USER MANUAL OF THE

WEIGHT DETERMINATION BY MANIFOLD REGULARIZATION CODE

1. INTRODUCTION

Weight Determination by Manifold Regularization (WDMR) may be used to construct nonlinear multi-proxy models for a broad scale of climate archives. In our paper we present a model to reconstruct water temperature from high resolution trace element measurements in the calcitic layer of blue mussel shells (*Mytilus edulis*). We encourage all paleontologists to explore new data using this method, in other to build new WDRM models for a broad scale of climate archives. We recommend redoing the exercise for *Mytilus edulis* to get experience for using the code before applying it to other data. In this user manual only the principal and the practical aspects of the WDMR-code are discussed. The mathematical details of the method are described in (Ohlsson et al., 2008; Ohlsson et al., 2009). To ensure optimal use of the WDMR toolbox we recommend contacting the authors, especially to ensure the optimization processes to chose the best proxy combination (as done in this paper), and explore the possibility of introducing dynamics and choosing the best turn-over time (for this paper we assumed a static relationship between proxy and environment, this however is often not the best choice).

2. TO CHECK THE RESULTS OF THE PAPER

All results presented in the paper can be checked using the m-file called ‘validation.m’ and the xls-file ‘TotalDataBeste2.xls’. Since the m-file ‘validation.m’ used some sub-files the Matlab current directory has to be on the file ‘WDMRforGMD’.

2.1. LINEAR VERSUS NONLINEAR

The file ‘validation.m’ contains only the code to build a WDMR-model. To construct the linear multiple regression models the Matlab ‘regress’ function can be used.

2.2. PROXY EVALUATION

The proxy combination can be changed by redefining the parameter ‘ProxyComb’. This parameter defines which proxy data will be used for the training and de validation of the model. This is done by defining a vector that identifies the columns of the proxy that users want to be used. For example to construct an MgSrPb-model the parameter ‘ProxyComb=[8 9 11], knowing that the columns of the proxies correspond with 8=Mg, 9=Sr, 10=Ba and 11=Pb.

3. TO USE THE WDMR-CODE ON ANOTHER DATA-SET

To use the WDMR code on another dataset the dataset has to be saved in the right form in an xls-file and several parameters have to be adapted to the new dataset in the m-file ‘validation.m’.

3.1. DATASET XLS-FILE

- For an example open the file ‘TotalDataBeste2.xls’
- New data have to be saved in a new excel file with a new file name. An xls-table has to be made for which the rows correspond with the subsequent data measurements and the colons with the deterrent proxies. The last column of the table should contain the data for the environmental parameter.
- If the dataset contains measurements on different specimens the data of the different specimens can be pasted in sequence. To indicate that de data are from another specimen, as many columns as specimens studied have to be placed before the measurements. To define which measurements correspond to which specimens, every column is filled with ones and zeros.

3.2. ADAPT M-FILE

Below here the m-file is copied. The sites where the file should be adapted to the new dataset are highlighted

```
for i=1:4
      %(4 corresponds with the number of xls-sheets (i.e. the number of tests
that will be analysed)

%% Data Preprosessing
    data0=xlsread('TotalDataBeste2.xls',i);
      %(change to your file name)

    % Here you can choose the proxy combination (8=Mg, 9=Sr, 10=Ba and 11=Pb)
    ProxyComb=[8 9 10 11]
      %( the columns in which the proxy measurements are given)
    data0=data0(:,[1:7 ProxyComb 12])
      %(7 is the number of specimens studied, 12 the column with the
environmental parameters)

    %Here you can choose how many times you want to repeat the data.
    %In the paper the system is supposed to be static and ‘herhaling’ is
    %chosen [-0 0]
    herhaling=[-0 0];

    [data1]=PrepaireData(data0,herhaling);
    C=find(data0(1,:)> 0);
    HerDat=data1(:,C(2):end);
    In=data1(:,1:C(2)-1);
    for a=1:7;
        OneShell=find(In(:,a)==1);
        p(a)=OneShell(1);
        l(a)=length(OneShell);
    end

%% Choosing training and test data
    TrainD=HerDat(1:p(5)-1,:);%training sett
    OptimD=HerDat(p(5):p(7)-1,:);%optimisation set
    TestD=HerDat(p(7):end,:);%validation set

%(here you choose which specimens are used to train/to validate the model,
this has to be adapted depending on number of specimens studied)

TrainD=[[sum(TrainD(:,1:4:end-1),2) sum(TrainD(:,2:4:end-1),2)…
sum(TrainD(:,3:4:end-1),2) sum(TrainD(:,4:4:end-1),2)]./4 TrainD(:,end)];

OptimD=[[sum(OptimD(:,1:4:end-1),2) sum(OptimD(:,2:4:end-1),2)…
sum(OptimD(:,3:4:end-1),2) sum(OptimD(:,4:4:end-1),2)]./4 OptimD(:,end)];

TestD=[[sum(TestD(:,1:4:end-1),2) sum(TestD(:,2:4:end-1),2)…
sum(TestD(:,3:4:end-1),2) sum(TestD(:,4:4:end-1),2)]./4 TestD(:,end)];

    [Nr,Nk]=size(TrainD);
    k=9;
    lambda=0.9;
    r=0.02;

%% Traing the model using WDMR
K=WDMR([TrainD(:,1:Nk-1)' OptimD(:,1:Nk-1)'],TestD(:,1:Nk-1)',lambda,k,r);
```

```
y_pred = K * [TrainD(:,Nk) ;OptimD(:,Nk); zeros(size(TestD(:,Nk)))];
[fi g]=fit(TestD(1:end,Nk).*std(data0(:,end))+mean(data0(:,end)),…
y_pred(end+1-length(TestD(:,Nk)):end).*std(data0(:,end))+mean(data0(:,end)))

%% Validation of the model
    figure(1);
    TEREC=TestD(1:end,Nk).*std(data0(:,end))+mean(data0(:,end));
    REC=y_pred(end+1- …
length(TestD(:,Nk)):end).*std(data0(:,end))+mean(data0(:,end));
    subplot(4,1,i);
      %(‘4’ and ‘i’ depend on the number of tests that are planned)
    plot(TEREC,'--k'); hold on;plot(REC,'k');
    MAE(i)=mean(abs(REC-TEREC));
    RMSE(i)=sqrt(mean((REC-TEREC).^2))
    FoutStd(i)=std(abs(REC-TEREC));
    text(1,18,['RMSE:',num2str(round(RMSE(i).*100)./100) ]);
    ylabel('Temperature'); xlabel('Sample');
end
```