

Regression Examples

Dave Lorenz

October 27, 2015

Abstract

These examples demonstrates the functions in the **smwrStats** package that aid linear regression analysis. The first set of examples uses the Haan (1977) dataset C14 from Helsel and Hirsch (2002) to replicate the analysis done in section 11.6 in their book. Please note that there are only 13 observations in the dataset and most practitioners would prefer more observations for this kind of multiple regression analysis; Harrell (2001) has some very good guidance in chapter 4 of his book. See section **Subset Selection Comments** about some specific for this example. The second set of examples uses the CuyhaogaTDS dataset in the **smwrData** package. Those examples demonstrate bias correction for a log-transformed response.

Contents

1	Introduction	3
2	Subset Selection	4
3	Model Diagnostics	6
4	Response vs. Fitted Plot	8
5	Scale-Location Plot	9
6	Probability Plot	11
7	Influence Plot	12
8	Residual Dependence Plot	13

9 Support Functions	14
10 Subset Selection Comments	16
11 Bias Correction	18

1 Introduction

These examples use data from the `smwrData` package. The data are created and retrieved in the following code.

```
> # Load the smwrStats package
> library(smwrStats)
> # Create the Haan dataset
> Haan1977 <- data.frame(
+ ROFF=c(17.38, 14.62, 15.48, 14.72, 18.37, 17.01, 18.2, 18.95, 13.94, 18.64,
+ 17.25, 17.48, 13.16),
+ PCIP=c(44.37, 44.09, 41.25, 45.5, 46.09, 49.12, 44.03, 48.71, 44.43, 47.72,
+ 48.38, 49, 47.03),
+ AREA=c(2.21, 2.53, 5.63, 1.55, 5.15, 2.14, 5.34, 7.47, 2.1, 3.89, 0.67,
+ 0.85, 1.72),
+ SLOPE=c(50, 7, 19, 6, 16, 26, 7, 11, 5, 18, 21, 23, 5),
+ LEN=c(2.38, 2.55, 3.11, 1.84, 4.14, 1.92, 4.73, 4.24, 2, 2.1, 1.15, 1.27,
+ 1.93),
+ PERIM=c(7.93, 7.65, 11.61, 5.31, 11.35, 5.89, 12.59, 12.33, 6.81, 9.87,
+ 3.93, 3.79, 5.19),
+ DI=c(0.91, 1.23, 2.11, 0.94, 1.63, 1.41, 1.3, 2.35, 1.19, 1.65, 0.62, 0.83,
+ 0.99),
+ Rs=c(0.38, 0.48, 0.57, 0.49, 0.39, 0.71, 0.27, 0.52, 0.53, 0.6, 0.48, 0.61,
+ 0.52),
+ FREQ=c(1.36, 2.37, 2.31, 3.87, 3.3, 1.87, 0.94, 1.2, 4.76, 3.08, 2.99,
+ 3.53, 2.33),
+ Rr=c(332, 55, 77, 68, 68, 230, 44, 72, 40, 115, 352, 300, 39)
+ )
> # load the data library and get the Cuyahoga data
> library(smwrData)
> data(CuyahogaTDS)
```

2 Subset Selection

The `allReg` function creates a data frame that contains candidate models and selection criteria. Note that the name of the response variable is taken from the column name for the `y` argument if it is rectangular, and from the argument name if it is not. Two general approaches for specifying the `x` and `y` arguments. The first uses the `with` function and is most useful when a subset of columns are explanatory variables, but it is probably more clear in identifying the variables. The second, which is commented out, requires less typing and can be useful when most columns are explanatory variables as in this case. Note the argument `lin.dep` is used to protect against potential linear dependencies—the default is to require at least 10 more observations than explanatory variables. For these data `lin.dep` is set to 1 for these data because there are 13 observations and 9 explanatory variables.

As a preliminary

```
> # Create the allReg output dataset
> HaanSub <- with(Haan1977, allReg(cbind(PCIP, AREA, SLOPE, LEN, PERIM,
+   DI, Rs, FREQ, Rr), ROFF, lin.dep=1))
> # An alternative call, note the use of the drop argument
> #HaanSub <- allReg(Haan1977[, -1], Haan1977[, 1, drop=FALSE], lin.dep=1)
> # What are the "best" 5 models by Cp
> head(HaanSub[order(HaanSub$Cp),])
```

	model.formula	nvars	stderr	R2	adjr2
13	ROFF ~ PCIP + PERIM + DI + FREQ + Rr	5	0.5157295	95.85716	92.89799
10	ROFF ~ PCIP + PERIM + DI + Rr	4	0.6201892	93.15309	89.72964
11	ROFF ~ PCIP + AREA + PERIM + Rr	4	0.6271707	92.99807	89.49710
12	ROFF ~ PCIP + PERIM + FREQ + Rr	4	0.6418703	92.66600	88.99900
7	ROFF ~ PCIP + PERIM + Rr	3	0.6880745	90.51866	87.35822
14	ROFF ~ PCIP + AREA + SLOPE + PERIM + Rr	5	0.5853334	94.66345	90.85162

	Cp	press
13	2.895526	6.907075
10	3.438170	7.956472
11	3.583937	7.665134
12	3.896182	9.514826
7	3.915331	9.968029
14	4.017979	7.249822

Helsel and Hirsch (2002) state "Based on C_p , the best model would be the 5 variable model having PCIP, PERIM, DI, FREQ and Rr as explanatory variables—the same model as selected by `allReg`. Remember that there is no guarantee that stepwise procedures regularly select the lowest C_p or PRESS models. The advantage of using an overall statistic like C_p is that options are

given to the scientist to select what is best. If the scientist decided AREA must be in the model, the lowest CP model containing AREA (the same four-variable model) could be selected. Cp and PRESS allow model choice to be based on multiple criteria such as prediction quality (PRESS), low VIF, cost, etc.”

To select a good model, Helsel and Hirsch (2002) describe several criteria in section 11.7. Those criteria need to be considered in addition to the assumptions of linear regression (section 9.1.1) and regression diagnostics (sections 9.5 and 11.5).

The output from `allReg` can be used to evaluate any of the selected models, by using the `as.formula` function on the contents of the `model.formula` column as in the following example.

```
lm(as.formula(HaanSub[13, "model.formula"]), data=Haan1977)
```

3 Model Diagnostics

The `multReg` function is designed to assist the user by performing many of the model diagnostic tests and plots suggested by Helsel and Hirsch (2002). This section will discuss the use of the `multReg` function to perform the selected model diagnostics and set up diagnostic plots for the model that Haan (1977) used. The **Support Functions** illustrates the individual functions the `smwrStats` package that aid linear regression analysis.

The code below specifies the model, created the regression model and prints the diagnostic tests.

```
> # Create the regression model
> Haan.lm <- lm(ROFF ~ PCIP + PERIM + Rr, data=Haan1977)
> # Create the diagnostic object and print it.
> Haan.reg <- multReg(Haan.lm)
> print(Haan.reg)
```

Call:

```
lm(formula = ROFF ~ PCIP + PERIM + Rr, data = Haan1977)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.0050	-0.4747	0.0203	0.5085	0.8361

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-9.64420	4.44080	-2.17	0.05795
PCIP	0.42957	0.09302	4.62	0.00126
PERIM	0.61658	0.07485	8.24	1.8e-05
Rr	0.01042	0.00201	5.19	0.00057

Residual standard error: 0.688 on 9 degrees of freedom

Multiple R-squared: 0.905, Adjusted R-squared: 0.874

F-statistic: 28.6 on 3 and 9 DF, p-value: 6.18e-05

press: 9.97

AIC: 32.4

BIC: 35.2

Anova Table (Type II tests)

Response: ROFF

	Sum Sq	Df	F value	Pr(>F)
PCIP	10.1	1	21.3	0.00126
PERIM	32.1	1	67.9	1.8e-05

```
Rr          12.8  1    26.9 0.00057
Residuals    4.3  9
```

Variance inflation factors

```
PCIP  1.30
PERIM 1.46
Rr     1.46
```

Test criteria

```
leverage  cooksD    dfits
      0.923    0.939    1.446
```

Observations exceeding at least one test criterion

```
ROFF yhat residb stnd.res stud.res leverage cooksD dfits
8  18.9 19.6 -0.683    -1.4    -1.50    0.500  0.492  -1.50*
13 13.2 14.2 -1.005    -1.8    -2.12    0.342  0.421  -1.53*
```

The printed results are comprised of several sections. The first section is the regression summary, consisting of the call, residual statistics, the coefficient table (without the significance stars), and statistics of the overall fit; the next section is the analysis of variance (ANOVA) table, which is most useful for assessing the overall significance of complex terms such as first- and second-order polynomials (**quadratic**) or sine and cosine transforms (**fourier**); the third section is a listing of the variance inflation factors (VIFs); and the last section shows the selected test criteria and the observations that exceed at least one of those criteria.

The following sections highlight selected diagnostic plots. When using the `plot` function in an interactive session, it is not necessary to specify which plot to create nor to set up a graphics device. The only call that would be necessary would be `plot(Haan.reg)`. Note that plot number 4 cannot be shown because it describes serial correlation and these data are not collected at specific points in time.

4 Response vs. Fitted Plot

The first diagnostic plot is response vs. fitted. The second is residuals vs fitted and is not shown. The basic difference is that the deviation shown by the smoothed line is exaggerated in the second plot! Each observation is plotted, the dashed line is the 1:1 fit and the solid line is a loess smooth (function `loess.smooth`) using the "symmetric" option for the `family` argument. The regression equation with the residual standard error.

```
> # setSweave is a specialized function that sets up the graphics page for
> # Sweave scripts. For interactive use, it should be removed and the
> # default setting for set.up can be used.
> setSweave("regplot01", 5, 5)
> plot(Haan.reg, which=1, set.up=FALSE)
> # Required call to close PDF output graphics
>
> graphics.off()
```

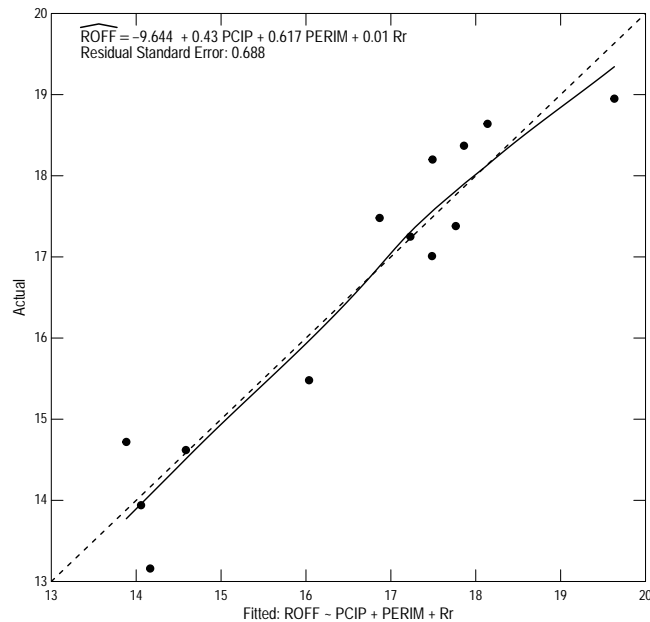


Figure 1. The residual vs. fitted diagnostic plot.

5 Scale-Location Plot

The third diagnostic plot is the scale-location plot, which plots the square root of the residuals vs. the fitted values. It is useful for diagnosing heteroscedasticity and is described by Cleveland (1993). Each observation is plotted, the dashed line is the theoretical mean, assuming a normal distribution, and the solid line is a loess smooth (function `loess.smooth` using the "symmetric" option for the `family` argument). Wooding's test for heteroscedasticity is also shown—it is a straightforward interpretation of the data, simply the results of the Spearman correlation of the data that are shown. The null hypothesis is that the residuals are homoscedastic.

```
> # setSweave is a specialized function that sets up the graphics page for
> # Sweave scripts. For interactive use, it should be removed and the
> # default setting for set.up can be used.
> setSweave("regplot02", 5, 5)
> plot(Haan.reg, which=3, set.up=FALSE)
> # Required call to close PDF output graphics
> graphics.off()
```

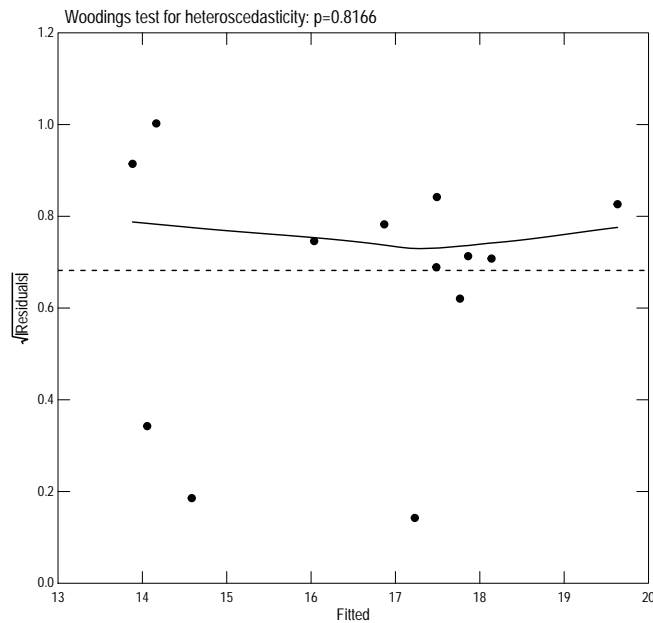


Figure 2. The scale-location diagnostic plot.

6 Probability Plot

The fifth diagnostic plot tests for the normality of the residuals. Each observation is plotted, the solid line is the theoretical fit, assuming a normal distribution. The PPCC test for normality is also shown. The null hypothesis is that the residuals are from a normal distribution.

```
> # setSweave is a specialized function that sets up the graphics page for  
> # Sweave scripts. For interactive use, it should be removed and the  
> # default setting for set.up can be used.  
> setSweave("regplot03", 5, 5)  
> plot(Haan.reg, which=5, set.up=FALSE)  
> # Required call to close PDF output graphics  
> graphics.off()
```

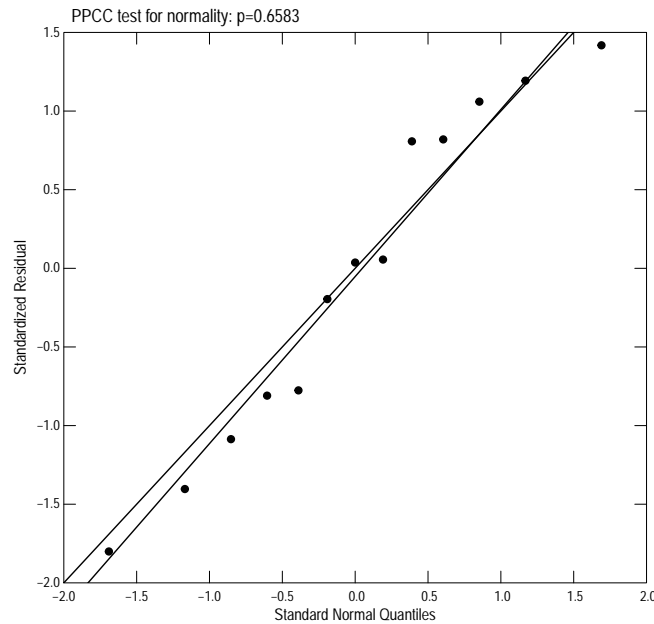


Figure 3. The normal probability diagnostic plot.

7 Influence Plot

The sixth diagnostic plot shows the approximate influence of each observation identified as exceeding one of the test criteria. Each observation is plotted, the solid line is the actual fit. Each identified observation is plotted in a different color and the fitted line with that observation removed is plotted in the same color. The seventh diagnostic plot is a plot of the studentized residual vs. the fitted value and is not shown in this vignette. Note that the label for observation number 8 is not shown in this example because it would be outside the range of the plot area.

```
> # setSweave is a specialized function that sets up the graphics page for  
> # Sweave scripts. For interactive use, it should be removed and the  
> # default setting for set.up can be used.  
> setSweave("regplot04", 5, 5)  
> plot(Haan.reg, which=6, set.up=FALSE)  
> # Required call to close PDF output graphics  
> graphics.off()
```

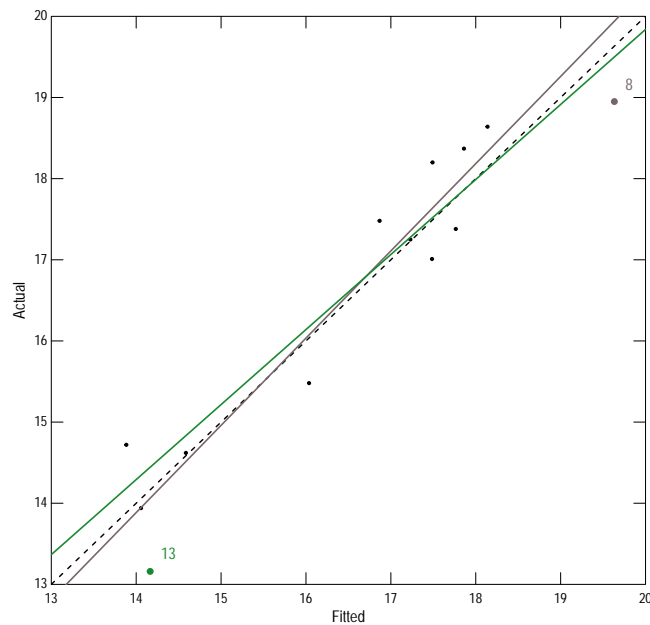


Figure 4. The influence diagnostic plot.

8 Residual Dependence Plot

The eighth diagnostic plot is actually a series of plots, one for each explanatory variable. But, a single explanatory variable can be selected instead of the series, as is shown in this example. Each observation is plotted, the dashed line is 0, the expected value of the residual for each observation and the solid line is a loess smooth (function `loess.smooth` using the "symmetric" option for the `family` argument). The results for a second order polynomial fit is also shown; it is the attained p-value of the squared explanatory variable added to the model.

```
> # setSweave is a specialized function that sets up the graphics page for
> # Sweave scripts. For interactive use, it should be removed and the
> # default setting for set.up can be used.
> setSweave("regplot05", 5, 5)
> plot(Haan.reg, which="PERIM", set.up=FALSE)
> # Required call to close PDF output graphics
> graphics.off()
```

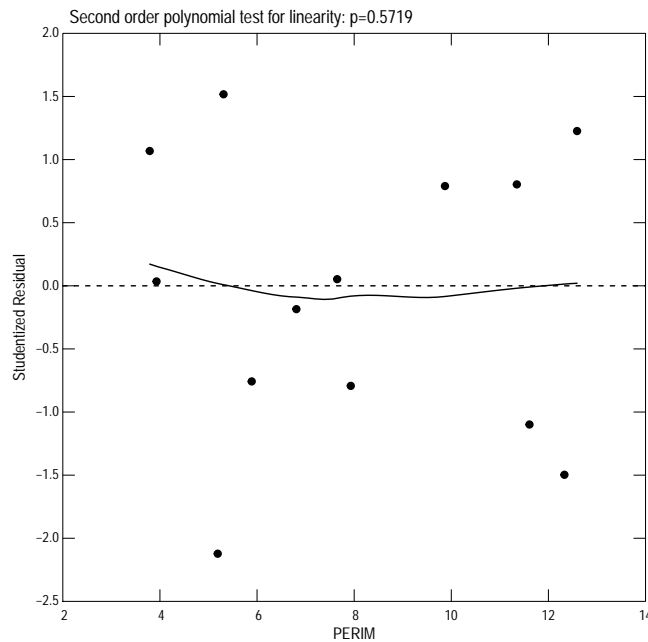


Figure 5. The residual dependence diagnostic plot.

9 Support Functions

The prediction error sum of squares (PRESS) is one of the best measures of the quality of a regression equation (Helsel and Hirsch (2002)). PRESS is a validation-type estimator of error; it sequentially drops a single observation then computes its value from the remaining observations and sums the squares of the differences. The `press` function will compute the PRESS statistic for any linear regression model created by `lm`. The following example uses the previously created regression model on the Haan data.

```
> # Compute the PRESS statistic
> press(Haan.lm)
```

```
[1] 9.968029
```

The `rmse` function is a very easy way to extract the root-mean-squared error or residual standard error from a regression model without running the `summary` function on the model.

```
> # The residual standard error is computed by the summary function.
> summary(Haan.lm)
```

Call:

```
lm(formula = ROFF ~ PCIP + PERIM + Rr, data = Haan1977)
```

Residuals:

	Min	1Q	Median	3Q	Max
	-1.00496	-0.47475	0.02029	0.50850	0.83609

Coefficients:

	Estimate	Std. Error	t value	Pr(> t)
(Intercept)	-9.644196	4.440796	-2.172	0.05795 .
PCIP	0.429570	0.093023	4.618	0.00126 **
PERIM	0.616580	0.074849	8.238	1.75e-05 ***
Rr	0.010421	0.002007	5.192	0.00057 ***

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

Residual standard error: 0.6881 on 9 degrees of freedom

Multiple R-squared: 0.9052, Adjusted R-squared: 0.8736

F-statistic: 28.64 on 3 and 9 DF, p-value: 6.182e-05

```
> # But can easily be computed using rmse:
> rmse(Haan.lm)
```

```
[1] 0.6880745
```

Helsel and Hirsch (2002) state "Concern over multi-collinearity should be strongest when the purpose is to make inferences about coefficients." The variance inflation factor is a good diagnostic for measuring multi-collinearity. The `vif` function computes the variance inflation factor for each variable in the model.

```
> # The variance inflation factors:  
> vif(Haan.lm)
```

	PCIP	PERIM	Rr
	1.297794	1.455248	1.460949

10 Subset Selection Comments

As noted in the opening paragraph, Harrell (2001) describes concerns about subset selection of explanatory variables and offers advice on how to approach building regression models. This example demonstrates one of his concerns: the individual significance of an explanatory variable is not appropriate in the context of many possible explanatory variables. This example uses the `cor.all` function to demonstrate that issue, but has applications beyond linear regression.

```
> # Create the correlation structure, and print it:
> Haan.cor <- cor.all(Haan1977)
> print(Haan.cor, digits=3)
```

Pearson's product-moment correlation

data: Haan1977

	ROFF	PCIP	AREA	SLOPE	LEN	PERIM	DI	Rs	FREQ
PCIP-cor	0.3865								
PCIP-r!=0	0.192								
PCIP-N	13								
AREA-cor	0.4709	-0.2470							
AREA-r!=0	0.104	0.416							
AREA-N	13	13							
SLOPE-cor	0.4092	0.0807	-0.1713						
SLOPE-r!=0	0.165	0.793	0.576						
SLOPE-N	13	13	13						
LEN-cor	0.4210	-0.3371	0.8958	-0.2059					
LEN-r!=0	0.152	0.260	<0.001	0.500					
LEN-N	13	13	13	13					
PERIM-cor	0.4647	-0.4149	0.9553	-0.0996	0.9183				
PERIM-r!=0	0.110	0.159	<0.001	0.746	<0.001				
PERIM-N	13	13	13	13	13				
DI-cor	0.3343	-0.1548	0.9083	-0.1614	0.6658	0.8105			
DI-r!=0	0.264	0.614	<0.001	0.598	0.013	0.001			
DI-N	13	13	13	13	13	13			
Rs-cor	-0.1532	0.4492	-0.2534	0.0460	-0.5750	-0.4089	0.1498		
Rs-r!=0	0.617	0.124	0.404	0.881	0.040	0.165	0.625		

Rs-N	13	13	13	13	13	13	13		
FREQ-cor	-0.3982	0.0406	-0.4804	-0.3011	-0.5340	-0.4774	-0.3241	0.2940	
FREQ-r!=0	0.178	0.895	0.097	0.317	0.060	0.099	0.280	0.330	
FREQ-N	13	13	13	13	13	13	13	13	
Rr-cor	0.3455	0.4188	-0.5176	0.7973	-0.5359	-0.5144	-0.5017	0.1756	-0.0762
Rr-r!=0	0.248	0.154	0.070	0.001	0.059	0.072	0.081	0.566	0.805
Rr-N	13	13	13	13	13	13	13	13	13

```

> # Now summarize the significance of the realtions between ROFF and the other variables
> summary(Haan.cor, variable="ROFF")

```

	Var1	Var2	Cor	Pval.holm	Counts
1	ROFF	PCIP	0.3865054	1.0000000	13
2	ROFF	AREA	0.4708535	0.9393878	13
3	ROFF	SLOPE	0.4091853	1.0000000	13
4	ROFF	LEN	0.4210102	1.0000000	13
5	ROFF	PERIM	0.4646590	0.9393878	13
6	ROFF	DI	0.3343175	1.0000000	13
7	ROFF	Rs	-0.1532346	1.0000000	13
8	ROFF	FREQ	-0.3981606	1.0000000	13
9	ROFF	Rr	0.3455404	1.0000000	13

The summary output shows the p-values adjusted for the significance level in the context of eight other variables. The p-values in the summary output have changed considerably from the p-values in the first column in the printed output.

11 Bias Correction

Helsel and Hirsch (2002) state that the mass of a constituent estimated using a log-transformed regression equation is not correctly estimated simply by back-transforming the predicted values. They describe two methods for bias correction, the Ferguson or maximum likelihood estimation method and Duan's smearing estimate. These are implemented in **smwrStats** in the **predictFerguson** and the **predictDuan** functions. Also included is the **predictMVUE** function which applies the minimum variance unbiased estimator described in Bradu and Mundlak (1970). This example build a log-transformed regression model and compares the sum of the estimates from all of the methods.

```
> # Create the regression model and print it:
> TDS.lm <- lm(log(TDS) ~ log(Q) + fourier(TIME), data=CuyahogaTDS)
> print(TDS.lm)
```

Call:

```
lm(formula = log(TDS) ~ log(Q) + fourier(TIME), data = CuyahogaTDS)
```

Coefficients:

(Intercept)		log(Q)	fourier(TIME)sin(k=1)
	7.99835	-0.29879	0.03962
fourier(TIME)cos(k=1)			
	0.06567		

```
> # The sum of the TDS data in the calibration dataset:
> sum(CuyahogaTDS$TDS)
```

```
[1] 35720
```

```
> # The sum of the simple back-transformed predictions
> sum(exp(predict(TDS.lm)))
```

```
[1] 35308.69
```

```
> # No the sume from each of the bais-corrected methods
> sum(predictFerguson(TDS.lm))
```

```
[1] 35735.3
```

```
> sum(predictDuan(TDS.lm))
```

[1] 35723.01

```
> sum(predictMVUE(TDS.lm))
```

[1] 35714.53

The back-transformation bias correction functions are necessary when the goal is to preserve mass or the mean estimate is needed. The estimates from the simple back-transformed values are valid for point estimates. Note that `predictFerguson` and `predictMVUE` can only be used for log-transformed regression model, but `predictDuan` can be used for any monotonic transformation.

References

- [1] Bradu, D. and Mundlak, Y., 1970, Estimation in the lognormal linear models: Journal of the American Statistical Association, v. 65, no. 329, p. 198-211.
- [2] Cleveland, W.S., 1993, Visualizing data: Summit, New Jersey, Hobart Press, 360 p.
- [3] Haan, C. T., 1977. Statistical methods in hydrology: Iowa State University Press, Ames, Iowa, 378 p.
- [4] Harrell, F.E., Jr., 2001, Regression modeling strategies with applications to linear models, logistic regression and survival analysis: New York, N.Y., Springer, 568 p.
- [5] Helsel, D.R. and Hirsch, R.M., 2002, Statistical methods in water resources: U.S. Geological Survey Techniques of Water-Resources Investigations, book 4, chap. A3, 522 p.