

### R for basic data analysis

Marcus W. Beck<sup>1</sup>   Todd D. O'Brien<sup>2</sup>

<sup>1</sup>ORISE, USEPA NHEERL Gulf Ecology Division  
Email: [beck.marcus@epa.gov](mailto:beck.marcus@epa.gov)

<sup>2</sup>NOAA/NMFS Copepod Project  
Email: [todd.obrien@noaa.gov](mailto:todd.obrien@noaa.gov)

# What you'll learn about

- Data organization
- Data exploration and visualization
  - ▶ Common functions
  - ▶ Graphing tools
- Data analysis and hypothesis testing
  - ▶ Common functions
  - ▶ Evaluation of output
  - ▶ Graphing tools

# Data organization

Start by opening R

The workspace is a group of objects that are loaded for our current session

Objects are loaded into the workspace by importing (or making within R) and assigning them to a variable object with a name of our choosing

We can see what's loaded in our workspace:

```
# create variable as a numeric vector  
a <- c(1, 2)  
  
# verify that its in our workspace  
ls()  
  
## [1] "a"
```

# Data organization

Here's a workflow for importing data from Excel:



- Column names should be simple
- Ensure all data will be easy to read
- File, Save as .csv
- Creates a comma separated file that looks like a spreadsheet
- One spreadsheet at a time
- header = T
- See ?read.csv for list of function options
- Remember to assign a name

# Data organization

Excel files can also be imported directly into R without converting to a .csv file

However, this is not as intuitive as one would expect since .xls files are a proprietary format

There are several packages for importing excel files: Here's a nice [summary](#)

Try using the [gdata](#) package on your own, this also requires an installation of [Strawberry Perl](#)

Our workshop will not use methods that require direct import of Excel files  
- we will always use a .csv or .txt format for simplicity

# Data organization

If the data you want to import are a text file... open it, how are the columns separated?

- comma... `sep = ','`
- tabs... `sep = '\t'`
- space... `sep = ' '`
- arbitrary character

Use the `read.table` function and identify the column delimiter :

```
# data not loaded, only 'a' from before
ls()

## [1] "a"

# load data as comma separated, assign to dat
# make sure you are in the correct working directory
# e.g., setwd('C:/my_directory')
dat <- read.table('dat_example.txt', sep = ',', header = T)
ls()

## [1] "a" "dat"
```

# Data exploration

Now that the data are in our workspace, let's explore!

View the first six rows

```
head(dat)
```

```
##           datetimestamp do_mgl depth
## 1 2011-01-01 00:00:00      NA  1.54
## 2 2011-01-01 00:15:00      NA  1.53
## 3 2011-01-01 00:30:00      NA  1.52
## 4 2011-01-01 00:45:00      NA  1.51
## 5 2011-01-01 01:00:00      NA  1.50
## 6 2011-01-01 01:15:00      NA  1.48
```

View the last six rows

```
tail(dat)
```

```
##           datetimestamp do_mgl depth
## 2971 2011-01-31 22:30:00    9.6  1.49
## 2972 2011-01-31 22:45:00    9.8  1.50
## 2973 2011-01-31 23:00:00   10.7  1.50
## 2974 2011-01-31 23:15:00   10.8  1.51
## 2975 2011-01-31 23:30:00   10.8  1.52
## 2976 2011-01-31 23:45:00   10.8  1.52
```

# Data exploration

What object class is the data?

```
class(dat)

## [1] "data.frame"
```

What are the dimensions of the data frame?

```
dim(dat)

## [1] 2976    3

nrow(dat)

## [1] 2976

ncol(dat)

## [1] 3
```

The data contain 2976 rows and 3 columns, is this correct?



# Data exploration

Can we get a summary of the data frame?

```
summary(dat)
```

```
##           datetimestamp           do_mgl           depth
## 2011-01-01 00:00:00:    1   Min.      : 7.5   Min.      :0.70
## 2011-01-01 00:15:00:    1   1st Qu.: 9.5   1st Qu.:1.02
## 2011-01-01 00:30:00:    1   Median :10.2   Median :1.21
## 2011-01-01 00:45:00:    1   Mean    :10.1   Mean    :1.19
## 2011-01-01 01:00:00:    1   3rd Qu.:10.7   3rd Qu.:1.36
## 2011-01-01 01:15:00:    1   Max.    :12.5   Max.    :1.69
## (Other)                :2970   NA's    :431   NA's    :2
```

Summary returns different information depending on the class of each column

The first column is considered a 'factor' and simple counts are returned

The other two columns are 'numeric' and five number summaries are returned, including the number of observations with NA (or missing) values

# Data exploration

Individual summaries of variables are also possible

How do we obtain variables of interest?

```
names(dat)

## [1] "datetimestamp" "do_mgl"          "depth"
```

We can get a variable directly using \$ or via indexing with [,]

```
# these all do the same thing

# get using $
dat$do_mgl

# get using [row, column] with variable name
dat[, 'do_mgl']

# get using [row, column] with column index
dat[, 2]
```

## Data exploration

Just as we had summaries of the data frame, we can get summaries of individual variables

```
summary(dat$do_mgl)
```

```
##      Min. 1st Qu.  Median    Mean 3rd Qu.    Max.   NA's  
##      7.5      9.5     10.2    10.1    10.7    12.5     431
```

Or specific information...

```
# note use of na.rm = T, you must specify how to handle missing values
```

```
mean(dat$do_mgl, na.rm = T)
```

```
## [1] 10.15
```

```
range(dat$do_mgl, na.rm = T)
```

```
## [1] 7.5 12.5
```

```
var(dat$do_mgl, na.rm = T)
```

```
## [1] 0.8342
```

# Data exploration

Text-based summaries of our data are nice, but we should also visualize:

- How are our data distributed?
- Are there any outliers or extreme observations?
- How do our variables compare?

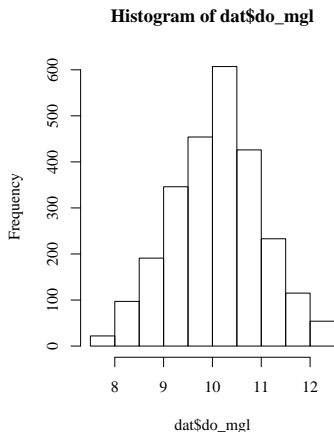
R has many built in functions for data exploration and plotting

- hist - plots a histogram (binned densities of continuous values)
- qqplot - comparison of a variable to a normal distribution
- barplot - for bar plots...
- plot - bivariate comparison of two variables
- Much, much more...

# Data exploration

Let's examine the distribution of dissolved oxygen measurements

```
hist(dat$do_mgl)
```

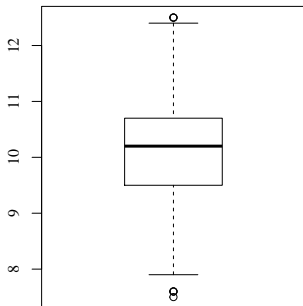


For example,  $\approx 600$  observations have DO values from, 10–10.5 mg L<sup>-1</sup>

# Data exploration

Boxplots are also useful for looking at a distribution

```
boxplot(dat$do_mgl)
```

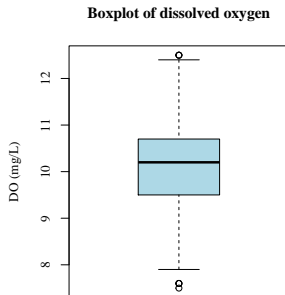


Let's make it look better...

# Data exploration

Boxplots are also useful for looking at a distribution

```
# a nicer looking boxplot  
boxplot(dat$do_mgl,  
  ylab = 'DO (mg/L)',  
  main = 'Boxplot of dissolved oxygen',  
  col = 'lightblue'  
)  
# see ?boxplot for all options
```



# Data exploration

Values beyond the whiskers in a boxplot are considered outliers

We can use the boxplot function to identify outliers...

```
# find the outliers using boxplot
myplot <- boxplot(dat$do_mgl, plot = F)
myplot$out

## [1] 7.6 7.6 7.5 7.6 12.5 12.5 12.5
```

This gives us the actual value, now we need to find them in our data

```
# find the rows of the outliers
outlier <- myplot$out
out_row <- which(dat$do_mgl %in% outlier)
out_row #these are the row number of the outliers

## [1] 704 1001 1002 1003 1322 1440 1441
```



# Data exploration

You can treat outliers as you wish

```
dat[out_row, ] # view the outliers
```

```
##           timestamp do_mgl depth
## 704  2011-01-08 07:45:00    7.6  1.20
## 1001 2011-01-11 10:00:00    7.6  1.32
## 1002 2011-01-11 10:15:00    7.5  1.31
## 1003 2011-01-11 10:30:00    7.6  1.30
## 1322 2011-01-14 18:15:00   12.5  1.28
## 1440 2011-01-15 23:45:00   12.5  1.38
## 1441 2011-01-16 00:00:00   12.5  1.37
```

Remove them...

```
dat[out_row, 'do_mgl'] <- NA
```

Replace with mean...

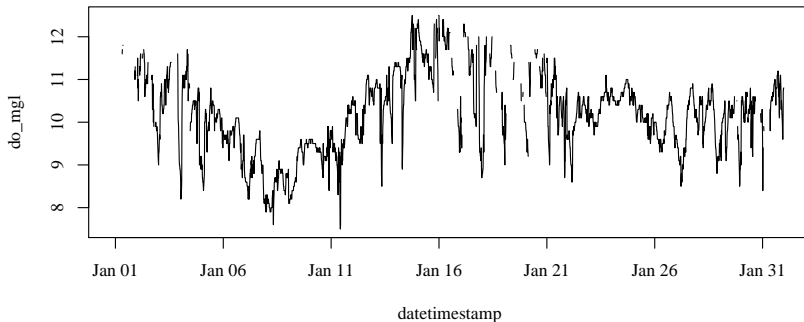
```
dat[out_row, 'do_mgl'] <- mean(dat$do_mgl, na.rm = T)
```

Or do nothing...

# Data exploration

The time series can be plotted to evaluate trends

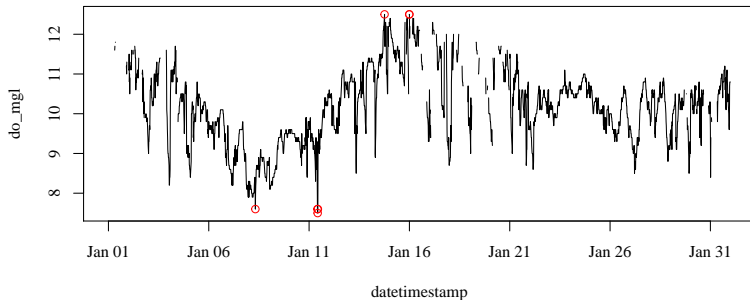
```
#first we have to convert the datetimestamp column  
dat$datetimestamp <- as.POSIXct(dat$datetimestamp)  
  
# plot the time series, y vs x syntax  
plot(do_mgl ~ datetimestamp, data = dat, type = 'l')
```



# Data exploration

We can also add our outliers to the plot

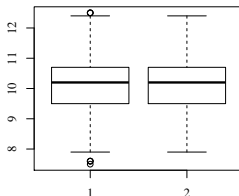
```
# plot the time series  
plot(do_mgl ~ datetimestamp, data = dat, type = 'l')  
  
# use the out_row object from earlier to subset  
x <- dat[out_row, 'datetimestamp']  
y <- dat[out_row, 'do_mgl']  
points(x, y, col = 'red')
```



# Data exploration

What effect do these outliers have on the mean DO value for the time series?

```
# the original dataset  
dat_orig <- dat$do_mgl  
  
# dataset with outliers removed  
dat_remove <- dat$do_mgl  
dat_remove[out_row] <- NA  
  
# a boxplot comparison of the two datasets  
boxplot(dat_orig, dat_remove)
```



## Data exploration

We can test these differences more formally using a standard statistical test

```
# a t-test, evaluates the null that the difference in means is zero
t.test(dat_orig, dat_remove)

##
##  Welch Two Sample t-test
##
## data:  dat_orig and dat_remove
## t = -0.05, df = 5081, p-value = 0.9602
## alternative hypothesis: true difference in means is not equal to 0
## 95 percent confidence interval:
##  -0.05128  0.04873
## sample estimates:
## mean of x mean of y
##      10.15      10.15
```

There is a 96.02% probability that the difference in means between the datasets is equal to zero, due to random chance

We should leave the outliers in the dataset...

# Conclusions

The previous examples are simple approaches to exploratory data analysis with R

These were designed to get you comfortable using the R command-line

The workshop will provide a comprehensive guide to exploring and working with time series data from SWMP

Please see the additional resources slide in 'intro\_to\_r.pdf' for more training information

Questions: contact Marcus Beck ([beck.marcus@epa.gov](mailto:beck.marcus@epa.gov)) or Todd O'Brien ([todd.obrien@noaa.gov](mailto:todd.obrien@noaa.gov))