

# GSqwsr: an R package to develop surrogate regressions for real-time water quality prediction

April 30, 2014

Steven Corsi, Laura DeCicco, Jessica Thompson, Austin Baldwin and Laura Hubbard  
USGS Wisconsin Water Science Center

# Introduction

- Many water-quality parameters are difficult or impossible to monitor in real time
- It is often desirable to estimate water-quality conditions in near real-time, and for unmeasured storm events
- Some of these constituents can be estimated based on easily-measured surrogates
- The goal of this work is the development of a standardized and widely available tool for development of surrogate regressions and evaluation of their goodness of fit

# Objectives

- A common situation is a site with measured real-time and discrete data
- Data from the real-time sensors is then used to build a surrogate regression model for prediction of the discrete constituents
- The process of merging data and developing these models can be time-consuming
- A real-world test case will be used for illustration of the surrogate regression utility highlighted in this talk

# Steps in the development of a surrogate regression model for a site

1. Retrieval of real-time and discrete data
2. Preparation and merging of datasets
3. Analysis of raw data to determine probable predictor (explanatory) variables
4. Development of regression models between predictor variables and the chosen response variable
5. Refinement of regression model fit
6. Repeat until satisfied with regression model

# Methods used by GSqwsr package

- Water-quality and real-time data is retrieved from U.S. Geological Survey web services
- Real-time and discrete data is merged and prepared for use in the regression model
- Censored data is represented as a range, visualized as a vertical line from the upper limit to zero on plots
- A general equation including all possible predictor variables is generated as the model upper bound
- Stepwise regression is used recursively to develop the surrogate regression model
- The R software environment is used to automate as many steps as possible in this process

# The R software environment

- The R environment for statistical computing is a free open-source software
- <http://www.r-project.org/>
- The environment I use for working in R is RStudio (<http://rstudio.org/>)
- R is a widely used environment, and there are many packages available, via CRAN, github, USGS, etc
- There is a very active user community and many available statistical packages

# The R software environment

## 1 Introduction to GSqwsr package

The GSqwsr (USGS water quality surrogate regressions) package was designed to simplify the process of gathering water quality sample data and unit surrogate data, running a stepwise regression using the USGSwsQW censReg regression function, and analyzing those results. This vignette will first show a general overview workflow (2), then a more detailed description of the workflow with working examples (3).

## 2 General Workflow

```
library("GSqwsr")

#Sample data included with package:
DTComplete <- StLouisDT
UV <- StLouisUV
QWcodes <- StLouisQWcodes
siteINFO <- StLouisInfo

investigateResponse <- "Ammonia.N"
transformResponse <- "lognormal"

DT <- DTComplete[c(investigateResponse,
                   getPredictVariables(names(UV)),
                   "decYear", "sinDY", "cosDY", "datetime")]
DT <- na.omit(DT)

predictVariables <- names(DT)[-which(names(DT)
```

# Retrieve real-time and discrete data

- For the USGS site of interest, retrieve the periods of record for available water-quality and real-time parameters
  - **whatUVNew**
  - **whatQW**

```
> UVcodes
```

	parameter cd	statcd	startdate	enddate	count	service
307	00010		2007-10-01	2014-04-24	2397	UV
308	00060		2007-10-01	2014-04-24	2397	UV
309	00065		2013-12-25	2014-04-24	120	UV
310	00095		2011-03-16	2014-04-24	1135	UV
311	00300		2007-10-01	2014-04-24	2397	UV
312	00400		2011-03-16	2014-04-24	1135	UV
313	63680		2011-03-17	2014-04-24	1134	UV

```
>
```

# Retrieve real-time and discrete data

- Choose desired water-quality and real-time parameters from those available, set period of analysis and retrieve data
  - **retrieveNWISqwData**
  - **makeQWObjects**
  - **getMultipleUV**

```
> UV <- getMultipleUV(site, startDate, endDate, UVP)
Getting data: 00010
Getting data: 00060
Getting data: 00095
Getting data: 00300
Getting data: 00400
Getting data: 63680

> dim(UV)
[1] 108472    16
>
```

# Prepare data for regression modeling

- Merge UV and QW datasets into one combined data frame
- This function alone saves significant time, allowing users to easily merge QW data with the nearest real-time value, as well as select an acceptable time gap (default of 2 hours)

## — mergeDatasets

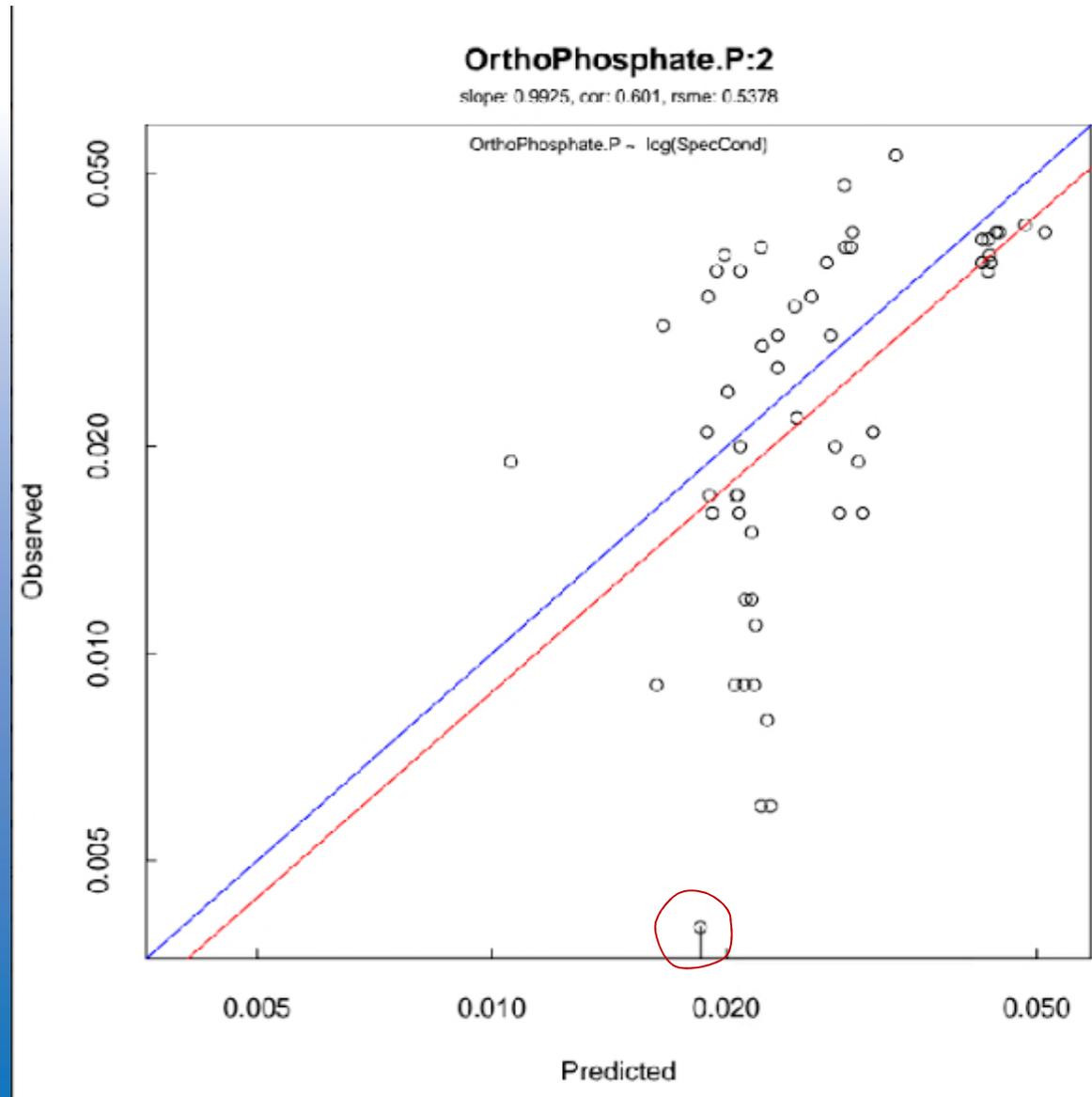
```
datetime wtemp Flow speecond DO pH Turb site Phosphorus_WW.P decYear
2011-04-05 08:05:00 7.8 1570 477 12.7 8.2 NA USGS-04137500 0.004 2011.258
2011-04-11 03:56:00 3.3 2250 314 12.5 8.2 1.5 USGS-04137500 0.005 2011.274
2011-04-11 15:56:00 3.3 2250 309 12.7 8.3 1.4 USGS-04137500 0.004 2011.275
2011-04-11 23:56:00 3.3 2250 307 12.7 8.3 1.4 USGS-04137500 0.005 2011.276
2011-04-12 03:56:00 3.3 2550 306 12.6 8.3 1.8 USGS-04137500 0.005 2011.276
2011-04-17 05:56:00 4.4 2540 306 12.6 8.2 1.4 USGS-04137500 0.004 2011.277
```

# Prepare data for regression modeling

- Choose a normal or lognormal distribution for the response variable (lognormal distributions are frequently used for environmental data, but may not be used if there are zero or negative values)
- For model development, there must not be any NA values
- Tools are provided to maximize rows or columns of data as desired when removing NAs
- Guidance is provided on saving data sets at any point, this being a typical place

# Censored data

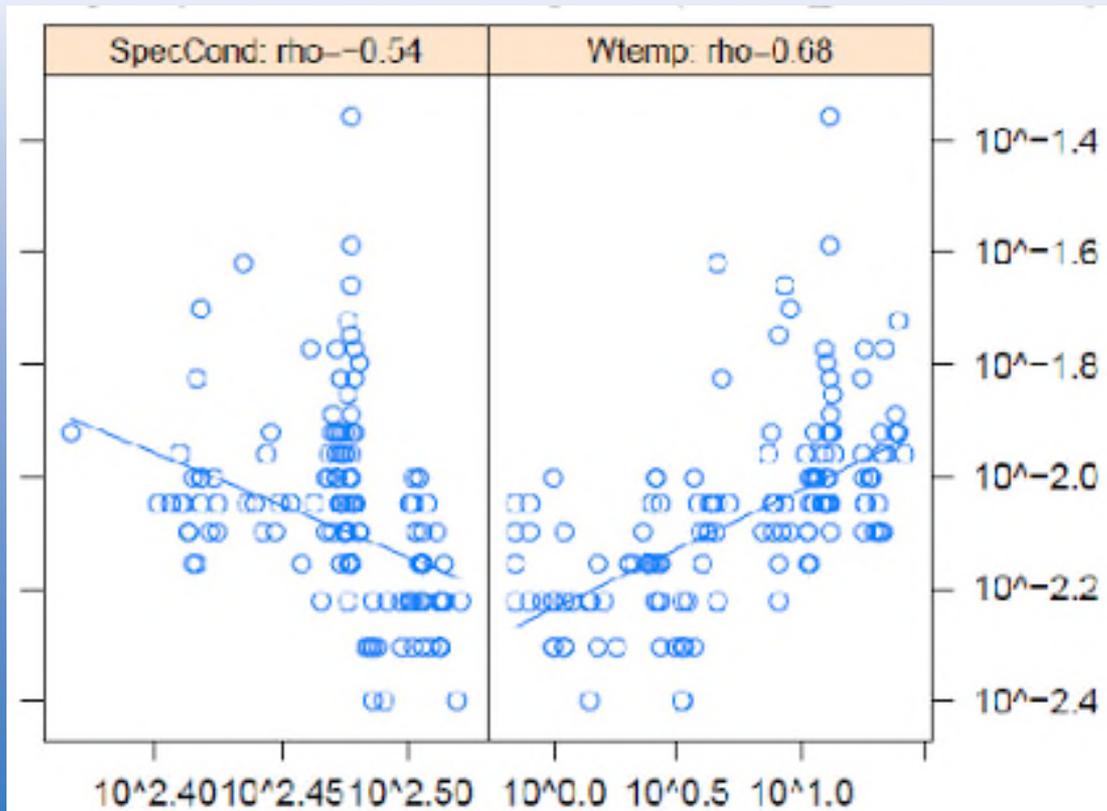
- Censored data is represented by a range
- For example, a value of  $<0.002$  is represented as  $0 - 0.002$



# Create preliminary graphs

- **plotQQTransforms**
- **predictVariableScatterPlots**

- Saved as pdf files
- Useful for initial exploration of possible predictor variables
- May be helpful in analysis of potential outliers
- Both linear and log distributions of predictor variables are graphed



# Run stepwise model

- Create a general regression formula containing all possible predictor variables as the upper bound
- Predictor variables decYear, sinDY and cosDY are added to represent seasonality and trend over time
- Normal and log transforms (for predictor variables with no zero or negative values) are added to the equation for each predictor variable

– **createFullFormula**

**Wtemp + Flow + SpecCond + DO + pH + sinDY + cosDY  
+ log(Wtemp) + log(Flow) + { log(SpecCond) + log(DO)**

# Run stepwise model

- Stepwise regression is used to choose model variables
- Choose from several options for parameter used for selection of variables
  - AIC (akaike information criterion) and BIC (Bayesian information criterion) are two commonly used options
- Output from the `prelimModelDev` function includes:
  - Variable selection steps
  - Information from the final model
  - Raw data used

**prelimModelDev**

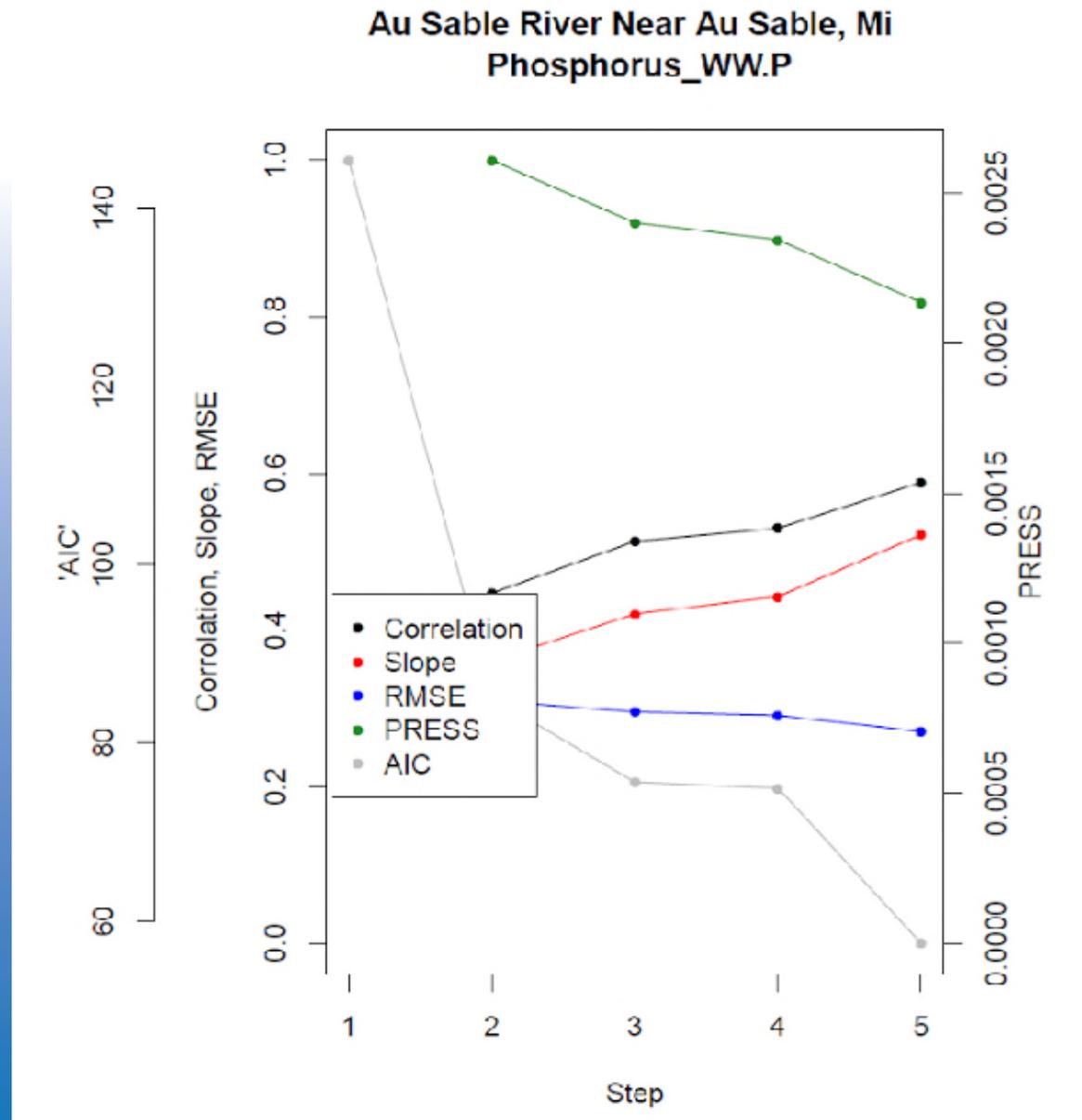
# Create plots and tables to visualize steps and resulting model

- **plotSteps**
- **analyzeSteps**

```
> plotSteps(steps, DT, transformResponse)
Phosphorus_WW.P ~ LU
Phosphorus_WW.P ~ DO + sINDY
Phosphorus_WW.P ~ DO + sINDY + cOSDY
Phosphorus_WW.P ~ DO + sINDY + cUSDY + FLOW
```

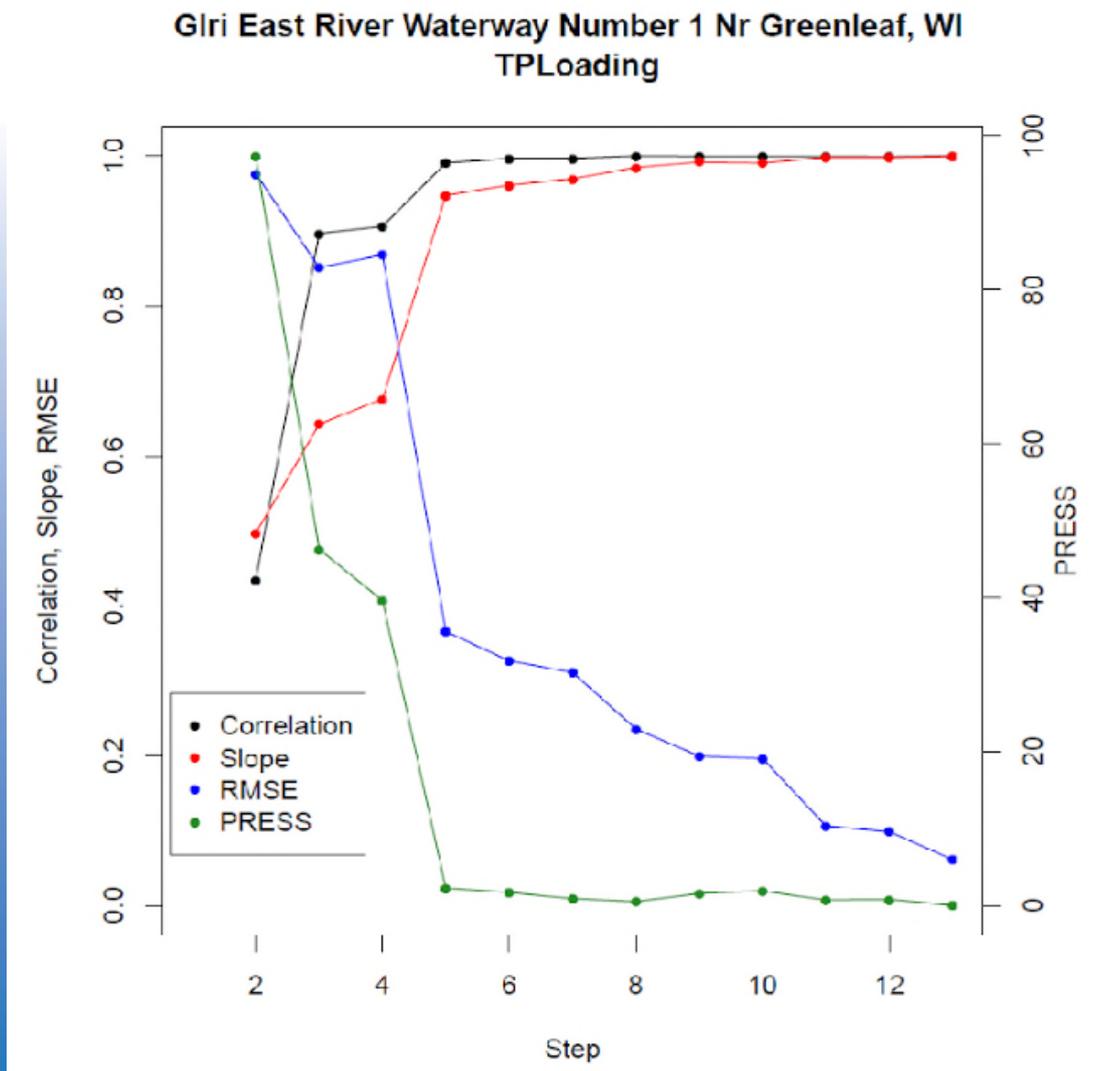
# analyzeSteps

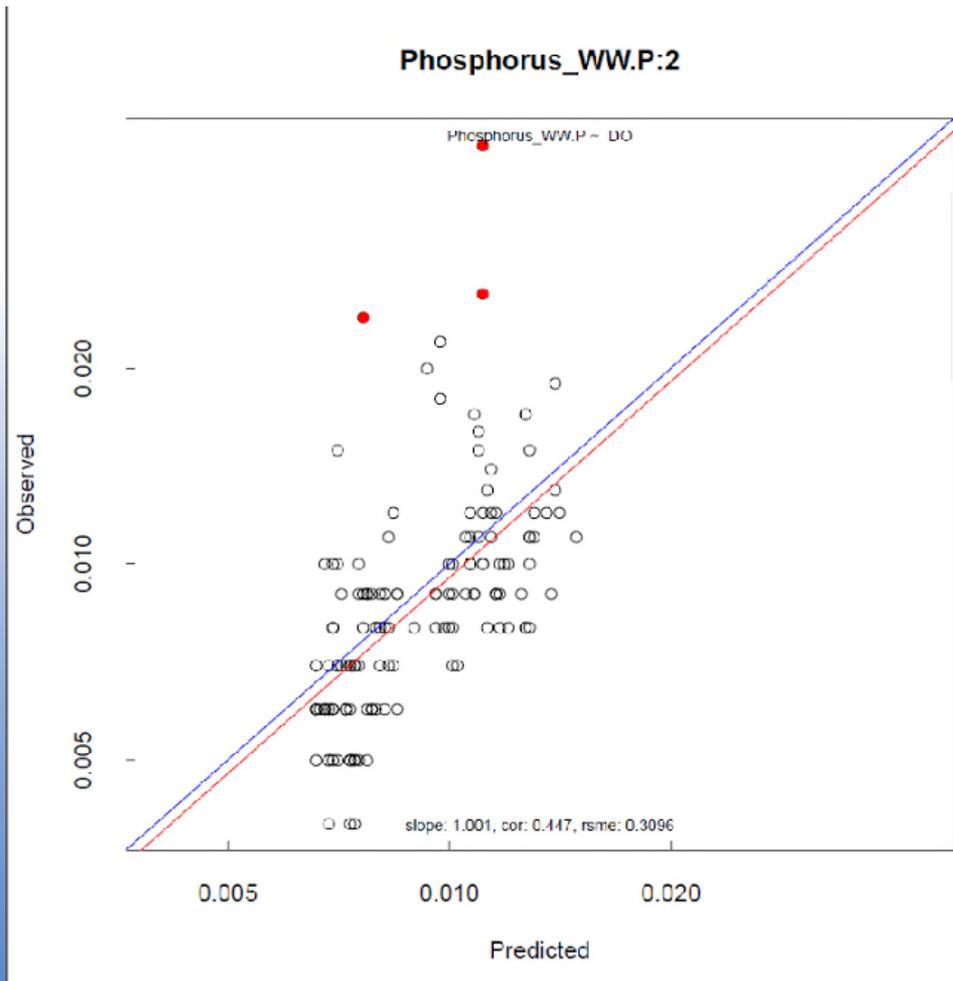
- Plot showing the change in model diagnostics at each step.
- This plot can be used to consider the added value of multiple predictor variables for the regression function.



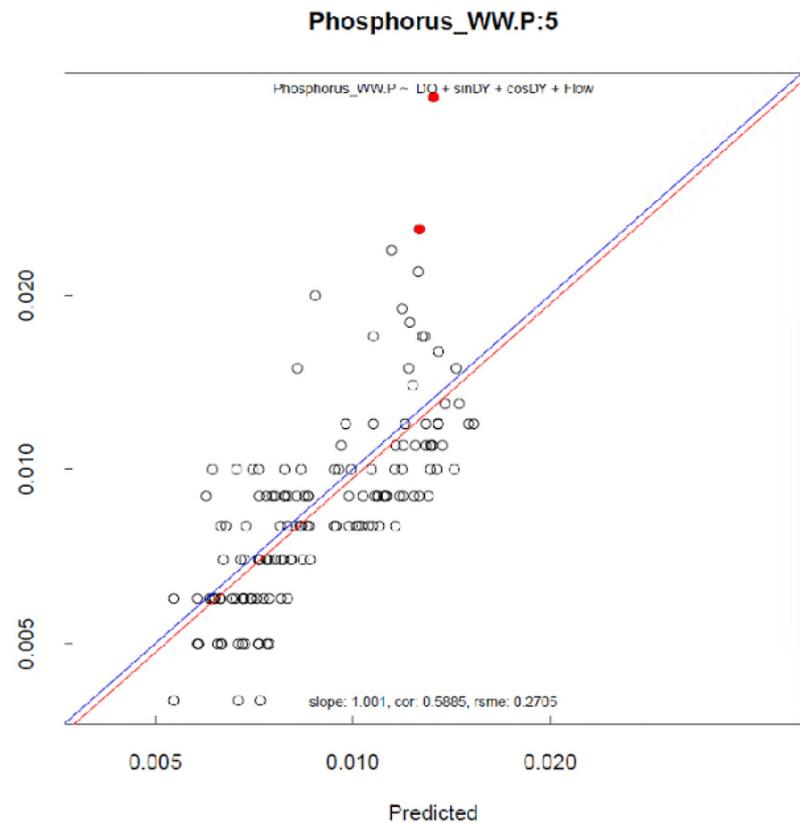
# analyzeSteps

This plot shows an extreme example where the stepwise regression kept refining the model for many steps that seem to provide limited improvement. The user may choose to use the model from step 5, for example.





Detail plots from Step 2 (left) and Step 5 (bottom) of the stepwise process. Possible outliers are shown in red. These are calculated based on external studentized residuals  $>3$  or  $<-3$



plotSteps

# Model Adjustments

- Examine possible outliers and remove them as desired

## findOutliers

```
>
> outliers <- findOutliers(modelReturn,DT,transformResponse)
> DT[outliers,]
  Phosphorus_WW.P Wtemp Flow SpecCond DO pH decYear   sinDY   cosDY      datetime
115          0.026   13 1380       300 9.6 8.3 2011.785 -0.9761927 0.2169052 2012-10-14 11:40:00
116          0.044   13 1650       300 9.6 8.3 2011.785 -0.9761150 0.2172543 2012-10-14 12:10:00
> head(DT)
  Phosphorus_WW.P Wtemp Flow SpecCond DO pH decYear   sinDY   cosDY      datetime
1          0.005   1.8 1520       322 12.7 8.2 2011.258 0.9988079 -0.04881395 2011-04-05 08:05:00
2          0.005   3.1 2250       314 12.5 8.2 2011.274 0.9888989 -0.14858968 2011-04-11 03:56:00
3          0.004   3.3 2250       309 12.7 8.3 2011.275 0.9875834 -0.13709563 2011-04-11 15:56:00
4          0.005   3.3 2250       307 12.7 8.3 2011.276 0.9866657 -0.16275983 2011-04-11 23:56:00
5          0.005   3.3 2560       306 12.6 8.3 2011.276 0.9861947 -0.16558994 2011-04-12 03:56:00
6          0.004   3.3 2590       306 12.6 8.2 2011.277 0.9859561 -0.16700448 2011-04-12 05:56:00
>
<
```

# Model Adjustments

- Adjust the step used, remove or add parameters or add interactions
  - ‘Scalar’ column can be adjusted to add or remove predictor variables
  - Other cells are used to add an interaction between two variables

**generateParamChoices**

**createFormulaFromDF**

variableNames	Scalar	Wtemp	Flow	SpecCond	DO	pH	sinDY	cosDY	log(Wtemp)	log(Flow)	log(SpecC)	log(DO)
Wtemp	0	0	0	0	0	0	0	0	0	0	0	0
Flow	1	0	0	0	0	0	0	0	0	0	0	0
SpecCond	0	0	0	0	0	0	0	0	0	0	0	0
DO	1	0	0	0	0	0	0	0	0	0	0	0
pH	0	0	0	0	0	0	0	0	0	0	0	0
sinDY	1	0	0	0	0	0	0	0	0	0	0	0
cosDY	0	0	0	0	0	0	0	0	0	0	0	0
log(Wtemp)	0	0	0	0	0	0	0	0	0	0	0	0
log(Flow)	0	0	0	0	0	0	0	0	0	0	0	0
log(SpecCond)	0	0	0	0	0	0	0	0	0	0	0	0
log(DO)	0	0	0	0	0	0	0	0	0	0	0	0

# Model Diagnostics

## resultPlots

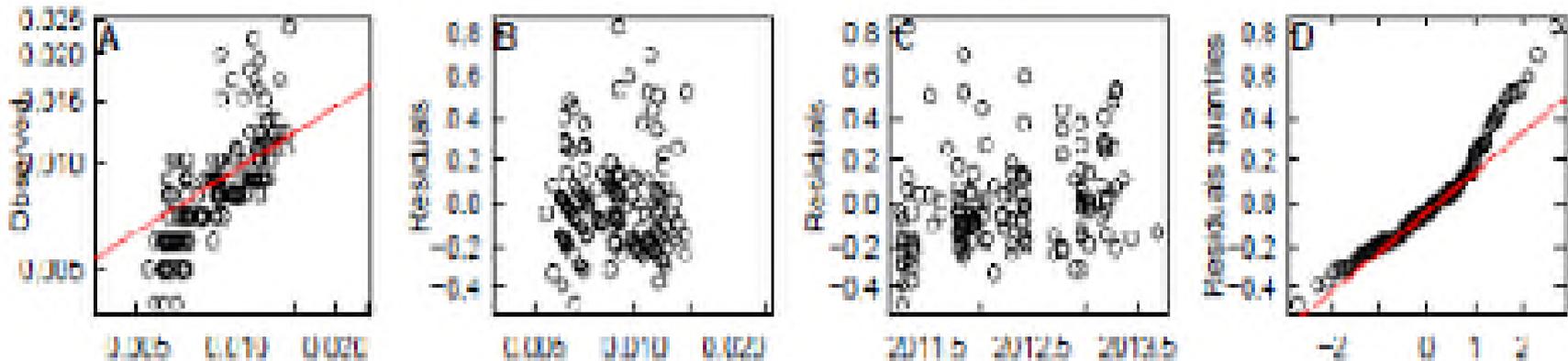
Observed  
vs  
Predicted

Residuals  
vs  
Predicted

Residuals  
vs  
Time

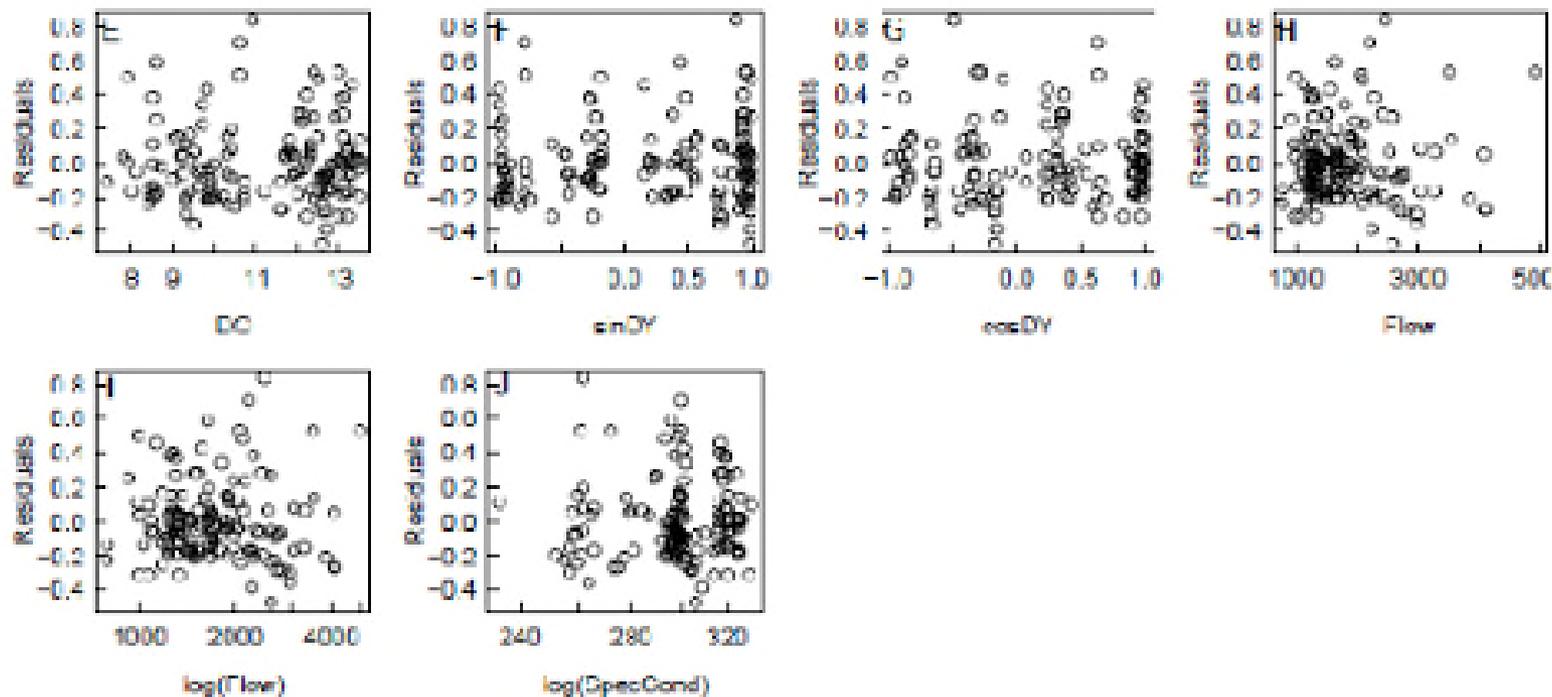
Residuals Quantiles  
vs  
Theoretical Quantiles

Phosphorus\_WWP at Au Sable River Near Au Sable, MI (04137500)



# Model Diagnostics

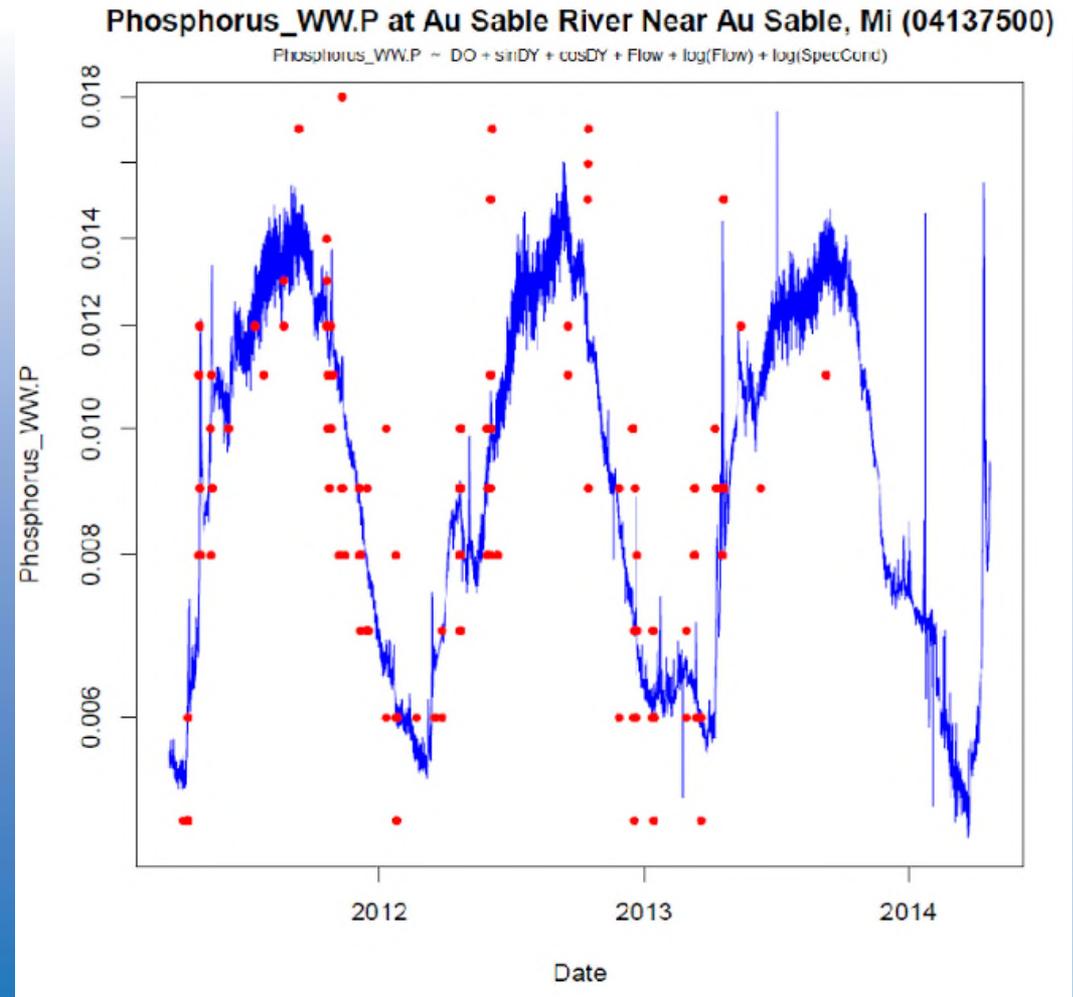
resultResidPlots



# Model Predictions

predictionPlot

- Create a plot of predicted values based on all available unit value data in blue with red dots representing actual measured data
- This plot illustrates strong seasonality



# Model Summary Information

- A text file with model summary information is also available
  - RMSE, RSQ, observations, degrees of freedom
  - Coefficient, standard error, p value and standard coefficient for each predictor variable chosen
  - Correlation matrix of coefficients

```
Phosphorus_WW.P at Au Sable River Near Au Sable, MI ( 04137500 )
Number of observations: 144
Distribution: lognormal
Method: ANLE
Degrees of freedom: 25
RMSE: 0.232661
RSQ: 58.0193
Number of censored values: 0
Phosphorus_WW.P ~ (Intercept) + DO + sindY + cosDY + Flow + log(Flow) + log(SpecCond)
```

	Term	Coefficient	StdError	pValue	StCoef
1	(Intercept)	5.925	2.778	0.030	2.133
2	DO	-0.122	0.043	0.005	-2.795
3	sindY	0.151	0.072	0.032	2.111
4	cosDY	0.099	0.113	0.378	0.861
5	Flow	0.000	0.000	0.000	3.655
6	log(Flow)	-0.687	0.253	0.006	-2.690
7	log(SpecCond)	-0.900	0.409	0.025	-2.202
8	logSigma	0.054	0.006	0.000	8.473

# Summary

- This work was funded by the GLRI project
- With 30 sites to manage, saving time was necessary
- Using this software utility, once setup was performed for the 30 sites, all of the initial regressions were created overnight
- By automating this initial work, scientists can focus time on refinement of individual regression models
- The utility is currently geared toward USGS NWIS data, but could be expanded to handle other sources
- Will be an upcoming publication on the GLRI sites using this technique

# Resources used in this talk

- <http://www.r-project.org/>
- <http://rstudio.org/>
- <http://github.com/USGS-R/>
- `install.packages("GSqwsr", repos="http://usgs-r.github.com/", type="source")`
- <https://github.com/USGS-R/surrogateRegression/raw/master/inst/doc/QWSR.pdf>
- <https://github.com/USGS-R/surrogateRegression/raw/master/inst/>