Appendix C
MATLAB Scripts

Two Matlab scripts *readadcp.m* and *movsov.m,* have been applied as part of the modelling process. Caluwaerts (2003) developed the first script in his thesis research for the Laboratory of Hydraulics at K.U.Leuven. It has been modified for the convenience of the present work. In other hand, *movsov.m* has been completely developed as part of this thesis research to compare velocity outputs from TELEMAC-2D with observed values to evaluate the model performance.

C.1 Readadcp MATLAB Script

This routine has been developed in MATLAB to read and calculate the depth-averaged velocity for every vertical data set recorded by ADCP. The raw data files are ASCII-format (produced by Win River) and contain the velocity profile records for every observed point. A printout version of the script can be found at the correspondent thesis document.

Some script lines have been added and/or modified from the original version, but keeping the main structure, to extract extra useful information. The printing of the date and time of the records has been implemented as well as the computation and printing of the velocity directions. The Trapezoidal rule (Soulsby, 1997) has replaced the previous formulation to compute a weighted depth-averaged velocity in order to account for the logarithmic velocity distribution in the profile encountered in flow along a wall (Graf, 1998).

The output matrix from this routine contains information about the date, time and position of the observed values (WGS84 expressed in Latitude and Longitude) as well as the correspondent East and North velocity component values with their calculated velocity magnitude (m/s) and direction (degrees). This data will be used as an input to the analysis tool developed in the present thesis (m*ovsov.m)* to compare with TELEMAC velocity outputs to evaluate the goodness of fit of the model.

C.2 Movsov MATLAB Script

The script has been developed to test the performance of the numerical 2D-Hydrodynamic model for the IJzer estuary efficiently. It compares every observed value with the model velocity output at the closest position in the model domain for the time step of interest. Movsov extracts the simulated velocity (East and North components) at every point within the model domain for the analysed time step to a matrix form (from ASCII result file of TELEMAC-2D 'modelresults.txt'). In this way, every model point during one time step can be compared with observed data available at different positions into the domain.

For a more convenient analysis, the observed data for every cross and longitudinal section must be stored in different files. The file name of the observed velocity data has been defined as 'measurements.txt'. Therefore, it is necessary to create folders for every section that contains the script, 'measurements.txt' and 'modelresults.txt' files. The files for the observed values must contain 4 columns: East and North geographic coordinates (UTM, 31N, ED50), observed velocity magnitude (m/s) and direction (degrees). The number of TELEMAC output variables, number of graphic printout periods and the time step to be compared has to be input externally by the user. The time is defined in terms of the graphic printout number correspondent to the present time of analysis (e.g. 3 days simulation period with a graphic printout period of thirty minutes have 144 printouts without counting the 0 time step). The user should define the section of analysis as a longitudinal or a transversal one.

Some plots to evaluate the model performance are generated: Model and Observed Velocity along a section (transversal or longitudinal) with their correspondent Scatter plots for observed versus model velocity magnitude and direction. For the first option the reference point in the plot is defined at the most left position at the transversal section, while for the longitudinal section it is fixed near the big reservoir mouth. In the scatter plots the velocity magnitude and direction at one model point is compared with the mean of the velocity magnitude and direction at closest measurement points. The standard deviation of the mean velocity observed values is displayed, as well as the RMSE. Some important output matrixes to be stored are 'plotpoints' and 'real_modeldata' containing the model and observed velocity data involved in the comparison process.

MATLAB SCRIPT TO COMPARE MODEL VELOCITY IN TELEMAC-2D WITH OBSERVED VALUES (MOVSOV)
Authors: Indira Nolivos and Ahbar Choudhury
Made in 2004 for KULeuven, Laboratory of Hydraulics

```
clear
close all

%READING TELEMAC ASCII DATA FILE
fid=fopen('modelresults.txt','r');
for n=1:20
   tline = fgetl(fid);                              %general information
end
numelem =  fscanf(fid,'%d \n',1);                   %number of mesh elements
numnods = fscanf(fid,'%d \n',1);                    %number of mesh nodes
total=numelem + numnods;                            %useless data
X=[];
Y=[];
timestep=[];
for n=1:(total+2)
    tline = fgetl(fid);                     %elements and nodes definition
end
datalinedata=fscanf(fid,'%lg \n',numnods);      %reading East coordinates
for n=1:numnods
    x=datalinedata(n);
    X=[X;x];                                    % nodes' East coordinate
end
datalinedata=fscanf(fid,'%lg \n',numnods);      %reading North coordinates
for n=1:numnods
    y=datalinedata(n);
    Y=[Y;y];                                    %nodes' North coordinates
end
U=[];
V=[];
w=input('number of TELEMAC output variables: ');
k=input('number of TELEMAC graphic printout periods…
                                                    without time step 0: ');
time=fscanf(fid,'%f \n',1);                         %reading '0' time step
datalinedata=fscanf(fid,'%f \n',(w*numnods));   %data scan at '0' time step

%East and north velocity components from telemac ascii data file
time=fscanf(fid,'%f \n',1);                         %first time step
for n=1:k
    timestep=[timestep;time];                       %Telemac printout periods
    datalinedata=fscanf(fid,'%f \n',(w*numnods));     %one time step data
    velocityU=[];
    velocityV=[];
    for n=1:numnods
        u=datalinedata(n);
        velocityU=[velocityU;u];            %East velocity component
    end
    U=[U velocityU];                        %East velocity component matrix
    for n=(numnods+1):(2*numnods)           %with data for all time steps
        v=datalinedata(n);
        velocityV=[velocityV;v];            %North velocity component
    end
```

```
    V=[V velocityV];                        %North velocity component matrix
    for n=(2*numnods+1):(7*numnods)         %with data for all time steps
        data=datalinedata(n);               %scanning other variables' data
    end
    time=fscanf(fid,'%f \n',1);             %reading the new time step
end

%READING OBSERVED VELOCITY DATA FILE
fid=fopen('measurements.txt','r');
data=fscanf(fid,'%f \n',inf);               %scanning the data file
columns=input('number of columns in the file of measurements: ');
if columns==4
    tstep=input('TELEMAC graphic printout period analysed: ');
    long=length(data);
    count=long/columns;                     %Data file rows
    Xo=[];
    Yo=[];
    Vel=[];
    Direc=[];
    k=1;
    for row=1:count
        xo=data(k);                         %Observed East coordinate
        yo=data(k+1);                       %Observed North coordinate
        vel=data(k+2);                      %Observed velocity magnitude
        direc=data(k+3);                    %Observed velocity direction
        Xo=[Xo;xo];                         %East coordinates
        Yo=[Yo;yo];                         %North coordinates
        Vel=[Vel;vel];                      %Velocity magnitudes
        Direc=[Direc;direc];                %Velocity directions
        k=k+4;                              %Position variable
    end

   %FINDING THE CLOSEST TELEMAC POINT TO THE OBSERVED POINTS IN THE SECTION
    x=[];
    y=[];
    compu=[];
    compv=[];
    for m=1:count
        Dist=[];
        for n=1:numnods
            dist=((Xo(m)-X(n))^2+(Yo(m)-Y(n))^2)^(0.5);
            %Distance between an observed point and all model points
            Dist=[Dist; dist];
        end
        [C,I] = min(Dist);              %TELEMAC closest point
        if ~V(I,tstep)==0               %conditional for TELEMAC wet points
            x=[x;X(I)];                 %East coordinate of model point
            y=[y;Y(I)];                 %North coordinate of model point
            compu=[compu;U(I,tstep)];
            %Vector for East velocity component
            compv=[compv;V(I,tstep)];
            %Vector for North velocity component
        else
            dum=Dist;                   %for non-wet points
            while V(I,tstep)==0
                dum(I)= max(Dist);
```

```
            [C,I] = min(dum);               %TELEMAC closest point
        end
        x=[x;X(I)];
        y=[y;Y(I)];
        compu=[compu;U(I,tstep)];
        compv=[compv;V(I,tstep)];
    end
end

%Calculating the magnitude and direction of the model velocities
velocities=[];
directions=[];
real_modeldata=[];
for runner=1:length(x)
    velocities=[velocities;sqrt(compu(runner)^2+compv(runner)^2)];
                                                %Velocity magnitude
    %Calculation of the velocity direction
    if compv(runner)~=0                         %Velocity direction
        if compu(runner)>=0 & compv(runner)>0
            calcdirection=atan(compu(runner)/compv(runner))*(180/pi);
        else
            if (compu(runner)>=0 & compv(runner)<0) |
               (compu(runner)<=0 & compv(runner)<0)
                calcdirection=atan(compu(runner)/…
                                        compv(runner))*(180/pi)+180;
            else
                calcdirection=atan(compu(runner)/…
                                        compv(runner))*(180/pi)+360;
            end
        end
        directions=[directions;calcdirection];
    else
        directions=[directions;99999];
                                    %This option is an error index
        display('bad telemac point was stored')
    end
    %General matrix containing the observed velocity data and the
    %model points to be compared with in one telemac time step
    real_modeldata=[real_modeldata;Xo(runner) Yo(runner) Vel(runner)…
                    Direc(runner) x(runner) y(runner) compu(runner)…
                compv(runner) velocities(runner) directions(runner)];
end

%MEAN AND STANDARD DEVIATION OF OBSERVED VELOCITY FOR EVERY MODEL POINT
DATA=real_modeldata;
vel=[];
meanvelmeasurements=[];
plotpoints=[];
meandirmeasurements=[];
telemacvelocities=[];
telemacdirections=[];
a=length(DATA);                             %number of rows
b=1;
totalvel=0;
n=1;
teta_o=[];
```

```
s=[];
c=[];
while b<a
   while DATA(n,9)==DATA(b,9)
       vel=[vel;DATA(b,3)];                %magnitudes (m/s)
       teta_o=[teta_o;DATA(b,4)];          %directions (degrees)
       s=[s;sin((pi*DATA(b,4))/180)];
       c=[c;cos((pi*DATA(b,4))/180)];
       if b<a
           b=b+1;
       else
           DATA(b,9)=9999;
       end
   end
   meanvelmeasurements=[meanvelmeasurements;mean(vel) std(vel)];
                          %Mean magnitude and standard deviation (m/s)
   S=mean(s);
   C=mean(c);
   teta=atan(S/C);        %Mean direction in radians
   if C~=0
        if C>0 & S>=0     %corrected directions in radians and degrees
            teta_degree=teta*(180/pi);      %first quadrant
        else
            if (S>=0 & C<0) | (S<=0 & C<0)
                teta=teta+pi;
                teta_degree=teta*(180/pi);  %second and third quadrant
            else
                teta=teta+(2*pi);
                teta_degree=teta*(180/pi);  %forth quadrant
            end
        end
    else
        display('error in measurements direction');
    end
   R=sqrt((C^2)+(S^2));               %residual from mean direction
   So=1-R;                            %variance from mean direction
   St_dev=sqrt(-2*log(1-So));         %standard deviation
   meandirmeasurements=[meandirmeasurements;teta_degree teta St_dev];
                                      %Degrees and radians
   telemacvelocities=[telemacvelocities;DATA(n,9)];          %(m/s)
   telemacdirections=[telemacdirections;DATA(n,10)…
                                           (DATA(n,10)*(pi/180))];
                                      %Degrees and radians
   plotpoints=[plotpoints;mean(vel) std(vel) teta_degree teta St_dev…
                          DATA(n,9) DATA(n,10) (DATA(n,10)*(pi/180))];
                                %Information used for the Scatter plots
   n=b;
   vel=[];
   teta_o=[];
   s=[];
   c=[];
end

%STATISTICS: ROOT MEAN SQUARE ERROR FOR THE MODEL VELOCITY (RMSE)
summag=0;
sumdir=0;
```

```
for n=1:length(meanvelmeasurements)
    summag=summag+(meanvelmeasurements(n,1)-telemacvelocities(n))^2;
    sumdir=sumdir+cos(meandirmeasurements(n,2)-telemacdirections(n,2));
end
rmse_vel=sqrt(summag/length(meanvelmeasurements));
rmse_dir=sqrt(-2*log(sumdir/length(meandirmeasurements)));
rmse1={'RMSE:',rmse_vel};
rmse2={'RMSE:',rmse_dir};

%PLOT OF THE MODEL AND REAL VELOCITIES ALONG A SECTION IN THE CANAL
z=input('choose between a cross section plot (1) or …
                    a longitudinal one (2) to show the results:');
if z==1
    %Determining the reference point for a cross section plot
    [C,I] = min(x);                    %most TELEMAC left position
    [D,J] = min(Xo);                   %most observed left position
    if C<D
        x1=x(I);                       %reference point in the plot.
        y1=y(I);
    else
        x1=Xo(J);
        y1=Yo(J);
    end
    Distmeas=[];
    Distmodel=[];
    r=length(x);
    for n=1:r
        dist1=((Xo(n)-x1)^2+(Yo(n)-y1)^2)^(0.5);
              %Distance of the observed points to the reference point
        dist2=((x(n)-x1)^2+(y(n)-y1)^2)^(0.5);
              %Distance of the TELEMAC points to the reference point
        Distmeas=[Distmeas; dist1];          %Observed point distances
        Distmodel=[Distmodel; dist2];        %Model point distances
    End
    %Plotting the model and observed velocities along the cross section
    figure(1)
    plot(Distmeas, Vel,'+r', Distmodel, velocities,'ok');
    axis([0 200 0 1]);
    legend('OBSERVED', 'MODEL');
    title('MODEL AND OBSERVED VELOCITY AT TRANSVERSAL SECTION',…
                                                    'Fontweight','bold');
    xlabel('DISTANCE (m)','Fontweight','bold');
    ylabel('VELOCITY (m/s)','Fontweight','bold');
    set(gca,'XTick',0:25:200);
    set(gca,'XTickLabel',{'0','25','50','75','100','125','150',…
                                  '175','200'},'Fontweight','bold');
    set(gca,'YTick',0:0.25:1);
    set(gca,'YTickLabel',{'0','0.25','0.50','0.75','1'},…
                                                    'Fontweight','bold');
else
    %Reference point for a longitudinal section plot
    x1=482100;                                 %reference point at the
    y1=5665900;                                %mouth of the big reservoir
    Distmeas=[];
    Distmodel=[];
    r=length(x);
```

```
    for n=1:r
        dist1=((Xo(n)-x1)^2+(Yo(n)-y1)^2)^(0.5);
               %Distance of the observed points to the reference point
        dist2=((x(n)-x1)^2+(y(n)-y1)^2)^(0.5);
               %Distance of the TELEMAC points to the reference point
        Distmeas=[Distmeas; dist1];          %Observed point distances
        Distmodel=[Distmodel; dist2];        %Model point distances
    end
    %Plotting the model and observed velocities along the section
    figure(1)
    plot(Distmeas, Vel,'+r', Distmodel, velocities,'ok');
    axis([0 1600 0 1]);
    legend('OBSERVED', 'MODEL');
    title('MODEL AND OBSERVED VELOCITY AT LONGITUDINAL SECTION',…
                                                    'Fontweight','bold');
    xlabel('DISTANCE (m)','Fontweight','bold');
    ylabel('VELOCITY (m/s)','Fontweight','bold');
    set(gca,'XTick',0:100:1600);
    set(gca,'XTickLabel',{'0','100','200','300','400','500','600',…
              '700','800','900','1000','1100','1200','1300','1400',…
                             '1500','1600'},'Fontweight','bold');
    set(gca,'YTick',0:0.25:1);
    set(gca,'YTickLabel',{'0','0.25','0.50','0.75','1'},…
                                                    'Fontweight','bold');
end

%SCATTER PLOTS OF THE OBSERVED VS MODEL VELOCITY
line=[0 1];
line2=[0 (2*pi)];
figure(2)
errorbar(telemacvelocities,meanvelmeasurements(:,1),…
                                      meanvelmeasurements(:,2),'.k');
hold on
plot(line,line,'--k','LineWidth',1);
axis square;
axis equal;
axis([0 1 0 1]);
title('OBSERVED VS MODEL VELOCITY MAGNITUDE','Fontweight','bold');
xlabel('MODEL MAGNITUDE (m/s)','Fontweight','bold');
ylabel('MEAN OBSERVED MAGNITUDE (m/s)','Fontweight','bold');
text(0.8,0.7,rmse1,'Fontweight','bold');
set(gca,'XTick',0:0.25:1);
set(gca,'XTickLabel',{'0','0.25','0.5','0.75','1'},'Fontweight','bold');
set(gca,'YTick',0:0.25:1);
set(gca,'YTickLabel',{'0','0.25','0.5','0.75','1'},'Fontweight','bold');
hold off
figure(3)
plot(telemacdirections(:,2),meandirmeasurements(:,2),'.k');
hold on
plot(line2,line2,'--k','LineWidth',1);
axis square;
axis equal;
axis([0 2*pi 0 2*pi]);
title('OBSERVED VS MODEL VELOCITY DIRECTION','Fontweight','bold');
xlabel('MODEL DIRECTION (radians)','Fontweight','bold');
ylabel('MEAN OBSERVED DIRECTION (radians)','Fontweight','bold');
```

```
   text(0.2,2,rmse2,'Fontweight','bold');
   set(gca,'XTick',0:(pi/2):(2*pi));
   set(gca,'XTickLabel',{'0','pi/2','pi','3pi/2','2pi'},…
                                                    'Fontweight','bold');
   set(gca,'YTick',0:(pi/2):(2*pi));
   set(gca,'YTickLabel',{'0','pi/2','pi','3pi/2','2pi'},…
                                                    'Fontweight','bold');
   hold off
else
   display ('Error: the measurements file must have 4 columns','r');
end
fclose all;
```