```
/* CalcESLs.sas */
/* ------------ */
/* Created 02 January 2006 by Sarah Gilman                    */
/* report errors to gilmans@u.washington.edu                  */

/* ------------------------------------------------------------ */
/* This is a sample SAS program to calculate ESL regressions over   */
/* a range of drop thresholds and wave height metrics.              */
/* It was written for SAS v9.1 for PC and should be compatible      */
/* with SAS v.8 as well.  To run on SAS v6 for Macintosh, some      */
/*  dataset and variable names may need to be shortened             */

/* The program assumes you have pairs of files contaning drop time  */
/* and height data of the type created by SiteParser and are        */
/* using hourly wave height data in the format used by the National */
/* Data Buoy Center (www.ndbc.noaa.gov) */

/* In most cases you should need to make changes only to Part 1.    */
/* Read through Part 1 carefully as it contains the information     */
/* for the program to find the input data files and locations to    */
/* save output    */

/* The general procedure is to submit each of the 4 parts to SAS    */
/* separately, checking  the log for errors between each part.      */
/* ------------------------------------------------------------ */


/* Part1 - Source & Output Files */
/* ----------------------------- */
/* Modify the following sections to point to the relevant file      */
/* locations on your computer                                       */

/* Output Files    */
/* ------------    */
/* The program will save the regression summary statistics to       */
/* a user-specified location.  The file will be saved in CSV        */
/* format and can be imported into Excel or other programs. This    */
/* data is also in a SAS dataset labelled 'reg_results'.  The                  */
/* raw data for regression is located in a SAS dataset labelled     */
/* 'toreg'                                                          */

/* replace with the path for your folder  */
%let fpath=C:\User\My Documents\myloggers\;
/* filename for regression statistics, omit extension  */
%let outname=ESLregs;

/* SiteParser file locations */
/* ----------------------    */
/* This program assumes a separate set of output files for          */
/* each year of data and drop threshold. If you have only one       */
/* year, set fyear and lyear to same value.  Similarly, if   */
/* there is only one drop threshold, set firstdrop and lastdrop     */
/* to same value */

%let firstdrop = 2; /* lowest drop threshold */
%let lastdrop = 7; /* highest drop threshold */
%let fyear=2000; /* first year of data */
%let lyear=2004; /*last year of data */

/* Actual file locations (use full path).  Use a separate entry for */
/* the droptime and droptide files for each year and drop value.    */
```

```
/* The variable names must be of the format tide_y####_d#, where    */
/* #### is the a year between fyear and lyear and # is an integer   */
/* between firstdrop and lastdrop.  Add more rows if necessary      */
%let tide_y2000_d2="C:\Documents and Settings\Administrator\Desktop\droptide_HMSMussel2002.txt";
%let time_y2000_d2="C:\Documents and Settings\Administrator\Desktop\droptime_HMSMussel2002.txt";


/* Buoy file locations */
/* ------------------- */

/* List the locations of all the NDBC buoy files (usually 1 per    */
/* calendar year) Add additional rows if necessary                 */
%let b1="C:\My Documents\NDBC data\46042h2000.txt";
%let b2="C:\My Documents\NDBC data\46042h2001.txt";

%let buoys=&b1 &b2;  /* add more variables to this list if needed */

/* CONGRATULATIONS!  You have made it to the end of Part 1!!!!     */


/* Part2 - Reading source files */
/* ---------------------------- */
/* Short routines to read in the SiteParser and NDBC data files    */

/* SiteParser data */
/* ------------    */

proc format;
        invalue twomiss 'N/A'=.
                        'No Data'=.m
                        -99999 - 99999=[best8.]
        ;
        invalue tmiss 'N/A'=.
                'No Data'=.m
                other=time5.
        ;
run;
%macro putlgvars (fnum, lnum, type);
        %if (&type=number) %then %do;
                %do k=&fnum %to &lnum;
                        log&k :twomiss.
                %end;
        %end;
        %else %if (&type=time) %then %do;
                %do k=&fnum %to &lnum;
                        log&k :tmiss.
                %end;
        %end;
%mend putlgvars;
%macro read1_sp (filei,varname,vtype);
        data line2var;
                format namelist $500.;
                format textname $30.;
                format logname $8.;
                infile &filei firstobs=3 obs=3 truncover ls=500 dsd;
                input namelist :$500.;
                ctr=1;
                textname=upcase(scan(namelist,ctr,'|'));
                do while(textname NE "");
                        logname="log"||trim(left(ctr));
                        output;
```

```
                        put logname= textname=;
                        ctr=ctr+1;
                        textname=upcase(scan(namelist,ctr,'|'));
                end;
                call symput("flog",trim(left(1)));
                call symput("llog",trim(left(ctr-1)));
                put firstlog= ctr=;
                keep logname textname;
        run;
        options errors=5;
        data readcol;
                format datetxt $20.;
                infile &filei dlm='|' firstobs=5 dsd missover;
                input datetxt :$20. %putlgvars(&flog,&llog,&vtype);
                dateval=input(datetxt,mmddyy10.);
                format dateval date7.;
                drop datetxt;
        run;
        proc transpose data=readcol out=long1(rename=(_name_=logname col1=&varname));
                by dateval;
        run;
        proc sort data=long1;
                by logname;
        run;
        proc sort data=line2var;
                by logname;
        run;
        data long&varname;
                merge line2var long1;
                by logname;
                attrib _all_ label="";
        run;
%mend read1_sp;
%macro build_pfiles;
        %do y=&fyear %to &lyear;
                %do d=&firstdrop %to &lastdrop;
                        %let timefile=&&time_y&y._d&d;
                        %let tidefile=&&tide_y&y._d&d;
                        %read1_sp (timefile,dtime,time);
                        %read1_sp (tidefile,dtide,number);
                        proc sort data=longdtime;
                                by dateval textname;
                        run;
                        proc sort data=longdtide;
                                by dateval textname;
                        run;
                        data oneset;
                                merge longdtime longdtide;
                                by dateval textname;
                                drop logname;
                                if (dtime=. or dtime=.M) then delete;
                                realtime=dhms(dateval,0,0,0) + dtime;
                                rthour=round(realtime,hms(1,0,0));
                                dropnum=&d;
                        run;
                        proc append base=alldrops new=oneset;
                        run;
                %end;
        %end;
        data alldrops;
                set alldrops;
```

```
                rename textname=logname;
        run;
%mend build_pfiles;
%build_pfiles;


/* buoy data       */
/* ------------    */

%macro elimdoubles(dataso,timevar,valuevar);
/* will delete duplicate time intervals in a dataset */
/* assume the time interval with the smaller data value is the one to delete */
/* this should default to removing missing values over non-missing values */

        proc sort data=&dataso;
                by &timevar descending &valuevar;
        run;
        data &dataso;
                set &dataso;
                lagtime=lag(&timevar);
                if lagtime=&timevar then delete;
                drop lagtime;
        run;

%mend elimdoubles;
%macro read_buoy(varlist,start,stop,fname,interpol);
        %put read_buoy called to read &varlist;
        %put from file &fname;
        %put start=&start stop=&stop;

        %let dirwind=%index(&varlist,winddir);
        %if (&dirwind) %then %let varlist= &varlist nwind ewind;

        filename bfile (&fname);

        data rawbuoy;
         infile bfile ;
         input Year mo Day hr winddir  wind GST  Wave  DPD   APD  MWD  BAR    tair  WTeMP  DEWP
VIS;
         if (hr=.) then delete;
         if winddir=999 then winddir=.;
         if wind=99. then wind=.;
         if gst=99. then gst=.;
         if wave=99. then wave=.;
         if dpd=99. then dpd=.;
         if apd=99. then apd=.;
         if mwd=999. then mwd=.;
         if bar=9999 then bar=.;
         if tair=999. then tair=.;
         if wtemp=999. then wtemp=.;
         if dewp=999. then dewp=.;
         if vis=99. then vis=.;
         realtime=dhms(mdy(mo,day,year),hr,0,0);
         /*convert from UTC to PST*/
         realtime=realtime-dhms(0,8,0,0);
         year=year(datepart(realtime));
         mo = month(datepart(realtime));
         day = day(datepart(realtime));
         hr= hour(timepart(realtime));
         min= minute(timepart(realtime));
         if (&dirwind) then do;
```

```
                nwind=cos(winddir*constant('PI')/180);
                ewind=sin(winddir*constant('PI')/180);
        end;
        keep year mo day hr min realtime &varlist;
       run;
       %elimdoubles(rawbuoy,realtime,&varlist);
       %if &interpol=yes %then %do;
                proc expand data=rawbuoy out=rbuoy3 from=hour.;
                  id realtime;
                  convert &varlist /method=join;
                run;

                %if (&dirwind) %then %do;
                        %let varlist2 = %sysfunc(tranwrd(&varlist,winddir,));
                        proc expand data=rbuoy3 out=rbuoy4 from=hour. to=minute10.;
                          id realtime;
                          convert &varlist2/method=join;
                          convert winddir/method=step;
                        run;
                %end;
                %else %do;
                        proc expand data=rbuoy3 out=rbuoy4 from=hour. to=minute10.;
                          id realtime;
                          convert &varlist/method=join;
                        run;
                %end;
                data buo_var;
                        set rbuoy4;
                        if (&start>realtime OR &stop<realtime) then delete;
                run;
       %end;
       %else %do;
                data buo_var;
                        set rawbuoy;
                        if (&start>realtime OR &stop<realtime) then delete;
                run;
       %end;
%mend read_buoy;
%read_buoy(wave,dhms(&fday,0,0,0),dhms(&lday,0,0,0),&buoys,no);
data obswave;
       set buo_var;
       rthour=realtime;
       keep rthour wave;
run;



/* Part3 - Regressions          */
/* -------------------          */

/* Calculate different wave time window */
proc expand data=obswave out=windows method=none;
       id rthour;
       convert wave=w01max;
       convert wave=w03max/transform=(movmax 3);
       convert wave=w06max/transform=(movmax 6);
       convert wave=w09max/transform=(movmax 9);
       convert wave=w12max/transform=(movmax 12);
       convert wave=w24max/transform=(movmax 24);
run;
proc sort data=windows;
```

```
        by rthour;
run;
proc sort data=alldrops;
        by rthour;
run;
data toreg;
        merge alldrops(in=sd2) windows(in=wd);
        by rthour;
        if sd2 and wd;
run;

/* calculate the mean wave for each time window */
/* necessary to calculate average run-up       */
proc sort data=toreg;
        by rthour;
run;
proc summary data=toreg noprint;
        by rthour;
        var w01max w03max w06max w09max w12max w24max;
        output out=ndb1 max=;
run;
proc summary data=ndb1 noprint;
        var w01max w03max w06max w09max w12max w24max;
        output out=ndb2 mean=;
run;
data _null_;
 set ndb2;
 call symput("avn01",trim(left(w01max)));
 call symput("avn03",trim(left(w03max)));
 call symput("avn06",trim(left(w06max)));
 call symput("avn12",trim(left(w12max)));
 call symput("avn09",trim(left(w09max)));
 call symput("avn24",trim(left(w24max)));
run;

/* run all possible regressions */
proc sort data=toreg;
        by dropnum logname;
run;
proc reg data=toreg outest=allregs outseb edf noprint;
        by dropnum logname;
        var dtide w01max w03max w06max w09max w12max w24max;
        w01max: model dtide=w01max;
        w03max: model dtide=w03max;
        w06max: model dtide=w06max;
        w09max: model dtide=w09max;
        w12max: model dtide=w12max;
        w24max: model dtide=w24max;
run;
data allregs2;
        set allregs;
        attrib _all_ label=" ";
run;
data allregs2a;
        set allregs2(where=(_type_="PARMS"));
        select (_model_);
                when('w01max') slope=w01max;
                when('w03max') slope=w03max;
                when('w06max') slope=w06max;
                when('w09max') slope=w09max;
                when('w12max') slope=w12max;
```

```
                when('w24max') slope=w24max;
                otherwise;
        end;
        nobs=_p_+_edf_;
        drop _type_ w01max w03max w03max w06max w09max w12max w24max _p_ _in_ _edf_;
run;
data allregs2b;
        set allregs2(where=(_type_="SEB"));
        select (_model_);
                when('w01max') se_slope=w01max;
                when('w03max') se_slope=w03max;
                when('w06max') se_slope=w06max;
                when('w09max') se_slope=w09max;
                when('w12max') se_slope=w12max;
                when('w24max') se_slope=w24max;
                otherwise;
        end;
        se_int=intercept;
        drop _type_ w01max w03max w06max w09max w12max w24max _p_ _in_ _edf_ intercept
          _rsq_ _rmse_;
run;
data reg_results;
        merge allregs2a allregs2b;
        by dropnum logname _model_;
        attrib _all_ label="";
        select (_model_);
                when('w01max') meanwave=&avn01;
                when('w03max') meanwave=&avn03;
                when('w06max') meanwave=&avn06;
                when('w09max') meanwave=&avn09;
                when('w12max') meanwave=&avn12;
                when('w24max') meanwave=&avn24;
                otherwise ;
        end;
        runup_avg=abs(slope*meanwave);
        wavehr=int(substr(_model_,2,2));
        format _rmse_ intercept _rsq_ slope se_slope se_int meanwave runup_avg  6.3;
run;

/* Part4 - Output              */
/* ----------------------------*/

/* Prints the top 6 regressions by logger and also plots   */
/* all R-squared values. Saves regression output to        */
/* the designated file location                            */

/* print the top six Rsquared values for each logger */
proc sort data=reg_results;
        by logname descending _rsq_;
run;
data topten;
        set reg_results;
        rank + 1;
        by logname;
        if first.logname then rank=1;
run;
proc print data=topten(where=(rank<6)) noobs;
        title "Top 6 Regressions (based on Rsq) by logger";
        var logname dropnum _model_ nobs intercept se_int slope se_slope _rsq_ runup_avg;
run;
title;
```

```
proc sort data=reg_results;
        by dropnum logname;
run;

/* Plot Rsquared values for all regression */
proc gplot data=reg_results;
        by dropnum;
        plot _rsq_*wavehr=logname/vaxis=0 to 1 by 0.1;
run;
quit;

/*Export the summary file */
proc sort data=reg_results;
        by dropname logname wavehr;
run;
proc export data=reg_results
        outfile="&fpath&outname..csv" dbms=csv replace;
run;
```