```
% Written by Vineet Yadav and Subhomoy Ghosh 3/25/2022
% To go with paper
% Metrics for assessing Linear Inverse Problems: A case study of a Trace
% Gas Inversion

% Load All The Relavant Data
% Please Change Paths Accordingly
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_3.2.mat'])
```

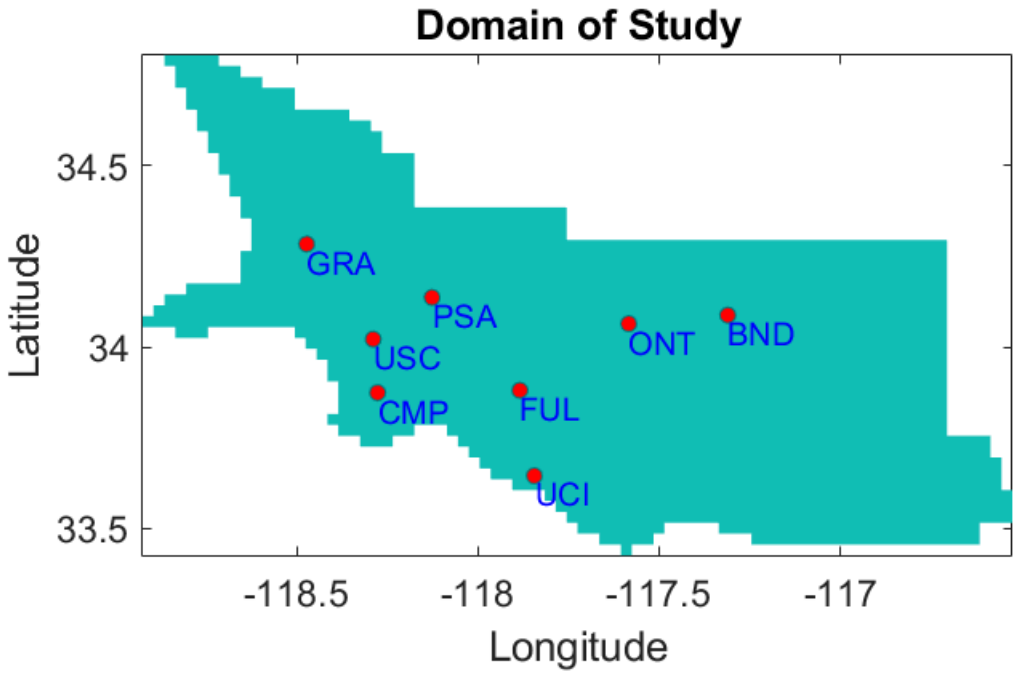
Coordinates of Sites that Measure Methane and other details about observations

```
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
```




```
[~,~,~,~,~,W1,~] = upscale(X(1:fluxD,1), ...
    uniqueLon',uniqueLat,index,2,'mean');

[outvecmat,outmat,lons_out,lats_out,index_out,W2,Wi]  = upscale(X(fluxD+1:end,1), ...
    uniqueLon',uniqueLat,index,2,'mean');
W=blkdiag(W1,W2);

Qa =  W* Q * W';
Xa = [W1 * X(1:fluxD,1); W2 * X(fluxD+1:end,1)];

%% for plotting replace shat_ridge_direct_av by shat
glona=repmat(lons_out',length(lats_out),1);
glata=repmat(lats_out,1,length(lons_out));

Ha=H*W';
```

Compute Fluxes Original Resolution

```
% COMPUTE FLUXES
% All the inputs are in the iput data file
% z are observations
% x is covariance
% R model data mismatch
% Q is prior error covariance
% H is Jacobian Matrix
```

```
% By using Eq. 13 & 14  we can compute fluxes as follow:
A=H*X;
psi=H*Q*H'+R;
ipsi=psi\speye(length(psi)); % This expression is
% equivalent to : inverse(psi)
omega=A'*ipsi*A;
iomega=omega\speye(length(omega)); % This expression is defined in Equation 15 in paper
% equivalent to : inverse(omega)
betaHat = iomega*A'*ipsi*z; % See Eq. 13 in paper
epsilon=Q*H'*ipsi*(z-H*X*betaHat);
shat = X*betaHat+epsilon;% See Eq. 12 in paper
modelRes=(X*iomega*A'*ipsi+Q*H'*ipsi-Q*H'*ipsi*A*iomega*A'*ipsi)*H;
DOFS=sum((diag(modelRes(1:fluxD,1:fluxD))+diag(modelRes(fluxD+1:fluxD*2,fluxD+1:fluxD*2
```

Compute Fluxes Coarser Resolution

```
X=Xa;
Q=Qa;
H=Ha;
A=H*X;
psi=H*Q*H'+R;
fluxD=501;
ipsi=psi\speye(length(psi)); % This expression is
% equivalent to : inverse(psi)
omega=A'*ipsi*A;
iomega=omega\speye(length(omega)); % This expression is defined in Equation 15 in paper
% equivalent to : inverse(omega)
betaHat = iomega*A'*ipsi*z; % See Eq. 13 in paper
epsilon=Q*H'*ipsi*(z-H*X*betaHat);
coarseShat = X*betaHat+epsilon;% See Eq. 12 in paper
modelResCoarse=(X*iomega*A'*ipsi+Q*H'*ipsi-Q*H'*ipsi*A*iomega*A'*ipsi)*H;
DOFSCoarse=sum((diag(modelResCoarse(1:fluxD,1:fluxD))+...
    diag(modelResCoarse(fluxD+1:fluxD*2,fluxD+1:fluxD*2)))/2);
```

```
disp(['*****************************************************'])
```

```
**************************************************
```

```
disp(['DOFS Original : ',num2str(DOFS)])
```

```
DOFS Original : 51.2386
```

```
disp(['*****************************************************'])
```

```
**************************************************
```

```
disp(['DOFS Coarse : ', num2str(DOFSCoarse)])
```

```
DOFS Coarse : 12.1443
```