

# Dataflow Output Standard Operating Procedure(SOP)

Joseph Stachelek

May 22, 2015

## Contents

<b>1</b>	<b>Introduction</b>	<b>2</b>
<b>2</b>	<b>Programs and Applications used in this SOP</b>	<b>2</b>
<b>3</b>	<b>Installing DataflowR</b>	<b>2</b>
3.1	Data . . . . .	3
<b>4</b>	<b>Cleaning, loading, interpolating, and plotting streaming data</b>	<b>4</b>
4.1	Clean incoming Dataflow and C6 files . . . . .	4
4.2	QA cleaned streaming data . . . . .	4
4.3	Loading previously cleaned streaming data . . . . .	5
4.4	Interpolating cleaned data files . . . . .	5
4.5	Plotting interpolated surfaces . . . . .	5
<b>5</b>	<b>Handling discrete grab sample data</b>	<b>6</b>
5.1	Cleaning grab sample records . . . . .	6
5.2	Loading previously cleaned grab data . . . . .	6
5.3	Plotting grab sample data . . . . .	7
<b>6</b>	<b>Data Analysis</b>	<b>7</b>
6.1	Calculating a difference-from-average surface . . . . .	7
6.2	Fit grab sample and streaming averages . . . . .	8
6.2.1	Calculate coefficients . . . . .	8
6.2.2	Generate corrected surfaces . . . . .	9

# 1 Introduction

This SOP details a workflow and introduces an R package to clean, load, interpolate, and plot streaming Dataflow output and associated discrete grab samples. The steps necessary to perform a number of more involved data analyses are also included.

## 2 Programs and Applications used in this SOP

- A spreadsheet program (MS Excel or similar)
- R (must have the **DataflowR** package and its dependencies installed)
- RStudio (optional; for more easily accessing documentation)
- ArcGIS (optional; alternatively use QGIS, Surfer, etc, for more easily adjusting final symbology)

## 3 Installing DataflowR

The R package **DataflowR** is distributed via a **.tar.gz** (analagous to **.zip**) package archive file. This package contains the source code for package functions as well as archived datasets for the SFWMD Florida Bay Dataflow Monitoring Program. In RStudio, it can be installed by navigating to **Tools -> Install Packages...** -> **Install from:** -> **Package Archive File**.

In most cases, **DataflowR** will be installed in the default user library in a folder named for your version of R. On Windows this is probably:

`C:/Users/<your_username>/Documents/R/win-library/<your_R_version>/DataflowR.`


The directory should have the following file structure:

```
/
├── doc
├── extdata
├── help
├── html
├── Meta
└── R
```

### 3.1 Data

All data is stored under `extdata` in the following subdirectories:

```
/
└─ extdata
   └─ DF_BaseFile
   └─ DF_FullDataSets
   └─ DF_Subsets
   └─ DF_Surfaces
   └─ DF_Validation
```

 **IMPORTANT!!** before continuing, copy the contents of `extdata` to a local folder such as: `C:\Documents\Data\Dataflow`.  
Next, update the file `localpath` to point to the location of this new folder.  
For example, the structure of `C:\Documents\Data\Dataflow` would look like:

```
/
└─ Dataflow
   └─ DF_BaseFile
   └─ DF_FullDataSets
   └─ DF_Subsets
   └─ DF_Surfaces
   └─ DF_Validation
```

Dataflow surveys in the archived data repository include the following:

Year	15	14	13	12	11	10	09	08	07	06
J										
F	x	x								
M										
A		x								
M	x		x							
J										
J		x								
A			x							
S										
O		x								
N			x							
D				x						

## 4 Cleaning, loading, interpolating, and plotting streaming data

### 4.1 Clean incoming Dataflow and C6 files

Incoming streaming data should be placed in `DF_FullDataSets/Raw/InstrumentOutput` in a folder named for the survey date in `yyyymm` format (e.g. Feb-2015 is 201502). Likewise, Dataflow ("`.txt`" or "`.TXT`") and C6 ("`.csv`" or "`.CSV`") files should start with the survey date in `yyyymmdd` format. For example, the raw data files for February 2015 would be placed under:

```
/
├── Dataflow
│   ├── DF_FullDataSets
│   │   ├── Raw
│   │   │   ├── InstrumentOutput
│   │   │   │   ├── 201502
│   │   │   │   │   ├── 201502_DF021115.txt
│   │   │   │   │   ├── ...
│   │   │   │   │   ├── 201502_C6_11FEB.csv
│   │   │   │   │   └── ...
```

👉 When working in a non-Windows environment, `.csv` files created under Windows may need to be opened and resaved as `text\csv` in order to avoid illegal "nul" characters.

Cleaning is accomplished via the `streamclean` function. The example below shows how to specify inputs to clean the data for the February 2015 survey. For this example we set the `tofile` parameter to `FALSE` but setting it to `TRUE` will save the cleaned output to `DF_FullDataSets` as a `.csv` file. The `streamclean` function will gather all the Dataflow records and merge them with the C6 data, remove leading and trailing records of all zeros, format GPS coordinates, check that conductivity to salinity calculations are correct (recalculate if necessary), and classify records based on fathom and CERP basin designations.

```
> dt<-streamclean(yearmon=201502,mmmin=7,c6pres=TRUE,tofile=FALSE,sep=",")
```

### 4.2 QA cleaned streaming data

The `streamclean` function does not perform detailed QA of individual parameter values or detect systematic bias. Systematic bias might occur because of sensor failure. In these cases there may be data but it is "junk data". The `streamqa` function creates

a series of diagnostic plots of the data. The user enters an interactive QA "session" to detect systematic biases and eliminate unrealistic data spikes.

```
> streamqa(yearmon=201502)
```

### 4.3 Loading previously cleaned streaming data

The `streamget` function will retrieve previously cleaned data. The function looks for full data sets in the `DF_FullDataSets` folder that match the specified survey date. An example for the February 2015 survey is shown below.

```
> dt<-streamget(yearmon=201502)
```

### 4.4 Interpolating cleaned data files

The `streaminterp` function will interpolate a dataset that has been loaded into memory from the `streamclean` or `streamget` functions. Interpolations are performed using functions in the `ipdw` R package (Stachelek 2014). Variables to be interpolated must be specified as inputs to the `paramlist` parameter. If you have loaded your dataset to memory under the name `dt`, use `names(dt)` to see the available parameters. Enter one or more parameters as arguments to a character list. For example, to interpolate salinity only (as below) use `c("sal")`. Additional parameters can be appended. For example, to interpolate salinity and temperature use `c("sal", "temp")`. Interpolation should take about 20 minutes plus about 2 minutes for each entry in `paramlist`. Raster surface output will be written to a subfolder of `DF_Surfaces` named for the year and month of the survey in question.

`streaminterp` will first attempt to split the full data set into training and validation datasets. If these already exist in the `DF_Subsets` and `DF_Validation` folders, a warning will be printed and the pre-existing datasets will be used. Next, `streaminterp` will attempt to create a dedicated folder to hold all the interpolated surfaces for the given survey. If this folder already exists, `streaminterp` will print a warning but the function should proceed as normal (the warning can be disregarded).

More details regarding the interpolation procedure can be found in Stachelek and Madden (2015).

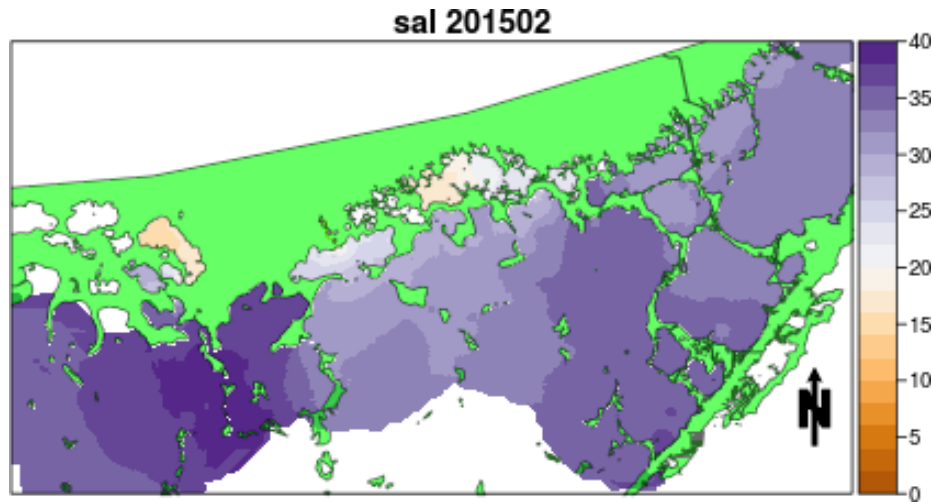
```
> streaminterp(dt,paramlist=c("sal"),yearmon)
```

### 4.5 Plotting interpolated surfaces

A quick visual inspection of interpolated outputs can be accomplished using the `surfplot` function. The `rnge` parameter takes either a single survey date or a list of two

survey dates to specify a date range for plotting. More detailed publication quality maps should be produced using a dedicated GIS program such as ArcGIS or QGIS.

```
> surfplot(rnge=c(201502),params=c("sal"))
```



## 5 Handling discrete grab sample data

### 5.1 Cleaning grab sample records

Incoming grab sample .csv data files should be placed in the `DF_GrabSamples/Raw` folder and their file names should have the survey date in `yyyymm` format preappended. These files can be cleaned using the `grabclean` function. The `grabclean` function formats column names, removes columns/rows of missing data, and calculates minute averages of the streaming data that correspond to the grab sample date/times. Output is saved to the `DF_GrabSamples` folder when `tofile` is set to `TRUE`. Suspect data records should be identified manually in the `flags` column. In this version of the SOP, there is only one flag `s`. This becomes important in Section 6.2 because suspect data records can create problems converting between extracted and fluoresced chlorophyll.

```
> grabclean(yearmon=201410,tofile=FALSE)
```

### 5.2 Loading previously cleaned grab data

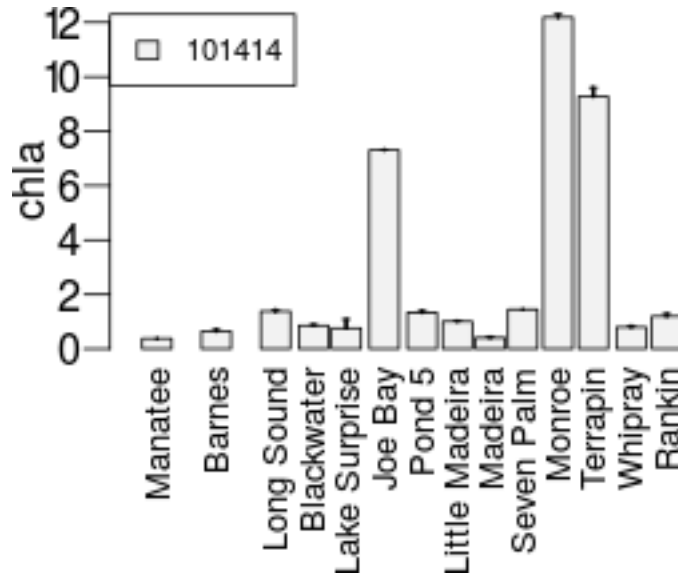
The `rnge` paramter takes either a single survey date or a list of two survey dates to specify a date range for retrieving cleaned grab data.

```
> grabs<-grabget(rnge=c(201402,201410))
```

### 5.3 Plotting grab sample data

Several plot types are defined in the `grabplot` function including "permit-style" vertical bar plots ("permitbarplot"), and. `grabplot` calls `grabget` internally.

```
> grabplot(rnge=201410,params=c("chla"),plottype="permitbarplot")
```

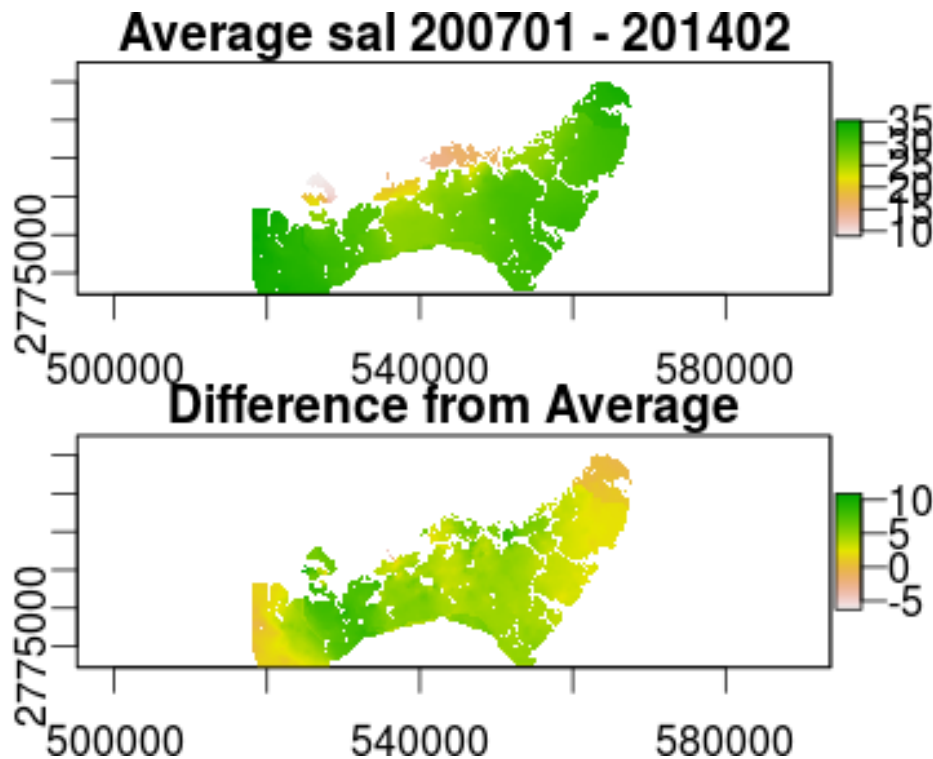


## 6 Data Analysis

### 6.1 Calculating a difference-from-average surface

The `avmap` function takes a survey date as input and searches the `DF_Surfaces` folder for interpolated surfaces of the same parameter within a specified number of months for each year. The number of months (tolerance) on either side of the input month is set using the `tolerance` parameter. The found surfaces often have different extents. The `percentcov` parameter controls the percent of all identified surveys required before a pixel is included in difference-from-average computations. Output surfaces are written to the current working directory unless the `tofile` parameter is set to `FALSE`.

```
> avmap(yearmon=201502,params="sal",tofile=TRUE,percentcov=0.6,tolerance=1)
```



## 6.2 Fit grab sample and streaming averages

### 6.2.1 Calculate coefficients

In order to generate maps of chlorophyll concentration, streaming fluorescence values (chlorophyll, algal pigments, cdom) must be statistically "fit" (regressed) against lab-derived extracted chlorophyll values. The `chlcoef` function searches the `DF_GrabSamples` folder for a cleaned grab dataset that matches the specified `yearmon` parameter value. First, the function generates a correlation matrix and identifies streaming variables that have at least a 0.4 correlation with extracted chlorophyll. The resulting variables are entered into a linear regression. If the R-squared value of the regression is less than 0.7, the variables are entered into a second degree polynomial regression. The regression (either the linear or the second degree polynomial) is subjected to a backward stepwise AIC model selection. The output of this step is usually a regression with a reduced number of parameters (citeMASS). The final regression is checked for multicollinearity by calculating variance inflation factors (Helsel and Hirsh 2002).

```
> chlcoef(yearmon=201502,remove.flags=TRUE)
```



### 6.2.2 Generate corrected surfaces

## References

- Stachelek, J. (2014). *ipdw: spatial interpolation by Inverse Path Distance Weighting*. R package version 0.2-2.
- Stachelek, J. and Madden, C. J. (2015). Application of inverse path distance weighting for high-density spatial mapping of coastal water quality patterns. *International Journal of Geographical Information Science*, pages 1–11.