Accompanying “SIREN” model for:

Trade-offs between growth and maturation: the cost of reproduction for surviving environmental extremes

Thomas M. Luhring[*a,b,c], and Ricardo M. Holdo[a]

[a] Division of Biological Sciences, University of Missouri, Columbia, Missouri, United States of America

[b] Savannah River Ecology Laboratory, Aiken, South Carolina, United States of America,

[c] Current Address: Fisheries and Wildlife, Michigan State University, Room 13 Natural Resources, 480 Wilson Rd., East Lansing, MI

***Corresponding Author:** tomluhring@gmail.com

Competitions Model Files:

Single-Species Sensitivity Analyses:

Siren lacertina

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
% This is the main program used to run the function%
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

clear global;

lCC=3;          % number levels used for Sts
lcC=3;          % number levels used for stS
lh=7;           % number levels used for h

simulation=lCC*lcC*lh; %each "simulation" has 100 replicated runs (each 5000
years) of identical stS, Sts, and h
%Creating vectors of each varying parameter (h, sTS, STs) to create all
%possible combinations at each location (63 possible combinations)

hhhh=[0 0.05 0.1 0.15 0.2 0.25 0.30];

iH=repmat(hhhh,1,9);     %Creates a vector of values for h 0 - 3.0
H=iH';

dsTS=[0 0 0 0 0 0 0 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0 1.0 1.0];
isTS=repmat(dsTS,1,3);   %Creates a vector of values for stS (set to zeros
for no competition)
sTS=isTS';

iSTs=[0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0 0.5 0.5 0.5 0.5 0.5 0.5 0.5
0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 0.5 1.0 1.0 1.0 1.0 1.0
1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0 1.0];  %Creates a
vector of values for Sts
STs=iSTs';



dlmwrite("Directory"\SirenModelh.txt',H) %creating text file with values for
h
dlmwrite("Directory"\SirenModellstGS.txt',sTS)
dlmwrite("Directory" \SirenModelGStls.txt',STs)

[MeanTV,MeannoN,Meantv,Meannon,stdTV,stdtv,stdnoN,stdnon]=L4SirenModelFinalSA
2(simulation);

%dlmwrite("Directory"/OutputExtinctions1.txt',MeanTV');
%dlmwrite("Directory"/OutputExtinctions2.txt',Meantv');
%dlmwrite("Directory"/OutputYearsZero1.txt',MeannoN');
%dlmwrite("Directory"/OutputYearsZero2.txt',Meannon');

Output=zeros(simulation,11);%creating matrix for dependent and independent
variables

Output(:,1)=H;
Output(:,2)=sTS;
Output(:,3)=STs;
Output(:,4)=MeanTV';
Output(:,5)=stdTV';
```

```
Output(:,6)=Meantv';
Output(:,7)=stdtv';
Output(:,8)=MeannoN';
Output(:,9)=stdnoN';
Output(:,10)=Meannon';
Output(:,11)=stdnon';

Output;

xlswrite("Directory"\VaryHandDemoOutput.xls',Output);
```

```
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
%      This is the function of the main program    %
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%

[MeanTV,MeannoN,Meantv,Meannon,stdTV,stdtv,stdnoN,stdnon]=L4SirenModelFinalSA
2(simulation)

%Leslie Matrix with adjustable values for 6 size classes (Greater Siren) and
4 size classes (Lesser Siren)

%Two Species of Sirens
%Parameters that differ between species are denoted in CAPS for Greater
%Siren, and lowercase for lesser siren
%This is intended to be run as a function of
"L4MainProgramVaryHandDemographics.m"

H=load("Directory"\SirenModelh.txt'); %loading H from main model
sTS=load("Directory"\SirenModellstGS.txt'); %loading stS from main model
STs=load("Directory"\SirenModelGStls.txt'); %loading Sts from main model

MeanTV=zeros(1,simulation);
Meantv=zeros(1,simulation);
MeannoN=zeros(1,simulation);
Meannon=zeros(1,simulation);

stdTV=zeros(1,simulation);
stdtv=zeros(1,simulation);
stdnoN=zeros(1,simulation);
stdnon=zeros(1,simulation);

for sim=1:simulation;
  %Parameters to remain constant within a simulation
  h=H(sim,:);
  stS=sTS(sim,:);
  Sts=STs(sim,:);

  runs=100;

  TV=zeros(1,runs);
  tv=zeros(1,runs);
  noN=zeros(1,runs);
  non=zeros(1,runs);

  rundr=zeros(1,runs); %tracking length of droughts

    for rep=1:runs; %This creates replication for each combination of h, Sts,
stS


        years=5000; %number of years to run model
        q=60; %drought threshold, whereas if length is greater than this
number, there is reduced survivorship (Sirens unaffected by short drought -->
Luhring and Todd 2010, Freeman 1958)
```

```
        y=250; %severe drought threshold, whereas if length is greater than
this number there is no recruitment and no growth (no advancing to next size
class) ***y cannot be greater than q!!!***
        fl=0.01; %chance of flood event leading to immigration per time step
        X0=[50;50;50;50;50;50]; %initial age distribution vector of siren in
population
        x0=[50;50;50;50]; %initial age distribution vector of siren in
population

%%text files included in supplementary material
        Par = load("Directory"\GreaterParameters.txt'); %%Greater Siren
Fecundity, Mass, and Survivorship Parameters
        par = load("Directory"\LesserParameters.txt');%%Lesser Siren
Fecundity, Mass, and Survivorship Parameters

        format short g; % makes Matlab use best of fixed or floating point
format

        %%Reserving space for arrays%%
        %imm=zeros(1,years);
        %im=zeros(1,years);

        %Greater Siren%
        X=zeros(6,years); %Reserving memory for (a,t) where a is #age classes
and t is number of time steps
        D=zeros(1,years); %Drought Occurance Matrix
        E=zeros(1,years); %Extinction Matrix
        V=zeros(1,years); %Extinction Event Occurance Matrix
        N=zeros(1,years); %Population size for 2nd through 6th age classes
        G=zeros(1,years); %Modifier for Carrying Capacity
        %Lesser Siren%
        x=zeros(4,years);
        d=zeros(1,years);
        e=zeros(1,years);
        %v=zeros(1,years);
        n=zeros(1,years);
        g=zeros(1,years);

        %%Placing values for starting conditions at t=1%%
        %Greater Siren%
        X(:,1)=X0; %placing the initial age distribution in the the first
column of the matrix X (X(:,1).... means "every row, first column)
        D(:,1)=0;  %no drought
        E(:,1)=0;  %no extinction in first year
        V(:,1)=0;  %no extinction events by first year
        G(:,1)=1; %no carrying capacity effect first year
        %Lesser Siren%
        x(:,1)=x0;
        d(:,1)=0;
        e(:,1)=0;
        v(:,1)=0;
        g(:,1)=1;

        %Greater Siren%
        M=zeros(1,6);   %Mass Values for Age Classes
        DR=zeros(1,6);  %Drought Resistance
```

```
        Pd=zeros(1,6);  %Survival Values For Drought
        P=zeros(1,6);   %Survival Values
        F=zeros(1,6);   %Fecundity Values

        M=normrnd(Par(1:6),Par(7:12));      %Drawing mass for each age class
and corresponding variance
        P=Par(13:18);                       %Drawing survivorship probability
for each age class
        F=normrnd(Par(19:24),Par(25:30));   %Drawing fecundity values for
each age class
        DR=0.0002.*(M.^2)+0.3701*M+145.63;  %Calculationg maximum length of
drought that each size class can survive (dependent on age-specific mass)

        %Lesser Siren%
        m=zeros(1,4);
        dR=zeros(1,4);
        pd=zeros(1,4);
        p=zeros(1,4);
        f=zeros(1,4);

        m=normrnd(par(1:4),par(5:8));
        p=par(9:12);
        f=normrnd(par(13:16),par(17:20));
        dR=0.0002.*(m.^2)+0.3701*m+145.63;

            %%Begin run of year 2 through final year%%
            for t=1:years
            dr(t)=exprnd(h)*(365); %creating a random drought length from a
exponential probability distribution function (most droughts short)

            Pd=(DR>=dr(t)).*P;
            pd=(dR>=dr(t)).*p;

                if dr(t)>145;  %% could this be rewritten as
j(t)=(dr(t)>=145)? (or something to that effect)
                j(t)=0; %this term prevents juveniles in the first age class
from surviving a drought that would kill a full grown 1st year siren (i.e.,
there is no holdover of recruits from the previous year)
                else
                j(t)=1;
                end

                if t==1;
                N(t)=sum(x0);%prevents N(t) from drawing on t=0
                n(t)=sum(x0);
                else
                N(t)=sum(X(2:6,(t-1))); %t-1 is population at previous time
step of all non-larvae
                n(t)=sum(x(2:4,(t-1)));
                end

                im=rand; %creating random variable to use for immigration
                imm(t)=(im<fl)*randi(50); %Creates immigrants to recolonize
wetland (both species equally recruited)
```

```
            B=0.00009;  %Greater Siren Constant for Strength of Density
Dependence
            b=0.00009;  %lesser siren constant for strength of density
dependence
            G(t)=exp(-B*((stS*n(t))+N(t))); %Greater siren modifier on
fecundity resulting from density dependence
            g(t)=exp(-b*((Sts*N(t))+n(t))); %Lesser siren modifier on
fecundity resulting from density dependence

            %%%LESLIE MATRICES%%%

            %%Greater Sirens%%
            %%Normal Leslie%%
            L=[G(t)*F(1) G(t)*F(2) G(t)*F(3) G(t)*F(4) G(t)*F(5) G(t)*F(6);
                P(1) 0 0 0 0 0;
                0 P(2) 0 0 0 0;
                0 0 P(3) 0 0 0;
                0 0 0 P(4) 0 0;
                0 0 0 0 P(5) P(6)]; %each group between ";'s" determines what
the total is for corresponding row in the next column

            %%Leslie Long Drought%%
            Lld=[0 0 0 0 0 0;
                0 Pd(2) 0 0 0 0;
                0 0 Pd(3) 0 0 0;
                0 0 0 Pd(4) 0 0;
                0 0 0 0 Pd(5) 0;
                0 0 0 0 0 Pd(6)]; % Leslie Matrix (Long Drought). P's are
shifted so that they remain in the same row in the subsequent time period (no
growth during drought = no "aging")

            %%Leslie Short Drought%%
            Lsd=[G(t)*F(1)*j(t) G(t)*F(2)*j(t) G(t)*F(3)*j(t) G(t)*F(4)*j(t)
G(t)*F(5)*j(t) G(t)*F(6)*j(t);
                Pd(1) 0 0 0 0 0;
                0 Pd(2) 0 0 0 0;
                0 0 Pd(3) 0 0 0;
                0 0 0 Pd(4) 0 0;
                0 0 0 0 Pd(5) Pd(6)]; %Leslie Matrix (Short Drought).
Progression to next age class

            %%Lesser Sirens%%
            %%Normal Leslie%%
            l=[g(t)*f(1) g(t)*f(2) g(t)*f(3) g(t)*f(4);
                p(1) 0 0 0;
                0 p(2) 0 0;
                0 0 p(3) p(4)]; %each group between ";'s" determines what the
total is for corresponding row in the next column

            %%Leslie Long Drought%%
            lld=[0 0 0 0;
                0 pd(2) 0 0;
                0 0 pd(3) 0;
                0 0 0 pd(4)]; % Leslie Matrix (Long Drought). P's are shifted
so that they remain in the same row in the subsequent time period (no growth
during drought = no "aging")
```

```
            %%Leslie Short Drought%%
            lsd=[g(t)*f(1)*j(t) g(t)*f(2)*j(t) g(t)*f(3)*j(t) g(t)*f(4)*j(t);
                pd(1) 0 0 0;
                0 pd(2) 0 0;
                0 0 pd(3) pd(4)]; %Leslie Matrix (Short Drought). Progression
to next age class

            %%%Drought Duration Determines The Type of Leslie Matrix Used%%%

            %%Greater Siren%%
                if dr(t)<q; %if the drought variable is less than a
threshold, then no drought (this could be looked at as wetland hydroperiod,
rainfall or other factors that cause wetland to go dry)
                X(:,t+1)=floor((L)*X(:,t)+imm(t)); %Normal Leslie Matrix
during non-drought years
                D(:,t)=0; %no drought
                elseif dr(t)>y;
                X(:,t+1)=floor((Lld)*X(:,t)); %LlD = Leslie long drought
matrix
                D(:,t)=2; %long drought
                else
                X(:,t+1)=floor((Lsd)*X(:,t)); %Drought Leslie Matrix during
drought years
                D(:,t)=1; %short drought
                end
            %%Lesser Siren%%
                if dr(t)<q; %if the drought variable is less than a
threshold, then no drought (this could be looked at as wetland hydroperiod,
rainfall or other factors that cause wetland to go dry)
                x(:,t+1)=floor((l)*x(:,t)+imm(t));
                d(:,t)=0;
                elseif dr(t)>y;
                x(:,t+1)=floor((lld)*x(:,t));
                d(:,t)=2;
                else
                x(:,t+1)=floor((lsd)*x(:,t));
                d(:,t)=1;
                end

            %%%Tracking Presence or Absence of Animals%%%

            %%Greater Siren%%
                if sum(X(1:6,t))==0 %if there are no animals
                E(:,t)=1; %put 1 in extinction matrix
                else
                E(:,t)=0; %else put a 0 in extinction matrix
                end
            %%Lesser Siren%%
                if sum(x(1:4,t))==0;
                e(:,t)=1;
                else
                e(:,t)=0;
                end
```

```
            %%%Tracking Extinction Events%%%

            %%Greater Siren%%
                if E(1,t)==1 && E(1,t-1)==0; %if there is a 1 in the
extinction matrix at the current time step (animals extinct) and there was a
0 at the previous time step (animals present)
                V(t)=1; %then there is an extinction event v=1 for that time
                E(1,t)==1 && E(1,t-1)==0;
                else
                V(t)=0; %if not, then v=0 (there was no extinction event)
                end
            %%Lesser Siren%%
                if e(1,t)==1 && e(1,t-1)==0; %if there is a 1 in the
extinction matrix at the current time step (animals extinct) and there was a
0 at the previous time step (animals present)
                v(t)=1; %then there is an extinction event v=1 for that time
                e(1,t)==1 && e(1,t-1)==0;
                else
                v(t)=0; %if not, then v=0 (there was no extinction event)
                end

            end
    %For each rep: Sums total number of extinctions, years in which a
    %population existed in the previous year and is subsequently equal to
zero (populations re-established via colonization/flooding)
    TV(:,rep)=sum(V);
    tv(:,rep)=sum(v);

    %For each rep: Sums all years that population is 0
    noN(:,rep)=sum(E); %within a single x-years period, the number of years
the population is equal to zero
    non(:,rep)=sum(e);

    end
    MeanTV(:,sim)=mean(TV);
    Meantv(:,sim)=mean(tv);
    MeannoN(:,sim)=mean(noN);
    Meannon(:,sim)=mean(non);

    stdTV(:,sim)=std(TV);
    stdtv(:,sim)=std(tv);
    stdnoN(:,sim)=std(noN);
    stdnon(:,sim)=std(non);

end
```

8
33
59
96
2
8.25
14.75
24
0.01
0.2
0.5
0.7
0
150
275
400
0
37.5
68.75
100

8
60
120
215
350
540
2
15
30
53.75
87.5
135
0.01
0.2
0.5
0.7
0.9
0.9
0
0
0
200
300
400
0
0
0
50
75
100

```
clear global;

runs=1000; %each "simulation" is comprised of set of x reps of a x-year
period with identical values for drought and competition
years=5000; %number of years to run each run

[TV,noN,MaxMass,M1,M2,M3,M4,M5,M6,P1,P2,P3,P4,P5,P6,F4,F5,F6]=GSfixedHSA(runs
,years);

Output=zeros(runs,19);%creating matrix for dependent and independent
variables

Output(:,1)=0.2;
Output(:,2)=TV;
Output(:,3)=noN;
Output(:,4)=MaxMass;
Output(:,5)=M1;
Output(:,6)=M2;
Output(:,7)=M3;
Output(:,8)=M4;
Output(:,9)=M5;
Output(:,10)=M6;
Output(:,11)=P1;
Output(:,12)=P2;
Output(:,13)=P3;
Output(:,14)=P4;
Output(:,15)=P5;
Output(:,16)=P6;
Output(:,17)=F4;
Output(:,18)=F5;
Output(:,19)=F6;


Output;

xlswrite("Directory"\GSOutputGlobalSA.xls',Output);
```

```
function
[TV,noN,MaxMass,M1,M2,M3,M4,M5,M6,P1,P2,P3,P4,P5,P6,F4,F5,F6]=GSfixedHSA(runs
,years)

%Leslie Matrix with adjustable values for 6 age classes
%Greater Sirens

   TV=zeros(runs,1);
   noN=zeros(runs,1);
   dr=zeros(runs,years);
   imm=zeros(runs,years);
   MaxMass=zeros(runs,1);
   M1=zeros(runs,1);
   M2=zeros(runs,1);
   M3=zeros(runs,1);
   M4=zeros(runs,1);
   M5=zeros(runs,1);
   M6=zeros(runs,1);
   P1=zeros(runs,1);
   P2=zeros(runs,1);
   P3=zeros(runs,1);
   P4=zeros(runs,1);
   P5=zeros(runs,1);
   P6=zeros(runs,1);
   F4=zeros(runs,1);
   F5=zeros(runs,1);
   F6=zeros(runs,1);


    for rep=1:runs; %Running Greater Sirens with fixed h
        h=0.2; %mean drought severity
        q=60; %drought threshold, whereas if length is greater than this
number, there is reduced survivorship (Sirens unaffected by short drought -->
Luhring and Todd 2010, Freeman 1958)
        y=250; %severe drought threshold, whereas if length is greater than
this number there is no recruitment and no growth (no advancing to next size
class) ***y cannot be greater than q!!!***
        fl=0.01; %chance of flood event leading to immigration per time step
        x0=[50;50;50;50;50;50]; %initial age distribution vector of siren in
population

        Par = load("Directory"\GSSA.txt'); %%Greater Siren Fecundity, Mass,
and Survivorship Parameters

        format short g; % makes Matlab use best of fixed or floating point
format

        %%Reserving space for arrays%%

        %Greater Siren%
        X=zeros(6,years); %Reserving memory for (a,t) where a is #age classes
and t is number of time steps
        D=zeros(1,years); %Drought Occurance Matrix
        E=zeros(1,years); %Extinction Matrix
```

```
        V=zeros(1,years); %Extinction Event Occurance Matrix
        N=zeros(1,years); %Population size for 2nd through 6th age classes
        G=zeros(1,years); %Modifier for Carrying Capacity

        %%Placing values for starting conditions at t=1%%
        %Greater Siren%
        X(:,1)=x0; %placing the initial age distribution in the the first
column of the matrix X (X(:,1).... means "every row, first column)
        D(:,1)=0;  %no drought
        E(:,1)=0;  %no extinction in first year
        V(:,1)=0;  %no extinction events by first year
        G(:,1)=1; %no carrying capacity effect first year

        %Greater Siren%
        M=zeros(1,6);   %Mass Values for Age Classes
        DR=zeros(1,6);  %Drought Resistance
        Pd=zeros(1,6);  %Survival Values For Drought
        P=zeros(1,6);   %Survival Values
        F=zeros(1,6);   %Fecundity Values

        M=normrnd(Par(1:6),Par(7:12));      %Drawing mass for each age class
and corresponding variance
        P=normrnd(Par(13:18),Par(19:24));         %Drawing survivorship
probability for each age class
        F=normrnd(Par(25:30),Par(31:36));                      %Drawing
fecundity values for each age class
        DR=0.0002.*(M.^2)+0.3701*M+145.63;  %Calculationg maximum length of
drought that each size class can survive (dependent on age-specific mass)
        MaxMass(rep)=max(M);

        M1(rep)=M(1,1);
        M2(rep)=M(2,1);
        M3(rep)=M(3,1);
        M4(rep)=M(4,1);
        M5(rep)=M(5,1);
        M6(rep)=M(6,1);
        P1(rep)=P(1,1);
        P2(rep)=P(2,1);
        P3(rep)=P(3,1);
        P4(rep)=P(4,1);
        P5(rep)=P(5,1);
        P6(rep)=P(6,1);
        F4(rep)=F(4,1); %F1 - F3 are all 0
        F5(rep)=F(5,1);
        F6(rep)=F(6,1);

        %%Begin run of year 2 through final year%%
            for t=1:years
            dr(t)=exprnd(h)*(365); %creating a random drought length from a
exponential probability distribution function (most droughts short)

            Pd=(DR>=dr(t)).*P;

                if dr(t)>145;  %% could this be rewritten as
j(t)=(dr(t)>=145)? (or something to that effect)
```

```
                j(t)=0; %this term prevents juveniles in the first age class
from surviving a drought that would kill a full grown 1st year siren (i.e.,
there is no holdover of recruits from the previous year)
                else
                j(t)=1;
                end

                if t==1;
                N(t)=sum(x0);%prevents N(t) from drawing on t=0

                else
                N(t)=sum(X(2:6,(t-1))); %t-1 is population at previous time
step of all non-larvae

                end

                im=rand; %creating random variable to use for immigration
                imm(t)=(im<fl)*randi(50); %Creates immigrants to recolonize
wetland (both species equally recruited)

            B=0.00009;  %Greater Siren Constant for Strength of Density
Dependence

            G(t)=exp(-B*(N(t))); %Greater siren modifier on fecundity
resulting from density dependence

            %%%LESLIE MATRICES%%%

            %%Greater Sirens%%
            %%Normal Leslie%%
            L=[G(t)*F(1) G(t)*F(2) G(t)*F(3) G(t)*F(4) G(t)*F(5) G(t)*F(6);
P(1) 0 0 0 0 0; 0 P(2) 0 0 0 0; 0 0 P(3) 0 0 0; 0 0 0 P(4) 0 0; 0 0 0 0 P(5)
P(6)]; %each group between ";'s" determines what the total is for
corresponding row in the next column
            %%Leslie Long Drought%%
            Lld=[0 0 0 0 0 0; 0 Pd(2) 0 0 0 0; 0 0 Pd(3) 0 0 0; 0 0 0 Pd(4) 0
0; 0 0 0 0 Pd(5) 0; 0 0 0 0 0 Pd(6)]; % Leslie Matrix (Long Drought). P's are
shifted so that they remain in the same row in the subsequent time period (no
growth during drought = no "aging")
            %%Leslie Short Drought%%
            Lsd=[G(t)*F(1)*j(t) G(t)*F(2)*j(t) G(t)*F(3)*j(t) G(t)*F(4)*j(t)
G(t)*F(5)*j(t) G(t)*F(6)*j(t); Pd(1) 0 0 0 0 0; 0 Pd(2) 0 0 0 0; 0 0 Pd(3) 0
0 0; 0 0 0 Pd(4) 0 0; 0 0 0 0 Pd(5) Pd(6)]; %Leslie Matrix (Short Drought).
Progression to next age class

            %%%Drought Duration Determines The Type of Leslie Matrix Used%%%

            %%Greater Siren%%
                if dr(t)<q; %if the drought variable is less than a
threshold, then no drought (this could be looked at as wetland hydroperiod,
rainfall or other factors that cause wetland to go dry)
                X(:,t+1)=floor((L)*X(:,t)+imm(t)); %Normal Leslie Matrix
during non-drought years
                D(:,t)=0; %no drought
                elseif dr(t)>y;
```

```
                X(:,t+1)=floor((Lld)*X(:,t)); %LlD = Leslie long drought
matrix
                D(:,t)=2; %long drought
                else
                X(:,t+1)=floor((Lsd)*X(:,t)); %Drought Leslie Matrix during
drought years
                D(:,t)=1; %short drought
                end

            %%%Tracking Presence or Absence of Animals%%%

            %%Greater Siren%%
                if sum(X(1:6,t))==0 %if there are no animals
                E(:,t)=1; %put 1 in extinction matrix
                else
                E(:,t)=0; %else put a 0 in extinction matrix
                end

            %%%Tracking Extinction Events%%%

            %%Greater Siren%%
                if E(1,t)==1 && E(1,t-1)==0; %if there is a 1 in the
extinction matrix at the current time step (animals extinct) and there was a
0 at the previous time step (animals present)
                V(t)=1; %then there is an extinction event v=1 for that time
                E(1,t)==1 && E(1,t-1)==0;
                else
                V(t)=0; %if not, then v=0 (there was no extinction event)
                end

            end
    %For each rep: Sums total number of extinctions, years in which a
    %population existed in the previous year and is subsequently equal to
zero (populations re-established via colonization/flooding)
    TV(rep,:)=sum(V);

    %For each rep: Sums all years that population is 0
    noN(rep,:)=sum(E); %within a single x-years period, the number of years
the population is equal to zero

    end


end
```

8
60
120
215
350
540
2
15
30
53.75
87.5
135
0.01
0.2
0.5
0.7
0.9
0.9
0.001
0.01
0.025
0.035
0.040
0.040
0
0
0
200
300
400
0
0
0
50
75
100

```
rm(list=ls())
gc()
setwd("Directory")
GSSA <- read.delim("GSSAOutput.txt", header=TRUE);
out <- rep(0,15*2)
dim(out) <- c(15,2)
X <- rep(0,861*15)
dim(X) <- c(861,15)
for (i in 1:15){
  X[,i] <- GSSA[[i+4]]
}
meanY <- mean(GSSA$Persist) # Y values
for (i in 1:15){ # iterate through X values
  fitY <- lm(GSSA$Persist ~ X[,i])
  out[i,1] <- summary(fitY)$adj.r.squared # calculate R-squared
  slope <- summary(fitY)$coefficients[2,1]
  interc <- summary(fitY)$coefficients[1,1]
  x1 <- mean(X[,i])-sd(X[,i])
  x2 <- mean(X[,i])+sd(X[,i])
  y1 <- interc+slope*x1
  y2 <- interc+slope*x2
  out[i,2] <-abs(y1-y2)*100/meanY # calculate Si
}
write.table(out, file = "GSSA_regressionsPersist.txt", row.names=FALSE, sep =
"\t")
```

```
clear global;

runs=1000; %each "simulation" is comprised of set of x reps of a x-year
period with identical values for drought and competition
years=5000; %number of years to run each run

[tv,non,MaxMass,m1,m2,m3,m4,p1,p2,p3,p4,f2,f3,f4]=LSfixedHSA(runs,years);

Output=zeros(runs,15);%creating matrix for dependent and independent
variables

Output(:,1)=0.1;
Output(:,2)=tv;
Output(:,3)=non;
Output(:,4)=MaxMass;
Output(:,5)=m1;
Output(:,6)=m2;
Output(:,7)=m3;
Output(:,8)=m4;
Output(:,9)=p1;
Output(:,10)=p2;
Output(:,11)=p3;
Output(:,12)=p4;
Output(:,13)=f2;
Output(:,14)=f3;
Output(:,15)=f4;


Output;

xlswrite("Directory"\LSOutputGlobalSA.xls',Output);
```

```
function
[tv,non,MaxMass,m1,m2,m3,m4,p1,p2,p3,p4,f2,f3,f4]=LSfixedHSA(runs,years)

%Leslie Matrix with adjustable values for 4 age classes
%Lesser Sirens

   tv=zeros(runs,1);
   non=zeros(runs,1);
   dr=zeros(runs,years);
   imm=zeros(runs,years);
   MaxMass=zeros(runs,1);
   m1=zeros(runs,1);
   m2=zeros(runs,1);
   m3=zeros(runs,1);
   m4=zeros(runs,1);
   p1=zeros(runs,1);
   p2=zeros(runs,1);
   p3=zeros(runs,1);
   p4=zeros(runs,1);
   f2=zeros(runs,1);
   f3=zeros(runs,1);
   f4=zeros(runs,1);


    for rep=1:runs; %Running Lesser Sirens with fixed h
        h=0.1; %mean drought severity
        q=60; %drought threshold, whereas if length is greater than this
number, there is reduced survivorship (Sirens unaffected by short drought -->
Luhring and Todd 2010, Freeman 1958)
        y=250; %severe drought threshold, whereas if length is greater than
this number there is no recruitment and no growth (no advancing to next size
class) ***y cannot be greater than q!!!***
        fl=0.01; %chance of flood event leading to immigration per time step
        x0=[50;50;50;50]; %initial age distribution vector of siren in
population

        Par = load("Directory"\LSSA.txt'); %%Lesser Siren Fecundity, Mass,
and Survivorship Parameters

        format short g; % makes Matlab use best of fixed or floating point
format

        %%Reserving space for arrays%%

        %Lesser Siren%
        x=zeros(4,years); %Reserving memory for (a,t) where a is #age classes
and t is number of time steps
        d=zeros(1,years); %Drought Occurance Matrix
        e=zeros(1,years); %Extinction Matrix
        v=zeros(1,years); %Extinction Event Occurance Matrix
        n=zeros(1,years); %Population size for 2nd through 6th age classes
        g=zeros(1,years); %Modifier for Carrying Capacity

        %%Placing values for starting conditions at t=1%%
```

```
        %Lesser Siren%
        x(:,1)=x0; %placing the initial age distribution in the the first
column of the matrix X (X(:,1).... means "every row, first column)
        d(:,1)=0;  %no drought
        e(:,1)=0;  %no extinction in first year
        v(:,1)=0;  %no extinction events by first year
        g(:,1)=1; %no carrying capacity effect first year

        %Greater Siren%
        m=zeros(1,4);   %Mass Values for Age Classes
        dr=zeros(1,4);  %Drought Resistance
        pd=zeros(1,4);  %Survival Values For Drought
        p=zeros(1,4);   %Survival Values
        f=zeros(1,4);   %Fecundity Values

        m=normrnd(Par(1:4),Par(5:8));       %Drawing mass for each age class
and corresponding variance
        p=normrnd(Par(9:12),Par(13:16));    %Drawing survivorship probability
for each age class
        f=normrnd(Par(17:20),Par(21:24));   %Drawing fecundity values for
each age class
        dR=0.0002.*(m.^2)+0.3701*m+145.63;  %Calculationg maximum length of
drought that each size class can survive (dependent on age-specific mass)
        MaxMass(rep)=max(m);

        m1(rep)=m(1,1);
        m2(rep)=m(2,1);
        m3(rep)=m(3,1);
        m4(rep)=m(4,1);
        p1(rep)=p(1,1);
        p2(rep)=p(2,1);
        p3(rep)=p(3,1);
        p4(rep)=p(4,1);
        f2(rep)=f(2,1); %f1 is 0
        f3(rep)=f(3,1);
        f4(rep)=f(4,1);

        %%Begin run of year 2 through final year%%
            for t=1:years
            dr(t)=exprnd(h)*(365); %creating a random drought length from a
exponential probability distribution function (most droughts short)

            pd=(dR>=dr(t)).*p;

                if dr(t)>145;
                j(t)=0; %this term prevents juveniles in the first age class
from surviving a drought that would kill a full grown 1st year siren (i.e.,
there is no holdover of recruits from the previous year)
                else
                j(t)=1;
                end

                if t==1;
                n(t)=sum(x0); %prevents n(t) from drawing on t=0
```

```
                else
                n(t)=sum(x(2:4,(t-1))); %t-1 is population at previous time
step of all non-larvae

                end

                im=rand; %creating random variable to use for immigration
                imm(t)=(im<fl)*randi(50); %Creates immigrants to recolonize
wetland (both species equally recruited)

            B=0.00009;  %Lesser Siren Constant for Strength of Density
Dependence

            g(t)=exp(-B*(n(t))); %Greater siren modifier on fecundity
resulting from density dependence

            %%%LESLIE MATRICES%%%

           %%Lesser Sirens%%
            %%Normal Leslie%%
            l=[g(t)*f(1) g(t)*f(2) g(t)*f(3) g(t)*f(4);
                p(1) 0 0 0;
                0 p(2) 0 0;
                0 0 p(3) p(4)]; %each group between ";'s" determines what the
total is for corresponding row in the next column

            %%Leslie Long Drought%%
            lld=[0 0 0 0;
                0 pd(2) 0 0;
                0 0 pd(3) 0;
                0 0 0 pd(4)]; % Leslie Matrix (Long Drought). P's are shifted
so that they remain in the same row in the subsequent time period (no growth
during drought = no "aging")


            %%Leslie Short Drought%%
            lsd=[g(t)*f(1)*j(t) g(t)*f(2)*j(t) g(t)*f(3)*j(t) g(t)*f(4)*j(t);
                pd(1) 0 0 0;
                0 pd(2) 0 0;
                0 0 pd(3) pd(4)]; %Leslie Matrix (Short Drought). Progression
to next age class

            %%%Drought Duration Determines The Type of Leslie Matrix Used%%%

            %%Lesser Siren%%
                if dr(t)<q; %if the drought variable is less than a
threshold, then no drought (this could be looked at as wetland hydroperiod,
rainfall or other factors that cause wetland to go dry)
                x(:,t+1)=floor((l)*x(:,t)+imm(t));
                d(:,t)=0;
                elseif dr(t)>y;
                x(:,t+1)=floor((lld)*x(:,t));
                d(:,t)=2;
                else
                x(:,t+1)=floor((lsd)*x(:,t));
```

```
                d(:,t)=1;
                end

            %%%Tracking Presence or Absence of Animals%%%

           %%Lesser Siren%%
                if sum(x(1:4,t))==0;
                e(:,t)=1;
                else
                e(:,t)=0;
                end

            %%%Tracking Extinction Events%%%

           %%Lesser Siren%%
                if e(1,t)==1 && e(1,t-1)==0; %if there is a 1 in the
extinction matrix at the current time step (animals extinct) and there was a
0 at the previous time step (animals present)
                v(t)=1; %then there is an extinction event v=1 for that time
                e(1,t)==1 && e(1,t-1)==0;
                else
                v(t)=0; %if not, then v=0 (there was no extinction event)
                end

            end
    %For each rep: Sums total number of extinctions, years in which a
    %population existed in the previous year and is subsequently equal to
zero (populations re-established via colonization/flooding)
    tv(rep,:)=sum(v);

    %For each rep: Sums all years that population is 0
    non(rep,:)=sum(e); %within a single x-years period, the number of years
the population is equal to zero

    end


end
```

```
rm(list=ls())
gc()
setwd("Directory")
LSSA <- read.delim("LSSAOutput.txt", header=TRUE);
out <- rep(0,11*2)
dim(out) <- c(11,2)
X <- rep(0,915*11)
dim(X) <- c(915,11)
for (i in 1:11){
  X[,i] <- LSSA[[i+4]]
}
meanY <- mean(LSSA$persist) # Y values
for (i in 1:11){ # iterate through X values
  fitY <- lm(LSSA$persist ~ X[,i])
  out[i,1] <- summary(fitY)$adj.r.squared # calculate R-squared
  slope <- summary(fitY)$coefficients[2,1]
  interc <- summary(fitY)$coefficients[1,1]
  x1 <- mean(X[,i])-sd(X[,i])
  x2 <- mean(X[,i])+sd(X[,i])
  y1 <- interc+slope*x1
  y2 <- interc+slope*x2
  out[i,2] <-abs(y1-y2)*100/meanY # calculate Si
}
write.table(out, file = "LSSA_regressionsPersist.txt", row.names=FALSE, sep =
"\t")
```