APPENDIX I: Used MatLab code

The same code was used for three groups of species, with exception of the species name which was named in “yNAME”.

```
yName = 'DIATOMO';
%yName = 'FILAMENT';
%yName = 'MACROVEG';

% fish species end variables are
%   GACMAX40    GACMIN40    PUNG    WHIBIN  VENBIN


% download data
% ----------------------------
data = importdata('data/majorlarvae_4matlab.txt');
rasterCovarNames = {'SHOPROFILE' 'DEPTH' 'BOTTOM' 'BOTTOMCOV'
'FETCH300W' 'FE300ME' ...
    'DIST20M' 'LINED' 'ISLANDNUM' 'DIS_SAND' 'DIS_SHALLO' 'PE900'
'PE3000' 'SAL09SPRS' ...  %'SAL10SPRS'
    'SAL910WIN' 'ICELAST09' 'ICEWIN09' 'EUTSTAT' ...   %'ICELAST10'
'ICEFIRST09' 'ICEFIRST10' 'ICEWIN10'
    'EKOSTAT' 'PHOSP' 'NITROG' 'CHL_A' 'SECCHI' 'RIVERS' 'BOTTOMCLS'
'SHAREA'};
spatNames = {'N_etrs89' 'E_etrs89'};

% --- TAKE x -----
% Raster variables
indRaster = false(1,size(data.data,2));
for i1=1:length(rasterCovarNames)
    indRaster(strcmp(data.textdata,rasterCovarNames{i1})) = true;
end
% Spatial coordinates
indSpat = false(1,size(data.data,2));
for i1=1:length(spatNames)
    indSpat(strcmp(data.textdata,spatNames{i1})) = true;
end
% Take environmental variables
% order [Raster Spatial ]
x = [data.data(:,indRaster) data.data(:,indSpat)];
xlabels = [data.textdata(indRaster) data.textdata(indSpat)];

% rescale the covariates
standind = 1:size(x,2)-2;      % columns of x to be standardized (all
other except co-ordinates)
mx = mean(x(:,standind)); stdx = std(x(:,standind));
x(:,standind) = (x(:,standind)-
repmat(mx,size(x,1),1))./repmat(stdx,size(x,1),1);
% scale co-ordinates to kilometers
x(:,end-1:end) = x(:,end-1:end)./1000;

% ------ TAKE y and Z -------------
y = data.data(:, strcmp(data.textdata,yName));     % the number of
count locations where species is precent
z = data.data(:, strcmp(data.textdata,'TOTAL_ED'));     % the total
number of count locations

% --------- define model and inference method-----
lik = lik_binomial;     % probit likelihood
```

```
cfc = gpcf_constant('constSigma2', 10, 'constSigma2_prior',
prior_fixed);          % The covariance function of the intercept term
cfs = gpcf_exp('magnSigma2', 2, 'lengthScale', [3 6],
'selectedVariables', size(x,2)-[1 0]  ,...
    'lengthScale_prior', prior_t('s2',100));
% The covariance function of the spatial component
cf_nn = gpcf_neuralnetwork('selectedVariables', 1:size(x,2)-2, ...
% The neural network covariance function for
    'weightSigma2', 10*ones(1,size(x,2)-2));
%          the nonlinear effect of covariates
%%%cf_se = gpcf_sexp('selectedVariables', 1:size(x,2)-2, ...
% The squared exponential covariance function for
%%%    'lengthScale', ones(1,size(x,2)-2));
%          the nonlinear effect of covariates
cf_li = gpcf_linear('selectedVariables', 1:size(x,2)-2);
% covariance function corresponding to linear model of covariates

% use neural network covariance, that is non-linear response of
covariates
gp = gp_set('lik', lik, 'cf', {cfc cf_nn cfs}, 'latent_method', 'EP');
% use squared exponential covariance, that is non-linear response of
covariates
%gp = gp_set('lik', lik, 'cf', {cfc cf_se cfs}, 'latent_method',
'EP');
% use linear covariance function, that is linear response of
covariates
%gp = gp_set('lik', lik, 'cf', {cfc cf_li cfs}, 'latent_method',
'EP');

% Infer the model
% optimize the covariance function parameters and find Gaussian
% approximation (EP) for the latent variables
opt=optimset('TolFun',1e-3,'TolX',1e-3,'Display','iter');   % Options
for optimization
gp=gp_optim(gp,x,y, 'z', z,'opt',opt);           % run optimization

%%
apcs = gp_avpredcomp(gp,x,y,'z',z)

figure; hold on
for i1=1:size(apcs.fs,2)
    prc = prctile(apcs.fsa(:,i1),[5 50 95]);
    plot([prc(1) prc(3)], [i1 i1])
    plot(prc(2),i1, 'kx', 'linewidth', 5)
end
plot([0 0],[0 25])
set(gca,'yTick',1:28,'yticklabel',xlabels)
%% Analyze results

% in case of linear model you can calculate the posterior of beta
% parameters
if strcmp(gp.cf{2},  'gpcf_linear')
    Efl =[]; Varfl = [];
    fprintf(' The posterior mean and 95%% credible intervals of
betas:\n')
    for i1 = 1:size(x,2)-2
        xtmp = zeros(1,size(x,2)); xtmp(i1)=1;
        [Eft, Vart] = gp_pred(gp,x,y, 'z',z, xtmp, 'predcf', 2);
        Efl(i1) = Eft; Varfl(i1) = Vart;
```

```
        fprintf('   %s: \t %.2f (%.2f, %.2f) \n', xlabels{i1}, Eft,
Eft-2*sqrt(Vart), Eft+2*sqrt(Vart))
    end
end
figure, hold on
set(gcf,'units','centimeters');
set(gcf,'defaultaxesfontsize',10)   %6 8
set(gcf,'defaulttextfontsize',10)   %6 8)
set(gcf,'defaultlinelinewidth',1.5)

% ind defines the locations at which we look at the response curve
ind = resampdet(ones(size(x,1),1),100,1);
npoints=20;
for i1=1:size(x,2)-2
    x_i1 = [linspace(min(x(:,i1)),max(x(:,i1)),npoints)' ; 0];
    xtemp = x(ind,:);
    xtemp = reshape( repmat(xtemp(:)',npoints+1,1),
(npoints+1)*length(ind),size(x,2));
    xtemp(:,i1) = repmat(x_i1,length(ind),1);
    xtemp(npoints+1:npoints+1:size(xtemp,1),i1) = x(ind,i1);
    %xtemp(npoints+1:npoints+1:size(xtemp,1),i1) =
x_i1(floor(npoints/2));
     [Ef,Varf] = gp_jpred(gp,x,y,xtemp, 'z',z,'predcf',
2:length(gp.cf)-1);
    Efmat = reshape(Ef,npoints+1,length(ind));
    %              Efmat = Efmat(1:npoints,:) - repmat(
Efmat(npoints+1,:), npoints,1);
    %              Efmat = Efmat - repmat( mean(Efmat,1), npoints,1);
    Efmat = Efmat(1:npoints,:);
    Efmat = Efmat - repmat( mean(Efmat,1), npoints,1);
    EF = mean(Efmat,2);
    Varf2=[];
    for v1=1:length(ind)
        vind = (1:npoints+1) + (v1-1)*(npoints+1);
        vartemp = Varf(vind,vind);
        %Varf2(:,v1) = diag(vartemp) + vartemp(end,end) -
2*vartemp(end,:)';
        wa=ones(length(x_i1),1)./npoints;wa(end)=0;
        Varf2(:,v1) = diag(vartemp(1:end,1:end)) + wa'*vartemp*wa -
2*vartemp(1:end,:)*wa;
    end
    Varf2=Varf2(1:end-1,:);
    VARF = mean(Varf2,2) + var(Efmat,[],2);
    x_i1=x_i1(1:end-1);

    %subplot(3,3,i1), hold on
    sp(i1) = subplot(ceil(sqrt(size(x,2)-2)),ceil(sqrt(size(x,2)-
2)),i1); hold on
    plot(repmat(x_i1*stdx(i1)+mx(i1),1,length(ind)), Efmat, 'k',
'color', [0.7 0.7 0.7], 'linewidth', 0.5), hold on
    plot(x_i1*stdx(i1)+mx(i1), EF, 'k')
    plot(x_i1*stdx(i1)+mx(i1), EF-2*sqrt(VARF), 'k--'), hold on
    plot(x_i1*stdx(i1)+mx(i1), EF+2*sqrt(VARF), 'k--'), hold on
    xlabel(xlabels{i1})
    xlim([min(x_i1*stdx(i1)+mx(i1)) max(x_i1*stdx(i1)+mx(i1))])
    ylim([min(EF-2*sqrt(VARF)) max(EF+2*sqrt(VARF))])
    grid on
    %     % define the prediction covariates (xtemp) so that one
covariate (i1)
%     % changes at ind locations and all other covariates are kept
constant.
```

```
%     x_i1 = linspace(min(x(:,i1)),max(x(:,i1)),npoints)';
%     xtemp = x(ind,:);
%     xtemp = reshape( repmat(xtemp(:)',npoints,1),
npoints*length(ind),size(x,2));
%     xtemp(:,i1) = repmat(x_i1,length(ind),1);
%
%     % Predict the response along the covariate i1
%     [ef,varf] = gp_pred(gp,x,y,xtemp, 'predcf', 2:length(gp.cf)-1);
%
%     % Reshape ef and substract the mean at each prediction point
%     efmat = reshape(ef,npoints,length(ind));
%     efmat = efmat - repmat( mean(efmat,1), npoints,1);
%     ef = mean(efmat,2);
%     varf = mean(reshape(varf,npoints,length(ind)),2) +
var(efmat,[],2);
%     %varf = var(efmat,[],2);
%
%     %subplot(3,3,i1), hold on
%     subplot(ceil(sqrt(size(x,2)-2)),ceil(sqrt(size(x,2)-2)),i1),
hold on
%     plot(repmat(x_i1*stdx(i1)+mx(i1),1,length(ind)), efmat, 'k',
'color', [0.7 0.7 0.7], 'linewidth', 0.5), hold on
%     plot(x_i1*stdx(i1)+mx(i1), ef, 'k')
%     plot(x_i1*stdx(i1)+mx(i1), ef-2*sqrt(varf), 'k--'), hold on
%     plot(x_i1*stdx(i1)+mx(i1), ef+2*sqrt(varf), 'k--'), hold on
%     xlabel(xlabels{i1})
%     xlim([min(x_i1*stdx(i1)+mx(i1)) max(x_i1*stdx(i1)+mx(i1))])
%     ylim([min(ef-2*sqrt(varf)) max(ef+2*sqrt(varf))])
%     grid on
%     if i1 == 1
%         text(1.5*max(x_i1*stdx(i1)+mx(i1)),1.2*max(ef+2*sqrt(varf)),
['The species modeled:' yName])
%     end
end
ylimits = [];
for i1=1:length(sp)
    ylimits(i1,:) = get(sp(i1),'ylim');
end
for i1=1:length(sp)
    set(sp(i1),'ylim', [min(ylimits(:,1)) max(ylimits(:,2))]);
end
subplot(ceil(sqrt(size(x,2)-2)),ceil(sqrt(size(x,2)-2)),2)
text(min(x(:,2)*stdx(2)+mx(2)),1.5*max(ylimits(:,2)), ['The species
modeled:' yName])

set(gcf, 'pos', [0 0 25 25])
set(gcf, 'paperposition', get(gcf,'pos'))


%% Train model without in-situ variables
% Define the name of end variable (Species name)
% Most interesting options are:
%    FILAMENT%, DIATOMO%, MACROVEG (charop, phaeoph)
yName = 'DIATOMO';
%yName = 'FILAMENT'

% fish species end variables are
%   GACMAX40    GACMIN40    PUNG    WHIBIN   VENBIN
```

```
% download data
% ----------------------------
data = importdata('data/majorlarvae_4matlab.txt');
rasterCovarNames = {'FETCH300W' 'FE300ME' 'DIST20M' 'LINED'
'ISLANDNUM' 'DIS_SAND' ...
    'DIS_SHALLO' 'PE900' 'PE3000' 'SAL09SPRS' 'SAL910WIN' 'ICELAST09'
'ICEWIN09' 'EUTSTAT' ...
    'EKOSTAT' 'PHOSP' 'NITROG' 'CHL_A' 'SECCHI' 'RIVERS' 'BOTTOMCLS'
'SHAREA'};
spatNames = {'N_etrs89' 'E_etrs89'};

% --- TAKE x -----
% Raster variables
indRaster = false(1,size(data.data,2));
for i1=1:length(rasterCovarNames)
    indRaster(strcmp(data.textdata,rasterCovarNames{i1})) = true;
end
% Spatial coordinates
indSpat = false(1,size(data.data,2));
for i1=1:length(spatNames)
    indSpat(strcmp(data.textdata,spatNames{i1})) = true;
end
% Take environmental variables
% order [Raster Spatial ]
x = [data.data(:,indRaster) data.data(:,indSpat)];
xlabels = [data.textdata(indRaster) data.textdata(indSpat)];

% rescale the covariates
standind = 1:size(x,2)-2;      % columns of x to be standardized (all
other except co-ordinates)
mx = mean(x(:,standind)); stdx = std(x(:,standind));
x(:,standind) = (x(:,standind)-
repmat(mx,size(x,1),1))./repmat(stdx,size(x,1),1);
% scale co-ordinates to kilometers
x(:,end-1:end) = x(:,end-1:end)./1000;

% ------ TAKE y -------------
% Try different tresholds (e.g. under 5%, 10% coverage is zero)
y = data.data(:, strcmp(data.textdata,yName));
y = 2*(y>0)-1;     % -1 = absence, 1 = presence

% --------- define model and inference method-----
lik = lik_probit;      % probit likelihood
cfc = gpcf_constant('constSigma2', 10, 'constSigma2_prior',
prior_fixed);           % The intercept term
cfs = gpcf_exp('magnSigma2', 2, 'lengthScale', [3 6],
'selectedVariables', size(x,2)-[1 0]  ,...
    'lengthScale_prior', prior_t('s2',100));
% The spatial covariance function
cf_nn = gpcf_neuralnetwork('selectedVariables', 1:size(x,2)-2, ...
% The nonlinear effect of covariates
    'weightSigma2', 10*ones(1,size(x,2)-2));
gp = gp_set('lik', lik, 'cf', {cfc cf_nn cfs}, 'latent_method', 'EP');

% Infer the model
% optimize the covariance function parameters and find Gaussian
% approximation (EP) for the latent variables
opt=optimset('TolFun',1e-3,'TolX',1e-3,'Display','iter');    % Options
for optimization
gp=gp_optim(gp,x,y,'z',z,'opt',opt);            % run optimization
```

```
%% Do the prediction
data_pred = importdata('data/G300_mod.txt');

% Rename few columns in the prediction data so that column names match
with
% training data column names
data_pred.textdata{strcmp(data_pred.textdata,'FE300W')} ='FETCH300W';
data_pred.textdata{strcmp(data_pred.textdata,'SAL09SPR')} =
'SAL09SPRS';
data_pred.textdata{strcmp(data_pred.textdata,'X')} ='E_etrs89';
data_pred.textdata{strcmp(data_pred.textdata,'Y')} = 'N_etrs89';
% pick up the right columns in right order from the prediction data
for i1=1:length(xlabels)
    otaInd(i1) = find(strcmp(data_pred.textdata,xlabels{i1}));
end
% create the prediction inputs
x_pred = data_pred.data(:,otaInd);
% rescale the covariates
standind = 1:size(x,2)-2;      % columns of x to be standardized (all
other except co-ordinates)
x_pred(:,standind) = (x_pred(:,standind)-
repmat(mx,size(x_pred,1),1))./repmat(stdx,size(x_pred,1),1);
% scale co-ordinates to kilometers
x_pred(:,end-1:end) = x_pred(:,end-1:end)./1000;
[Ef,Varf,lpy] =
gp_pred(gp,x,y,x_pred,'z',z,'yt',ones(size(x_pred,1),1),'zt',ones(size
(x_pred,1),1));

% Write the prediction into the file
pred = [x_pred(:,end).*1000 x_pred(:,end-1).*1000 exp(lpy)];   % scale
to intensity per 100m^3

fid = fopen([yName '_prediction.txt'],'w');
fprintf(fid, 'X\tY\tZ\n');
for tmpi = 1:size(pred,1)
    fprintf(fid, '%f\t%f\t%f\n',
pred(tmpi,1),pred(tmpi,2),pred(tmpi,3));
end
fclose(fid);
%save([yName '_prediction.txt'], 'pred', '-ascii', '-double', '-tabs')
```

APPENDIX I: Response curves for all the variables

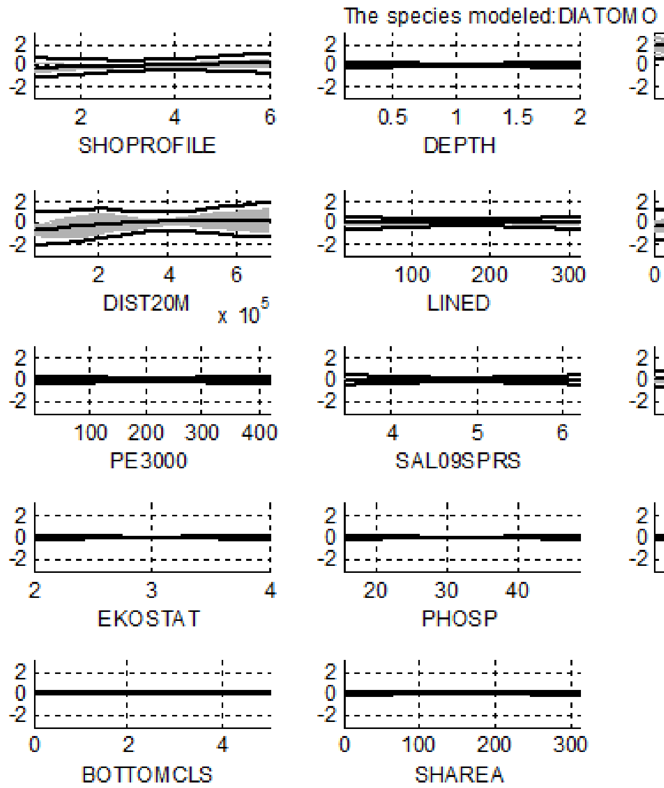

Figure 10: DIATOM response curves of all the variables.

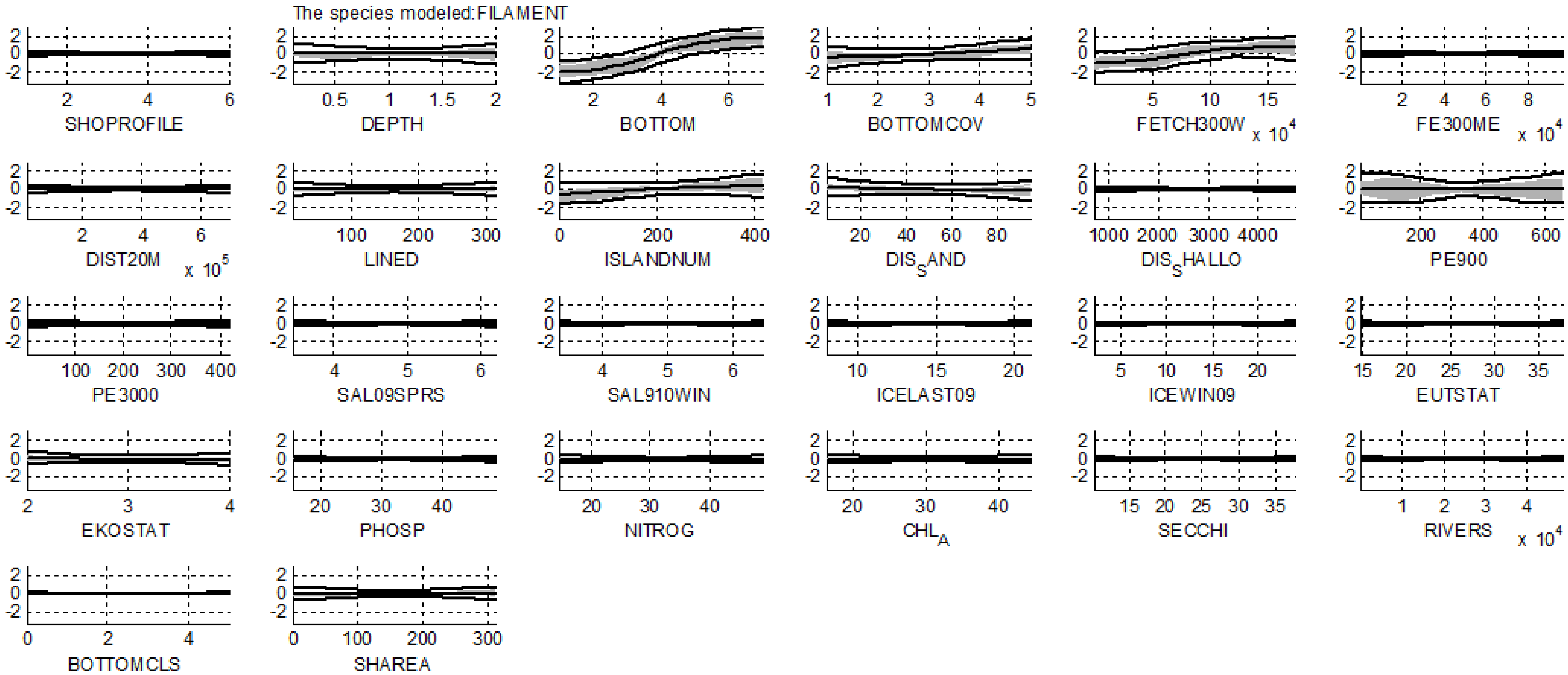

Figure 11: FILAMENT response curves of all the variables.

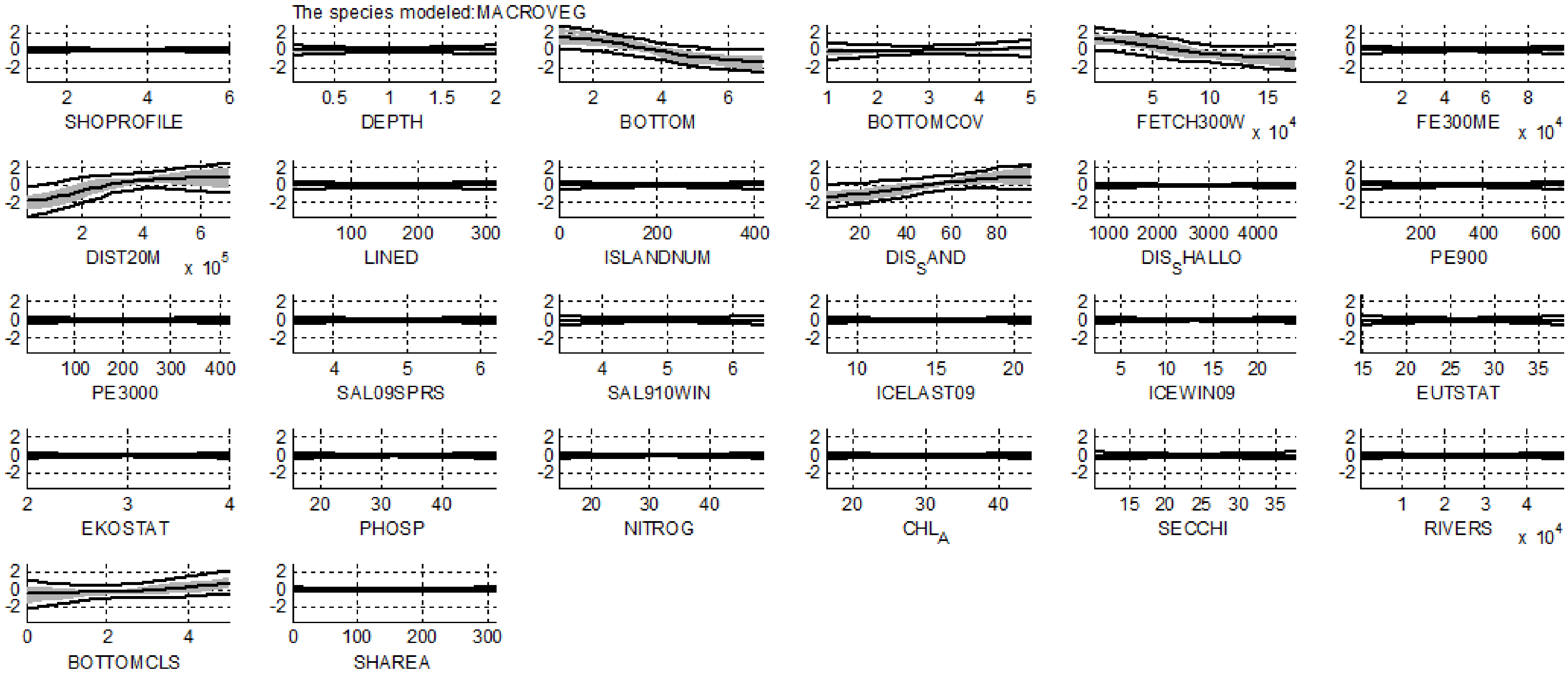

Figure 12: MACROVEG response curves of all the variables.