

Model Selection and Diagnostics

Nathaniel E. Helwig

Assistant Professor of Psychology and Statistics
University of Minnesota (Twin Cities)



Updated 04-Jan-2017

Copyright © 2017 by Nathaniel E. Helwig

Outline of Notes

1) Model Selection:

- Overview of problem
- p value methods
- Adjusted R^2
- Information criteria
- Prediction based
- Example

2) Model Diagnostics:

- Normality assumption
- Linearity assumption
- Homogeneity of variance
- Equal influence
- Multicollinearity
- Example

Model Selection

The Problem: Which Variables??

The problem of **model selection** asks the question: which variables should be included in a multiple regression model?

We do not want to include too many predictors.

- Problem of over-fitting data
- Solution may not cross-validate

We do not want to include too few predictors.

- Miss important relationships in data
- Misinterpret relationships in data

All Possible Models

We need to consider ALL possible models that could be formed.

If we have p predictors, then (according to binomial theorem) there are

$$\sum_{j=1}^p \binom{p}{j} = 2^p$$

possible models we could choose.

Model Selection Strategies

We can use different statistical model selection strategies to choose which predictors to include.

There are a variety of strategies we can use:

- p value based methods (not so good)
- Adjusted R^2 (better)
- Information criteria (best)
- Prediction/cross-validation (best)

Overview of p Value Model Selection

p value model selection strategies choose which terms to include based on the significance of the terms (i.e., p -values of F tests).

There are three popular p -value based selection strategies:

- Backwards elimination
- Forward selection
- Stepwise selection

There is no guarantee that these selection strategies will produce a reasonable (or the same) model!

Backwards Elimination Algorithm

Given a threshold α^* , **backwards elimination algorithm** is:

- 1 Begin with all possible predictors in model
- 2 Remove predictor with largest p-value above α^*
- 3 Refit and repeat step 2 until all p-values below α^*

Note that α^* doesn't have to be the magical 0.05; typically set α^* larger (e.g., 0.10 or 0.15) if ultimately interested in prediction.

- Do not want to miss important predictors

Forward Selection Algorithm

Given a threshold α^* , **forward selection algorithm** is:

- 1 Begin with no predictors in model
- 2 Add predictor with smallest p-value below α^*
- 3 Refit and repeat step 2 until no new p-values below α^*

Note that α^* doesn't have to be the magical 0.05; typically set α^* larger (e.g., 0.10 or 0.15) if ultimately interested in prediction.

- Do not want to miss important predictors

Stepwise Selection Algorithm

Given thresholds α_F^* and α_B^* , **stepwise selection algorithm** is:

- 1 Begin with no predictors in model
- 2 Forward step: add predictor with smallest p-value below α_F^*
- 3 Backward step: remove predictor with largest p-value above α_B^*
- 4 Repeat steps 2–3 until convergence (or max steps reached)

Note that α_F^* and α_B^* do not have to be the magical 0.05; typically set α^* larger (e.g., 0.10 or 0.15) if ultimately interested in prediction.

- α_F^* and α_B^* do not have to be equal

Coefficient of Multiple Determination (revisited)

Consider the MLR model $y_i = b_0 + \sum_{j=1}^p b_j x_{ij} + e_i$ with $e_i \stackrel{\text{iid}}{\sim} N(0, \sigma^2)$.

Remember: the **coefficient of multiple determination** is defined as

$$\begin{aligned} R^2 &= \frac{SSR}{SST} \\ &= 1 - \frac{SSE}{SST} \end{aligned}$$

and gives the amount of variation in y_i that is explained by the linear relationships with x_{i1}, \dots, x_{ip} .

Adjusted R^2 (revisited)

Including more predictors in a MLR model can artificially inflate R^2 :

- Capitalizing on spurious effects present in noisy data
- Phenomenon of **over-fitting** the data

The **adjusted R^2** is a relative measure of fit:

$$\begin{aligned} R_a^2 &= 1 - \frac{SSE/df_E}{SST/df_T} \\ &= 1 - \frac{\hat{\sigma}^2}{s_Y^2} \end{aligned}$$

where $s_Y^2 = \frac{\sum_{i=1}^n (y_i - \bar{y})^2}{n-1}$ is the sample estimate of the variance of Y .

Adjusted R^2 for Model Selection

If p is not too large, could calculate R_a^2 for all 2^p possible models.

- Pick model with largest R_a^2 .

Implemented in `leaps` function (`leaps` package).

- Branch-and-bound search through all possible subsets
- Use `method="adjr2"` option to select via adjusted R^2

Likelihood Function (revisited)

Remember that $(\mathbf{y}|\mathbf{X}) \sim N(\mathbf{X}\mathbf{b}, \sigma^2\mathbf{I}_n)$, which implies that \mathbf{y} has pdf

$$f(\mathbf{y}|\mathbf{X}, \mathbf{b}, \sigma^2) = (2\pi)^{-n/2}(\sigma^2)^{-n/2} e^{-\frac{1}{2\sigma^2}(\mathbf{y}-\mathbf{X}\mathbf{b})'(\mathbf{y}-\mathbf{X}\mathbf{b})}$$

As a result, the log-likelihood of (\mathbf{b}, σ^2) given (\mathbf{y}, \mathbf{X}) is

$$\ln\{L(\mathbf{b}, \sigma^2|\mathbf{y}, \mathbf{X})\} = -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\sigma^2) - \frac{1}{2\sigma^2}(\mathbf{y} - \mathbf{X}\mathbf{b})'(\mathbf{y} - \mathbf{X}\mathbf{b})$$

Maximized Likelihood Functions

Remember that the MLEs of \mathbf{b} and σ^2 are

$$\begin{aligned}\hat{\mathbf{b}} &= (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\mathbf{y} \\ \tilde{\sigma}^2 &= SSE/n\end{aligned}$$

where $SSE = (\mathbf{y} - \mathbf{X}\hat{\mathbf{b}})'(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}})$ is the sum-of-squared errors.

As a result, the **maximized log-likelihood** of (\mathbf{b}, σ^2) given (\mathbf{y}, \mathbf{X}) is

$$\begin{aligned}\ln\{L(\hat{\mathbf{b}}, \tilde{\sigma}^2|\mathbf{y}, \mathbf{X})\} &= -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\tilde{\sigma}^2) - \frac{1}{2\tilde{\sigma}^2}(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}})'(\mathbf{y} - \mathbf{X}\hat{\mathbf{b}}) \\ &= -\frac{n}{2}\ln(2\pi) - \frac{n}{2}\ln(\tilde{\sigma}^2) - \frac{n}{2}\end{aligned}$$

Likelihoods and Information Criteria

Information criteria define model fit using maximized likelihoods that are penalized according to model complexity.

Defining $\hat{\mathcal{L}} = \ln\{L(\hat{\mathbf{b}}, \hat{\sigma}^2 | \mathbf{y}, \mathbf{X})\}$, Akaike's (1974) **AIC** is defined as

$$AIC = -2\hat{\mathcal{L}} + 2k$$

where k is number of parameters; note that AIC stands for **An Information Criterion**, but people typically refer to it as Akaike's.

The **Bayesian Information Criterion** (**BIC**; Schwarz, 1978) is

$$BIC = -2\hat{\mathcal{L}} + \ln(n)k$$

Information Criteria in Regression

Using the definition $\hat{\mathcal{L}} = -\frac{n}{2} \ln(2\pi) - \frac{n}{2} \ln(\tilde{\sigma}^2) - \frac{n}{2}$, we have that

$$AIC = n + n \ln(2\pi) + n \ln(\tilde{\sigma}^2) + 2k$$

$$BIC = n + n \ln(2\pi) + n \ln(\tilde{\sigma}^2) + \ln(n)k$$

where $k = p + 1$ is the number of columns of the model design matrix.

In some cases the constant $n + n \ln(2\pi)$ is dropped, such as

$$AIC^* = n \ln(\tilde{\sigma}^2) + 2k$$

$$BIC^* = n \ln(\tilde{\sigma}^2) + \ln(n)k$$

Information Criteria and Model Selection

AIC and BIC are theoretical optimal criteria for model selection.

- Smaller AIC (or BIC) means better model.
- $AIC < BIC$ whenever $n \geq 8 \implies AIC$ tends to pick larger models

AIC is optimal model selection criterion if trying to find model that best describes data among possible candidate models

- True model is unknown and not one of candidate models

BIC is optimal model selection criterion if trying to find true model among possible candidate models

- True model is one of candidate models

AIC and BIC Model Selection in R

You can perform AIC and BIC model selection using `step` function.

Default is stepwise AIC selection (`direction="both"` and `k=2`)

- Use `direction="backward"` or `direction="forward"` to change selection algorithm
- Set `k=log(n)` to perform BIC selection

Prediction and Model Selection

If we are ultimately interested in prediction, we can use prediction-based criteria to select our model.

Idea: minimize prediction SSE (instead of SSE for given data).

Most implementations do exhaustive (or branch-and-bound) searches, but you could use these criterion in a stepwise fashion too.

Mallow's C_p

Consider the model $\mathbf{y} = \mathbf{X}\mathbf{b} + \mathbf{e}$ where \mathbf{X} is $n \times m$ and $\mathbf{e} \sim N(\mathbf{0}, \sigma^2 \mathbf{I})$.

If we want to estimate the mean-squared prediction error (MSPE)

$$\frac{1}{\sigma^2} \sum_{i=1}^n E \left\{ [\hat{y}_i - E(y_i | \mathbf{x}_i)]^2 \right\}$$

we can use Mallow's (1973) C_p

$$C_p = \frac{SSE_p}{\hat{\sigma}^2} - n + 2p$$

where

- SSE_p is the SSE with $p < m$ columns of \mathbf{X} used in fit
- $\hat{\sigma}^2 = SSE/(n - m)$ is the MSE of full model

Mallow's C_p in R

Implemented in `leaps` function (`leaps` package).

- Branch-and-bound search through all possible subsets
- Use default `method="Cp"` option to select via Mallow's C_p

We could also use the `drop1` and `add1` functions

- These functions drop/add one predictor to a model
- If `sumF=summary(Fmod)` where `Fmod` is the full model, then use the input `scale=sumF$sigma^2` to get Mallow's C_p

Predicted Residual Sum-of-Squares (PRESS)

The Predicted Residual Sum-of-Squares (PRESS) statistic is

$$\text{PRESS} = \sum_{i=1}^n (y_i - \hat{y}_{[-i]})^2 = \sum_{i=1}^n \left(\frac{\hat{e}_i}{1 - h_{ii}} \right)^2$$

where

- $\hat{y}_{[-i]} = \mathbf{x}_i \hat{\mathbf{b}}_{[-i]}$ and $\hat{\mathbf{b}}_{[-i]}$ is estimate of \mathbf{b} without i -th observation
- \hat{e}_i is i -th estimated residual from full model
- h_{ii} is i -th leverage score from full model

PRESS Statistic in R

```
getpress <- function(ix,y,x){  
  if(any(ix)){  
    linmod=lm(y~.,data=as.data.frame(x[,ix]))  
  } else {  
    linmod=lm(y~1)  
  }  
  sum((linmod$residuals/(1-hatvalues(linmod)))^2)  
}
```

```
presslm <- function(x,y){  
  x=as.data.frame(x)  
  np=ncol(x)  
  xlist=vector("list",np)  
  for(j in 1:np){xlist[[j]]=c(TRUE,FALSE)}  
  xall=expand.grid(xlist)  
  allpress=apply(xall,1,getpress,y=y,x=x)  
  list(which=as.matrix(xall),press=allpress)  
}
```

R State Facts Data

The `state.x77` matrix contains 8 variables (columns) collected from the 50 states (rows) during the early-to-mid 1970s

- `Population`: estimate of state population (1975)
- `Income`: per capita income (1974)
- `Illiteracy`: percent illiterate (1970)
- `Life Exp`: life expectancy (1969–1971)
- `Murder`: murder rate per 100,000 people (1976)
- `HS.Grad`: percent high-school graduates (1970)
- `Frost`: mean number of days with minimum temperature below freezing (1931–1960)
- `Area`: land area in square miles

Look at State Facts Data

```
> states = data.frame(state.x77, row.names=state.abb)
> states[1:15,]
```

	Population	Income	Illiteracy	Life.Exp	Murder	HS.Grad	Frost	Area
AL	3615	3624	2.1	69.05	15.1	41.3	20	50708
AK	365	6315	1.5	69.31	11.3	66.7	152	566432
AZ	2212	4530	1.8	70.55	7.8	58.1	15	113417
AR	2110	3378	1.9	70.66	10.1	39.9	65	51945
CA	21198	5114	1.1	71.71	10.3	62.6	20	156361
CO	2541	4884	0.7	72.06	6.8	63.9	166	103766
CT	3100	5348	1.1	72.48	3.1	56.0	139	4862
DE	579	4809	0.9	70.06	6.2	54.6	103	1982
FL	8277	4815	1.3	70.66	10.7	52.6	11	54090
GA	4931	4091	2.0	68.54	13.9	40.6	60	58073
HI	868	4963	1.9	73.60	6.2	61.9	0	6425
ID	813	4119	0.6	71.87	5.3	59.5	126	82677
IL	11197	5107	0.9	70.14	10.3	52.6	127	55748
IN	5313	4458	0.7	70.88	7.1	52.9	122	36097
IA	2861	4628	0.5	72.56	2.3	59.0	140	55941

State Data: Full Model

```
> fullmod = lm(Murder ~ . , data=states)
> summary(fullmod)      # I deleted some output
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.222e+02	1.789e+01	6.831	2.54e-08	***
Population	1.880e-04	6.474e-05	2.905	0.00584	**
Income	-1.592e-04	5.725e-04	-0.278	0.78232	
Illiteracy	1.373e+00	8.322e-01	1.650	0.10641	
Life.Exp	-1.655e+00	2.562e-01	-6.459	8.68e-08	***
HS.Grad	3.234e-02	5.725e-02	0.565	0.57519	
Frost	-1.288e-02	7.392e-03	-1.743	0.08867	.
Area	5.967e-06	3.801e-06	1.570	0.12391	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.746 on 42 degrees of freedom
 Multiple R-squared: 0.8083, Adjusted R-squared: 0.7763
 F-statistic: 25.29 on 7 and 42 DF, p-value: 3.872e-13

State Data: Adjusted R^2 Selection

```
> X = states[,-5]
> arsqmod = leaps(x=X, y=states$Murder, method="adjr2")
> widx = which.max(arsqmod$adjr2)
> xidx = (1:ncol(X))[arsqmod$which[widx,]]
> Xin = data.frame(X[,xidx])
> arsqmod = lm(states$Murder ~ . , data=Xin)
> summary(arsqmod)      # I deleted some output
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.202e+02	1.718e+01	6.994	1.17e-08	***
Population	1.780e-04	5.930e-05	3.001	0.00442	**
Illiteracy	1.173e+00	6.801e-01	1.725	0.09161	.
Life.Exp	-1.608e+00	2.324e-01	-6.919	1.50e-08	***
Frost	-1.373e-02	7.080e-03	-1.939	0.05888	.
Area	6.804e-06	2.919e-06	2.331	0.02439	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.712 on 44 degrees of freedom
 Multiple R-squared: 0.8068, Adjusted R-squared: 0.7848
 F-statistic: 36.74 on 5 and 44 DF, p-value: 1.221e-14

State Data: Stepwise AIC Selection

```
> smod = lm(states$Murder ~ . , data=states)
> aicmod = step(smod, trace=0)
> summary(aicmod)      # I deleted some output
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.202e+02	1.718e+01	6.994	1.17e-08	***
Population	1.780e-04	5.930e-05	3.001	0.00442	**
Illiteracy	1.173e+00	6.801e-01	1.725	0.09161	.
Life.Exp	-1.608e+00	2.324e-01	-6.919	1.50e-08	***
Frost	-1.373e-02	7.080e-03	-1.939	0.05888	.
Area	6.804e-06	2.919e-06	2.331	0.02439	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.712 on 44 degrees of freedom
 Multiple R-squared: 0.8068, Adjusted R-squared: 0.7848
 F-statistic: 36.74 on 5 and 44 DF, p-value: 1.221e-14

State Data: Stepwise BIC Selection

```
> smod = lm(states$Murder ~ . , data=states)
> bicmod = step(smod, k=log(50), trace=0)
> summary(bicmod)      # I deleted some output
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.387e+02	1.369e+01	10.136	3.40e-13	***
Population	1.581e-04	5.944e-05	2.660	0.010778	*
Life.Exp	-1.837e+00	1.946e-01	-9.442	3.04e-12	***
Frost	-2.204e-02	5.299e-03	-4.160	0.000141	***
Area	7.387e-06	2.962e-06	2.494	0.016374	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.75 on 45 degrees of freedom
 Multiple R-squared: 0.7937, Adjusted R-squared: 0.7754
 F-statistic: 43.28 on 4 and 45 DF, p-value: 7.106e-15

State Data: Mallow's C_p Selection

```
> X = states[,-5]
> cpmod = leaps(x=X, y=states$Murder, method="Cp")
> widx = which.min(cpmod$Cp)
> xidx = (1:ncol(X))[cpmod$which[widx,]]
> Xin = data.frame(X[,xidx])
> cpmod = lm(states$Murder ~ . , data=Xin)
> summary(cpmod)          # I deleted some output
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.202e+02	1.718e+01	6.994	1.17e-08	***
Population	1.780e-04	5.930e-05	3.001	0.00442	**
Illiteracy	1.173e+00	6.801e-01	1.725	0.09161	.
Life.Exp	-1.608e+00	2.324e-01	-6.919	1.50e-08	***
Frost	-1.373e-02	7.080e-03	-1.939	0.05888	.
Area	6.804e-06	2.919e-06	2.331	0.02439	*

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.712 on 44 degrees of freedom
 Multiple R-squared: 0.8068, Adjusted R-squared: 0.7848
 F-statistic: 36.74 on 5 and 44 DF, p-value: 1.221e-14

State Data: PRESS Selection

```
> X = states[,-5]
> prmod = presslm(x=X, y=states$Murder)
> widx = which.min(prmod$press)
> xidx = (1:ncol(X)) [prmod$which[widx,]]
> Xin = as.data.frame(X[,xidx])
> prmod = lm(states$Murder ~ . , data=Xin)
> summary(prmod)          # I deleted some output
```

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)	
(Intercept)	1.259e+02	1.777e+01	7.083	8.64e-09	***
Population	1.946e-04	6.078e-05	3.202	0.00254	**
Illiteracy	1.912e+00	7.620e-01	2.509	0.01587	*
Life.Exp	-1.757e+00	2.491e-01	-7.053	9.57e-09	***
HS.Grad	7.626e-02	4.369e-02	1.746	0.08786	.
Frost	-1.011e-02	7.199e-03	-1.404	0.16719	

Signif. codes: 0 '***' 0.001 '**' 0.01 '*' 0.05 '.' 0.1 ' ' 1

Residual standard error: 1.755 on 44 degrees of freedom

Multiple R-squared: 0.797, Adjusted R-squared: 0.7739

F-statistic: 34.54 on 5 and 44 DF, p-value: 3.565e-14

State Data: Summary of Results

```

> xnames = colnames(states)[-5]
> xtab = matrix(0,5,7)
> rownames(xtab) = c("Ra^2", "AIC", "BIC", "Cp", "PRESS")
> colnames(xtab) = xnames
> xlist = list(arsqmod, aicmod, bicmod, cpmod, prmod)
> for(j in 1:5){
+   ix = match(names(attr(xlist[[j]]$terms, "dataClasses"))[-1], xnames)
+   xtab[j,ix] = 1
+ }
> xtab

```

	Population	Income	Illiteracy	Life.Exp	HS.Grad	Frost	Area
Ra^2	1	0	1	1	0	1	1
AIC	1	0	1	1	0	1	1
BIC	1	0	0	1	0	1	1
Cp	1	0	1	1	0	1	1
PRESS	1	0	1	1	1	1	0

Model Diagnostics

Visualizing Non-Normality

Two visualizations (plots) useful for examining normality:

- **QQ-plot**: plots empirical (estimated) quantiles against theoretical normal quantiles
- **Histogram**: plots empirical (estimated) distribution of data

It is often helpful to add references lines to the plots:

- QQ-plot: add 45° line and/or qq-line (i.e., quantile-quantile line)
- Histogram: add empirical density for MLE normal

Testing for Non-Normality

QQ-plots and histograms provide nice visualizations, but are not formal tests of whether X follows a normal distribution.

To formally test the normality assumption, we could use the **Shapiro-Wilk normality test**.

- Test is $H_0 : Y \sim N(\mu, \sigma^2)$ versus $H_1 : Y \not\sim N(\mu, \sigma^2)$
- Reject H_0 if observed W is too small
- Implemented in the `shapiro.test` function in R

Solutions for Non-Normality

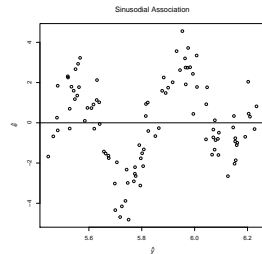
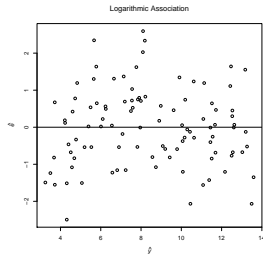
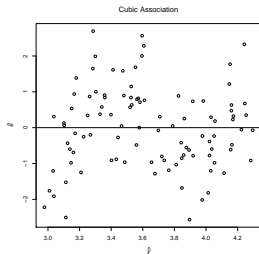
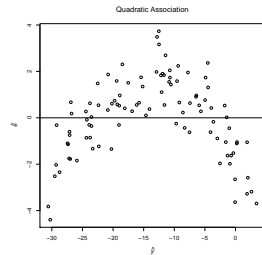
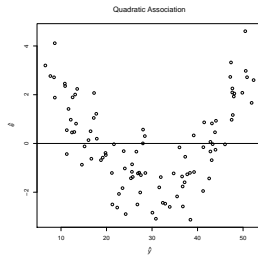
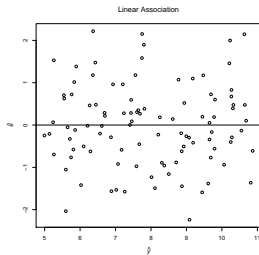
Many possible solutions to deal with non-normality in regression:

- Bootstrap to get SE estimates for regression coefficients
- Least-squares with rank transformed data (see below reference)
- Use generalized linear model (if data is exponential family)
- Nonparametric regression

Conover, W. J., & Iman, R. L. (1981). Rank transformations as a bridge between parametric and nonparametric statistics. *The American Statistician*, 35, 124–129.

Visualizing Non-Linearity

To visualize the linearity assumption, plot \hat{y}_i versus \hat{e}_i :



Testing for Non-Linearity

Box-Cox transformation:

- Finds optimal power transformation of y_i for MLR model
- Test significance of power transformation coefficient λ

Polynomial regression:

- Refit model with polynomial terms
- Test significance of higher order effects

Nonparametric regression:

- Fit nonparametric regression model
- Test significance of non-linear effects

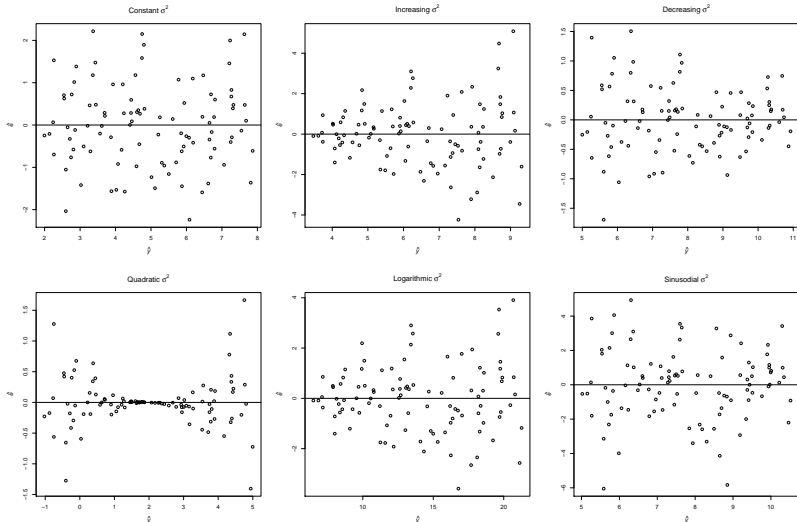
Solutions for Non-Linearity

If you have data with a nonlinear relationship, you could

- Transform data to have more linear relationship (e.g., Box-Cox)
- Fit polynomial regression model
- Fit nonparametric regression model
- Use other nonparametric approach (e.g., analyze rank data)

Visualizing Non-Constant Error Variance

To visualize the constant variance assumption, plot \hat{y}_i versus \hat{e}_i :



Testing for Non-Constant Error Variance

Consider the auxiliary model predicting the squared error terms

$$e_i^2 = \gamma_0 + \sum_{j=1}^p \gamma_j x_{ij} + \tilde{e}_i$$

where $\gamma = (\gamma_0, \gamma_1, \dots, \gamma_p)'$ are the auxiliary coefficients, and $\tilde{\mathbf{e}} = (\tilde{e}_1, \dots, \tilde{e}_n)'$ is the corresponding auxiliary error vector.

To test $H_0 : V(e_i) = \sigma^2$ vs. $H_1 : V(e_i) \neq \sigma^2$ use **Breusch-Pagan test**:

$$\chi_{BP}^2 = n\tilde{R}^2$$

where \tilde{R}^2 is coefficient of multiple determination from auxiliary model.

- As $n \rightarrow \infty$, we have $\chi_{BP}^2 \rightarrow \chi_p^2$
- Reject H_0 if $\chi_{BP}^2 > \chi_{p(\alpha)}^2$

Solutions for Non-Constant Error Variance

If $E(e_i^2) = \sigma_i^2$, then we have **heteroskedasticity**.

Weighted Least Squares

- Assumes that $\mathbf{e} \sim N(\mathbf{0}, \mathbf{W}^{-1})$ with $\mathbf{W} = \text{diag}(1/\sigma_1^2, \dots, 1/\sigma_n^2)$
- WLS solution: $\hat{\mathbf{b}}_w = (\tilde{\mathbf{X}}'\tilde{\mathbf{X}})^{-1}\tilde{\mathbf{X}}'\tilde{\mathbf{y}}$, where $\tilde{\mathbf{y}} = \mathbf{W}^{1/2}\mathbf{y}$ and $\tilde{\mathbf{X}} = \mathbf{W}^{1/2}\mathbf{X}$

Sandwich Standard Error Estimates

- $V(\hat{\mathbf{b}}) = \sigma^2(\mathbf{X}'\mathbf{X})^{-1}$ because we assume $\mathbf{e} \sim N(\mathbf{0}_n, \sigma^2\mathbf{I}_n)$
- $V_S(\hat{\mathbf{b}}) = (\mathbf{X}'\mathbf{X})^{-1}\mathbf{X}'\hat{E}(\mathbf{e}\mathbf{e}')\mathbf{X}(\mathbf{X}'\mathbf{X})^{-1}$ is sandwich estimate

Visualizing Equal Influence in Regression

Remember $\hat{\mathbf{e}} \sim N(\mathbf{0}_n, \sigma^2(\mathbf{I}_n - \mathbf{H})) \iff V(\hat{e}_i) = \sigma^2(1 - h_{ii})$.

- h_{ii} are the **leverage** values (diagonals of \mathbf{H})
- $\hat{y}_i = \sum_{j=1}^n h_{ij}y_j$, so large leverage may have large influence

Rule of thumb: leverages larger than $2\bar{h}$ should be looked at more closely, where $\bar{h} = \frac{1}{n} \sum_{i=1}^n h_{ii}$ is the mean leverage.

Plot leverage for each subject (along with $2\bar{h}$) to visualize influence.

Testing for Equal Influence in Regression

To test for unequal influence, we can use **Cook's distance**

- $\hat{\mathbf{b}}_{(i)}$ is OLS estimate of \mathbf{b} holding out i -th observation
- $(\hat{\mathbf{y}}_{(i)} - \hat{\mathbf{y}})'(\hat{\mathbf{y}}_{(i)} - \hat{\mathbf{y}}) = (\hat{\mathbf{b}}_{(i)} - \hat{\mathbf{b}})' \mathbf{X}' \mathbf{X} (\hat{\mathbf{b}}_{(i)} - \hat{\mathbf{b}})$ where $\hat{\mathbf{y}}_{(i)} = \mathbf{X} \hat{\mathbf{b}}_{(i)}$

Cook's (1977) distance D_i is defined as

$$D_i = \frac{(\hat{\mathbf{b}}_{(i)} - \hat{\mathbf{b}})' \mathbf{X}' \mathbf{X} (\hat{\mathbf{b}}_{(i)} - \hat{\mathbf{b}})}{(p+1)\hat{\sigma}^2} = \frac{\hat{e}_i^2}{(p+1)\hat{\sigma}^2} \left[\frac{h_{ii}}{(1+h_{ii})^2} \right] \sim F_{p+1, n-p-1}$$

- Note if $D_i \approx F_{p+1, n-p-1}^{(0.5)}$, then holding out i -th observation moves OLS estimate to edge of 50% confidence region
- Typically want $\hat{\mathbf{b}}_{(i)}$ to say within 5–10% (or less) region.

Solutions for Unequal Influence

Many possible solutions to deal with unequal influence:

- Rank (or other) transformation of data
- IRWLS (`r1m` function in `MASS` package)
- Regression trees (`cv.tree` function in `tree` package)
- Minimize L_1 norm (`lqnorm` function in `VGAM` package)
- Quantile regression (`rq` function in `quantreg` package)

Defining Multicollinearity

Consider the MLR model $y_i = b_0 + \sum_{j=1}^p b_j x_{ij} + e_i$ with $e_i \stackrel{\text{iid}}{\sim} N(0, \sigma^2)$; if the x_{ij} are highly correlated with one another, we have **multicollinearity**.

This is a problem because interpretation becomes difficult. . .

- b_j is expected change in Y holding other predictors constant
- If predictors are highly correlated, how do we interpret b_j ??

Multicollinearity is also a problem for model estimation. . .

- If predictors are highly correlated, the inverse $(\mathbf{X}'\mathbf{X})^{-1}$ is unstable
- Can not trust the resulting parameter and SE estimates

Quantifying Multicollinearity

Pairwise Correlations

- Examine correlation matrix or scatterplot matrix
- Smaller (in absolute magnitude) correlations are better

Part Correlation

- Let $\mathcal{X} = \text{span}\{1, X_1, \dots, X_p\}$ and define $\mathcal{D}_j = \mathcal{X} \ominus \mathcal{X}_j$
- Part correlation is $r_{Y(X_j, \mathcal{D}_j)} = \sqrt{R_{\mathcal{X}}^2 - R_{\mathcal{D}_j}^2}$, where $R_{\mathcal{X}}^2$ and $R_{\mathcal{D}_j}^2$ denote the R^2 with and without X_j

Variance Inflation Factors

- **Variance inflation factor** (VIF) is defined as: $(\text{VIF})_j = \frac{1}{1 - R_j^2}$
- Note that R_j^2 is the coefficient of multiple determination for predicting X_j from remaining predictors

Solutions for Multicollinearity

Remove one (or more) predictors

- Remove predictors with small part correlations
- Remove predictors with large VIFs correlations
- If possible, use theory to select most sensible predictors
- Otherwise use some model selection strategy

State Data: Normality Assumption

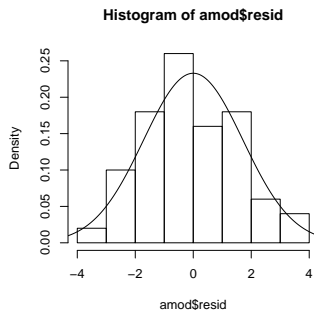
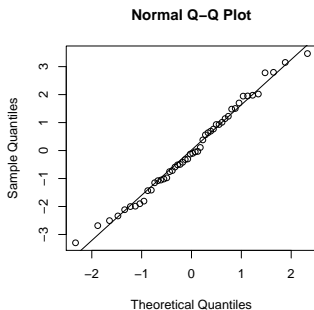
```
> amod = lm(Murder~Population+Illiteracy+Life.Exp+Frost+Area, data=states)
> shapiro.test(amod$resid)
```

Shapiro-Wilk normality test

```
data:  amod$resid
W = 0.986, p-value = 0.8116
```

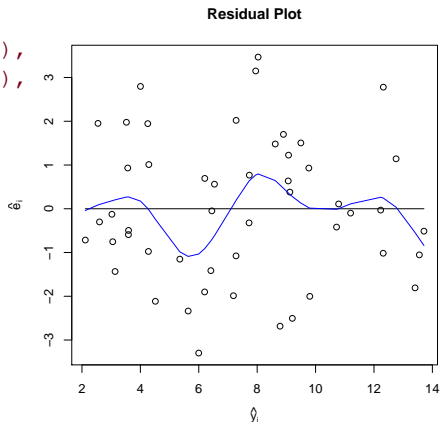
```
> par(mfrow=c(1,2))
> qqnorm(amod$resid)
> qqline(amod$resid)
> hist(amod$resid, freq=F)
> xseq=seq(-5,5,length=200)
> lines(xseq,dnorm(xseq,sd=summary(amod)$sigma))
```

State Data: Normality Assumption (continued)



State Data: Linearity Assumption

```
> yhat=amod$fit
> ehat=amod$resid
> plot(yhat,ehat,
+       xlab=expression(hat(y)[i]),
+       ylab=expression(hat(e)[i]),
+       main="Residual Plot")
> lines(range(yhat),c(0,0))
> smod=smooth.spline(yhat,ehat)
> lines(smod,col="blue")
```



State Data: Homogeneity of Variance

```
BPtest=function(mymod) {  
  mymod$model[,1]=(mymod$resid)^2  
  newmod=lm(formula(mymod),data=mymod$model)  
  modsum=summary(newmod)  
  Rsq=modsum$r.squared  
  BPstat=Rsq*(dim(mymod$model)[1])  
  pval=1-pchisq(BPstat,modsum$df[1]-1)  
  list(BP=BPstat,df=modsum$df[1]-1,pval=pval)  
}  
  
> BPtest(amod)  
$BP  
[1] 6.09788  
  
$df  
[1] 5  
  
$pval  
[1] 0.296811
```

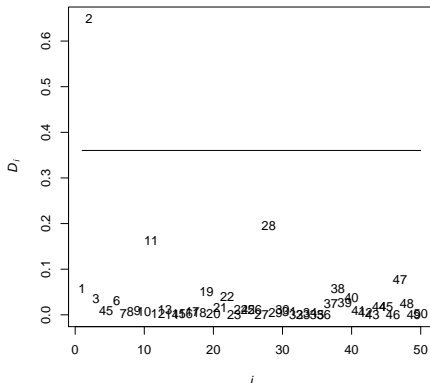
State Data: Equal Influence

```
cookplot<-function(mymod,k=NULL,alpha=0.1,ptext=TRUE,...){
  nx=dim(mymod$model)[1]
  np=length(mymod$coef)
  cdist=cooks.distance(mymod)
  if(is.null(k)){k=qf(alpha,np,nx-np)}
  ylim=range(cdist)
  if(ylim[1]>k){ylim[1]=k} else if(ylim[2]<k){ylim[2]=k}
  if(ptext){
    plot(1:nx,cdist,type="n",xlab=expression(italic(i)),ylim=ylim,
         ylab=expression(italic(D[i])),main="Cook's Distance Plot")
    text(1:nx,cdist,1:nx)
  } else{plot(1:nx,cdist,xlab=expression(italic(i)),ylim=ylim,
              ylab=expression(italic(D[i])),main="Cook's Distance Plot")}
  lines(c(1,nx),c(k,k),...)
}
> cookplot(amod)
```

State Data: Equal Influence (continued)

```
> rownames(states)[c(2,11,28)]
[1] "AK" "HI" "NV"
```

Cook's Distance Plot



May want to refit model without Alaska, which is highly influential.

State Data: Multicollinearity

```
> library(faraway)
> X=model.matrix(amod)[,-1]
> X[1:4,]
```

	Population	Illiteracy	Life.Exp	Frost	Area
AL	3615	2.1	69.05	20	50708
AK	365	1.5	69.31	152	566432
AZ	2212	1.8	70.55	15	113417
AR	2110	1.9	70.66	65	51945

```
> vif(X)
```

Population	Illiteracy	Life.Exp	Frost	Area
1.171232	2.871577	1.625921	2.262943	1.036358

```
> round(cor(X), 3)
```

	Population	Illiteracy	Life.Exp	Frost	Area
Population	1.000	0.108	-0.068	-0.332	0.023
Illiteracy	0.108	1.000	-0.588	-0.672	0.077
Life.Exp	-0.068	-0.588	1.000	0.262	-0.107
Frost	-0.332	-0.672	0.262	1.000	0.059
Area	0.023	0.077	-0.107	0.059	1.000