% Remote Sensing - Model Comparison Toolkit

% -----------------------------------------

% The Matlab routines in this folder are developed to perform a visual and statistical comparison between gridded model data and a set of remote sensing images.

% The 2 main routines are:

% 1) Prepare\_RS\_Model\_Comparison.m => Preprocessing and visualization of the data.

% 2) RS\_Model\_Comparison.m => Calculate statistics and plot Target diagram.

%

% Note that the routines are written and tested in Matlab version 8.2.0.701 (R2013b).

% It is possible that some of the functions (especially the ones that handle the visualization)

% will not function properly in more recent Matlab versions. In case any problems arise,

% it is possible to skip the visualization since it is not the essential part of the routines.

%

% The routines are developed in the framework of the FP7 INFORM project

% (http://www.copernicus-inform.eu/), Grant Agreement n° 606865 – INFORM.

%

% Authors: Dirk Lauwaet (main developer) and Jean-Luc De Kok (VITO)

% Version 1.0: September 2016

%

%

% Prepare\_RS\_Model\_Comparison.m

% -----------------------------

% This routine preprocesses model and remote sensing data to facilitate a

% visual and statistical comparison. The data are interpolated to a

% user-defined regular lon-lat grid and data quality information (flags) is

% used to exclude suspicious data points. The resulting maps can be visualized and

% the resulting data is stored in Matlab matrices.

clear all;

close all;

clc;

% INPUT SPECIFICATIONS

% --------------------

% MODEL DATA

File\_Model = '../../../FRESHMON/Markermeer\_modeldata/TIM.nc';

VarName\_Model = 'TIM';

%File\_Model = '../../../FRESHMON/Markermeer\_modeldata/CHLF.nc';

%VarName\_Model = 'CHlfa';

% REMOTE SENSING DATA

Filenames\_RS=importdata('../../../SERVIJ\_MERISdata/RAW/filenames.txt');

% choose variable: 2=CHLF, 3=TSM

variable=3;

% SET UP RS ARRAYS

% time is an array of the corresponding time index in the model data matrix

time=[29,57,72,104,107,120,124,130,131,134,159,162,168,181,184,185,193,194,197,200,206,207,225,254,255,263,264,282,289,321];

% datearr and name are used for plotting and saving images

datearr={'29.01.2006';'26.02.2006';'13.03.2006';'14.04.2006';'17.04.2006';'30.04.2006';'04.05.2006';'10.05.2006';'11.05.2006'; ...

'14.05.2006';'08.06.2006';'11.06.2006';'17.06.2006';'30.06.2006';'03.07.2006';'04.07.2006';'12.07.2006';'13.07.2006'; ...

'16.07.2006';'19.07.2006';'25.07.2006';'26.07.2006';'13.08.2006';'11.09.2006';'12.09.2006';'20.09.2006';'21.09.2006'; ...

'09.10.2006';'16.10.2006';'17.11.2006'};

name={'20060129';'20060226';'20060313';'20060414';'20060417';'20060430';'20060504';'20060510';'20060511'; ...

'20060514';'20060608';'20060611';'20060617';'20060630';'20060703';'20060704';'20060712';'20060713'; ...

'20060716';'20060719';'20060725';'20060726';'20060813';'20060911';'20060912';'20060920';'20060921'; ...

'20061009';'20061016';'20061117'};

% SET UP REGULAR LON-LAT GRID

% ---------------------------

nx=260;

ny =200;

res=0.002;

for i=1:nx

lon(i,1:ny) = 4.96+i\*res;

end

for j=1:ny

lat(1:nx,j) = 52.30+j\*res;

end

nlat = size(lat,1);

nlon = size(lon,2);

% READ MODEL DATA

% ---------------

Lat\_Model = ncread(File\_Model,'lat');

Lon\_Model = ncread(File\_Model,'lon');

ncdisp(File\_Model,VarName\_Model);

Variable\_Model = ncread(File\_Model,char(VarName\_Model));

NoDataValue = ncreadatt(File\_Model,char(VarName\_Model),'\_FillValue');

nlat\_model = size(Lat\_Model,1);

nlon\_model = size(Lon\_Model,2);

n\_time\_model = size(Variable\_Model,3);

% START LOOP OVER ALL RS IMAGES

% -----------------------------

for t=1:size(time,2)

date=datearr{t}

i\_time = time(t);

% SELECT MODEL VALUES

% -------------------

for i=1:nlat\_model

for j =1:nlon\_model

z\_model(i,j) = Variable\_Model(i,j,i\_time);

end

end

z\_model(z\_model<0)= NaN ;

id = find(~isnan(z\_model));

n\_data\_model = size(id,1);

% READ RS AND FLAG DATA

% ---------------------

File\_RS = load(strcat('../../../SERVIJ\_MERISdata/RAW/',Filenames\_RS{t},'.mat'))

Lat\_RS = File\_RS.lat;

Lon\_RS = File\_RS.lon;

Variable\_RS = File\_RS.c(:,:,variable);

Flag\_RS = File\_RS.l2\_flags;

nlat\_RS = size(Lat\_RS,1);

nlon\_RS = size(Lon\_RS,2);

n\_time\_RS = 1;

% FIRST INTERPOLATE RS AND FLAGS TO REGULAR GRID, BEFORE APPLYING FLAGS

% (scatteredInterpolant can't correctly handle NaN data)

% REGRID RS DATA TO REGULAR GRID

% ------------------------------

index = 0;

for i=1:nlat\_RS

for j =1:nlon\_RS

if (~isnan(Variable\_RS(i,j)))

index=index+1;

u(index,1) = Lat\_RS(i,j);

v(index,1) = Lon\_RS(i,j);

w(index,1) = Variable\_RS(i,j);

end

end

end

InterpolationMethod = 'nearest';

ExtrapolationMethod = 'none';

F = scatteredInterpolant(double(u),double(v),double(w),char(InterpolationMethod),char(ExtrapolationMethod));

Variable\_RS\_Regridded=F(lat,lon);

% REGRID FLAG DATA TO REGULAR GRID

% --------------------------------

index = 0;

for i=1:nlat\_RS

for j =1:nlon\_RS

if (~isnan(Flag\_RS(i,j)))

index=index+1;

u1(index,1) = Lat\_RS(i,j);

v1(index,1) = Lon\_RS(i,j);

w1(index,1) = Flag\_RS(i,j);

end

end

end

InterpolationMethod = 'nearest';

ExtrapolationMethod = 'none';

F = scatteredInterpolant(double(u1),double(v1),double(w1),char(InterpolationMethod),char(ExtrapolationMethod));

Flag\_RS\_Regridded=F(lat,lon);

% PUT NaN DATA TO BAD FLAG (meris\_mask.m can't handle NaN)

% ------------------------

for i=1:nlat

for j=1:nlon

if (isnan(Flag\_RS\_Regridded(i,j)))

Flag\_RS\_Regridded(i,j) = 8388616;

end

end

end

% FILTER OUT BAD VALUES (Deltares info and code (meris\_mask.m, meris\_flags.m, setproperty.m))

% ---------------------

mask=meris\_mask(Flag\_RS\_Regridded,[22 23]);

for i=1:nlat

for j=1:nlon

if (variable==3)

if ( isnan(mask(i,j)) || Variable\_RS\_Regridded(i,j) < 0.1 || Variable\_RS\_Regridded(i,j) > 200) %TSM

Variable\_RS\_Regridded(i,j)=NaN ;

end

else

if ( isnan(mask(i,j)) || Variable\_RS\_Regridded(i,j) < 0.01 || Variable\_RS\_Regridded(i,j) > 150) %CHLF

Variable\_RS\_Regridded(i,j)=NaN ;

end

end

end

end

% REGRID MODEL DATA TO REGULAR GRID

% ---------------------------------

index = 0;

for i=1:nlat\_model

for j =1:nlon\_model

if (~isnan(z\_model(i,j)))

index=index+1;

u2(index,1) = Lat\_Model(i,j);

v2(index,1) = Lon\_Model(i,j);

w2(index,1) = z\_model(i,j);

end

end

end

InterpolationMethod = 'nearest';

ExtrapolationMethod = 'none';

F = scatteredInterpolant(double(u2),double(v2),double(w2),char(InterpolationMethod),char(ExtrapolationMethod));

Variable\_Model\_Regridded=F(lat,lon);

% FILTER OUT BAD VALUES (same as for the RS data)

% ---------------------

for i=1:nlat

for j=1:nlon

if ( isnan(Variable\_RS\_Regridded(i,j)) || isnan(Variable\_Model\_Regridded(i,j)) )

Variable\_Model\_Regridded(i,j)=NaN ;

Variable\_RS\_Regridded(i,j)=NaN;

end

end

end

% PUT DATA IN OUTPUT MATRICES

% ---------------------------

model(t,:,:)=Variable\_Model\_Regridded(:,:);

obs(t,:,:)=Variable\_RS\_Regridded(:,:);

% VISUALIZATION

% -------------

Map1=Variable\_RS\_Regridded;

Map2=Variable\_Model\_Regridded;

LonMin = min(min(lon));

LonMax = max(max(lon));

LatMin = min(min(lat));

LatMax = max(max(lat));

figure( 'Position', get(0,'Screensize') );

for k = 1:2

subplot( 1, 2, k);

if (k == 1)

Map=Map1;

Bins= [0 10 20 30 40 50 60 500];

else

Map=Map2;

Bins= [0 10 20 30 40 50 60 500];

end

set(gcf, 'Position', get(0,'Screensize'));

Map\_reclassed = reclassify(Map,Bins,NoDataValue);

m\_proj('Transverse Mercator','longitude',[LonMin-0.01 LonMax+0.01],'latitude',[LatMin-0.01 LatMax+0.01]);

m\_pcolor(lon,lat,Map\_reclassed);

caxis([1 7])

hold on;

% ADD GEO FEATURES

m\_gshhs\_h('save','gumby');

m\_usercoast('gumby','patch','w');

m\_grid('xtick',10,'ytick',10,'FontSize',12);

hold on

axis equal;

% ADD COLORBAR

Bins = [0 10 20 30 40 50 60 500];

nclasses = size(Bins,2) - 1;

colormap(jet(nclasses));

lcolorbar({'0-10 ','10-20','20-30','30-40','40-50','50-60','>60'});

% ADD TITLE

if (k == 1)

if (variable==3)

title(['MERIS (regridded) Total Suspended Matter ' date]);

else

title(['MERIS (regridded) Chlorophyll-a ' date]);

end

else

if (variable==3)

title(['Modelled (regridded) Total Inorganic Matter ' date]);

else

title(['Modelled (regridded) Chlorophyll-a ' date]);

end

end

% ADD RULER

offset = 0.2;

[x\_ruler1,y] = m\_ll2xy(LonMin+offset\*(LonMax-LonMin),LatMin+0.1\*(LatMax-LatMin));

[x\_ruler2,y] = m\_ll2xy(LonMax,LatMin+0.1\*(LatMax-LatMin));

[dummy, ymin] = m\_ll2xy(LonMin,LatMin);

distance = m\_lldist([(LonMin+offset\*(LonMax-LonMin)) LonMax],[LatMin+0.1\*(LatMax-LatMin) LatMin+0.1\*(LatMax-LatMin)]);

scalefactor = 5/distance;

x\_ruler2= x\_ruler1+scalefactor\*(x\_ruler2-x\_ruler1);

line([x\_ruler1 x\_ruler2],[y y ],'Color','k','Linewidth',1);

line([x\_ruler1 x\_ruler1],[(y - 0.1\*(y-ymin)) (y + 0.1\*(y-ymin))],'Color','k','Linewidth',1);

line([x\_ruler2 x\_ruler2],[(y - 0.1\*(y-ymin)) (y + 0.1\*(y-ymin))],'Color','k','Linewidth',1);

text(x\_ruler1+0.4\*(x\_ruler2-x\_ruler1),y + 0.2\*(y-ymin),'5 km','Color','black','FontSize',10,'FontWeight','bold','FontName','Arial','vertical','top');

end

% PRINT MAP TO JPEG

if (variable==3)

ShowMapName = strcat('TSM\_regridded\_',name{t})

else

ShowMapName = strcat('CHL\_regridded\_',name{t})

end

set(gcf,'PaperPositionMode','auto')

print(char(ShowMapName),'-djpeg');

end % for loop over time

% SAVE OUTPUT MATRICES

% --------------------

if (variable==3)

save('model\_tsm.mat', 'model');

save('obs\_tsm.mat','obs');

else

save('model\_chl.mat', 'model');

save('obs\_chl.mat','obs');

end

% Remote Sensing - Model Comparison Toolkit

% -----------------------------------------

% The Matlab routines in this folder are developed to perform a visual and statistical comparison between gridded model data and a set of remote sensing images.

% The 2 main routines are:

% 1) Prepare\_RS\_Model\_Comparison.m => Preprocessing and visualization of the data.

% 2) RS\_Model\_Comparison.m => Calculate statistics and plot Target diagram.

%

% Note that the routines are written and tested in Matlab version 8.2.0.701 (R2013b).

% It is possible that some of the functions (especially the ones that handle the visualization)

% will not function properly in more recent Matlab versions. In case any problems arise,

% it is possible to skip the visualization since it is not the essential part of the routines.

%

% The routines are developed in the framework of the FP7 INFORM project

% (http://www.copernicus-inform.eu/), Grant Agreement n° 606865 – INFORM.

%

% Authors: Dirk Lauwaet (main developer) and Jean-Luc De Kok (VITO)

% Version 1.0: September 2016

%

% RS\_Model\_Comparison.m

% ---------------------

% This routine reads in model and remote sensing matrices that are prepared

% with Prepare\_RS\_Model\_Comparison.m and calculates a number of statistics.

% Finally a Target Diagram is plotted to visualize the comparison.

% Based on: Jolliff, J.K., Kindle, J.C., Shulman, I., Penta, B., Friedrichs, M.A.M., Helber, R., Arnone, R.A., 2009.

% Summary diagrams for coupled hydrodynamic-ecosystem model skill assessment. Journal of Marine Systems 76, 64-82.

clear all;

close all;

clc;

% LOAD MODEL AND RS DATA (structure=[time,x,y])

% ----------------------

%load('obs\_chl.mat');

%load('model\_chl.mat');

load('obs\_tsm.mat');

load('model\_tsm.mat');

nx = size(obs,2);

ny = size(obs,3);

nt = size(obs,1);

% Print name of target plot

ShowMapName='TSM\_target\_all';

% START LOOP OVER TIME

% --------------------

for t=1:nt

n=1;

for i=1:nx

for j=1:ny

if (~isnan(obs(t,i,j)))

o(n)=obs(t,i,j);

m(n)=model(t,i,j);

n=n+1;

end

end

end

% OPTIONAL: calculate the fraction of available data points to weigh the

% statistics.

n=n-1

nn(t)=n;

% VALIDATION STATISTICS

% ---------------------

std\_obs(t) = std( o ); % Standard deviation observations, normalised to N-1, unmatched

avg\_obs(t) = mean( o ); % Mean value observations

std\_mod(t) = std( m ); % Standard deviation model

avg\_mod(t) = mean( m ); % Mean value model

rmse(t) = sqrt( mean( ( m - o ).^2 ) ); % Root Mean Square Error

bias(t) = mean( m - o ); % Bias

absbias(t) = abs( bias(t) ); % Absolute Bias

C = corrcoef( m, o );

r2(t) = C(1,2).\*C(2,1); % Coefficient of determination

r(t) = C(1,2); % Correlation coefficient

rrmse(t) = rmse(t) ./ avg\_obs(t); % Relative RMSE

nmb(t) = avg\_mod(t) ./ avg\_obs(t) - 1; % Normalized mean bias

mfb(t) = mean( 2.\*( m - o )./( m + o ) ); % Mean fractional bias

mfe(t) = mean( 2.\* abs( m - o )./ ( m + o ) ); % Mean fractional error

nmsd(t) = ( std\_mod(t) - std\_obs(t) ) ./ std\_obs(t); % Normalized mean standard deviation

ioa(t) = 1 - n .\* rmse(t).^2 ./ sum( ( abs(m-avg\_obs(t))+(o-avg\_obs(t)) ).^2 ); % Index of agreement

am(t) = std\_mod(t)/std\_obs(t); % Mismatch in amplitude

me(t) = 1 - ( sum((m - o).^2) / sum((o - avg\_obs(t)).^2) ); % Model Efficiency

% TARGET VALUE

crmse(t) = sqrt( mean( ( ( m - avg\_mod(t) ) - ( o - avg\_obs(t) ) ).^2 ) ); % Centered RMSE

target(t) = sqrt( ( bias(t) ./ std\_obs(t) ).^2 + ( crmse(t) ./ std\_obs(t) ).^2 ); % Target Value

% TARGET X AND Y FOR TARGET DIAGRAM

targ\_y(t) = bias(t) ./ std\_obs(t);

targ\_x1 = crmse(t) ./ std\_obs(t);

% calculate target plot sign

targ\_sgn = 1;

if r < 1

f = nmsd ./ sqrt( 2.\*(1-r));

if f > 1,

% SD dominates the error

targ\_sgn = 1;

else

% R dominates the error

targ\_sgn = -1;

end

end

targ\_x(t) = targ\_sgn \* targ\_x1;

end % for-loop over time

% TARGET DIAGRAM

% --------------

% list of symbols and names for target plot.

% Available colours: y,m,c,b,g,k,r.

% Available symbols: o,s,d,+,\*

symbol={'yo','mo','co','bo','go','ko','ro', ...

'ys','ms','cs','bs','gs','ks','rs', ...

'yd','md','cd','bd','gd','kd','rd', ...

'y+','m+','c+','b+','g+','k+','r+', ...

'y\*','m\*'};

names={'29.01.06 ','26.02.06 ','13.03.06 ','14.04.06 ','17.04.06 ','30.04.06 ','04.05.06 ','10.05.06 ','11.05.06 ', ...

'14.05.06 ','08.06.06 ','11.06.06 ','17.06.06 ','30.06.06 ','03.07.06 ','04.07.06 ','12.07.06 ','13.07.06 ', ...

'16.07.06 ','19.07.06 ','25.07.06 ','26.07.06 ','13.08.06 ','11.09.06 ','12.09.06 ','20.09.06 ','21.09.06 ', ...

'09.10.06 ','16.10.06 ','17.11.06 '};

set(0, 'DefaultAxesFontName', 'Calibri' );

scrsz = get(0,'ScreenSize');

hFig = figure( 'Position',[scrsz(3)/4 scrsz(4)/4 6\*scrsz(3)/4 6\*scrsz(4)/4] );

targplot( targ\_x, targ\_y, 'inputMode', 'coords', ...

'targetMax', 3, ...

'catSymbol', symbol , 'catLabel', names );

title( 'Target plot' );

set(gcf,'PaperPositionMode','auto')

print(char(ShowMapName),'-djpeg');