Design Matrices without Full Column Rank

The Problem

In our derivation of the least squares estimates for linear regression, we took the derivatives of the residual sum of squares with respect to each of $\beta_0, \dots, \beta_p$, set the results equal to 0, and figured out how to write the results in terms of matrices. We arrived at this point:

$$(X'X)\hat{\beta} = X'y$$

In order to solve for beta, we multiplied on the left by $(X'X)^{-1}$ to obtain

$$\hat{\beta} = (X'X)^{-1}X'y$$

This is only possible if $X'X$ has full rank, which is the case if and only if X has full column rank.

So, what are some examples of settings where X *doesn't* have full rank?

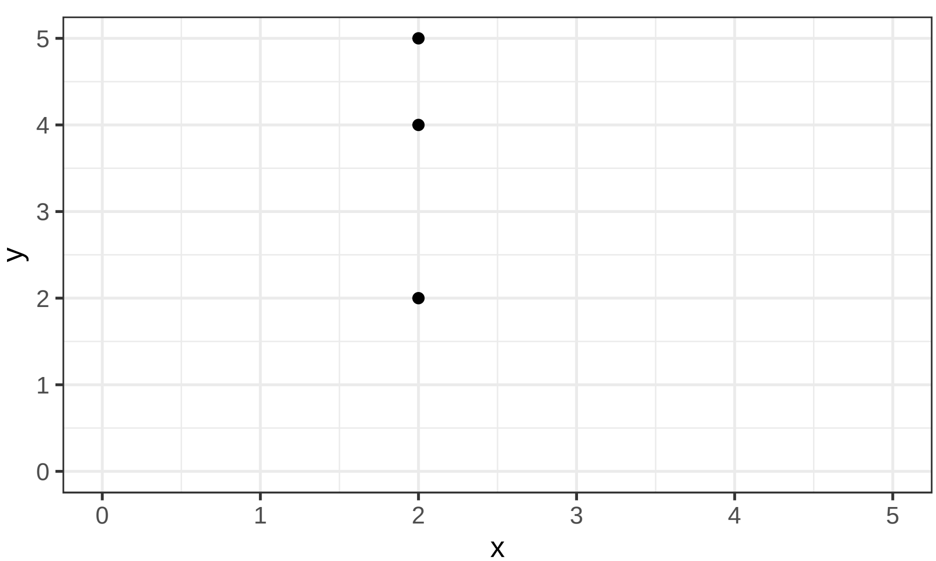
Example 1: Not enough distinct values of x

Suppose that we have the following data set, and we want to fit a simple linear regression model:

```
example_data
```

```
##   x y
## 1 2 2
## 2 2 4
## 3 2 5
```

```
ggplot(data = example_data, mapping = aes(x = x, y = y)) +
  geom_point() +
  xlim(c(0, 5)) +
  ylim(c(0, 5)) +
  theme_bw()
```



What happens if we try to do estimation?

```
lm_fit <- lm(y ~ x, data = example_data)
summary(lm_fit)
```

```
##
## Call:
## lm(formula = y ~ x, data = example_data)
##
## Residuals:
##       1       2       3
## -1.6667  0.3333  1.3333
##
## Coefficients: (1 not defined because of singularities)
##             Estimate Std. Error t value Pr(>|t|)
## (Intercept)   3.6667     0.8819   4.158   0.0533 .
## x                 NA         NA      NA       NA
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
##
## Residual standard error: 1.528 on 2 degrees of freedom
```

```
X <- model.matrix(lm_fit)
t(X) %*% X
```

```
##             (Intercept)  x
## (Intercept)           3  6
## x                     6 12
```

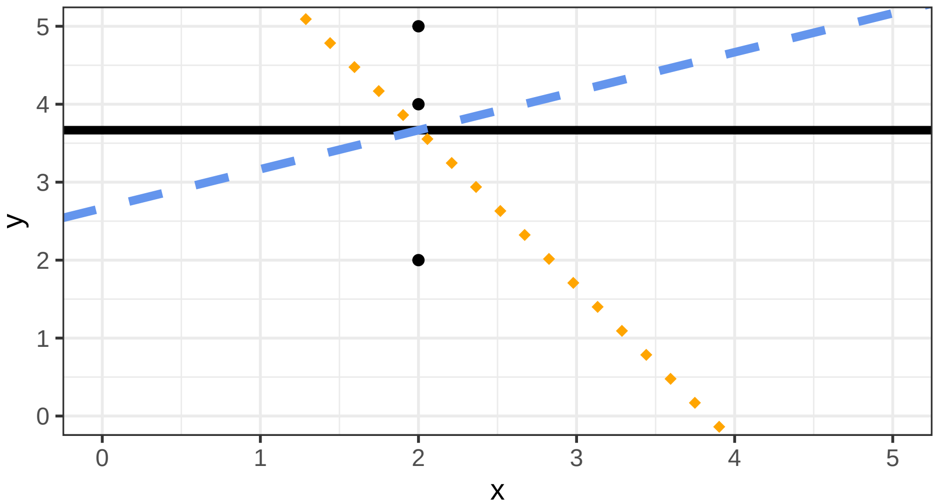
```
solve(t(X) %*% X)
```

```
## Error in solve.default(t(X) %*% X): Lapack routine dgesv: system is exactly singular: U[2,2] = 0
```

What's going on?

- There are many possible lines that fit the data
- The slope and intercept parameters are not **identifiable** from the data we have

```
ggplot(data = example_data, mapping = aes(x = x, y = y)) +
  geom_point() +
  geom_abline(intercept = 3.6667, slope = 0, size = 1.5) +
  geom_abline(intercept = 2.6667, slope = 0.5, color = "cornflowerblue", linetype = 2, size = 1.5) +
  geom_abline(intercept = 7.6667, slope = -2, color = "orange", linetype = 3, size = 1.5) +
  xlim(c(0, 5)) +
  ylim(c(0, 5)) +
  theme_bw()
```



Notice that the fitted values are the same for all of these three lines:

```
beta_hat1 <- matrix(c(3.6667, 0))
y_hat1 <- X %*% beta_hat1
y_hat1
```

```
##      [,1]
## 1 3.6667
## 2 3.6667
## 3 3.6667
```

```
beta_hat2 <- matrix(c(2.6667, 0.5))
y_hat2 <- X %*% beta_hat2
y_hat2
```

```
##      [,1]
## 1 3.6667
## 2 3.6667
## 3 3.6667
```

```
beta_hat3 <- matrix(c(7.6667, -2))
y_hat3 <- X %*% beta_hat3
y_hat3
```

```
##      [,1]
## 1 3.6667
## 2 3.6667
## 3 3.6667
```

In turn, the residual sums of squares are the same for all three lines too:

```
sum((example_data$y - y_hat1)^2)
```

```
## [1] 4.666667
```

```
sum((example_data$y - y_hat2)^2)
```

```
## [1] 4.666667
```

```
sum((example_data$y - y_hat3)^2)
```

```
## [1] 4.666667
```

Example 2: Multiple Regression with Redundant Covariates

The Current Population Survey (CPS) is used to supplement census information between census years. These data consist of a random sample of persons from the 1985 CPS, with information on wages and other characteristics of the workers, including sex, number of years of education, years of work experience, occupational status, region of residence and union membership.

Suppose we fit a model for `wage` (in US dollars per hour) based on the following explanatory variables:

- `educ` number of years of education
- `age` age in years
- `exper` number of years of work experience (inferred from `age` and `educ`)
- `married` a factor with levels `Married`, `Single`
- `sector` a factor with levels `clerical`, `const`, `manag`, `manuf`, `other`, `prof`, `sales`, `service`

```r
head(CPS85)
```

```
##   wage educ age exper married   sector
## 1  9.0   10  43    27 Married    const
## 2  5.5   12  38    20 Married    sales
## 3  3.8   12  22     4  Single    sales
## 4 10.5   12  47    29 Married clerical
## 5 15.0   12  58    40 Married    const
## 6  9.0   16  49    27 Married clerical
```

```r
lm_fit <- lm(wage ~ educ + age + exper + married + sector, data = CPS85)
summary(lm_fit)
```

```
## 
## Call:
## lm(formula = wage ~ educ + age + exper + married + sector, data = CPS85)
## 
## Residuals:
##     Min      1Q  Median      3Q     Max 
## -10.456  -2.841  -0.712   1.876  34.115 
## 
## Coefficients: (1 not defined because of singularities)
##               Estimate Std. Error t value Pr(>|t|)    
## (Intercept)   -4.31556    1.59566  -2.705  0.00706 ** 
## educ           0.65293    0.09605   6.798 2.91e-11 ***
## age            0.09436    0.01753   5.382 1.11e-07 ***
## exper               NA         NA      NA       NA    
## marriedSingle -0.39608    0.42309  -0.936  0.34962    
## sectorconst    3.00449    1.09875   2.734  0.00646 ** 
## sectormanag    3.97363    0.76582   5.189 3.04e-07 ***
## sectormanuf    1.67249    0.71946   2.325  0.02047 *  
## sectorother    2.13155    0.71153   2.996  0.00287 ** 
## sectorprof     2.67365    0.67853   3.940 9.24e-05 ***
## sectorsales   -0.17695    0.84902  -0.208  0.83499    
## sectorservice -0.12186    0.67318  -0.181  0.85642    
## ---
## Signif. codes:  0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1
## 
## Residual standard error: 4.421 on 522 degrees of freedom
## Multiple R-squared:  0.2752, Adjusted R-squared:  0.2613 
## F-statistic: 19.82 on 10 and 522 DF,  p-value: < 2.2e-16
```

```r
X <- model.matrix(lm_fit)
head(X)
```

```
##   (Intercept) educ age exper marriedSingle sectorconst sectormanag sectormanuf sectorother sectorpro
## 1           1   10  43    27             0           1           0           0           0
## 2           1   12  38    20             0           0           0           0           0
## 3           1   12  22     4             1           0           0           0           0
## 4           1   12  47    29             0           0           0           0           0
## 5           1   12  58    40             0           1           0           0           0
## 6           1   16  49    27             0           0           0           0           0
```

```r
solve(t(X) %*% X)
```

```
## Error in solve.default(t(X) %*% X): system is computationally singular: reciprocal condition number
```

What's going on?

`exper` was inferred from `age` and `educ`, assuming everyone started school at age 6 and started getting job experience immediately after leaving school:

```r
exper_is_made_up <- cbind(
  CPS85$exper,
  CPS85$age - CPS85$educ - 6
)

head(exper_is_made_up)
```

```
##      [,1] [,2]
## [1,]   27   27
## [2,]   20   20
## [3,]    4    4
## [4,]   29   29
## [5,]   40   40
## [6,]   27   27
```

This means the fourth column of X is equal to a linear combination of the first three columns:

```r
X_is_not_full_rank <- cbind(
  X[, 3] - X[, 2] - 6 * X[, 1],
  X[, 4]
)
head(X_is_not_full_rank)
```

```
##   [,1] [,2]
## 1   27   27
## 2   20   20
## 3    4    4
## 4   29   29
## 5   40   40
## 6   27   27
```

Once again, this means that we can get the same fitted values (and therefore the same residual sum of squares) from different coefficients for those variables:

```r
cbind(beta_hat1, beta_hat2)
```

```
##              [,1]        [,2]
##  [1,] -4.31555602 -3.74940308
##  [2,]  0.65293313  0.74729196
##  [3,]  0.09435882  0.00000000
##  [4,]  0.00000000  0.09435882
##  [5,] -0.39608164 -0.39608164
##  [6,]  3.00448959  3.00448959
##  [7,]  3.97363158  3.97363158
##  [8,]  1.67248889  1.67248889
##  [9,]  2.13154511  2.13154511
## [10,]  2.67364689  2.67364689
## [11,] -0.17694859 -0.17694859
## [12,] -0.12185820 -0.12185820
```

```r
y_hat1 <- X %*% beta_hat1
y_hat2 <- X %*% beta_hat2

same_fitted_values <- cbind(y_hat1, y_hat2)
head(same_fitted_values)
```

```
##        [,1]      [,2]
## 1  9.275694  9.275694
## 2  6.928328  6.928328
## 3  5.022505  5.022505
## 4  7.954506  7.954506
## 5 11.996943 11.996943
## 6 10.754956 10.754956
```

Summary

A few ways to think about when you might have a design matrix that isn't full rank:

1) There is some redundancy in the explanatory variables
2) There isn't enough information in your data to learn about the relationship you're interested in (e.g. we can't separate the effects of several closely related variables because they are linear functions of each other).
3) Multiple different coefficient values can explain the observed data equally as well (same fitted values, so same RSS).
    - The model parameters are not **identifiable**.

Roughly, model parameters are identifiable if there is a unique set of parameter values that explains the observed data best.

In the case of linear regression, model parameters are identifiable if there is a unique set of parameter values that minimize RSS.