R Minicourse Workshop
White River Case Study

Presented to the
Washington State Deptment of Ecology
September 2–3, 2014

Dr. Robin Matthews, Institute for Watershed Studies
Dr. Geoffrey Matthews, Computer Science Department
Western Washington University

White River Case Study

Preliminaries

- The R scripts for this lecture are called `whiteriver01.r` and `whiteriver02.r`.
- The first script does one reach, distance, and parameter, interactively.
- The second does all combinations of reach, distance, and several parameters and saves the results into files. `whiterivertext.txt` and `whiteriverplot.pdf`
- Greg Pelletier provided a spreadsheet `Q2KW_White_River_observed_and_predicted_data.xlsx`
- I saved the two pages of the spreadsheet into two csv files:
  - `Q2KW_White_River_observed_data.csv`
  - `Q2KW_White_River_predicted_data.csv`

White River Case Study

Loading the Data

```
predicted <- read.table("Q2KW_White_River_predicted_data.csv",
                        header=T,
                        sep=",",
                        stringsAsFactors=F)
observed <- read.table("Q2KW_White_River_observed_data.csv",
                       header=T,
                       sep=",",
                       stringsAsFactors=F)
# We want to load "5/5/12" as a string, rather than a factor,
# to make subsequent conversion to dates a bit easier.
# Now replace date strings with actual dates:
predicted$Date.time <- strptime(predicted$Date.time,
                                format="%m/%d/%y %H:%M")
observed$Date.time <- strptime(observed$Date.time,
                               format="%m/%d/%y %H:%M")
```

White River Case Study

Large size of the dataset makes it hard to view interactively.

```
dim(observed)
[1] 47265    54
dim(predicted)
[1] 48450    79
names(observed)
[1] "Reach.number"
[2] "Distance..Km."
[3] "Date.time"
[4] "Temperature..degC."
[5] "Conductivity..uS.cm.25C." ...
names(predicted)
[1] "Reach"
[2] "Distance..Km."
[3] "Date.time"
[4] "Temperature..degC."
[5] "Conductivity..uS.cm.25C." ...
```

White River Case Study

Observed *vs.* predicted reaches

- We want to use reach and distance as a site.
- Examining reaches in both observed and predicted shows that the observed reaches are a subset of the predicted reaches.
- So we can use the same reach, *e.g.* 23, for both files.

```
observed.reaches <- unique(sort(observed$Reach.number))
observed.reaches
[1]  3  8 13 20 21 22 23 25 27 28 31 33
predicted.reaches <- unique(sort(predicted$Reach))
predicted.reaches
[1]  0  1  2  3  4  5  6  7  8  9 10 11 12 13 14 15 16 17 18 19 20 21 22 23 24 25 26 27 28
my.reach <- 23
```

White River Case Study

Observed *vs.* predicted distances

- Distances are not the same in the two files.
- We will have to find a closest distance in the predicted file to match any distance in the observed file.

```
observed.distances
[1]  0.03574664  2.15476664  5.81305664  7.90000000  9.37570664 12.01200000 13.31981664 16
[9] 24.72141000 31.85200000 39.69900000
predicted.distances
[1]  0.04  1.12  2.15  3.35  4.51  5.81  7.09  8.28  9.38 10.72 11.83 13.32 15.25 16.20 17
[18] 20.75 22.08 23.44 24.72 26.12 27.46 28.73 30.23 31.85 33.58 34.85 36.57 38.21 39.70 4
my.observed.distance <- observed.distances[6]
min.difference.index <- which.min(abs(predicted.distances - my.observed.distance))
my.predicted.distance <- predicted.distances[min.difference.index]
```

White River Case Study

Parameter names and columns

```
observed.parameter.name <- "pH"
observed.parameter.column <- which(names(observed) == observed.parameter.name)
predicted.parameter.name <- "pH"
predicted.parameter.column <- which(names(predicted) == predicted.parameter.name)
```

- Now we can use any of the following:
  - `observed[[observed.parameter.column]]`
  - `observed[,observed.parameter.column]`
  - `observed$pH`
  - `observed[[observed.parameter.name]]`
- And similarly for `predicted`

White River Case Study

Creating an ID for selected parameter, reach, distance

- We have now selected a parameter, a reach, and a distance.
- It will be helpful to print these out here and there, so we make a string with all this information in it.

```
my.id <- paste( observed.parameter.name,
                " reach:", my.reach,
                " distance:", my.observed.distance)
cat(paste("Sample ID: ", my.id, "\n"))
```

White River Case Study

Selecting the rows

- We want only the data from the selected reach and distance, which also has a valid date and a valid datum.
- So we select just those rows.
- If the number of rows is too small, we can try a different reach and distance.
- Note that reach has different names in the two datasets.

```
observed.rows <- (observed$Reach.number == my.reach) &
  (observed$Distance == my.observed.distance) &
  (!is.na(observed$Date.time)) &
  (!is.na(observed[,observed.parameter.column]))

predicted.rows <- (predicted$Reach == my.reach) &
  (predicted$Distance == my.predicted.distance) &
  (!is.na(predicted$Date.time))&
  (!is.na(predicted[,predicted.parameter.column]))
```

White River Case Study

Subsetting the rows and columns of the data

- We can now select just the rows and columns we want.
- This will be the data we analyze, it has been "cleaned" so that we don't have to worry about missing numbers, *etc.*

```
date.name <- "Date.time"
observed.date.column <- which(names(observed) == date.name)
predicted.date.column <- which(names(predicted) == date.name)

observed.columns <- c(observed.date.column, observed.parameter.column)
predicted.columns <- c(predicted.date.column, predicted.parameter.column)

observed.subset <- observed[observed.rows, observed.columns]
predicted.subset <- predicted[predicted.rows, predicted.columns]
```

White River Case Study

Plotting observed and predicted data

- We want plots of observed and predicted to have the same x and y range, so we find the biggest range needed for each.

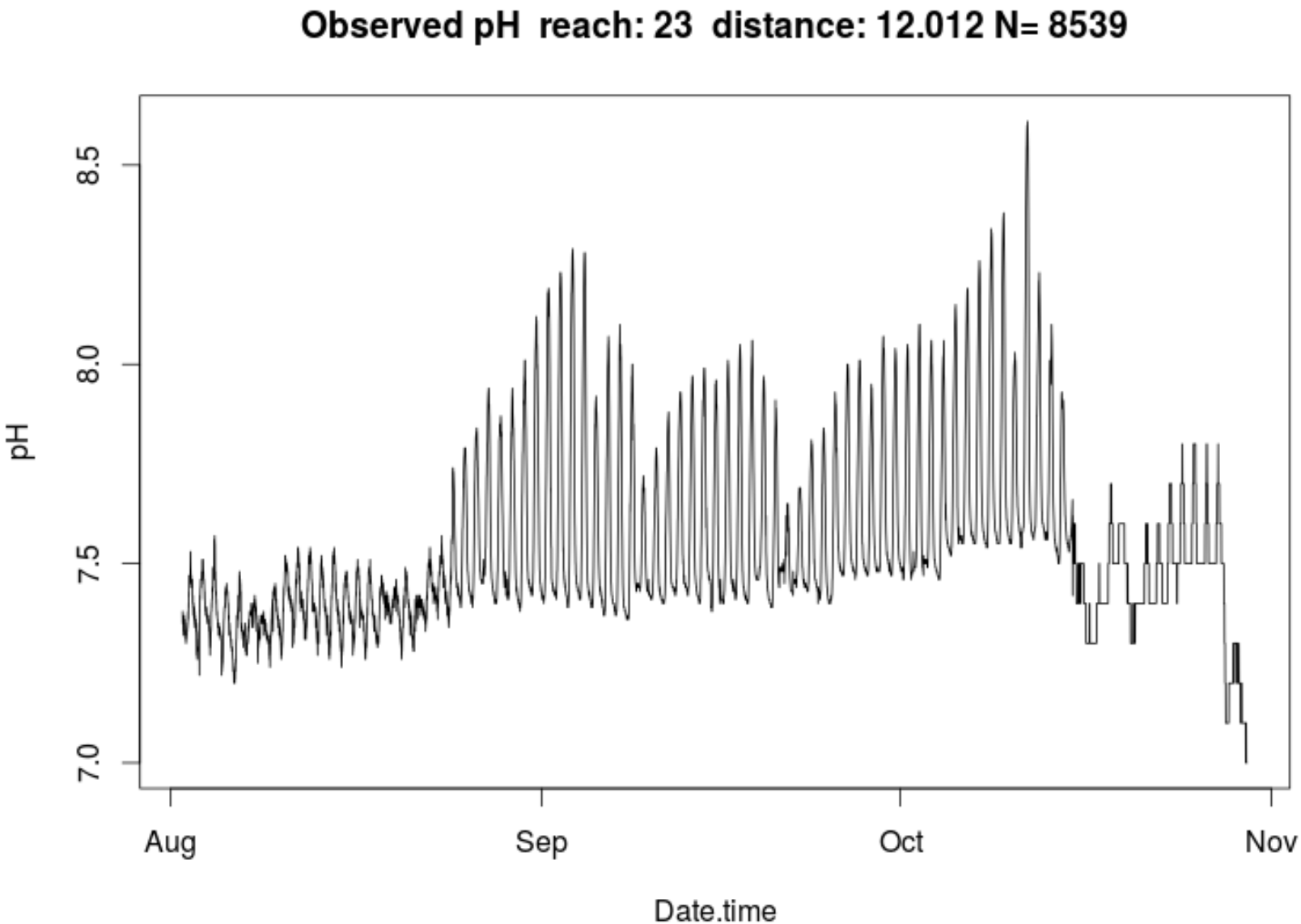
```
x.range <- as.numeric(range(c(observed.subset$Date.time,
                              predicted.subset$Date.time)))
y.range <- range(c(observed.subset[[observed.parameter.name]],
                   predicted.subset[[predicted.parameter.name]]))
```

- Now we can plot the data.

```
plot(observed.subset,
     type="l",
     xlim=x.range,
     ylim=y.range,
     main=paste("Observed", my.id,
                "N=",length(which(observed.rows))))
```

White River Case Study

Observed plot



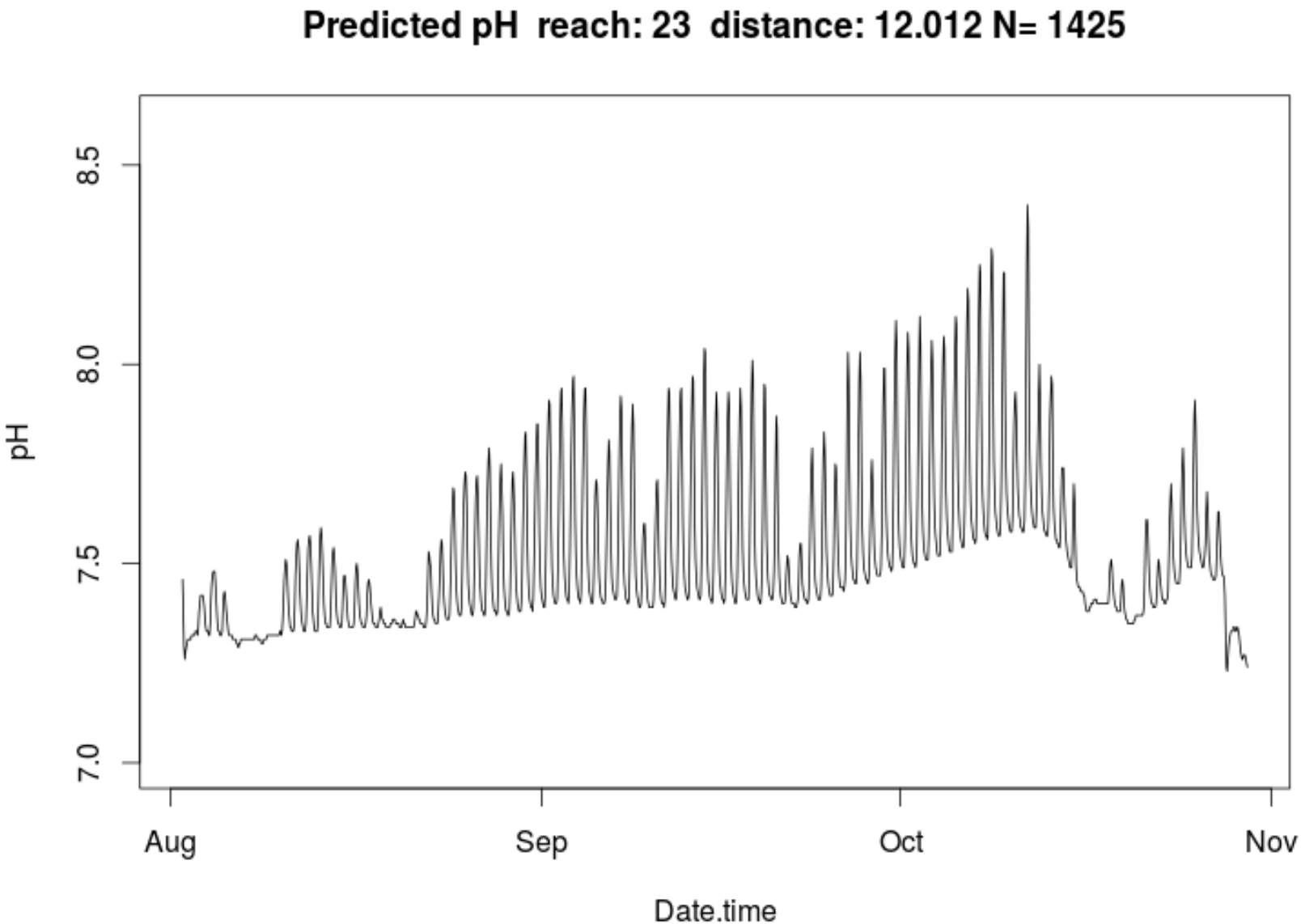
White River Case Study

Predicted plot

```
plot(predicted.subset,
     type="l",
     xlim=x.range,
     ylim=y.range,
     main=paste("Predicted", my.id,
                "N=",length(which(predicted.rows))))
```

White River Case Study

Predicted plot



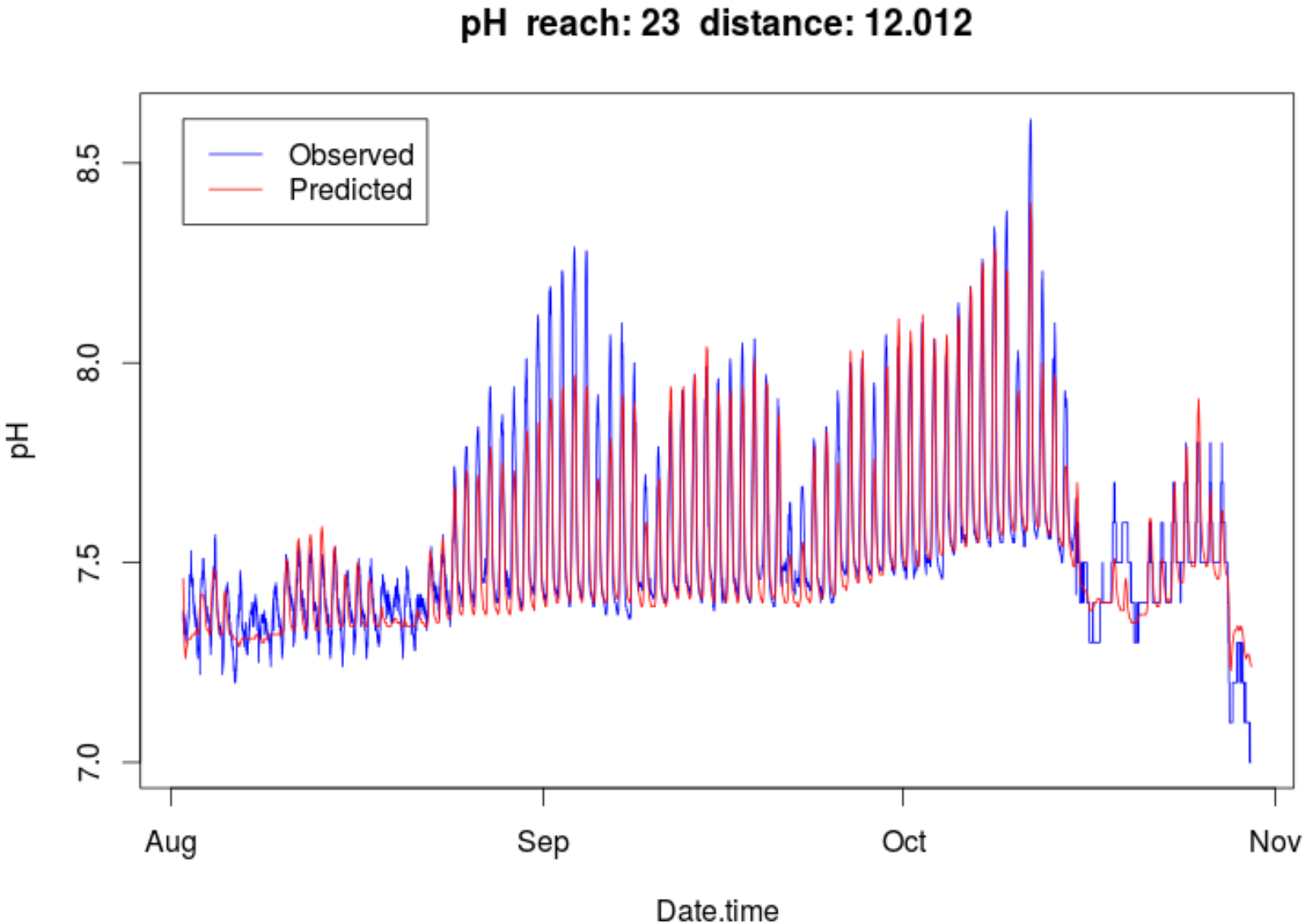
White River Case Study

Predicted and observed plot

```
plot(observed.subset,
     type="l",
     xlim=x.range,
     ylim=y.range,
     col="blue",
     main=paste(my.id))
lines(predicted.subset,
     type="l",
     col="red")
legend(x=min(x.range), y=ypos, yjust=yjus,
       lty=1,
       col=c("blue","red"),
       legend=c("Observed","Predicted"))
```

White River Case Study

Predicted and observed plot



White River Case Study

Subsample the data

- There are tens of thousands of rows for some of these plots.
- We make analysis more tractable by subsampling.

```
n <- min(200, nrow(observed.subset))
sample.rows <- sort(sample(1:nrow(observed.subset), n))
observed.sample <- observed.subset[sample.rows,]
predictions <- rep(0,n)
for (row in 1:n) {
  obs.time <- observed.sample$Date.time[row]
  pred.row <- which.min(abs(predicted.subset$Date.time-obs.time))
  predictions[row] <- predicted.subset[[predicted.parameter.name]][pred.row]
}
```

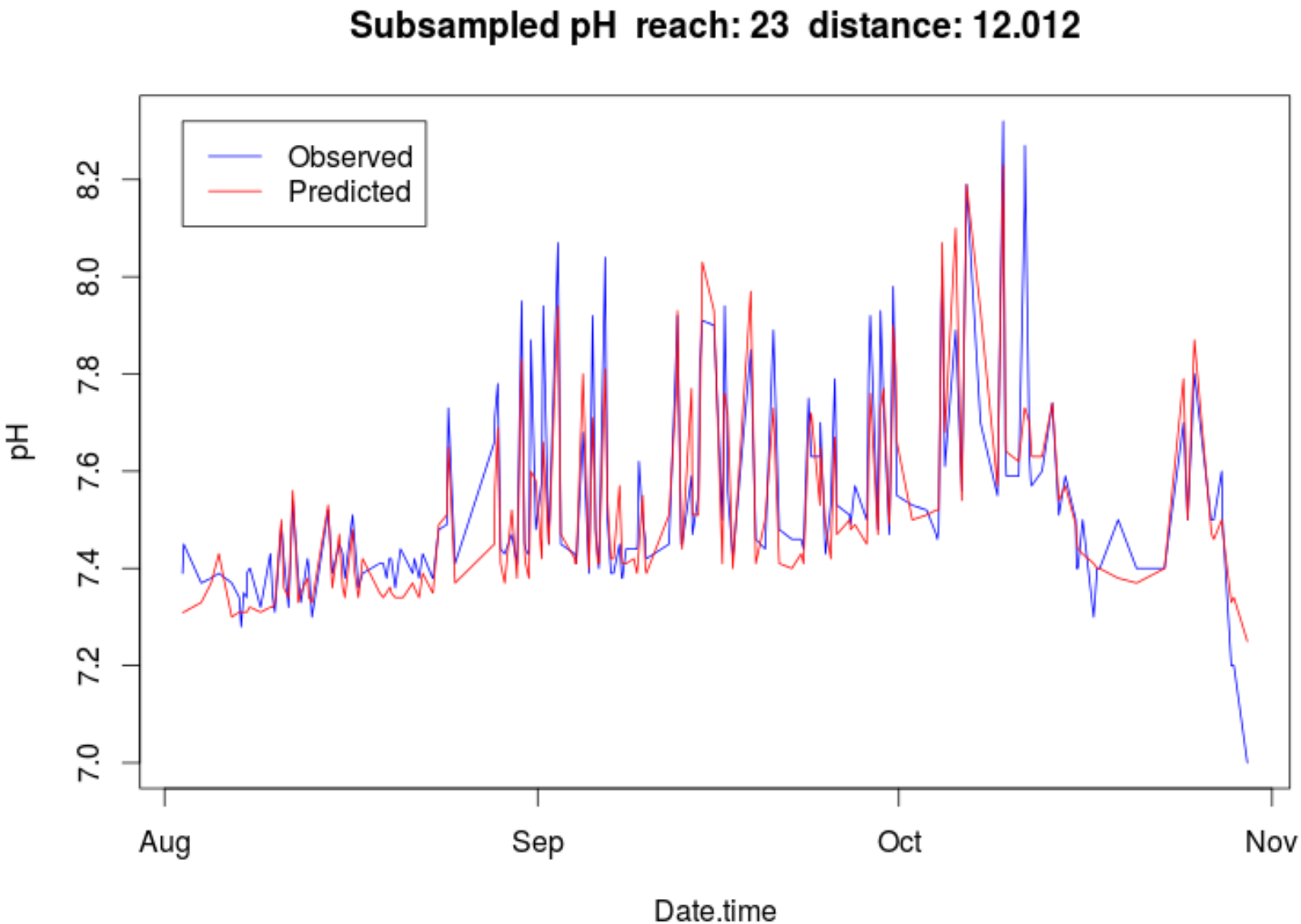
White River Case Study

Plot the observed and predicted subsample

```
plot(observed.sample,
     type="l",
     xlim=x.range,
     ylim=y.range,
     col="blue",
     main=paste("Subsampled",my.id))
lines(observed.sample$Date.time,predictions,
      type="l",
      col="red")
legend(x=min(x.range), y=ypos, yjust=yjus,
       lty=1,
       col=c("blue","red"),
       legend=c("Observed","Predicted")
       )
```

White River Case Study

Plot the observed and predicted subsample



White River Case Study

Fit a linear model

```
my.model <- lm(predictions ~ observed.sample[[observed.parameter.name]])
summary(my.model)
Call:
lm(formula = predictions ~ observed.sample[[observed.parameter.name]])

Residuals:
     Min       1Q   Median       3Q      Max
-0.25823 -0.03804 -0.00541  0.04324  0.24426

Coefficients:
                                          Estimate Std. Error t value Pr(>|t|)
(Intercept)                                1.26902    0.19073   6.653 2.74e-10 ***
observed.sample[[observed.parameter.name]]  0.82893    0.02526  32.815  < 2e-16 ***
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1

Residual standard error: 0.07581 on 198 degrees of freedom
Multiple R-squared:  0.8447, Adjusted R-squared:  0.8439
F-statistic:  1077 on 1 and 198 DF,  p-value: < 2.2e-16
```

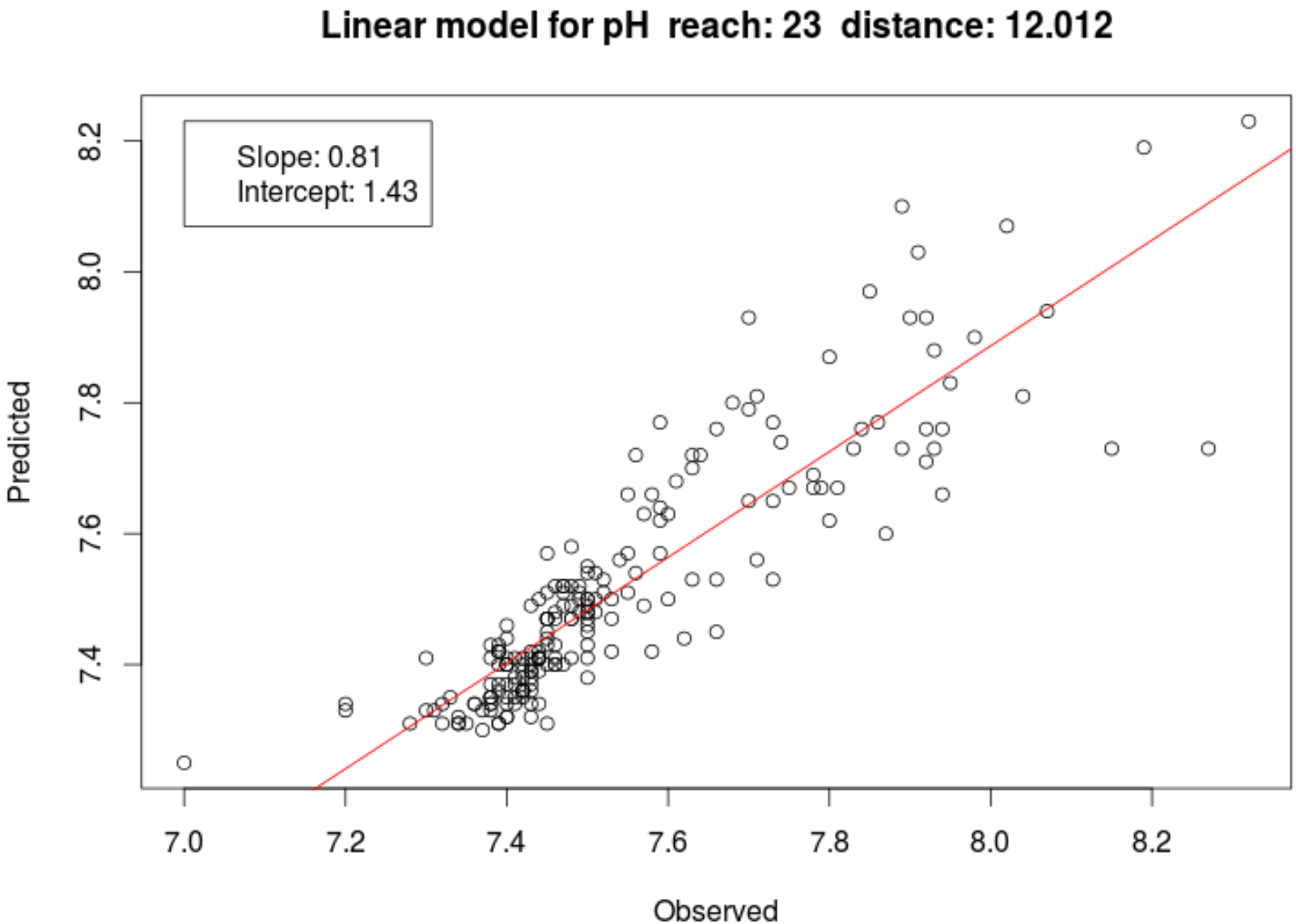
White River Case Study

Plot the observed *vs.* predicted and the linear fit

```
plot(observed.sample[[observed.parameter.name]], predictions,
     xlab="Observed",
     ylab="Predicted",
     main=paste("Linear model for", my.id))
# Add the best fit line and legend
abline(my.model,col="red")
legend(x=min(observed.sample[[observed.parameter.name]]),
       y=max(predictions),
       legend=c(paste("Slope:", round(my.model$coefficients[2],2)),
                paste("Intercept:", round(my.model$coefficients[1],2))))
```

White River Case Study

Plot the observed *vs.* predicted and the linear fit



White River Case Study

ANOVA on the model

```
anova(my.model)
Analysis of Variance Table

Response: predictions
                                          Df Sum Sq Mean Sq F value    Pr(>F)
observed.sample[[observed.parameter.name]]  1 6.1881  6.1881  1076.8 < 2.2e-16 ***
Residuals                                 198 1.1378  0.0057
---
Signif. codes:  0 *** 0.001 ** 0.01 * 0.05 . 0.1   1
```

White River Case Study

Plot the residuals for the linear model

```
par(mfrow=c(2,2))
plot(my.model, main=my.id)
par(mfrow=c(1,1))
```

White River Case Study

Plot the residuals for the linear model