MTH 4230 R Notes 3

1 Regression Through the Origin

- To fit a regression model ***through the origin*** (i.e. the model with **no intercept**),

$$Y_i = \beta_1 X_i + \epsilon_i$$

  we use the *formula* `y ~ -1 + x` in the call to the `lm()` function.

- For example, consider the data in these vectors `x` and `y`:

```
x <- c(22, 15, 7, 19, 20, 9, 15, 10, 19, 21)
y <- c(12.2, 9.5, 6.7, 5.9, 10.0, 8.9, 11.5, 10.0, 9.9, 10.1)
```

  We fit the model with **no intercept** by typing:

```
my.reg <- lm(y ~ -1 + x)
```

  Then we look at the results in the usual manner, using `summary()`:

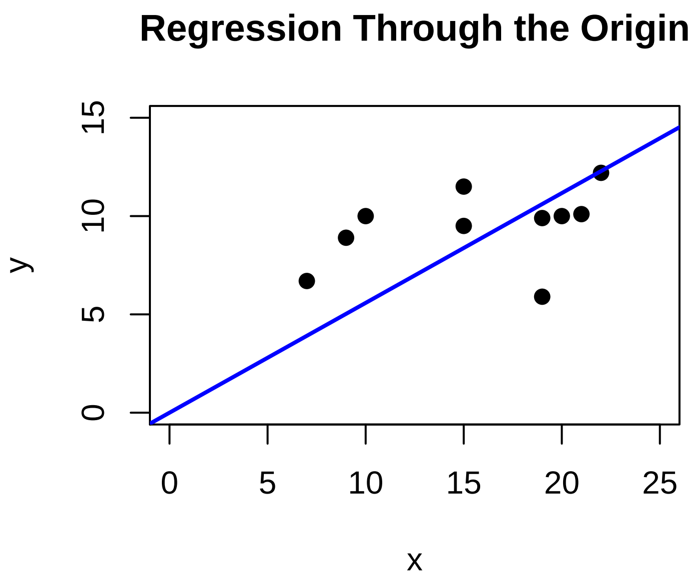
```
summary(my.reg)

##
## Call:
## lm(formula = y ~ -1 + x)
##
## Residuals:
##    Min     1Q Median     3Q    Max
## -4.709 -1.053  0.520  3.041  4.416
##
## Coefficients:
##   Estimate Std. Error t value Pr(>|t|)
## x   0.5584     0.0571   9.779 4.31e-06
##
## Residual standard error: 2.982 on 9 degrees of freedom
## Multiple R-squared:  0.914,Adjusted R-squared:  0.9044
## F-statistic: 95.62 on 1 and 9 DF,  p-value: 4.31e-06
```

  Notice that there's **no intercept term** in the output, just the slope term. A plot of the data with the fitted regression line is shown below. Notice that **the line passes through the origin**.

```
plot(x, y, pch = 19, xlim = c(0,25), ylim = c(0,15),
    main = "Regression Through the Origin")

abline(my.reg, lwd = 2, col = "blue")
```



2 Matrix Approach to Regression

2.1 Obtaining the Design Matrix

- To obtain the $n \times p$ ***design matrix X*** used in a regression analysis, we use the function:

```
model.matrix()       # Returns the design matrix X used in a linear
                     # regression analysis carried out by lm()
```

  The `model.matrix()` function takes as its main argument either an *lm* object or a *formula* indicating a regression model. It returns the $n \times p$ *design matrix* $\boldsymbol{X}$.

- For example:

```
x <- c(22, 15, 7, 19, 20, 9, 15, 10, 19, 21)
y <- c(12.2, 9.5, 6.7, 5.9, 10.0, 8.9, 11.5, 10.0, 9.9, 10.1)
```

```
my.reg <- lm(y ~ x)
```

```
X <- model.matrix(my.reg)
```

```
X

##    (Intercept)  x
## 1            1 22
## 2            1 15
## 3            1  7
## 4            1 19
## 5            1 20
## 6            1  9
## 7            1 15
## 8            1 10
## 9            1 19
## 10           1 21
## attr(,"assign")
## [1] 0 1
```

The matrix returned by `model.matrix()` comes equipped with several *attributes*. In R, an ***attribute*** is bit of extra information (so-called ***meta data***) that some *classes* of data objects, including *matrices*, contain.

The `"assign"` attribute is a *vector* with a 0 representing the intercept and 1 the predictor. For models with more predictors, the `"assign"` *vector* would have elements `2`, `3`, ..., `p` for the additional predictors. To see all of `X`'s *attributes*, type `attributes(X)`.

We can verify that `X` is *matrix* and look at its dimensions using:

```
is.matrix(X)

## [1] TRUE

dim(X)

## [1] 10  2
```

We see that `X` indeed a *matrix* and it has $\boldsymbol{n = 10}$ rows and $\boldsymbol{p = 2}$ columns.

- Another way to obtain the ***model matrix*** is by passing a *formula* to `model.matrix()`:

```
X <- model.matrix(y ~ x)
```

2.2 Performing Computations with the Design Matrix

- All of the usual matrix functions and operators (e.g. `t()`, `%*%`, `solve()`, `det()`, etc.) can be used with the design matrix $\boldsymbol{X}$.

- For example, we can use matrix operations to obtain the ***vector of estimated coefficients***

$$\boldsymbol{b} = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}$$

given by

$$\boldsymbol{b} = \left(\boldsymbol{X}^T\boldsymbol{X}\right)^{-1}\boldsymbol{X}^T\boldsymbol{Y}$$

from the *matrix* `X` and *vector* `y` created earlier by typing:

```
b <- solve(t(X) %*% X) %*% t(X) %*% y
b

##                  [,1]
## (Intercept) 7.3189622
## x           0.1370088
```

We see that the ***estimated intercept*** is $\boldsymbol{b_0 = 7.319}$ and the ***estimated slope*** is $\boldsymbol{b_1 = 0.137}$.

2.3 The (Estimated) Variance-Covariance Matrix of b_0 and b_1

- The (estimated) ***variance-covariance matrix*** $\boldsymbol{s^2\{b\}}$ of the *estimated coefficient vector*

$$\boldsymbol{b} = \begin{bmatrix} b_0 \\ b_1 \end{bmatrix}$$

is obtained using the function:

```
vcov()     # Returns the (estimated) variance-covariance matrix of the
           # regression coefficient estimates in a linear regression
           # analysis carried out by lm()
```

The `vcov()` function takes an *lm* object (such as `my.reg`) as its main argument and returns a *matrix* containing the ***variances*** on the diagonal and ***covariances*** on the off-diagonals.

- For example, using `my.reg` created earlier:

```
vcov(my.reg)

##             (Intercept)          x
## (Intercept)   3.7119109 -0.2137037
## x            -0.2137037  0.0136117
```

```
vcov.tmp <- vcov(my.reg)
```

We see that:

- The (estimated) ***variance*** of $\boldsymbol{b_0}$ is $\boldsymbol{s^2\{b_0\} = 3.712}$, so the (estimated) ***standard error*** of $\boldsymbol{b_0}$ is $\boldsymbol{s\{b_0\} = \sqrt{3.712} = 1.927}$.
- The (estimated) ***variance*** of $\boldsymbol{b_1}$ is $\boldsymbol{s^2\{b_1\} = 0.014}$, so the (estimated) ***standard error*** of $\boldsymbol{b_1}$ is $\boldsymbol{s\{b_1\} = \sqrt{0.014} = 0.117}$.
- The (estimated) ***covariance*** between $\boldsymbol{b_0}$ and $\boldsymbol{b_1}$ is $\boldsymbol{s^2\{b_0, b_1\} = -0.214}$.

- Note that the **variance-covariance matrix** of $\boldsymbol{b}$ is MSE $\cdot$ $(\boldsymbol{X}^T\boldsymbol{X})^{-1}$, so we could also have obtained it using the MSE from `lm()` and the design matrix `X`.