

```

%friver_classification
%Imports reflectance data and classifies river pixels as snow, ice, mixed
%icewater and open water
%
%Inputs: Daily MODIS images, river mask, metadata file
%Output: Daily images with classified pixels: 0 = land, 1 = water, 2 = mixed
%ice/water, 3 = ice, 4 = snow
%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%%
cd(fp_folder);
load('input_info.mat'); %metadata
load('mask.mat'); %river mask
river_mask=mask;
cd(reflFolder);
land_ind=find(river_mask==0);
river_ind=find(river_mask==1);
geotiffs=dir('*.tif');
k=length(geotiffs); %number of days
d1=input_info(39,year_ind+2);
d2=input_info(40,year_ind+2);
days=d1:1:d2;

%Remove missing/bad days
if input_info(2,year_ind+2)~=0
    for n=1:input_info(2,year_ind+2)
        ch=input_info(2+n,year_ind+2);
        days(days==ch)=[];
    end
end

snow_tol=0.73; %Snow > 0.73
ice_tol=0.5; %Ice > 0.5
mix_tol=0.1; %Mixed Ice/Water > 0.1

l1=input_info(1,1);
l2=input_info(1,2);
classified_reflectance=zeros(l1,l2,k); %size of input image

for i=1:k
    filename = geotiffs(i).name;
    [sample_tiff, R] = geotiffread(filename);
    info=geotiffinfo(filename);
    input_tiff = double(sample_tiff);
    %correct for different sized images
    if length(input_tiff(:,1))~=l1 && length(input_tiff(:,2))==l2
        %reflectance_tif=zeros(l1,l2);
        reflectance_tif=input_tiff(1:3376,:);
    end
    if length(input_tiff(1,:))~=l2
        reflectance_tif=zeros(l1,l2);
        reflectance_tif(1:2896,1:1274)=input_tiff;
    end
    if length(input_tiff(:,1))==l1 && length(input_tiff(1,:))~=l2
        reflectance_tif = .0001.*input_tiff;
    end
    reflectance_tif(land_ind)=0; %all non-river to 0
end

```

[illegible]

```

%fcloud_input
%Inputs cloud mask and determines which segments are clear
%
%Inputs: Cloud coordinates, river mask and groups, metadata, MODIS daily
%cloud mask images
%Outputs: Binary cloud mask for each segment for each day
cd(fp_folder);
load('input_info.mat'); %metadata
load('cloud_coord_10k.mat'); %coordinates of each group, group length
load('group_mask_10k.mat'); %river group
cd(cloudFolder);
geotiffs=dir('*.tif');
gr_coord=cloud_coord;
group_length=cloud_length;
p=input_info(2,1); %number of groups
t=length(geotiffs)/6; %number of days
cloud_mask=NaN*ones(p,t);
cl_dat=[-1 0 1 2 3]; %cloud references
index=1:6:600; %used to get only 1st image
for i=1:t
    w=index(i);
    filename = geotiffs(w).name;
    [tiff_read, Rcloud] = geotiffread(filename);
    %convert into matlab format
    tiff_read=double(tiff_read);
    ind=find(tiff_read<0);
    tiff_read(ind)=256+tiff_read(ind);
    for j=1:p
        l=group_length(j);
        xworld=gr_coord(1:l,1,j);
        yworld=gr_coord(1:l,2,j);
        [I,J]=worldToDiscrete(Rcloud,xworld,yworld); %puts river points into
        %pixel coordinates
        cloud_input=zeros(l,1);
        for k=1:l
            if isnan(I(k))==0 && isnan(J(k))== 0
                cloud = tiff_read(I(k),J(k));

                bit1=bitget(cloud,1);
                bit2=bitget(cloud,2);
                bit3=bitget(cloud,3);
                if bit1==0
                    cloud_input(k)=-1; %no quality data
                else
                    if bit2==0 && bit3==0
                        cloud_input(k)=0; %cloud
                    end
                    if bit2==0 && bit3==1
                        cloud_input(k)=1; %probably cloud
                    end
                    if bit2==1 && bit3==0
                        cloud_input(k)=2; %probably clear
                    end
                    if bit2==1 && bit3==1
                        cloud_input(k)=3; %confident clear
                    end
                end
            end
        end
    end
end

```

```

        else
            cloud_input(k)=-1;
        end

    end

    sum_cloud=zeros(5,1);
    for k=1:5
        sum_cloud(k)=sum(cloud_input==cl_dat(k)); %sums the number of
pixels of each type
    end
    total_cloud=sum_cloud(1)+sum_cloud(2)+sum_cloud(3); %no data plus
cloudy plus probably cloudy
    if total_cloud>.5*1 %cloud tolerance for pixels
        cloud_mask(j,i)=0; %group segment cloudy
    else
        cloud_mask(j,i)=1; %group segment clear
    end
end
end

d1=input_info(39,year_ind+2);
d2=input_info(40,year_ind+2);
days=d1:1:d2;
%remove missing days for cloud data
if input_info(26,year_ind+2)~=0
    for n=1:input_info(26,year_ind+2)
        ch=input_info(26+n,year_ind+2);
        days(days==ch)=[];
    end
end
if t~=length(days)
    disp('error wrong number of days');
end

%remove missing days for reflectance data
if input_info(2,year_ind+2)~=0
    for n=1:input_info(2,year_ind+2)
        ch=input_info(2+n,year_ind+2);
        ind=find(days==ch);
        if isempty(ind)==0
            cloud_mask(:,ind)=[];
        end
    end
end

%set bad data to zero (cloudy)
if input_info(16,year_ind+2)~=0
    for n=1:input_info(16,year_ind+2)
        ch2=input_info(16+n,year_ind+2);
        ind2=find(days==ch2);
        if isempty(ind2)==0
            cloud_mask(:,ind2)=zeros(p,1);
        end
    end
end

```

```
end
end
```

[illegible]

```

%fbreakup_analysis
%Determines the breakup date
%
%Inputs: River mask and 10 km segments, metadata, cloud mask,
%classified river images
%Outputs: Breakup dates
cd(fp_folder)
load('input_info.mat');
load('group_mask_10k.mat');
load('reference.mat'); %reference mat to give R
cd(reflFolder);
load('river_class.mat'); %classified tif
classified_reflectance=double(classified_reflectance);
load('cloud_mask_10k.mat'); %output cloud mask
p=input_info(2,1); %number of groups
d=length(classified_reflectance(1,1,:)); %number of days
clgr=[1 2 3 4 -1]; %1 = water, 2 = mixed ice/water, 3 = ice, 4 = snow
%-1 = bad data
sum_frequency=zeros(5,d,p);
breakup_days=zeros(p,1);

river_group=r_group;

d1=input_info(39,year_ind+2);
d2=input_info(40,year_ind+2);
days=d1:1:d2+1;

%remove missing/bad days
if input_info(2,year_ind+2)~=0
    for n=1:input_info(2,year_ind+2)
        ch=input_info(2+n,year_ind+2);
        days(days==ch)=[];
    end
end

for j=1:p %looping through groups
    group_ind=find(river_group==j);

    for i=1:d %looping through days
        ref=classified_reflectance(:, :, i);
        dat=ref(group_ind);
        if cloud_mask(j,i)==1 %only look at clear segments
            for k=1:5
                sum_frequency(k,i,j)=sum(dat==clgr(k)); %sum numbers of pixels
                %of each type
            end
        end
        clear ref
        clear dat
    end

    water=sum_frequency(1, :, j); %sum of water pixels

    date4=find(water > .75*length(group_ind)); %first day 75% open water

    %if breakup is never detected
    if isempty(date4)==1;

```

[illegible]

```
%error_analysis
%Corrects breakup dates for cloud error and determines the window of cloud
%uncertainty
%
%Inputs: Breakup dates, cloud mask
%Outputs: Corrected breakup dates, cloud uncertainty
cd(fp_folder);
load('input_info.mat');
p=input_info(2,1);
cd(reflFolder)
load('breakup_dates_10k.mat');
load('cloud_mask_10k.mat');

d1=input_info(39,year_ind+2);
d2=input_info(40,year_ind+2);
days=d1:1:d2+1;

%Remove missing/bad days
if input_info(2,year_ind+2)~=0
    for n=1:input_info(2,year_ind+2)
        ch=input_info(2+n,year_ind+2);
        days(days==ch)=[ ];
    end
end

error_days=zeros(p,3);

for i=1:p
    day=breakup_days(i);
    ind_day=find(days==day);
    if day==days(end);
        cloud=find(cloud_mask(i,1:ind_day-2)==1);
    else
        %find number of days between detected breakup date and previous
        %non-cloudy day
        cloud=find(cloud_mask(i,1:ind_day-1)==1);
    end
    if isempty(cloud)==1
        last_day=ind_day;
    else
        last_day=cloud(end);
        error_days(i,1)=ind_day-last_day;
        error_days(i,2)=(ind_day-last_day-1)/2; %window (+/-) days
        error_days(i,3)=ceil(day-error_days(i,2)); %corrected breakup date
    end
end

save('breakup_werror.mat','error_days');
cd('C:/Users/swcooley/Desktop/codes/fcodes');
```

```

%frun
%run mackenzie
cd('C:/Users/swcooley/Desktop/codes/fcodes');

run create_ob_folders
breakup_dates=zeros(51,15);
bdates_w_error=zeros(51,15);
error_range=zeros(51,15);
breakup_dates_20k=zeros(25,15);
bdates_w_error_20k=zeros(25,15);
error_range_20k=zeros(25,15);
for year_ind=1:15
    reflFolder=or3_folders(year_ind,:);
    fp_folder=or_fp_folders(3,:);
    cloudFolder=or3c_folders(year_ind,:);
    run friver_classification
    run fcloud_input
    run fbreakup_analysis
    breakup_dates(:,year_ind)=breakup_days;
    run error_analysis
    bdates_w_error(:,year_ind)=error_days(:,3);
    error_range(:,year_ind)=error_days(:,2);
    run fcloud_input_20k
    run fbreakup_analysis_20k
    breakup_dates_20k(:,year_ind)=breakup_days;
    run error_analysis_20k
    bdates_w_error_20k(:,year_ind)=error_days(:,3);
    error_range_20k(:,year_ind)=error_days(:,2);
    clearvars -except breakup_dates or3_folders or_fp_folders or3c_folders
year_ind years fp_folder bdates_w_error error_range error_range_20k
breakup_dates_20k bdates_w_error_20k

end
cd(fp_folder);
save('breakup_dates_new.mat','breakup_dates','bdates_w_error','error_range','
breakup_dates_20k','error_range_20k','bdates_w_error_20k');

cd('C:/Users/swcooley/Desktop/codes/fcodes');

```