Obtaining Data

For our linear model examples, we will use two types of data:

- Artificial data generated with R
- Actual data loaded from a file

Obtaining Data

For our linear model examples, we will use two types of data:

- Artificial data generated with R
- Actual data loaded from a file

We will use synthetic data when we want data that:

- Exactly satisfies the assumitions of the model
- Has specific, known values for the coefficients (β values) and residual error

Obtaining Data

For our linear model examples, we will use two types of data:

- Artificial data generated with R
- Actual data loaded from a file

We will use synthetic data when we want data that:

- Exactly satisfies the assumitions of the model
- Has specific, known values for the coefficients (β values) and residual error

We will use actual data to illustrate practical applications.

Reading Data

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

Reading Data

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

You can download this file to your computer and open it as a spreadsheet for quick reference.

Reading Data

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

You can download this file to your computer and open it as a spreadsheet for quick reference.

For actual statistical analysis, we will be using R, for several reasons:

- R is available as a free download
- R can be (legally) installed on any computer (including your own)
- When you are not on campus, you can use R on your own computer
- When the day comes that you are no longer a Stonehill student, you can still use R

The Bad News about R

Admitedly, most people find R harder to use than most of the alternatives, for several reasons:

- The default interface is a command line (there are some alternatives) - OK, now what do I do?
- There is definitely a learning curve with R
- The output tends to be rather terse, showing only the minimum unless you specifically ask for more

The Bad News about R

Admitedly, most people find R harder to use than most of the alternatives, for several reasons:

- The default interface is a command line (there are some alternatives) - OK, now what do I do?
- There is definitely a learning curve with R
- The output tends to be rather terse, showing only the minimum unless you specifically ask for more

Fortunately R is getting to be widely used and there is a lot of help information available on the internet.

The Bad News about R

Admitedly, most people find R harder to use than most of the alternatives, for several reasons:

- The default interface is a command line (there are some alternatives) - OK, now what do I do?
- There is definitely a learning curve with R
- The output tends to be rather terse, showing only the minimum unless you specifically ask for more

Fortunately R is getting to be widely used and there is a lot of help information available on the internet.

Getting Help in R

If you know the name of a command in R, you can get help by preceding the command with a question mark "?"

Getting Help in R

If you know the name of a command in R, you can get help by preceding the command with a question mark "?"

To get help with the $means$ command, type $?means$

Getting Help in R

If you know the name of a command in R, you can get help by preceding the command with a question mark "?"

To get help with the $means$ command, type $?means$

Most R commands have a number of parameters, some required and some optional.

Getting Help in R

If you know the name of a command in R, you can get help by preceding the command with a question mark "?"

To get help with the $means$ command, type $?means$

Most R commands have a number of parameters, some required and some optional.

In most cases you can simply omit the optional parameters.

Simple Regression with Artificial Data

First we will perform some experiments with the linear model

$$Y_i = \beta_0 + \beta_1 X_i + e_i$$

where X is a continuous variable, β_0 and β_1 are parameters, and $e_i \sim N(0, \sigma_e)$

Simple Regression with Artificial Data

First we will perform some experiments with the linear model

$$Y_i = \beta_0 + \beta_1 X_i + e_i$$

where X is a continuous variable, β_0 and β_1 are parameters, and $e_i \sim N(0, \sigma_e)$

In this case,

- The procedure is called *(simple) linear regression*
- β_0 is called the *intercept*
- β_1 is called the *slope*
- σ_e is called the *residual standard error*

Simple Regression with Artificial Data

First we will generate data fitting the model

$$Y_i = \beta_0 + \beta_1 X_i + e_i$$

having parameters:

- The X values are 1,000 random numbers uniformly distributed between 1 and 100
- $\beta_0 = 4$
- $\beta_1 = 2$
- $\sigma_e = 5$

Simple Regression with Artificial Data

The R commands to generate data fitting the model

$$Y_i = \beta_0 + \beta_1 X_i + e_i$$

with $\beta_0 = 4$, $\beta_1 = 2$, and $\sigma_e = 5$ are:

- *x<-100*runif(1000)*
- *beta0<-4*
- *beta1<-2*
- *e<-rnorm(1000,0,5)*
- *y<-beta0+beta1*x+e*

Simple Regression with Artificial Data

The R commands to run the regression and print the summary of the results are:

$$Y_i = \beta_0 + \beta_1 X_i + e_i$$

- *lmtest<-lm(y x)*
- *summary(lmtest)*

Simple Regression with Artificial Data

The output should look something like this: Call:
lm(formula = y x)
Residuals:

Min	1Q	Median	3Q	Max
-16.36900	-3.27713	0.08972	3.42257	16.61158

Coefficients:

	Estimate	Std. Error	t value	Pr(>|t|)
(Intercept)	**3.930758**	0.318934	11.92	<2e-16 ***
x	**2.004834**	0.005482	365.72	<2e-16 ***

Residual standard error: **4.905** on 998 degrees of freedom
Multiple R-squared: 0.9926, Adjusted R-squared: 0.9926
F-statistic: 1.337e+05 on 1 and 998 DF, p-value: < 2.2e-16

Simple Regression with Artificial Data

Now let's consider the case where the variable X actually has no predictive value ($\beta_1 = 0$). Our model reduces to

$$Y_i = \beta_0 + 0 \cdot X_i + e_i \;=\; \beta_0 + e_i$$

Now to rund the model with $\beta_0 = 4$, $\beta_1 = 0$, and $\sigma_e = 5$ enter:

- *beta1<-0*
- *y<-beta0+beta1*x+e*

Simple Regression with Artificial Data

The R commands to run the regression and print the summary of the results are the same:

$$Y_i = \beta_0 + \beta_1 + e_i$$

- *lmtest<-lm(y x)*
- *summary(lmtest)*

Simple Regression with Artificial Data

Note the coefficient of X this time: Call:
lm(formula = y x)
Residuals:
Min 1Q Median 3Q Max
-16.36900 -3.27713 0.08972 3.42257 16.61158
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) 3.800758 0.318934 11.917 <2e-16 ***
x 0.004834 0.005482 0.882 0.378

Residual standard error: 4.905 on 998 degrees of freedom
Multiple R-squared: 0.0007785, Adjusted R-squared:
-0.0002227 F-statistic: 0.7775 on 1 and 998 DF, p-value:
0.3781

Simple Regression with Artificial Data

The interpretation is that the parameter β_1, the slope, is not significantly different from zero.

Saying that a parameter is zero is equivalent to eliminating it from the model.

Simple Regression with Artificial Data

The interpretation is that the parameter β_1, the slope, is not significantly different from zero.

Saying that a parameter is zero is equivalent to eliminating it from the model.

Now let's consider the case where the variable X has predictive value ($\beta_1 \neq 0$) but the intercept of the regression line is zero ($\beta_0 = 0$). Our model reduces to

$$Y_i = 0 + \beta_1 \cdot X_i + e_i \;=\; \beta_1 X_i + e_i$$

Now to rund the model with $\beta_0 = 0$, $\beta_1 = 2$, and $\sigma_e = 5$ enter:

- *beta0<-0*
- *beta1<-2*

- *y<-beta0+beta1*x+e*

Simple Regression with Artificial Data

Note the coefficient of (Intercept) this time:

```
Call:
lm(formula = y   x)
Residuals:
Min 1Q Median 3Q Max
-16.36900 -3.27713 0.08972 3.42257 16.61158
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.199242 0.318934 -0.625 0.532
x 2.004834 0.005482 365.718 <2e-16 ***
Residual standard error: 4.905 on 998 degrees of freedom
Multiple R-squared: 0.9926, Adjusted R-squared: 0.9926
F-statistic: 1.337e+05 on 1 and 998 DF, p-value: < 2.2e-16
```

Simple Regression with Artificial Data

Note the coefficient of (Intercept) this time:

```
Call:
lm(formula = y   x)
Residuals:
Min 1Q Median 3Q Max
-16.36900 -3.27713 0.08972 3.42257 16.61158
Coefficients:
Estimate Std. Error t value Pr(>|t|)
(Intercept) -0.199242 0.318934 -0.625 0.532
x 0.004834 0.005482 0.882 0.378
Residual standard error: 4.905 on 998 degrees of freedom
Multiple R-squared: 0.9926, Adjusted R-squared: 0.9926
F-statistic: 1.337e+05 on 1 and 998 DF, p-value: < 2.2e-16
```

Simple Regression with Artificial Data

The interpretation is that the parameter β_0, the intercept, is not significantly different from zero, but the slope β_1 is.

Simple Regression with Artificial Data

The interpretation is that the parameter β_0, the intercept, is not significantly different from zero, but the slope β_1 is.

Now let's consider the case where the variable X has no predictive value ($\beta_1 = 0$) and the intercept of the regression line is zero ($\beta_0 = 0$). Our model reduces to

$$Y_i = 0 + \beta_1 \cdot X_i + e_i \; = \; e_i$$

Now to rund the model with $\beta_0 = 0$, $\beta_1 = 0$, and $\sigma_e = 5$ enter:

- *beta0<-0*
- *beta1<-0*
- *y<-beta0+beta1*x+e*

Reading a .csv file into R

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

Reading a .csv file into R

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

You can download this file to your computer and open it as a spreadsheet for quick reference.

Reading a .csv file into R

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

You can download this file to your computer and open it as a spreadsheet for quick reference.

Time does not allow us to explore the many options for loading data into R, so we will make use the $read.table$ function, the obvious choice in this case.

Reading a .csv file into R

We will be using the 2009 EPA mileage data, which is stored on the website as a comma-delimited (.csv) file.

You can download this file to your computer and open it as a spreadsheet for quick reference.

Time does not allow us to explore the many options for loading data into R, so we will make use the $read.table$ function, the obvious choice in this case.

You can display the help information for this command by entering $?read.table$

Reading a .csv file into R

Go to the course web page, then the *Notes and Handouts* section.

Reading a .csv file into R

Go to the course web page, then the *Notes and Handouts* section.

Right click on the *2009 EPA Mileage Data* link and select *copy link location*

Reading a .csv file into R

Go to the course web page, then the *Notes and Handouts* section.

Right click on the *2009 EPA Mileage Data* link and select *copy link location*

This should copy the URL for the EPA .csv data file, which is:

http://www.sandgquinn.org/stonehill/MA225/notes/09tstcar.csv

Reading a .csv file into R

Go to the course web page, then the *Notes and Handouts* section.

Right click on the *2009 EPA Mileage Data* link and select *copy link location*

This should copy the URL for the EPA .csv data file, which is:

http://www.sandgquinn.org/stonehill/MA225/notes/09tstcar.csv

Carefully type the following command in R, but don't hit enter:

epa<-read.table("",sep=",",fill=TRUE,header=TRUE)

Reading a .csv file into R

Now paste the URL between the two consecutive double quotes and hit enter:

epa<-
read.table("http://www.sandgquinn.org/stonehill/MA225/notes/0

Reading a .csv file into R

Now paste the URL between the two consecutive double quotes and hit enter:

epa<-
read.table("http://www.sandgquinn.org/stonehill/MA225/notes/0

If there are no errors, this should load the EPA data into a data frame called epa.

Reading a .csv file into R

Now paste the URL between the two consecutive double quotes and hit enter:

epa<-
read.table("http://www.sandgquinn.org/stonehill/MA225/notes/0

If there are no errors, this should load the EPA data into a data frame called epa.

The following optional step makes the colums of the *epa* data frame available as vectors:

attach(epa)

Reading a .csv file into R

Now paste the URL between the two consecutive double quotes and hit enter:

epa<-
read.table("http://www.sandgquinn.org/stonehill/MA225/notes/0

If there are no errors, this should load the EPA data into a data frame called epa.

The following optional step makes the colums of the *epa* data frame available as vectors:

attach(epa)

If this all worked, you should see an abbreviated list of the contents of the data frame named *epa*.

Reading a .csv file into R

We will need to know the column names that were read from the file, so type:

labels(epa)

Reading a .csv file into R

We will need to know the column names that were read from the file, so type:

labels(epa)

If you opened the .csv file in a spreadsheet, these would be the column headings.

Reading a .csv file into R

We will need to know the column names that were read from the file, so type:

labels(epa)

If you opened the .csv file in a spreadsheet, these would be the column headings.

In some cases (but not this one) the first column might be a name associated with the row. To display rownames, enter

rownames(epa)

Reading a .csv file into R

We will need to know the column names that were read from the file, so type:

labels(epa)

If you opened the .csv file in a spreadsheet, these would be the column headings.

In some cases (but not this one) the first column might be a name associated with the row. To display rownames, enter

rownames(epa)

This should just show row numbers because this file has no rownames

Using Data Frames

The data.frame structure is a bit like a two dimensional array, the first dimension being the row, the second the column. To display the first 20 rows of *epa* showing only the first 10 columns, type:

epa(1:20,1:10)

Using Data Frames

The data.frame structure is a bit like a two dimensional array, the first dimension being the row, the second the column. To display the first 20 rows of *epa* showing only the first 10 columns, type:

epa(1:20,1:10)

To display the first 50 rows of the *mpg* column, type:

epa$mpg[1:50] or just *mpg[1:50]* if you have attached *epa*

Using Data Frames

The data.frame structure is a bit like a two dimensional array, the first dimension being the row, the second the column. To display the first 20 rows of *epa* showing only the first 10 columns, type:

epa(1:20,1:10)

To display the first 50 rows of the *mpg* column, type:

epa$mpg[1:50] or just *mpg[1:50]* if you have attached *epa*

To display the entire *mpg* column, type:

epa$mpg or just *mpg* if you have attached *epa*

Using Data Frames

The data.frame structure is a bit like a two dimensional array, the first dimension being the row, the second the column. To display the first 20 rows of *epa* showing only the first 10 columns, type:

epa(1:20,1:10)

To display the first 50 rows of the *mpg* column, type:

epa$mpg[1:50] or just *mpg[1:50]* if you have attached *epa*

To display the entire *mpg* column, type:

epa$mpg or just *mpg* if you have attached *epa*

To display the mean of the *mpg* column, type

mean(epa$mpg) or just *mean(epa)* if you have attached *epa*