The following is motivated by a physics problem, in which interest was in the slope of a function at 0; in particular, the radius of a proton can be estimated as $\sqrt{-6f^j(0)}$ where $f(x)$ is some function and we observe $Y_i = f(X_i) + s_i$. The physicists hoped to use the fact that the function is approximately linear near 0 to estimate $f^j(0)$. In this problem, the physicists selected a model by AIC and then constructed a confidence interval for the parameter based on the selected model, which was linear in the predictor.

Consider the ordinary least squares model

$$Y_i = -0.0001255 - 3.2258405X_i + 6.324418X_i^2 + s_i,$$
$$s_i \sim \text{Normal}(0,\ 0.001886^2).$$

First, draw $(X_1, Y_1), \ldots, (X_{240}, Y_{240})$ from this model, with $X_i \sim \text{Uniform}(0.01, 0.02)$; plot the data, the equation above, and the fit of a linear model with no quadratic term.

Conduct a simulation experiment by simulating $(X_1, Y_1), \ldots, (X_{240}, Y_{240})$ from this model, with $X_i \sim \text{Uniform}(0.01, 0.02)$. Repeat the following steps 1000 times:

(i) Simulate the data $X_1, \ldots, X_{240}$ and $Y_1, \ldots, Y_{240}$ from the model.

(ii) Fit the models

$$Y_i = \beta_0 + \beta_1 X_i + s_i,$$
$$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + s_i.$$

with $s_i \sim \text{Normal}(0, \sigma^2)$.

(iii) Select between these two models using AIC.

(iv) Based on the selected model, construct a point estimate and 95% confidence interval for β_1.

(v) Construct a point estimate and 95% confidence interval for β_1 based on the model including X^2.

(vi) Construct an AIC model averaged estimator of β_1,

$$\hat{\beta}_{\text{avg}} = w_{\text{linear}}\hat{\beta}_{1,\text{linear}} + w_{\text{quadratic}}\hat{\beta}_{1,\text{quadratic}},$$

with the Akaike weights given in the slides. *Hint:* to avoid overflow errors, compute the weights using:

```
AkaikeWeights<-function(aics)        {
  log_weights<--aics/2

  log_denominator<-max(log_weights)+
    log(sum(exp(log_weights-max(log_weights))))

  return(exp(log_weights-log_denominator))
}
```

In machine learning, normalizing on the log-scale in this fashion is sometimes referred to as the log-sum-exp trick.

(a) Based on your plot of the data with the true function and a linear fit, does the function appear to be approximately linear?

(b) How often did AIC select the correct model?

(c) What was the coverage of the confidence interval based on using AIC? Comment on the results

(d) What was the coverage of the interval based on using the model with X^2?

(e) For each of the three point estimates (linear, quadratic, and model averaged), compute the mean squared error

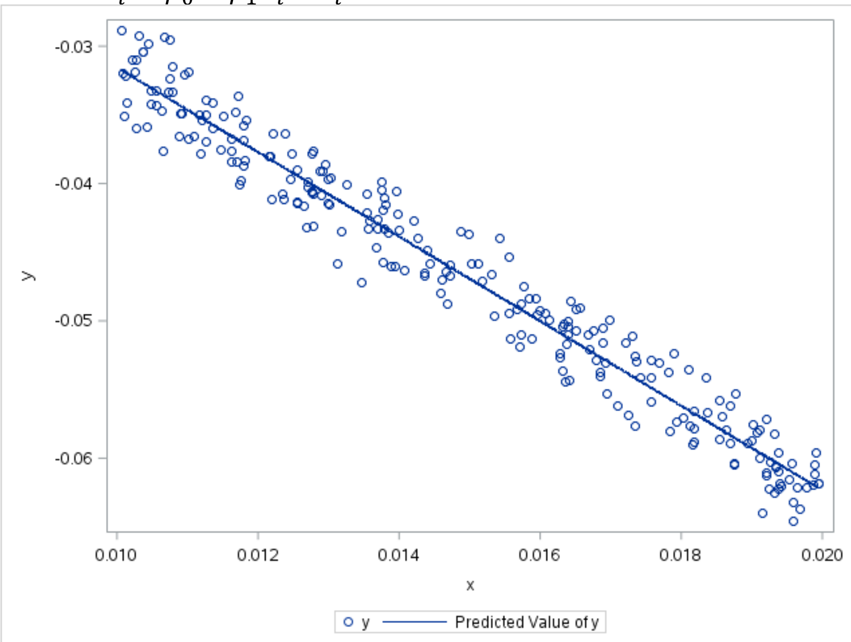
$$\text{MSE} = \frac{1}{1000} \sum_{j=1}^{1000} (\beta_1 - \hat{\beta}_{1j})^2.$$

Which estimator has the smallest MSE? Which estimator has the largest? What conclusions can you draw from this?

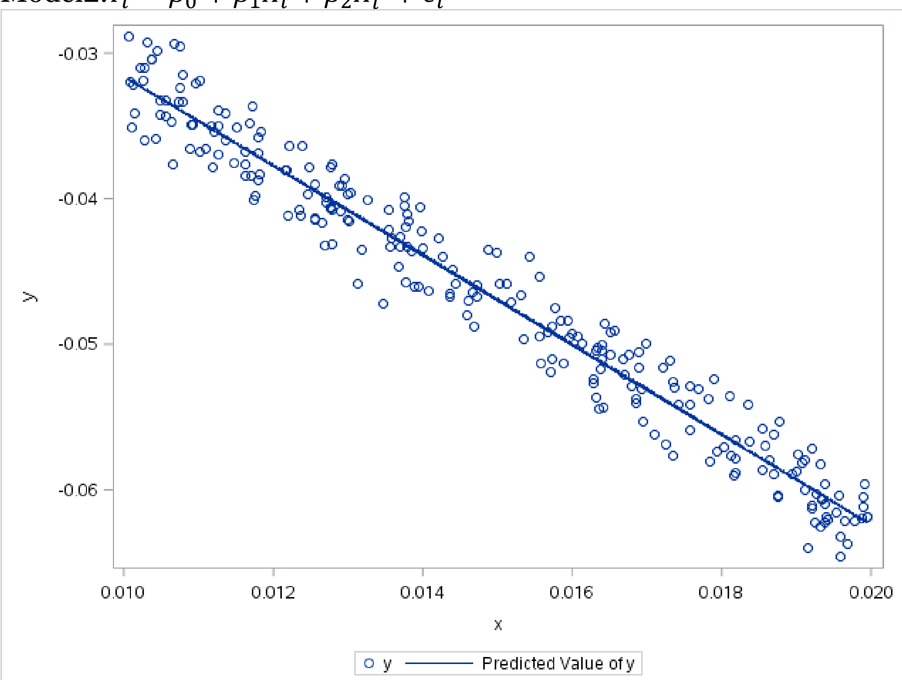
(f) Repeat the experiment with $n = 100000$ (X, Y) pairs (replicate 100 times), computing only AIC and BIC. How often does AIC select the true model? BIC? Repeat this, but set $\beta_2 = 0$, and again explain the differences in the performance of AIC and BIC. *Note:* in R, BIC can be computed as AIC(fit, k = log(n)).

Extra credit

Mingyuan Wang
(a)
This is the scatter plot of data with fit line of model1.
Model1:$Y_i = \beta_0 + \beta_1 X_i + \epsilon_i$



This is the scatter plot of data with fit line of model2.
Model2:$Y_i = \beta_0 + \beta_1 X_i + \beta_2 X_i^2 + \epsilon_i$



(b)

Obs	AIC_RIGHT_RATE
1	0.181

Base on the simulation of 1000 runs and 240 pairs. The correction rate of AIC is 0.181.
(c)

AIC_CI_LOW	AIC_CI_UP
-3.38965711143622	-2.89878323776869

The coverage of confidence interval of β_1 based on AIC is very small. In other words, it is precise. And model selected by AIC has low coefficient SD.

AIC_CI_ACCUR
0.15

But in 1000 runs, only 0.15 of times the true β_1 fall into the AIC CI.

(d)

X2_CI_LOW	X2_CI_UP
-4.19664864857487	-2.24684260910501

The coverage of confidence interval of β_1 based on model2 is bigger than that based on AIC. In other words, it is not as precise as CI based on AIC.

X2_CI_ACCUR
0.949

In 1000 runs, 0.949 of times the true β_1 fall into the CI with quadratic model.

(e)

LINEAR_MSE	QUAD_MSE	MEAN_MSE
0.03719409967747	0.23412903415352	0.03719409967747

Linear and model average estimator have the smallest MSE. Quadratic estimator gas the largest MSE. Although the true model is a quadratic one. It seems that the estimator of β_1 from simple linear model is more close to the true β_1.

(f)

When we set runtimes to100 and pairs to 100000, AIC and BIC both have 100% correct rate.

Obs	AIC_RIGHT_RATE	BIC_RIGHT_RATE
1	1	1

When we set the β_2=0, AIC and BIC both have 89% correct rate.

Obs	AIC_RIGHT_RATE	BIC_RIGHT_RATE
1	0.89	0.89

Code:

```
/*graph*/
data simulation;
do i= 1 to 240;
 x=rand("Uniform")*0.01+0.01;
 e=rand("Normal",0,0.001886);
 xsq=x**2;
 y=-0.0001255-3.2258405*x+6.324418*(x**2)+e;
 output;
end;
run;
ods graphics on;
proc reg data=simulation ;
```

```
model y=x ;
output out=draw1 predicted=predict;
run;
proc sgplot data=draw1;
scatter x=x y=y;
series x=x y=predict;
run;
proc reg data=simulation ;
model y=x xsq;
output out=draw2 predicted=predict;
run;
proc sgplot data=draw2;
scatter x=x y=y;
series x=x y=predict;
run;
proc sgplot data=draw2;
scatter x=x y=y;
series x=xsq y=predict;
run;
/*simulation*/
%macro simulate(runtime,pair,beta2,bic);

%let aiccorrect=0;
%let biccorrect=0;
%let aiclow=0;
%let aicup=0;
%let x2low=0;
%let x2up=0;
%let beta1E=0;
%let beta2E=0;
%let betamE=0;
%let aicCI=0;
%let x2CI=0;
%let aicCIrate=0;
%let x2CIrate=0;

%do n= 1 %to &runtime;
/*sample*/
data simulation;
do i= 1 to &pair;
 x=rand("Uniform")*0.01+0.01;
 e=rand("Normal",0,0.001886);
 xsq=x**2;
 y=-0.0001255-3.2258405*x+&beta2*(x**2)+e;
 output;
end;
run;
/*regression*/
proc reg data=simulation outest=result1 tableout alpha=0.05 noprint;
model y=x/aic &bic;
run;
proc transpose data=result1 out=r1;
id _type_;
```

```
var x;
run;
proc reg data=simulation outest=result2 tableout alpha=0.05 noprint;
model y=x xsq/aic &bic;
run;
proc transpose data=result2 out=r2;
id _type_;
var x;
run;
/***take statistics out***/
%if &bic=bic %then %do;
data _null_;/*bIC1*/
set result1(obs=1) ;
call symputx("bic1", _BIC_,L);
call symputx("aic1", _AIC_,L);
run;
data _null_;/*bIC2*/
set result2(obs=1) ;
call symputx("bic2", _BIC_,L);
call symputx("aic2", _AIC_,L);
run;
/*BIC correct chose*/
%if &beta2=0 %then %do;
%let biccorrect1=%sysevalf(&bic1<&bic2);
%let biccorrect=&biccorrect+&biccorrect1;
%end;
%else %do;
%let biccorrect1=%sysevalf(&bic1>&bic2);
%let biccorrect=&biccorrect+&biccorrect1;
%end;
%end;
%else %DO;
data _null_;/*AIC1*/
set result1(obs=1) ;
call symputx("aic1", _AIC_,L);
run;
data _null_;/*AIC2*/
set result2(obs=1) ;
call symputx("aic2", _AIC_,L);
run;
%end;

%IF %SYSEVALF(&aic1>&aic2) %then %do;
data _null_;/*ci*/
set r2;
call symputx("aiclow1",L95B,L);
call symputx("aicup1",U95B,L);
run;
%end;
%else %DO;
data _null_;/*ci*/
set r1;
call symputx("aiclow1",L95B,L);
```

```
call symputx("aicup1",U95B,L);
run;
%end;
/*get beta hat*/
data _null_;
set r1;
call symputx("betahat1",PARMS,L);
run;
data _null_;
set r2;
call symputx("betahat2",PARMS,L);
call symputx("x2low1",L95B,L);
call symputx("x2up1",U95B,L);
run;
/*CI AIC*/
%let aiclow=&aiclow+&aiclow1;
%let aicup=&aicup+&aicup1;
%let aicCI1=%sysevalf(&aiclow1<-3.2258405 and -3.2258405<&aicup1);
%let aicCI=&aicCI+&aicCI1;
/*CI x2*/
%let x2low=&x2low+&x2low1;
%let x2up=&x2up+&x2up1;
%let x2CI1=%sysevalf(&x2low1<-3.2258405 and -3.2258405<&x2up1);
%let x2CI=&x2CI+&x2CI1;
/*average beta*/
%let max=%sysfunc(max(-&aic1/2,-&aic2/2));
%let logw1=%sysevalf(-&aic1/2);
%let logw2=%sysevalf(-&aic2/2);
%let exp1=%sysfunc(exp(&logw1-&max));
%let exp2=%sysfunc(exp(&logw2-&max));
%let logw=%sysfunc(log(&exp1+&exp2));
%let weight1=%sysfunc(exp(&logw1-(&max+&logw)));
%let weight2=%sysfunc(exp(&logw2-(&max+&logw)));
%let betamean=%sysevalf(&betahat1*&weight1+&betahat2*&weight2);
/*AIC correct*/
%if &beta2=0 %then %do;
%let aiccorrect1=%sysevalf(&aic1<&aic2);
%let aiccorrect=&aiccorrect+&aiccorrect1;
%end;
%else %do;
%let aiccorrect1=%sysevalf(&aic1>&aic2);
%let aiccorrect=&aiccorrect+&aiccorrect1;
%end;
/*MSE*/
%let beta1E1=%sysevalf((-3.2258405-&betahat1)**2);
%let beta1E=&beta1E+&beta1E1;
%let beta2E1=%sysevalf((-3.2258405-&betahat2)**2);
%let beta2E=&beta2E+&beta2E1;
%let betamE1=%sysevalf((-3.2258405-&betahat1)**2);
%let betamE=&betamE+&betamE1;

%end;
/*CI MSE*/
```

```
%let beta1SE=%sysevalf(&beta1E);
%let beta1MSE=%sysevalf(&beta1SE/&runtime);
%let beta2SE=%sysevalf(&beta2E);
%let beta2MSE=%sysevalf(&beta2SE/&runtime);
%let betamSE=%sysevalf(&betamE);
%let betamMSE=%sysevalf(&betamSE/&runtime);
%let aiclowb=%sysevalf(&aiclow);
%let aiclowb=%sysevalf(&aiclowb/&runtime);
%let aicupb=%sysevalf(&aicup);
%let aicupb=%sysevalf(&aicupb/&runtime);
%let x2lowb=%sysevalf(&x2low);
%let x2lowb=%sysevalf(&x2lowb/&runtime);
%let x2upb=%sysevalf(&x2up);
%let x2upb=%sysevalf(&x2upb/&runtime);
%let aicCIrate=%sysevalf(&aicCI);
%let aicCIrate=%sysevalf(&aicCIrate/&runtime);
%let x2CIrate=%sysevalf(&x2CI);
%let x2CIrate=%sysevalf(&x2CIrate/&runtime);
%let aiccorrectrate=%sysevalf(&aiccorrect);
%let aiccorrectrate=%sysevalf(&aiccorrectrate/&runtime);
%if &bic=bic %then %do;
%let biccorrectrate=%sysevalf(&biccorrect);
%let biccorrectrate=%sysevalf(&biccorrectrate/&runtime);
%end;
/*print result*/
%if &bic=bic %then %do;
data show;
AIC_RIGHT_RATE=symget("aiccorrectrate");
BIC_RIGHT_RATE=symget("biccorrectrate");
run;
%end;
%else %do;
data show;
AIC_RIGHT_RATE=symget("aiccorrectrate");
AIC_CI_LOW=symget("aiclowb");
AIC_CI_UP=symget("aicupb");
X2_CI_LOW=symget("x2lowb");
X2_CI_UP=symget("x2upb");
AIC_CI_ACCUR=symget("aicCIrate");
X2_CI_ACCUR=symget("x2CIrate");
LINEAR_MSE=symget("beta1MSE");
QUAD_MSE=symget("beta2MSE");
MEAN_MSE=symget("betamMSE");
run;
%end;
proc print data=show;
run;

%mend simulate;

/*conduct simulation*/
%simulate(1000,240,6.324418,/*bic*/)
%simulate(100,100000,6.324418,bic)
```

%simulate(100,100000,0,bic)