Final Review - Multiple Linear Regression

Multiple Linear Regression

Example: We want to do physical experiment to measure the gravitational acceleration $g \approx 9.8$ m / s^2. We have several weights of 10g each. We use springs to measure their gravitational forces:

In[12]:=
```
X = {10, 20, 30, 40, 50, 60, 70, 80};
```

In[∘]:=
```
Residual = RandomVariate[NormalDistribution[0, 0.05], Length[X]]
```

Out[∘]=
```
{-0.0115325, 0.0157878, 0.0294766,
 -0.0597483, 0.0173925, 0.0349411, 0.0903353, 0.122338}
```

In[14]:=
```
Y = 0.0098 * X + Residual
```

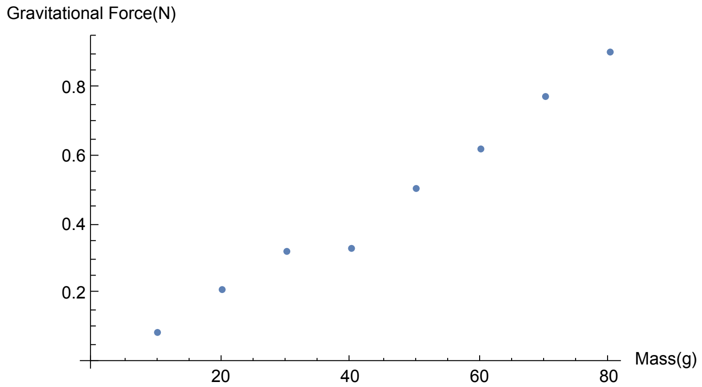
Out[14]=
```
{0.0864675, 0.211788, 0.323477, 0.332252, 0.507393, 0.622941, 0.776335, 0.906338}
```

In[128]:=
```
Data = Transpose[{X, Y}]
```

Out[128]=
```
{{10, 0.0864675}, {20, 0.211788}, {30, 0.323477}, {40, 0.332252},
 {50, 0.507393}, {60, 0.622941}, {70, 0.776335}, {80, 0.906338}}
```

In[16]:=
```
ListPlot[Data, AxesLabel → {"Mass(g)", "Gravitational Force(N)"}]
```

Out[16]=



Settings and Assumptions

To discuss the model

$$Y \mid x = \beta_0 + \beta_1 x_1 + \beta_2 x_2 + \ldots + \beta_p x_p + E$$

We gives the same assumptions, and define

$$\boldsymbol{Y} = \begin{pmatrix} y_1 \\ y_2 \\ \vdots \\ y_n \end{pmatrix}, \boldsymbol{X} = \begin{pmatrix} 1 & x_{11} & \cdots & x_{p1} \\ 1 & x_{12} & & \\ \vdots & \vdots & \ddots & \vdots \\ 1 & x_{1\,n} & & x_{pn} \end{pmatrix}, \boldsymbol{\beta} = \begin{pmatrix} \beta_0 \\ \beta_1 \\ \vdots \\ \beta_p \end{pmatrix}, \boldsymbol{E} = \begin{pmatrix} e_1 \\ e_2 \\ \vdots \\ e_n \end{pmatrix}$$

We have $\boldsymbol{Y} = \boldsymbol{X}\,\boldsymbol{\beta} + \boldsymbol{E}$. Our assumptions are similar:

- $\mathrm{E}[\boldsymbol{E}] = \mathbf{0}$,
- $\mathrm{Var}[\boldsymbol{E}] = \mathrm{Var}[\boldsymbol{Y}] = \sigma^2\, \mathbb{1}_n$ is constant.
- $\boldsymbol{E}$ is independent of the elements of $\boldsymbol{X}$.

Least Squares Estimation

To minimize $SS_E = (Y - X\,b)^T\,(Y - X\,b)$, we have $b = \left(X^T\,X\right)^{-1} X^T\,Y$.

Example: I want to check whether gravitational force is related to the square of mass, so I fit my data to the model $y = b_0 + b_1\,x + b_2\,x^2$. I calculate

In[129]:=
```
y = Transpose[Data][[2]];
x = Transpose[Table[Function[x, x^k] /@ Transpose[Data][[1]], {k, 0, 2}]];
{MatrixForm[x], MatrixForm[y]}
```

Out[131]=

$$\left\{ \begin{pmatrix} 1 & 10 & 100 \\ 1 & 20 & 400 \\ 1 & 30 & 900 \\ 1 & 40 & 1600 \\ 1 & 50 & 2500 \\ 1 & 60 & 3600 \\ 1 & 70 & 4900 \\ 1 & 80 & 6400 \end{pmatrix}, \begin{pmatrix} 0.0864675 \\ 0.211788 \\ 0.323477 \\ 0.332252 \\ 0.507393 \\ 0.622941 \\ 0.776335 \\ 0.906338 \end{pmatrix} \right\}$$

In[132]:=
```
b = Inverse[Transpose[x].x].Transpose[x].y;
MatrixForm[b]
```

Out[133]//MatrixForm=

$$\begin{pmatrix} 0.0350763 \\ 0.00664771 \\ 0.0000535885 \end{pmatrix}$$

So the result become $y = 0.035 + 0.0066\,x + 0.00005\,x^2$.

In[22]:=
```
lmQuadratic = LinearModelFit[{x, y}]
```

Out[22]=
```
FittedModel[ 0.0350763 #1 + 0.00664771 #2 + 0.0000535885 #3 ]
```

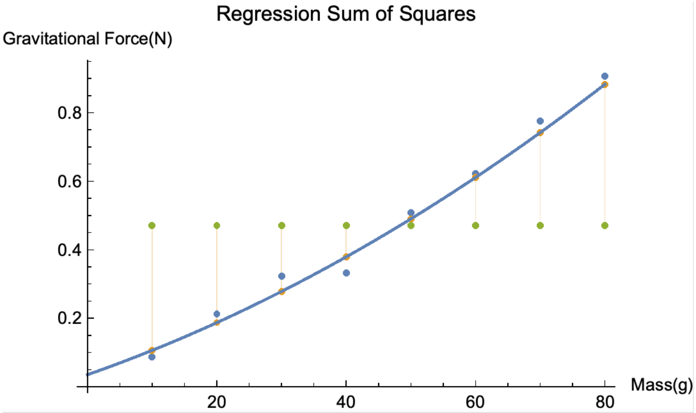
Error Analysis

We define an ***orthogonal projection*** matrix

$$P = \frac{1}{n}\begin{pmatrix} 1 & 1 & \cdots & 1 \\ 1 & 1 & & \\ \vdots & \vdots & \ddots & \vdots \\ 1 & 1 & & 1 \end{pmatrix}$$

which can map a vector in $\mathbb{R}^n$ to the mean of its entries. It has the nice property that $P^2 = P$, $P^T = P$.

Out[•]=



We can then write $SS_T = \langle(\mathbb{1}_n - P)\,Y, (\mathbb{1}_n - P)\,Y\rangle = \langle Y, (\mathbb{1}_n - P)\,Y\rangle$. Furthermore, we define the hat matrix $H = X(X^T X)^{-1} X^T$, then $\hat{Y} = X\,b = X(X^T X)^{-1} X^T\,Y = H\,Y$.

$$\begin{aligned} SS_T &= \textstyle\sum_{i=1}^n \left(Y_i - \overline{Y}\right)^2 = \langle Y, (\mathbb{1}_n - P)\,Y\rangle \\ &= \langle Y, (\mathbb{1}_n - H)\,Y\rangle + \langle Y, (H - P)\,Y\rangle \\ &= SS_E + SS_R \\ &= \textstyle\sum_{i=1}^n \left(Y_i - \hat{Y}_i\right)^2 + \sum_{i=1}^n \left(\hat{Y}_i - \overline{Y}\right)^2 \end{aligned}$$

where SS_R is the ***regression sum of squares***. Then the coefficient of multiple determination is

$$R^2 = \frac{SS_T - SS_E}{SS_T} = \frac{SS_R}{SS_T}$$

In[24]:=
```
lmQuadratic["RSquared"]
```

Out[24]=
```
0.98973
```

Important Note. If you want to get R^2 directly from `model["RSquared"]`, use `LinearModelFit` instead of `NonlinearModelFit`.

Distribution of Sum of Squares Error

- SS_E / σ^2 follows a chi-squared distribution with $n - p - 1$ degrees of freedom.
- If $\beta_1 = \beta_2 = \cdots = \beta_p = 0$, then SS_R / σ^2 follows a chi-squared distribution with p degrees of freedom.
- SS_R and SS_E are independent.
- The estimator for variance $S^2 = \frac{SS_E}{n-p-1}$ is unbiased.

In[∘]:=
```
lmQuadratic["EstimatedVariance"]
```
Out[∘]=
```
0.00115691
```

F-test for significance of Regression

Testing Parameter	Null Hypothesis	Test Statistics
$\beta_1, \beta_2, \ldots, \beta_p$	$H_0 : \beta_1 = \beta_2 = \cdots = \beta_p = 0$	$F_{p,n-p-1} = \frac{SS_R/p}{SS_E/(n-p-1)} = \frac{SS_R/p}{S^2} = \frac{n-p-1}{p}\frac{R^2}{1-R^2}$

We reject H_0 if $F_{p,n-p-1} > f_{\alpha,p,n-p-1}$.

In[137]:=
```
(Length[Data] - 2 - 1)/2 lmQuadratic["RSquared"]/(1 - lmQuadratic["RSquared"])
```
Out[137]=
```
240.921
```

In[28]:=
```
FStat = Mean[lmQuadratic["ANOVATableFStatistics"]]
```
Out[28]=
```
240.921
```

In[30]:=
```
1 - CDF[FRatioDistribution[2, 8 - 2 - 1], FStat]
```
Out[30]=
```
0.00001068946254299
```

Distribution of Least-Squares Estimators

- We have $E[\boldsymbol{b}] = \boldsymbol{\beta}$, meaning that it is unbiased, and
- $\text{Var}[\boldsymbol{b}] = \sigma^2 (\boldsymbol{X}^T \boldsymbol{X})^{-1}$, where $\text{Var}[\boldsymbol{b}] = \begin{pmatrix} \text{Var}[b_0] & \text{Cov}[b_0, b_1] & \cdots & \text{Cov}[b_0, b_p] \\ \text{Cov}[b_0, b_1] & \text{Var}[b_1] & \ddots & \vdots \\ \vdots & \ddots & \ddots & \vdots \\ \text{Cov}[b_0, b_p] & \cdots & \cdots & \text{Var}[b_p] \end{pmatrix}$ is the covariance matrix. The variance of a parameter will be $\text{Var}[B_i] = \xi_{ii}\,\sigma^2$, where ξ_{ii} is the $(i+1)^{\text{th}}$ diagonal element of $(\boldsymbol{X}^T \boldsymbol{X})^{-1}$.
- The random vector $\boldsymbol{b}$ follows normal distribution.
- The statistic $(n - p - 1)\, S^2 / \sigma^2 = SS_E / \sigma^2$ is independent of $\boldsymbol{b}$.

In[135]:=
```
lmQuadratic["EstimatedVariance"] Inverse[Transpose[x].x] // MatrixForm
```
Out[135]//MatrixForm=

$$\begin{pmatrix} 0.00225183 & -0.000105361 & 1.03295\times10^{-6} \\ -0.000105361 & 5.85339\times10^{-6} & -6.19771\times10^{-8} \\ 1.03295\times10^{-6} & -6.19771\times10^{-8} & 6.88634\times10^{-10} \end{pmatrix}$$

In[46]:=
```
lmQuadratic["CovarianceMatrix"] // MatrixForm
```
Out[46]//MatrixForm=

$$\begin{pmatrix} 0.00225183 & -0.000105361 & 1.03295\times10^{-6} \\ -0.000105361 & 5.85339\times10^{-6} & -6.19771\times10^{-8} \\ 1.03295\times10^{-6} & -6.19771\times10^{-8} & 6.88634\times10^{-10} \end{pmatrix}$$

Confidence Interval of Least-Squares Estimators

The $100\,(1-\alpha)\,\%$ confidence intervals for the model parameters are

$$\beta_j = b_j \pm t_{\alpha/2,n-p-1}\, S\, \sqrt{\xi_{jj}}\,, \qquad j = 0,\ \ldots,\ p$$

In[50]:=
```
lmQuadratic["ParameterConfidenceIntervalTable", ConfidenceLevel → 0.95]
```

Out[50]=

	Estimate	Standard Error	Confidence Interval
#1	0.0350763	0.0474535	{-0.0869068, 0.157059}
#2	0.00664771	0.00241938	{0.000428496, 0.0128669}
#3	0.0000535885	0.0000262418	{-0.0000138683, 0.000121045}

Distribution of Estimated Mean

The $100\,(1-\alpha)\,\%$ ***confidence interval*** for the conditional mean is

$$\hat{\mu}_{Y|\mathbf{x_0}} \pm t_{\alpha/2,n-p-1}\, S\, \sqrt{\mathbf{x_0}^T \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \mathbf{x_0}}$$

With $100\,(1-\alpha)\,\%$ chance, the conditional mean $\mu_{Y|\mathbf{x_0}}$ will lie in this interval.

The $100\,(1-\alpha)\,\%$ ***prediction interval*** for the observed value is

$$\hat{\mu}_{Y|\mathbf{x_0}} \pm t_{\alpha/2,n-p-1}\, S\, \sqrt{1 + \mathbf{x_0}^T \left(\boldsymbol{X}^T \boldsymbol{X}\right)^{-1} \mathbf{x_0}}$$

With $100\,(1-\alpha)\,\%$ chance, the newly observed value $Y\,|\,\mathbf{x_0}$ will lie in this interval.

In[109]:=
```
xpredict = 15;
yhat = Normal[lmQuadratic] /. {#1 → 1, #2 → xpredict, #3 → xpredict^2}
CI = lmQuadratic["MeanPredictionBands", ConfidenceLevel → 0.95] /.
   {#1 → 1, #2 → xpredict, #3 → xpredict^2}
PI = lmQuadratic["SinglePredictionBands", ConfidenceLevel → 0.95] /.
   {#1 → 1, #2 → xpredict, #3 → xpredict^2}
```

Out[110]= 0.146849

Out[111]= {0.0899842, 0.203714}

Out[112]= {0.04255, 0.251149}

T-Test for Model Sufficiency

Testing Parameter	Null Hypothesis	Test Statistics
β_j	$H_0 : \beta_j = 0$	$T_{n-p-1} = \dfrac{b_j}{S\sqrt{\xi_{jj}}}$

We reject H_0 if $|T_{n-p-1}| > t_{\alpha/2,n-p-1}$.

In[48]:=
```
lmQuadratic["ParameterTable"]
```

Out[48]=

	Estimate	Standard Error	t-Statistic	P-Value
#1	0.0350763	0.0474535	0.739173	0.493021
#2	0.00664771	0.00241938	2.74769	0.0404209
#3	0.0000535885	0.0000262418	2.0421	0.0966097

Partial F-Test for Model Sufficiency

Suppose we have two models:

- A ***full model*** with $p+1$ predictor variables

$$\mu_{Y|x_1,\ldots,x_p} = \beta_0 + \beta_1 x_1 + \ldots + \beta_p x_p$$

- A ***reduced model*** with $m+1$ predictor variables

$$\mu_{Y|\tilde{x}_1,\ldots,\tilde{x}_m} = \tilde{\beta}_0 + \tilde{\beta}_1 \tilde{x}_1 + \ldots + \tilde{\beta}_m \tilde{x}_m$$

where $\{\tilde{x}_1, \ldots \tilde{x}_m\} \subset \{x_1, \ldots, x_p\}$.

Null Hypothesis	Test Statistics
H_0 : reduced model is sufficient	$F_{p-m,n-p-1} = \frac{n-p-1}{p-m} \frac{SS_{E,\text{reduced}} - SS_{E,\text{full}}}{SS_{E,\text{full}}} = \frac{n-p-1}{p-m} \frac{SS_{R,\text{reduced}} - SS_{R,\text{full}}}{SS_{R,\text{full}}}$

We reject H_0 if $F_{p-m,n-p-1} > f_{\alpha,p-m,n-p-1}$.

Test whether we need a quadratic model instead of a simple linear model to fit this data.

- **Step 1**: Calculate the full and reduced model.

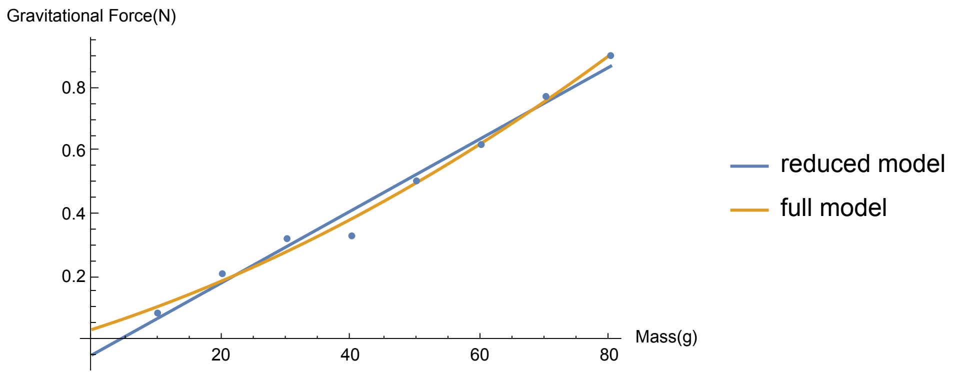
In[54]:=
```
lm = LinearModelFit[Data, m, m]
```

Out[54]=
```
FittedModel[ -0.0453065+0.0114707 m ]
```

In[52]:=
```
lmQuadratic
```

Out[52]=
```
FittedModel[ 0.0350763 #1+0.00664771 #2+0.0000535885 #3 ]
```

Out[65]=



- **Step 2**: Calculate the SS_E for both models.

In[59]:=
```
SSEreduced = Total[lm["FitResiduals"]^2]
```
Out[59]= `0.010609`

In[60]:=
```
SSEfull = Total[lmQuadratic["FitResiduals"]^2]
```
Out[60]= `0.00578453`

- **Step 3**: Calculate the test statistics and critical value.

In[66]:=
```
n = Length[Data];
p = 2; m = 1;
```
$$\texttt{FTestStat} = \frac{\texttt{n - p - 1}}{\texttt{p - m}} \frac{\texttt{SSEreduced - SSEfull}}{\texttt{SSEfull}}$$

Out[68]= `4.17018`

In[70]:=
```
InverseCDF[FRatioDistribution[p - m, n - p - 1], 0.95]
```
Out[70]= `6.60789`

Since $4.17 < 6.61$, there is no evidence that the full model is needed.

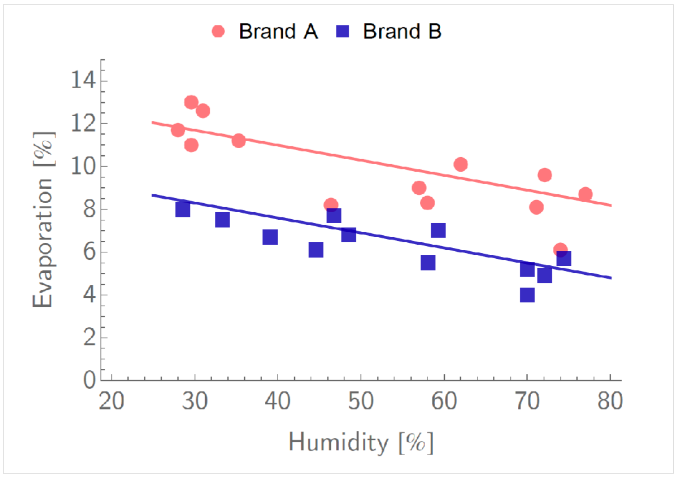
Indicator Variable

We can use $\ell - 1$ ***indicator variables*** to model ℓ levels. For example, we can define

$$(x_2, x_3) = \begin{cases} (0, 0) & \text{predictor is of type } A, \\ (1, 0) & \text{predictor is of type } B, \\ (0, 1) & \text{predictor is of type } C, \end{cases}$$

Suppose we have one numeric predictor x_1, we can set our model as

$$\hat{\mu}_{Y|x_1, x_2, x_3} = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3$$



if we want the indicator to only affect intercept, or

$$\hat{\mu}_{Y|x_1, x_2, x_3} = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + b_4 x_1 x_2 + b_5 x_1 x_3$$

if we want the indicator to also affect slope of x_1.

If we use $\hat{\mu}_{Y|x_1, x_2, x_3} = b_0 + b_1 x_1 + b_2 x_2 + b_3 x_3 + b_4 x_1 x_2 + b_5 x_1 x_3$, what will be our

estimation of $\mu_{Y|x_1,x_2,x_3}$ if the predictor is of type B?

Simply plug in $(x_2, x_3) = (1, 0)$ gives us $\hat{\mu}_{Y|x_1,x_2,x_3} = b_0 + b_1 x_1 + b_2 + b_4 x_1 = (b_0 + b_2) + (b_1 + b_4) x_1$.

Model Selection

Goal: Choose the model that fits the data, and do well in predictions.

Methods:

- **Forward Selection**:
  Start with the model only with β_0.
  For each step, find the one variable that improves the model's R^2 the most, and add it to the model.
  Stop the algorithm when the newly added parameter is not significant.
- **Backward Elimination**:
  Start with the full model.
  For each step, find the one variable that affects the model's R^2 the most, and delete it from the model.
  Stop the algorithm when the latest deleted parameter is significant.
- **Stepwise Method** (Not recommended).
- Minimize PRESS, maximize adjusted R^2, split data into training & test sets ......

Common Mistakes in the Assignment

Exercise 8.2 iii)

Exercise 8.2

In the experiment "Simple Harmonic Motion: Oscillations in Mechanical Systems" of the course Vp141 Physics Lab I, the spring coefficient is measured by using a Jolly balance. A spring is attached to the Jolly balance and weights are added to extend the spring. The extension L of the Jolly balance (not the actual spring extension) is recorded. For one spring the data (rounded) was obtained by two groups:

Group 1		Group 2	
L[cm]	m[g]	L[cm]	m[g]
4.88	0	4.95	0
6.92	4.7	7.00	4.7
8.99	9.5	9.10	9.5
11.09	14.3	11.20	14.3
13.18	19.1	13.30	19.1
15.26	23.9	15.41	24.0
17.39	28.7	17.51	28.7

Use Mathematica to do the following exercises:

i) For the given data, perform a simple linear regression for the random variable L as a function of the (non-random) parameter m. Plot the regression line.
(2 Marks)

ii) Calculate the value of R^2 and check for significance of regression.
(2 Marks)

iii) Perform a test for lack of fit. Is the linear model appropriate?
(2 Marks)

(Many thanks to Li Yingyu, Teachng Assistant for Vp241, for providing the data and advice on the experiment.)

In this case, the data $\{23.9, 15.26\}$, $\{24.0, 15.41\}$ should NOT be considered as repeated measurements. So k should be 8, the number of distinct value of m.

Thank you all very much for this wonderful semester.

Good luck with the Final!