4.319399 0.84987966  -5.082365  3.889526e-07
```

Linear regression with several x

```
> summary(mm)$r.squared
[1] 0.171043
> summary(mm)$adj.r.squared
[1] 0.1706488
```

Interaction

- to include interaction, use *:

```
> mm<-lm(bmxwt~bmxht*gender,data=ii)
```

- R will automatically include the main effects and the interaction term

Interaction

```
> mm<-lm(bmxwt~bmxht*gender,data=ii)
> mm$coefficients
    (Intercept)           bmxht      genderMale
    -77.5229836       0.9693521     -34.4747482
bmxht:genderMale
      0.1809367
```

Interaction

```
> confint(mm)
                      2.5 %      97.5 %
(Intercept)      -97.25098676 -57.7949804
bmxht              0.84652191   1.0921824
genderMale       -62.52228596  -6.4272104
bmxht:genderMale   0.01294417   0.3489292
```

Interaction

```
> summary(mm)$coefficients
                  Estimate  Std. Error   t value     Pr(>|t|)
(Intercept)    -77.5229836 10.06259546 -7.704074 1.633169e-14
bmxht            0.9693521  0.06265160 15.472105 1.466730e-52
genderMale     -34.4747482 14.30611218 -2.409792 1.600432e-02
bmxht:genderMale 0.1809367  0.08568736  2.111591 3.478038e-02
```

Interaction

```
> summary(mm)$r.squared
[1] 0.171921
> summary(mm)$adj.r.squared
[1] 0.1713302
```

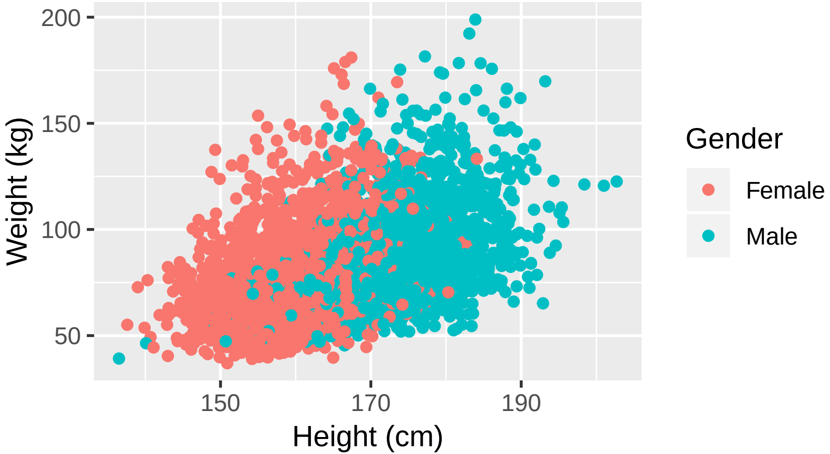
Scatter plots with a 3rd dimension

- to add color as a 3rd dimension, modify `aes()` in the first part, adding a `color` argument:

```
ggplot(mapping=aes(x=bmxht,y=bmxwt,color=gender),data=ii)+
  geom_point()+
  xlab('Height (cm)')+
  ylab('Weight (kg)')+
  labs(color='Gender')
```

- the addition of `labs()` with the `color` argument simply renames the label for the legend

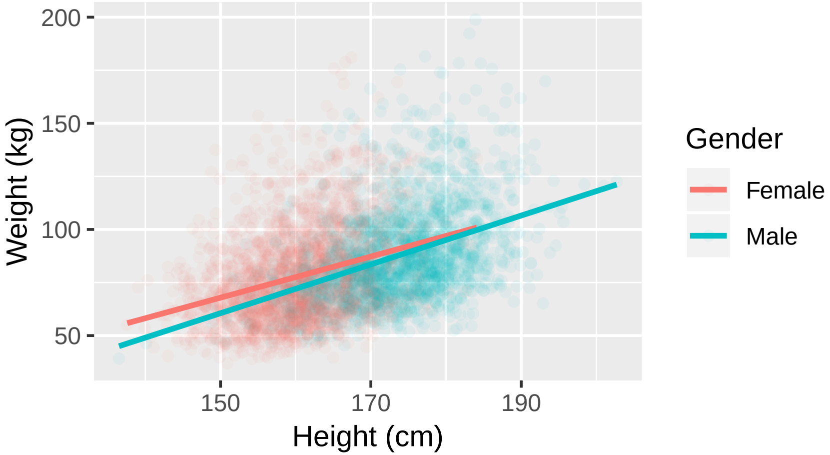
Scatter plots with a 3rd dimension



Scatter plots with a 3rd dimension

```
ggplot(mapping=aes(x=bmxht,y=bmxwt,color=gender),data=ii)+
  geom_point(alpha=0.05)+
  geom_smooth(method='lm',formula=y~x,se=FALSE)+
  xlab('Height (cm)')+ylab('Weight (kg)')+
  labs(color='Gender')
```

Scatter plots with a 3rd dimension



Model comparison and prediction

Model comparison

- for model comparison, use anova():

```
> mm1<-lm(bpxdi1~bmxbmi,data=ii)
> mm2<-lm(bpxdi1~bmxbmi+poly(ridageyr,2),data=ii)
> anova(mm2,mm1)
Analysis of Variance Table

Model 1: bpxdi1 ~ bmxbmi + poly(ridageyr, 2)
Model 2: bpxdi1 ~ bmxbmi
  Res.Df    RSS Df Sum of Sq      F    Pr(>F)
1   4019 541062
2   4021 596843 -2    -55781 207.17 < 2.2e-16 ***
---
Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
```

Prediction

- to obtain predictions, you must use the data argument when you fit the model:

```
mm<-lm(bpxdi1~bmxbmi,data=ii)
```

Prediction

- the "new data" must have all of the x variable names that are in the model
- in this example, this means that `bmxbmi` must be in the new data:

```
dd<-data.frame(bmxbmi=seq(20,80,by=10))
```

Prediction

```
> dd
  bmxbmi
1     20
2     30
3     40
4     50
5     60
6     70
7     80
```

Prediction

- to obtain the predictions, use `predict()` with the `newdata` argument:

```
> predict(mm,newdata=dd)
       1        2        3        4        5        6        7
68.52373 70.55697 72.59020 74.62343 76.65667 78.68990 80.72313
```

Prediction

- optionally adding the predictions to the new data:

```
> dd$prediction<-predict(mm,dd)
> dd
  bmxbmi  prediction
1      20    68.52373
2      30    70.55697
3      40    72.59020
4      50    74.62343
5      60    76.65667
6      70    78.68990
7      80    80.72313
```

Prediction

- with 2 x variables:

```
mm<-lm(bpxdi1~bmxbmi+hs,data=ii)
dd<-data.frame(bmxbmi=rep(seq(20,80,by=20),2),
               hs=c(rep('History of smoking',4),rep('No',4)))
```

Prediction

```
> dd
  bmxbmi                 hs
1     20 History of smoking
2     40 History of smoking
3     60 History of smoking
4     80 History of smoking
5     20                 No
6     40                 No
7     60                 No
8     80                 No
```

Prediction

```
> dd$prediction<-predict(mm,dd)
> dd
  bmxbmi                 hs prediction
1     20 History of smoking   69.35505
2     40 History of smoking   73.36493
3     60 History of smoking   77.37480
4     80 History of smoking   81.38468
5     20                 No   68.01184
6     40                 No   72.02171
7     60                 No   76.03159
8     80                 No   80.04146
```

Exercise

Exercise

You may work in groups of 2–3. Save your code in a script.

1. Use `plot()` to create a scatter plot of LDL cholesterol (`lbdldl`) and systolic blood pressure (`bpxsy1`), with systolic blood pressure on the y axis.
2. Use ggplot2 to create the plot. Add a LOESS line.
3. Use ggplot2 to create the plot. Add a simple linear regression line.
4. Compute Pearson's correlation coefficient for the association between LDL cholesterol (`lbdldl`) and systolic blood pressure (`bpxsy1`).

More on the next slide...

Exercise

5. Use ggplot2 to create a box plot of history of smoking (`hs`) and systolic blood pressure (`bpxsy1`), with systolic blood pressure on the y axis.
6. Use a t-test to test the association between systolic blood pressure (`bpxsy1`) and history of smoking (`hs`), with systolic blood pressure as the y variable.
7. Using the t-test above, write code that will output just the 95% confidence interval for the difference in means. Round the numbers to 1 decimal place and express your interval using a dash. You should obtain the following output: `0.6--2.2`. Technical note: the confidence interval is, by default, for the level-1-minus-level-2 difference.

More on the next slide. . .

Exercise

8. Use ggplot2 to produce a scatter plot of age (`ridageyr`) and LDL cholesterol (`lbdldl`), with LDL cholesterol on the y axis and color shading for gender (`gender`).
9. Fit a linear regression model for LDL cholesterol (`lbdldl`), with age (`ridageyr`) and gender (`gender`) as x variables.
10. Use the above model to predict LDL cholesterol for the following individuals: (a) 52 years old and male, (b) 35 years old and female, (c) 47 years old and female, and (d) 61 years old and male.