```
% Load All The Relavant Data
% Please Change Paths Accordingly
% Senstivity with respect to observations
clc % clear console
clear all % clear all variables from the workspace
% Note windows kind of paths
% loads a matrix that contains prior and lat lon domain of inversion
dataPath = uigetdir(path);
% Addpath for code files
addpath(genpath(dataPath))
% repeat covariate as we only have annual
% covariate that is like invariant prior
% Load forward  operator, observations and parameters for Q and R
load([dataPath,'\','data_section_3.1.mat'])
```

*Coordinates of Sites that Measure Methane and

other details about observations*

```
towerNames={'ONT', 'FUL', 'CMP', 'GRA','USC', 'UCI','PSA','BND'};
timePeriods=2;
% Observation time is stored in amap variable
obsTime=[amap_ONT(:,1) 1*ones(size(amap_ONT,1),1);...% 1 represents ONT
    amap_FUL(:,1) 2*ones(size(amap_FUL,1),1);...% 2 represents FUL
    amap_CMP(:,1) 3*ones(size(amap_CMP,1),1);...
    amap_GRA(:,1) 4*ones(size(amap_GRA,1),1);...
    amap_USC(:,1) 5*ones(size(amap_USC,1),1);...
    amap_UCI(:,1) 6*ones(size(amap_UCI,1),1);...
    amap_PSA(:,1) 7*ones(size(amap_PSA,1),1);...
    amap_BND(:,1) 8*ones(size(amap_BND,1),1)];
% Number of observations available from each tower
towerSize=[size(amap_ONT,1) size(amap_FUL,1) size(amap_CMP,1) ...
    size(amap_GRA,1) size(amap_USC,1) size(amap_UCI,1) size(amap_PSA,1) ...
    size(amap_BND,1)];
% tower coordinates that measures Methane CH4
towerCoord = [34.064167 -117.583611 % Ontario
     33.880417 -117.884122 % Fullerton
     33.873792 -118.276806 % Compton
     34.283889 -118.4725 % Granada Hills
     34.021447 -118.288844 % University of Souther California
     33.644422 -117.844181 % University of California Irvine
     34.1366 -118.12641 % Pasadena
     34.087686 -117.310167]; % San Bernardino
% Time When Observations Were Taken
obsTimePre=[linspace(1,size(H,1),size(H,1))' ...
    obsTime datevec(obsTime(:,1))];
obsTowers=num2cell(obsTime(:,2));
% This is just list tower name with each observation time
obsTowers(obsTime(:,2)==1)={'ONT'}; % Ontario
obsTowers(obsTime(:,2)==2)={'FUL'}; % Fullerton
obsTowers(obsTime(:,2)==3)={'CMP'}; % Compton
obsTowers(obsTime(:,2)==4)={'GRA'}; % Granada Hills
obsTowers(obsTime(:,2)==5)={'USC'}; % University of Souther California
obsTowers(obsTime(:,2)==6)={'UCI'}; % University of California Irvine
obsTowers(obsTime(:,2)==7)={'PSA'}; % Pasadena
obsTowers(obsTime(:,2)==8)={'BND'}; % San Bernardino
```

Plot Spatial Domain or the region of The Study

DOMAIN OF THE STUDY VARIABLES [PLOTTING: NOTHING RELATED TO EQUATIONS]

```
fluxD=size(H,2)/2;% total no of flux grid cells. Two 4 day time periods
% Create grid of latitude and longitude
% Unique latitudes
uniqueLat=unique(latlon(:,2));
% Unique Longitudes
uniqueLon=[unique(latlon(:,1))]';
% Grid of Latitude and Longitudes
gridlon1=repmat(uniqueLon,length(uniqueLat),1);
gridlat1=repmat(uniqueLat,1, length(uniqueLon));
% Now we get indices where data would be plotted
% This is the mask
index=zeros(fluxD,2);
for i = 1:fluxD
    [~,col]=min(abs(latlon(i,1)-gridlon1(1,:)));
    [~,row]=min(abs(latlon(i,2)-gridlat1(:,1)));
    index(i,1) = row;
    index(i,2) = col;
end
% This is our plotting grid
mapgrid=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
for i = 1: fluxD
    mapgrid(index(i,1),index(i,2))=1;
end
titles ='Domain of Study';
h=pcolor(gridlon1,gridlat1,mapgrid);
set(h, 'EdgeColor', 'none');
shading flat; % do not interpolate pixels
axis on;      % display axis
axis tight;   % no white borders
axis image;   % real x,y scaling
set(gca,'fontsize',14)
ylabel('Latitude')
xlabel('Longitude')
title(titles,'FontSize', 14,'Fontname','Arial')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'o','MarkerEdgeColor',[0 .5 .5],...
    'MarkerFaceColor','red' );
text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
    'top','FontSize', 12,'Fontname','Arial','Color','blue')
hold off
```

Compute IOAMI or Jaccard Index and Jensen-Shannon Matrices

for Each Observation See variables areaCover and jenShannon note we convert sparse matrix to full matrix to speed up computation. do not do this for large matrices This is not optimized code for computing IOAMI or JSD. construct H;

```
H=[H_ONT;H_FUL;H_CMP;H_GRA;H_USC;H_UCI;H_PSA;H_BND];
H=full(H);
% Empty IOAMI CORRELATION Matrix to Fill it with Values
IOAMI_CORR=NaN*ones(size(H,1));
% Empty IOAMI Covariance Matrix to Fill it with Values
IOAMI_COV=NaN*ones(size(H,1));
% Empty Jensen Shannon DISTANCE Matrix to Fill it with Values
JSD=NaN*ones(size(H,1));
% Go through footprint matrix row by row. Footprint of each
% observation is a row in a matrix
for i=1:size(H,1)
    for j=1:size(H,1)
        % IOAMI
        [IOAMI_CORR(i,j),~,~]=ioami_jaccard(H(i,:),H(j,:),'normalized');
        % IOAMI
        [IOAMI_COV(i,j),~,~]=ioami_jaccard(H(i,:),H(j,:),'nonnormalized');
        % We have to normalize H for JSD so that it becomes a
        % Probability Distribution
        normlazed_H_1=H(i,:)./sum(H(i,:),2);
        normlazed_H_2=H(j,:)./sum(H(j,:),2);
        % Jensen Shannon using log with base 2 so it ranges between
        % 0 and 1
        JSD(i,j)=jsd(normlazed_H_1,normlazed_H_2,'log_2');
        %[,JSD(i,j)=ioami_jaccard(normlazed_H_1,normlazed_H_2,
        %'nonnormalized');
    end
end
% Note IOAMI is correlation matrix
JSD=1-JSD; %convert JSD distance matrix to correlation matrix
% Note to get IOAMI correlation matrix to distance matrix do
% IOAMI = 1- IOAMI
% Eigen Spectrum of IOAMI & JSD. It is check for positive
% semidefiniteness
eigen_IOAMI_CORR=eig(IOAMI_CORR);
eigen_IOAMI_COV=eig(IOAMI_COV);
eigen_JSD=eig(JSD);
% Test for correlation matrix
% (1) Symmetric
% (2) Ranges between -1 to 1. JSD only ranges between 0 and 1 and IOAMI by
% Definition should also be between 0 and 1 for footprints or Jacobian
% (3) It should be positive semidefinite
```

*Plot Eigenvalues of the IOAMI and JSD Correlation Matrices of H to

see they are Positive Definite*

```
close all % close all existing figures
% Note All Eigenvalues should be >=0 which shows that it is Positive
% Definite Matrix
```

```
plot(eigen_IOAMI_CORR); % line plot of eigen values of IOAMI
hold on % hold to plot another line
plot (eigen_JSD); % line plot of eigen values of IOAMI
% Labels and Title for Plots
ylabel('Eigen Values',"FontSize",14)
xlabel('Eigen Value No',"FontSize",14)
legend({'IOAMI','JSD'},'Location','northwest',"FontSize",14)
title('Eigenspectrum','FontSize',14)
grid on % show grid on the plot
set(gca,'FontSize',14)
hold off
```

Plot IOAMI Correlation Matrix of the Jacobian (H) to see its Structure

Empty display to create Gap between Figures and Tables of Matrices

```
disp(' ');
disp(' ');
disp(' ');
% Image Display of Correlation Matrix
imagesc(IOAMI_CORR)
ylabel('Col No',"FontSize",14)
xlabel('Row No',"FontSize",14)
axis off;
title('IOAMI Correlation Matrix','FontSize',14)
color=colorbar; color.Label.String = 'Correlation';
colormap Jet;
set(gca,'FontSize',14)
```

PLOT JSD Correlation Matrix of the Jacobian (H) to see its Structure

Empty display to create Gap between Figures and Tables of Matrices

```
disp(' ');
disp(' ');
disp(' ');
% Image Display of Correlation Matrix
imagesc(JSD)
ylabel('Col No',"FontSize",14)
xlabel('Row No',"FontSize",14)
title('JSD Correlation Matrix','FontSize',14)
color=colorbar; color.Label.String = 'Correlation'; color.FontSize=14;
colormap Jet;
set(gca,'FontSize',14)
```

Compare IOAMI and JSD Correlation Matrices

The matrix with lower condition number is better and same goes with respect to the Norm. Especially, if it is used as representative of spatio-temporal correlation in model data mismatch matrix variations

```
cond_IOAMI=cond(IOAMI_CORR);
cond_JSD=cond(JSD);
% compute frobenius norm
```

```
norm_IOAMI=norm(IOAMI_CORR,'fro');
norm_JSD=norm(JSD,'fro');
% compute distance between two correlation matrices formula from
% Correlation Matrix Distance, a Meaningful Measurefor Evaluation of
% Non-Stationary MIMO Channels
correlation_distance=1-(trace(IOAMI_CORR*JSD)/(norm_IOAMI*norm_JSD));
% note range of correlation distance is between 0 - 1 with 0 being same
% and 1 being completely dissimilar. For details see:
% See: Herdin, Markus, et al. "Correlation matrix distance, a
% meaningful measure for evaluation of non-stationary MIMO channels."
% 2005 IEEE 61st Vehicular
% Technology Conference. Vol. 1. IEEE, 2005.
% Smaller condition number means better behaved matrix for constructing a
% coveriance matrix
disp(['*****************************************************'])
disp(['Distance between IOAMI and JSD Correlation Matrix is ::', ...
    num2str(correlation_distance)])
disp(['*****************************************************'])
disp(['condition Number of IOAMI Correlation Matrix is ::', ...
    num2str(cond_IOAMI)])
disp(['condition Number of JSD Correlation Matrix is ::', ...
    num2str(cond_JSD)])
disp(['*****************************************************'])
disp(['Frobenius Norm of IOAMI Matrix is ::', num2str(norm_IOAMI)])
disp(['Frobenius NORM of JSD Matrix is ::', num2str(norm_JSD)])
disp(['*****************************************************'])
```

Plot Small Sub-Matrix of IOAMI in the units of Jacobian and Check

full IOAMI in the units of the Jacobian is Positive Definite Empty display to create Gap between Figures and Tables of Matrices

```
disp(' ');
disp(' ');
disp(' ');
disp(['Outputting first 5 rows and Columns of IOAMI in Units of', ...
    'ppm/micromoles m-2 sec-1 ::'])
IOAMI_COV(1:5,1:5)
% Check if whole IOAMI SubMatrix is Positive Definite by Plotting its Eigen
% Values
close all % close all existing figures
% Note All Eigenvalues should be >=0 which shows that it is Positive
% Definite Matrix
% Empty display to create Gap between Figures and Tables of Matrices
disp(' ');
disp(' ');
disp(' ');
plot(eigen_IOAMI_COV); % line plot of eigen values of IOAMI
ylabel('Eigen Values',"FontSize",14)
xlabel('Eigen Value No',"FontSize",14)
legend({'IOAMI in units of Jacobian'},'Location','northwest',"FontSize",14)
title({'Eigenspectrum of IOAMI', 'in the units of Jacobian'},'FontSize',14)
grid on % show grid on the plot
```

```
set(gca,'FontSize',14)
hold off
```

*Put IOAMI and Jaccard Index Matrix in Cell Array for Checking Time

of Observation* Put Area Stat in Cell Array With Tower Names

```
tempAreaCover=NaN*ones(size(IOAMI_CORR,1)+2, size(IOAMI_CORR,2)+2);
tempAreaCover(3:end,3:end)=IOAMI_CORR;
tempAreaCover=num2cell(tempAreaCover);
for i=1:size(H,1)
    tempAreaCover{i+2,2}=datestr(obsTimePre(i,2),'yyyymmddHHMM');
    tempAreaCover{2,i+2}=datestr(obsTimePre(i,2),'yyyymmddHHMM');
end
tempAreaCover(3:end,1)=obsTowers(:,1);
tempAreaCover(1,3:end)=obsTowers(:,1)';
tempAreaCover{1,1}='TOWERS';
tempAreaCover{2,2}='TIME';
tempAreaCover{1,2}=[];
tempAreaCover{2,1}=[];
% Empty display to create Gap between Figures and Tables of Matrices
disp(' ');
disp(' ');
disp(' ');
disp('IOAMI CORRELATION MATRIX SEE VARIABLE TEMP AREA COVER')
disp(tempAreaCover);
```

*Plot footprint Ioami and JSD correlation of observation from ONT site

collected 2015-10-23::1900 with all other observations* **See varable tempAreaCover for observation time with IOAMI correlation matrix**

Observation time is same for both JSD and IOAMI correlation matrix

```
close all % close all existing figures
% Note All correlation is between 0 and 1
% Note Correlation with Itself is one;
% Empty display to create Gap between Figures
disp(' ');
disp(' ');
disp(' ');
plot(IOAMI_CORR(1:end,1)); % line plot of IOAMI correlation of ONT obs
% with all other obs
hold on % hold to plot another line
plot (JSD(1:end,1)); % line plot of IOAMI correlation of ONT obs with
% all other obs
% Labels and Title for Plots
ylabel('Correlation',"FontSize",14)
xlabel({'Observation (see first column of','variable tempAreaCover for',...
    'time and towers'},"FontSize",14)
legend({'IOAMI','JSD'},'Location','northwest',"FontSize",14)
title({'Correlation between footprint of an ONT Observation',...
    ' (2015-10-23::1900) with other footprints'}','FontSize',14)
```

```
grid on % show grid on the plot
set(gca,'FontSize',14)
hold off
```

COMPUTE SPATIO-TEMPORAL AREA OF DOMINANCE (STAD)

STAD: We WILL APPLY STAD ON MEAN SENSTIVITY DUE TO TOTAL NO OF OBSERVATIONS THAT VARIES BY SITE Compute Mean Senstivity

```
obsTowers(obsTime(:,2)==1)={'ONT'}; % Ontario
obsTowers(obsTime(:,2)==2)={'FUL'}; % Fullerton
obsTowers(obsTime(:,2)==3)={'CMP'}; % Compton
obsTowers(obsTime(:,2)==4)={'GRA'}; % Granada Hills
obsTowers(obsTime(:,2)==5)={'USC'}; % University of Souther California
obsTowers(obsTime(:,2)==6)={'UCI'}; % University of California Irvine
obsTowers(obsTime(:,2)==7)={'PSA'}; % Pasadena
obsTowers(obsTime(:,2)==8)={'BND'}; % San Bernardino
% Mean H Column Wise
H_Mean_Senstivity=[mean(H_ONT)' mean(H_FUL)' mean(H_CMP)' mean(H_GRA)...
    ' mean(H_USC)' ...
    mean(H_UCI)' mean(H_PSA)' mean(H_BND)'];
% we have two time period for which we are computing fluxes therefore we
% sum senstivities to get it for one time-period that would cover a 4-day
% overlapping time period
H_Mean_Senstivity=full(H_Mean_Senstivity(1:fluxD,:)+...
    H_Mean_Senstivity(fluxD+1:end,:));
% Find Gridcells of Dominance
loopLimit=size(H_Mean_Senstivity,2);
storeDominanceIndex=NaN*ones(fluxD,1);
maxH=full(max(H_Mean_Senstivity,[],2));
for i=1:loopLimit
    zeroH=maxH-H_Mean_Senstivity(:,i);
    iR=find(zeroH==0);
    storeDominanceIndex(iR,1)=i;
end
```

PLOT STAD

This is our plotting grid

```
mapgrid=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
for i = 1: fluxD
    mapgrid(index(i,1),index(i,2))=storeDominanceIndex(i);
end
titles ='STAD FOR ALL SITES';
h=pcolor(gridlon1,gridlat1,mapgrid);
set(h, 'EdgeColor', 'none');
shading flat; % do not interpolate pixels
axis on;      % display axis
axis tight;   % no white borders
axis image;   % real x,y scaling
set(gca,'fontsize',16)
ylabel('Latitude')
```

```
xlabel('Longitude')
title(titles,'FontSize', 16,'Fontname','Arial')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'o','MarkerEdgeColor',[0 .5 .5],...
    'MarkerFaceColor','red' );
text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
    'top','FontSize', 16,'Fontname','Arial','Color','#EDB120')
hold off
```

```
clear grid* IOAMI* index i j JSD* eig* correlatio* dataPath* ...
    normalized* mapgrid norm* ans col color cond* iR* ...
    H_Mean_Senstivity latlon loopLimit titles store* temp* ...
    time* no h max* unique* zero* nFluxes Area dates_ ...
    fluxD nfluxes mask ...
    noTowers row X obs* tower*
```