```
% 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));
% Load forward  operator, observations and parameters for Q and R
%load([dataPath,'SnspaperData_test_2015_2016_69_70.mat'])
load([dataPath,'\','data_section_3.2.mat'])
```

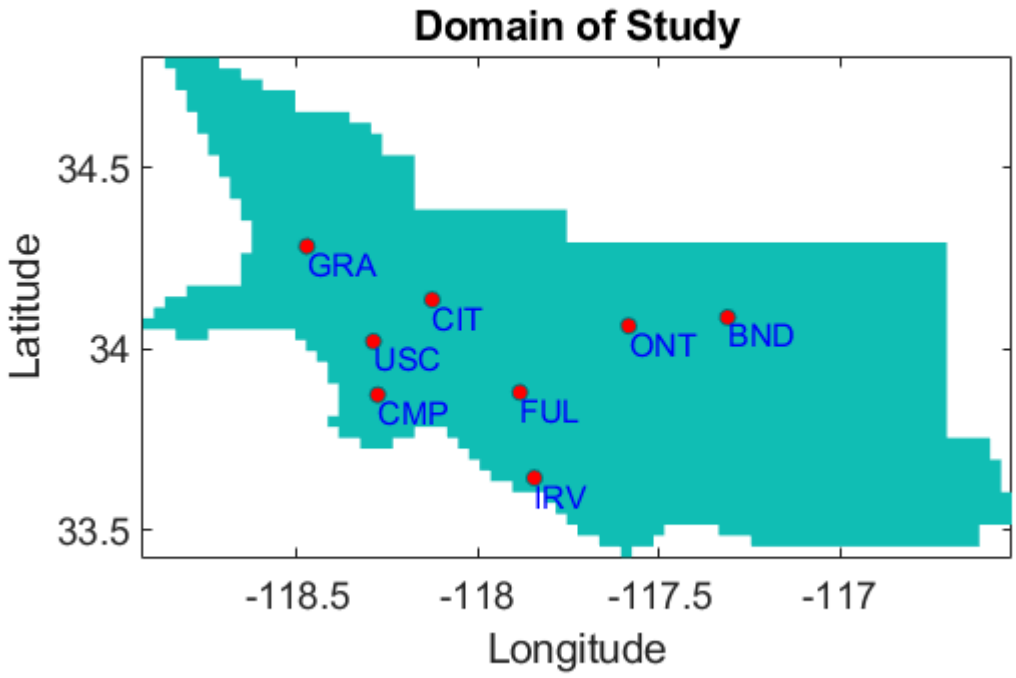
Coordinates of Sites that Measure Methane and other details about observations

```
towerNames={'ONT', 'FUL', 'CMP', 'GRA','USC', 'IRV','CIT','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))];
% This would be updated after inversion
obsTimePost=obsTimePre;
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

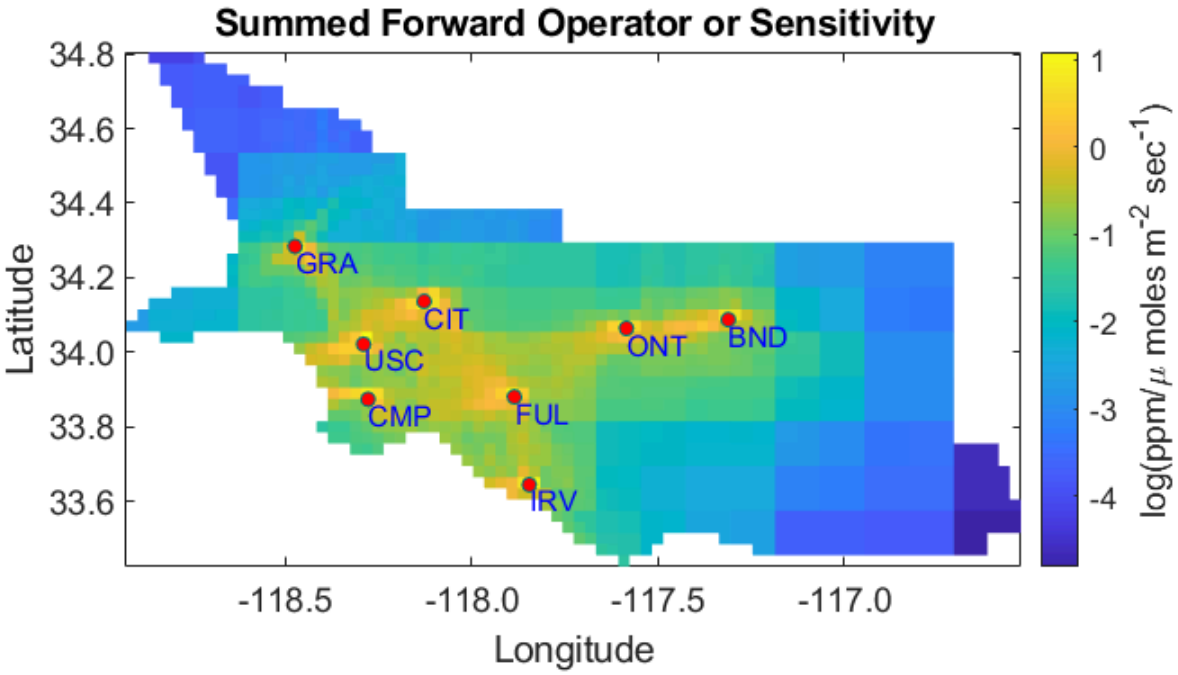
```
% DOMAIN OF THE STUDY VARIABLES [PLOTTING: NOTHING RELATED TO EQUATIONS]
fluxD=size(H,2)/2;% total no of flux grid cells. Two 4 day time periods
% Create grid of latitude and longitude
% Unique latitudes
uniqueLat=unique(latlon(:,2));
% Unique Longitudes
uniqueLon=unique(latlon(:,1))';
% Grid of 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
```



Plot Sensitivity

```
sumH=sum(H);
sumH=sumH(:,1:fluxD)+sumH(:,fluxD+1:end);
sumH=sumH';
mapgrid=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
for i = 1: fluxD
    mapgrid(index(i,1),index(i,2))=log(sumH(i));
end
titles ='Summed Forward Operator or Sensitivity';
figure('Renderer', 'painters', 'Position', [10 10  700  400])
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
colormap("parula")
m=colorbar;
ylabel(m, 'log(ppm/\mu moles m^{-2} sec^{-1})',...
    'Fontname','Arial','FontSize', 14)
set(gca,'fontsize',14)
ylabel('Latitude')
xlabel('Longitude')
hold on
title(titles,'FontSize', 14,'Fontname','Arial')
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')
fmt=ytickformat;
ytickformat('%.1f')
fmt=xtickformat;
xtickformat('%.1f')
hold off
```




```
%print([dataPath,'\','senstivities.png'],'-dpng','-r500')
```

```
% Geostat Estimation Equation for Fluxes
% Reference:
% Michalak, Anna M., Lori Bruhwiler, and Pieter P. Tans.
% A geostatistical approach to surface flux
% estimation of atmospheric trace gases. Journal
% of Geophysical Research: Atmospheres 109.D14 (2004).
```

$$\hat{\mathbf{s}} = \mathbf{X}\hat{\beta} + \mathbf{Q}\mathbf{H}^T\Psi^{-1}(\mathbf{z} - \mathbf{H}\mathbf{X}\hat{\beta})$$

(Equation 11 in paper)

```
% Definition of Symbols
```

$\hat{\mathbf{s}}$ = fluxes to be estimated (units : micromoles $m^{-2}sec^{-1}$)

$\mathbf{H}$ = forward operator or a jacobian $\left(\text{units} : \frac{\text{ppm}}{\text{micromoles } m^{-2}sec^{-1}}\right)$

$\mathbf{Q}$ = prior error covariance matrix (units : micromoles $m^{-2}sec^{-1})^2$

$\mathbf{X}$ = covariates related to fluxes

$\hat{\beta}$ = weights on covariates $i.e, \mathbf{X}$

$\mathbf{z}$ = observations (units : ppm)

$\mathbf{R}$ = observational error covariance or model data mismatach (units : ppm^2)

```
% Dimensions of the Each Quantity/Symbols in the
% Equation
```

$\hat{\mathbf{s}}$ is $(m, 1)$, $\mathbf{H}$ is (n, m), $\mathbf{Q}$ is (k, k), $\mathbf{R}$ is (n, n),
$\mathbf{X}$ is (m, p), $\mathbf{z}$ is $(n, 1)$ and β is $(p, 1)$

```
% We further define:
```

$\mathbf{A} = \mathbf{HX}$, $\Omega = \mathbf{A^T}\Psi^{-1}\mathbf{A}$ and $\Psi^{-1} = \mathbf{HQH}^T + \mathbf{R}$

```
% Replacing Beta in Equation 1 by Equation 2
```

$\hat{\beta} = \Omega^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{z}$

(Equation 12 in Paper)

```
% We get
```

$$\frac{\partial \hat{\mathbf{s}}_G}{\partial \mathbf{z}} = \mathbf{X}\Omega^{-1}\,\mathbf{A}^T\,\Psi^{-1}\mathbf{z} + \mathbf{QH}^T\Psi^{-1}(z - \mathbf{A}\Omega^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{z})$$

(Equation 16)

$$\epsilon = \mathbf{QH}^T\Psi^{-1}(z - \mathbf{A}\Omega^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{z})$$

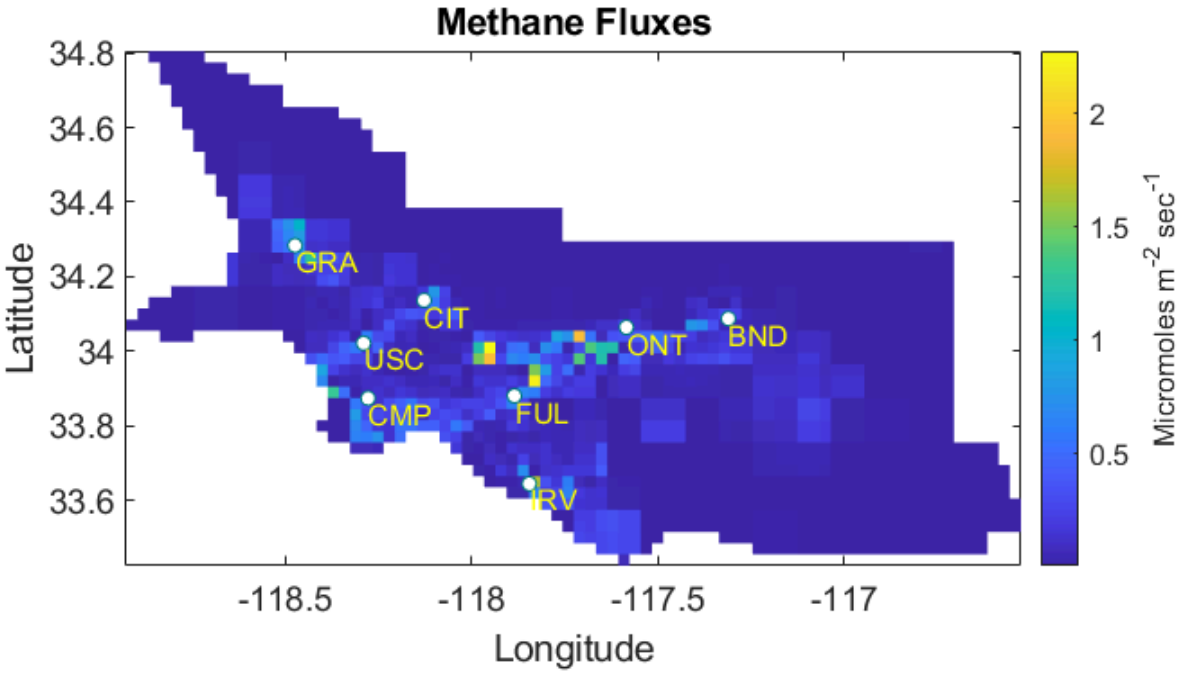
Compute Fluxes From Geostat Method

```
% 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
```

Plot Estimated Fluxes

```
% Plot Fluxes [PLOTTING: NOTHING RELATED TO EQUATIONS]
% This is our plotting grid
% For demonstration we plot absolute value of shat as some values of shat
% is negative whereas the CH4 fluxes cannot go negative
mapgrid=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
for i = 1: fluxD
    mapgrid(index(i,1),index(i,2))=abs(shat(i+1826));
end
titles ='Methane Fluxes';
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)
caxis([min(abs(shat(1826:end))) max(abs(shat(1826:end)))])
colormap("default")
m=colorbar;
ylabel(m, 'Micromoles m^{-2} sec^{-1}','FontSize', 12,'Fontname','Arial')
set(m,'fontsize',12);
ylabel('Latitude')
xlabel('Longitude')
title(titles,'FontSize', 14,'Fontname','Arial')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'o','MarkerEdgeColor',[0 .5 .5],...
    'MarkerFaceColor','auto' );
text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment','top',...
    'FontSize', 12,'Fontname','Arial','Color','y')
hold off
```



Compute Senstivity of Estimated Fluxes To Observations $\frac{\partial \hat{\mathbf{s}}_G}{\partial \mathbf{z}}$

$$\frac{\partial \hat{\mathbf{s}}_G}{\partial \mathbf{z}} = \mathbf{X}\Omega^{-1}\mathbf{A}^T\Psi^{-1} + \mathbf{Q}\mathbf{H}^T\Psi^{-1} - \mathbf{Q}\mathbf{H}^T\Psi^{-1}\mathbf{A}\Omega^{-1}\mathbf{A}^T\Psi^{-1} = \Lambda$$

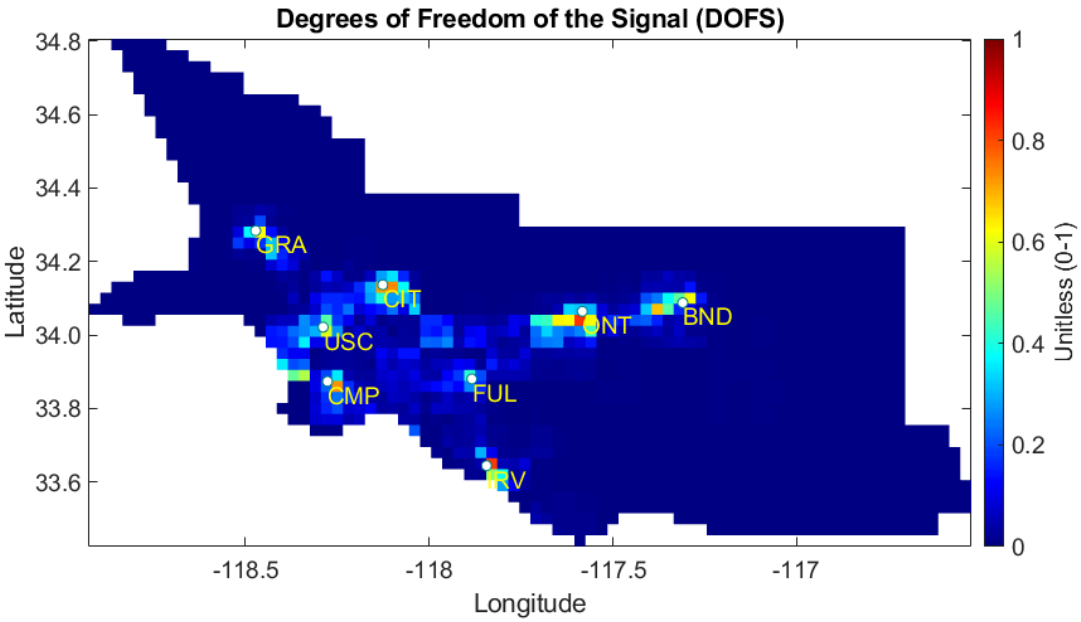
(Equation 16 in paper)

```
% UNITS: (μ Moles m2 sec-1)/ppm
delsG_delZ=(X*iomega*A'*ipsi+Q*H'*ipsi-Q*H'*ipsi*A*iomega*A'*ipsi);
```

Compute and Plot Model Resolution Matrix

```
% Compute
modelRes=(X*iomega*A'*ipsi+Q*H'*ipsi-Q*H'*ipsi*A*iomega*A'*ipsi)*H;
DOFS=(diag(modelRes(1:fluxD,1:fluxD))+diag(modelRes(fluxD+1:fluxD*2,fluxD+1:fluxD*2)))/2;
% PLot
mapgrid=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
for i = 1: fluxD
    mapgrid(index(i,1),index(i,2))=DOFS(i);
end
titles ='Degrees of Freedom of the Signal (DOFS)';
figure('Renderer', 'painters', 'Position',[16 16 1000 500])
t = tiledlayout(1,1,'TileSpacing','Compact','Padding','Compact');
nexttile
```

```
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)
caxis([0 1])
colormap("jet")
m=colorbar;
ylabel(m, 'Unitless (0-1)','FontSize', 14,'Fontname','Arial')
ytickformat('%,.1f')
set(m,'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','auto' );
text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment','top',...
    'FontSize', 14,'Fontname','Arial','Color','y')
hold off
print('basinFluxes.png','-dpng','-r500')
```



Compute $\frac{\partial \hat{\mathbf{s}}_G}{\partial \mathbf{X}}$

$$\frac{\partial \widehat{\mathbf{s}_G}}{\partial \mathbf{X}} = \mathbf{K}_z \otimes (\mathbf{I} + (\mathbf{M}\mathbf{A}^T - \mathbf{X}\boldsymbol{\Omega}^{-1}\mathbf{A}^T - \mathbf{Q}\mathbf{H}^T)\boldsymbol{\Psi}^{-1}\mathbf{H}) + (\mathbf{X}\boldsymbol{\Omega}^{-1} - \mathbf{M}) \otimes \left(\mathbf{F}_z - \mathbf{K}\mathbf{A}^T\boldsymbol{\Psi}^{-1}\mathbf{H}\right)$$

where

$$\mathbf{K}_z = \mathbf{z}^T\boldsymbol{\Psi}^{-1}\mathbf{A}\boldsymbol{\Omega}^{-1}$$

$$\mathbf{M} = \mathbf{Q}\mathbf{H}^T\boldsymbol{\Psi}^{-1}\,\mathbf{A}\boldsymbol{\Omega}^{-1}$$

$$\mathbf{F}_z = \mathbf{z}^T\boldsymbol{\Psi}^{-1}\mathbf{H}$$

Equation(20) in paper

Note we do not $\frac{\partial \widehat{\mathbf{s}_G}}{\partial \mathbf{X}_{ij}}$ version of the equation as for this smaill problem $\frac{\partial \widehat{\mathbf{s}_G}}{\partial \mathbf{X}}$ can be directly obtained by using Kronecker Form

```
Kz=z'*ipsi*A*iomega;
M=Q*H'*ipsi*A*iomega;
Fz=z'*ipsi*H;
part1=eye(fluxD*timePeriods)+(M*A'-X*iomega*A'-Q*H')*ipsi*H;
part2=X*iomega-M;
part3=Fz-Kz*A'*ipsi*H;
delsG_delX=kron(Kz,part1)+kron(part2,part3);
delsG_delX=full(delsG_delX);
```

Compute $\frac{\partial \widehat{\mathbf{s}_G}}{\partial \gamma_i}$ or senstivity with respect to Q parameters.

$$\frac{\partial \widehat{\mathbf{s}_G}}{\partial \gamma_i} = \left(-\mathbf{X}\boldsymbol{\Omega}^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{H} + \mathbf{I}_k - \mathbf{Q}\mathbf{H}^T\Psi^{-1}\mathbf{H} + \mathbf{Q}\mathbf{H}^T\Psi^{-1}\mathbf{A}\Omega^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{H}\right)\frac{\partial \mathbf{Q}}{\partial \gamma_i}\mathbf{H}^T\Psi^{-1}(\mathbf{z} - \mathbf{A}\Omega^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{z})$$

Equation 32 in the paper

```
% Compute delsG_delQ Q Units:
% micromoles m^2 sec^-1/(micromoles m^2 sec^-1)^2
% parameters number of parameters for Q
delsG_delGamma=NaN*ones(size(shat,1),size(parameters,1)-length(towerSize));
for i = 1:size(parameters,1)-length(towerSize)
    Q_Qi=eye(length(Q));
    delsG_delGamma(:,i)=(-X*iomega*A'*ipsi*H+eye(size(H,2))-Q*H'*ipsi*H+...
        Q*H'*ipsi*A*iomega*A'*ipsi*H)*Q_Qi*H'*ipsi*(z-A*iomega*A'*ipsi*z);
end
% For an arbitrary full prior covariance matrix a entry by entry
% senstivity can be given as:
```

```
%Note derivative of scalar * identity matrix is just identity matrix
% as given below. This is the form of Q used in this live script and paper
```

$$\mathbf{Q} = \gamma \begin{bmatrix} 1 & 0 & 0 \\ 0 & 1 & 0 \\ 0 & 0 & 1 \end{bmatrix}$$

whre γ is a scaling factor optimized through restricted maximum likelihood (REML).

Units of α are $(\text{micromoles } m^{-2}\text{sec}^{-1})^2$

Compute $\frac{\partial \widehat{\mathbf{s}}_G}{\partial \alpha_i}$ or senstivity with respect to R parameters.

$$\frac{\partial \widehat{\mathbf{s}}_G}{\partial \alpha_i} = \left(-\mathbf{X}\Omega^{-1}\mathbf{A}^T - \mathbf{B} + \mathbf{C}\mathbf{A}\Omega^{-1}\mathbf{A}^T\right)\Psi^{-1}\frac{\partial \mathbf{R}}{\partial \alpha_i}\Psi^{-1}(\mathbf{z} - \mathbf{A}\Omega^{-1}\mathbf{A}^T\Psi^{-1}\mathbf{z})$$

$$\mathbf{B} = \mathbf{Q}\mathbf{H}^T$$

$$\mathbf{C} = \mathbf{B}\Psi^{-1}$$

Equation (34 in the paper)

```
% In this study R for a case study is specified as:
```

$$\mathbf{R} = \begin{bmatrix} \sigma^2_{\text{Tower } A} & 0 & 0 & 0 & 0 & 0 \\ 0 & \sigma^2_{\text{Tower } A} & 0 & 0 & 0 & 0 \\ 0 & 0 & \sigma^2_{\text{Tower } B} & 0 & 0 & 0 \\ 0 & 0 & 0 & \sigma^2_{\text{Tower } B} & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & \sigma^2_{\text{Tower } n} \end{bmatrix}$$

```
% Given the structure of delsG_delGamma in Eq. 31 Senstivity
% with respect to particular parameter in R (i.e. delR/delRi) or R_Ri
% like variance for a particular tower specified along a
% diagonal can be specified as:
```

$$\frac{\partial \mathbf{R}}{\partial \alpha_i} = \begin{bmatrix} 1 & 0 & 0 & 0 & 0 & 0 \\ 0 & 1 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \\ 0 & 0 & 0 & 0 & \dots & 0 \\ 0 & 0 & 0 & 0 & 0 & 0 \end{bmatrix}$$

```
% CODE to compute del shat/del Ri Units: micromoles
% m^2 sec^-1/(ppm)^2
cumtowersize=cumsum(towerSize);
% cumulative sum of total tower observations
```

```
delsG_delalpha=NaN*ones(size(shat,1),length(cumtowersize));
B=Q*H';
C=B*ipsi;
for i = 1: length(towerSize)
    if i == 1 % this step is necessary to get appropriate
        % indices to create derivative matrices
        beginMeas=1;
        endMeas=cumtowersize(i);
    elseif i>1
        beginMeas=cumtowersize(i-1)+1;
        endMeas=cumtowersize(i);
    end
    R_Ri=zeros(size(R,1),1);
    R_Ri(beginMeas:endMeas)=1;
    R_Ri=diag(R_Ri);
    delsG_delalpha(:,i)=(-X*iomega*A'-B+C*A*iomega*A')*...
        ipsi*R_Ri*ipsi*...
        (z-A*iomega*A'*ipsi*z);
end
```

Compute $\frac{\partial \hat{\mathbf{s}}_G}{\partial \beta}$

$$\frac{\partial \hat{\mathbf{s}}_G}{\partial \beta} = \mathbf{X} - \mathbf{CA}$$

Equation 20 in the paper

```
delsG_delB=X-C*A;
```

Return Importances Seperately

```
weights_delsdelz = return_importance(delsG_delZ,shat,'sorted');
weights_delsdelR = return_importance(delsG_delalpha,shat,'sorted');
weights_delsdelX = return_importance(delsG_delX,shat,'sorted');
```

Rank Towers

```
cumtowersize=cumsum(towerSize);
indices_ranking=[[1 cumtowersize(1:end-1)+1]' cumtowersize(1:end)'];
totalSum=cell(length(towerSize),3);
totalSum(:,1)=towerNames;
for i=1:length(towerSize)
    iR=find(weights_delsdelz(:,1)>=indices_ranking(i,1) & ...
        weights_delsdelz(:,1)<=indices_ranking(i,2));
    totalSum{i,2}=sum(weights_delsdelz(iR,2));
    totalSum{i,3}=[iR weights_delsdelz(iR,2)];
end
Ranking_Towers=sortrows(totalSum, -2);
```

Create empty matrix for storing data for Creating Boxplot

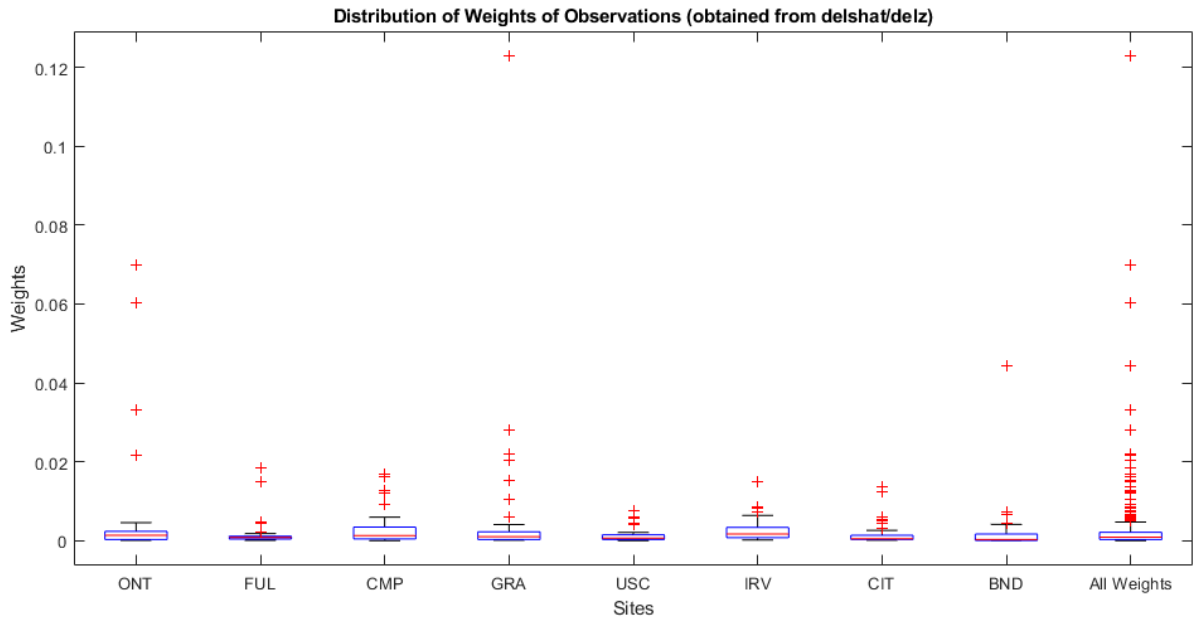
```
emptyBox=NaN*ones(40,length(towerSize));
for i=1:length(towerSize)
    extent=length(totalSum{i, 3}(:,end));
    emptyBox(1:extent,i)=sortrows((totalSum{i, 3}(:,end)),-1);
end
```

Create Boxplot

```
boxplotData=NaN*ones(length(emptyBox(:)), size(emptyBox,2)+1);
boxplotData(1:40,1:size(emptyBox,2))=emptyBox;
boxplotData(:,end)=emptyBox(:);
boxplot(boxplotData,'Labels',{'ONT', 'FUL', 'CMP', 'GRA','USC', ...
```

```
Warning: Unable to set 'Position', 'InnerPosition', 'OuterPosition', or 'ActivePositionProperty' for objects
 in a TiledChartLayout
```

```
    'IRV','CIT','BND','All Weights'})
xlabel('Sites')
ylabel('Weights')
title('Distribution of Weights of Observations (obtained from delshat/delz)')
```



Scale and Combine to find which one of delsdelZ, delsdelR, delsdelX

```
% Note we first scale all the derivatives between 0 and 1 and then sum them
% and once again scale them
```

Scaled derivative of $\frac{\partial \hat{\mathbf{s}}_G}{\partial \mathbf{z}}$ only one column for z

```
derivative=delsG_delZ;
derivative_scaledZ=(derivative-repmat(min(derivative),size(derivative,1),1)) ./ ...
    (repmat(max(derivative),size(derivative,1),1)-...
    repmat(min(derivative),size(derivative,1),1));
derivative_summed=sum(derivative_scaledZ,2);
delsdelZ_scaled=(derivative_summed-...
    repmat(min(derivative_summed),size(derivative_summed,1),1)) ./...
    (repmat(max(derivative_summed),size(derivative_summed,1),1)-...
    repmat(min(derivative_summed),size(derivative_summed,1),1));
```

Scaled derivative of $\frac{\partial \hat{\mathbf{s}}_G}{\partial \alpha_i}$ only one column for R

```
derivative=delsG_delalpha;
derivative_scaledR=(derivative-repmat(min(derivative),size(derivative,1),1)) ./ ...
    (repmat(max(derivative),size(derivative,1),1)-...
    repmat(min(derivative),size(derivative,1),1));
derivative_summed=sum(derivative_scaledR,2);
delsdelR_scaled=(derivative_summed-repmat(min(derivative_summed),...
    size(derivative_summed,1),1)) ./...
    (repmat(max(derivative_summed),size(derivative_summed,1),1)-...
    repmat(min(derivative_summed),size(derivative_summed,1),1));
```

Scaled derivative of $\frac{\partial \hat{\mathbf{s}}_G}{\partial \mathbf{X}}$ only one column for X

```
%% Scaled derivative of R
derivative=delsG_delX;
derivative_scaledX=(derivative-repmat(min(derivative),size(derivative,1),1)) ./ ...
    (repmat(max(derivative),size(derivative,1),1)-...
    repmat(min(derivative),size(derivative,1),1));
derivative_summed=sum(derivative_scaledX,2);
delsdelX_scaled=(derivative_summed-repmat(min(derivative_summed),...
    size(derivative_summed,1),1)) ./...
    (repmat(max(derivative_summed),size(derivative_summed,1),1)-...
    repmat(min(derivative_summed),size(derivative_summed,1),1));
```

Scaled derivative of $\frac{\partial \hat{\mathbf{s}}_G}{\partial \beta}$ only one column for Beta

```
derivative=delsG_delB;
derivative_scaledB=(derivative-repmat(min(derivative),size(derivative,1),1)) ./ ...
    (repmat(max(derivative),size(derivative,1),1)-...
    repmat(min(derivative),size(derivative,1),1));
derivative_summed=sum(derivative_scaledB,2);
delsdelB_scaled=(derivative_summed-repmat(min(derivative_summed),...
    size(derivative_summed,1),1)) ./...
```

```
    (repmat(max(derivative_summed),size(derivative_summed,1),1)-...
    repmat(min(derivative_summed),size(derivative_summed,1),1));
```

Scaled derivative of $\frac{\partial \hat{\mathbf{s}}_G}{\partial \gamma_i}$ only one column for Q

```
derivative=delsG_delGamma;
derivative_scaledQ=(derivative-repmat(min(derivative),size(derivative,1),1)) ./ ...
    (repmat(max(derivative),size(derivative,1),1)-...
    repmat(min(derivative),size(derivative,1),1));
derivative_summed=sum(derivative_scaledQ,2);
delsdelQ_scaled=(derivative_summed-repmat(min(derivative_summed),...
    size(derivative_summed,1),1)) ./...
    (repmat(max(derivative_summed),size(derivative_summed,1),1)-...
    repmat(min(derivative_summed),size(derivative_summed,1),1));
```

Create Independent Variables

```
independent_variables=[delsdelZ_scaled delsdelR_scaled delsdelX_scaled ...
    delsdelQ_scaled delsdelB_scaled];

% independent_variables=[delsdelZ_scaled delsdelR_scaled...
%     delsdelQ_scaled delsdelB_scaled];
correlation_=corrcoef([shat independent_variables]);
correlation_names={'variables','shat','delz','delR', 'delX','delQ','delBeta'};
correlation_evaluation=cell(length(correlation_names));
correlation_evaluation(1,1:length(correlation_names))=correlation_names;
correlation_evaluation(2:end,1)=correlation_names(2:end)';
correlation_evaluation(2:end,2:end)=num2cell(correlation_)
```

correlation_evaluation = 7×7 cell

	1	2	3	4	5	6	7
1	'variables'	'shat'	'delz'	'delR'	'delX'	'delQ'	'delBeta'
2	'shat'	1	0.2985	-0.5594	-0.5004	-0.2159	0.2443
3	'delz'	0.2985	1	-0.0770	-0.3766	-0.1075	0.4695
4	'delR'	-0.5594	-0.0770	1	-0.0726	-0.1797	0.0437
5	'delX'	-0.5004	-0.3766	-0.0726	1	0.2833	-0.3585
6	'delQ'	-0.2159	-0.1075	-0.1797	0.2833	1	-0.2532
7	'delBeta'	0.2443	0.4695	0.0437	-0.3585	-0.2532	1

Get overall importance of z, R, X, Beta, Q

```
% Note index 1 is z, 2 is R, 3 is X, 4 is Beta, 5 is Q
weights_overall = return_importance(independent_variables,shat,'unsorted');
disp('Overall Importance')
```

Overall Importance

```
Overall_Importance={'z' weights_overall(1,2); 'R' weights_overall(2,2); ...
    'X' weights_overall(3,2); 'Q' weights_overall(4,2);'B' weights_overall(5,2)}
```

Overall_Importance = 5×2 cell

	1	2
1	'z'	0.0496
2	'R'	0.5370
3	'X'	0.3134
4	'Q'	0.0646
5	'B'	0.0353

```
Overall_Importance=sortrows(Overall_Importance,-2);
```

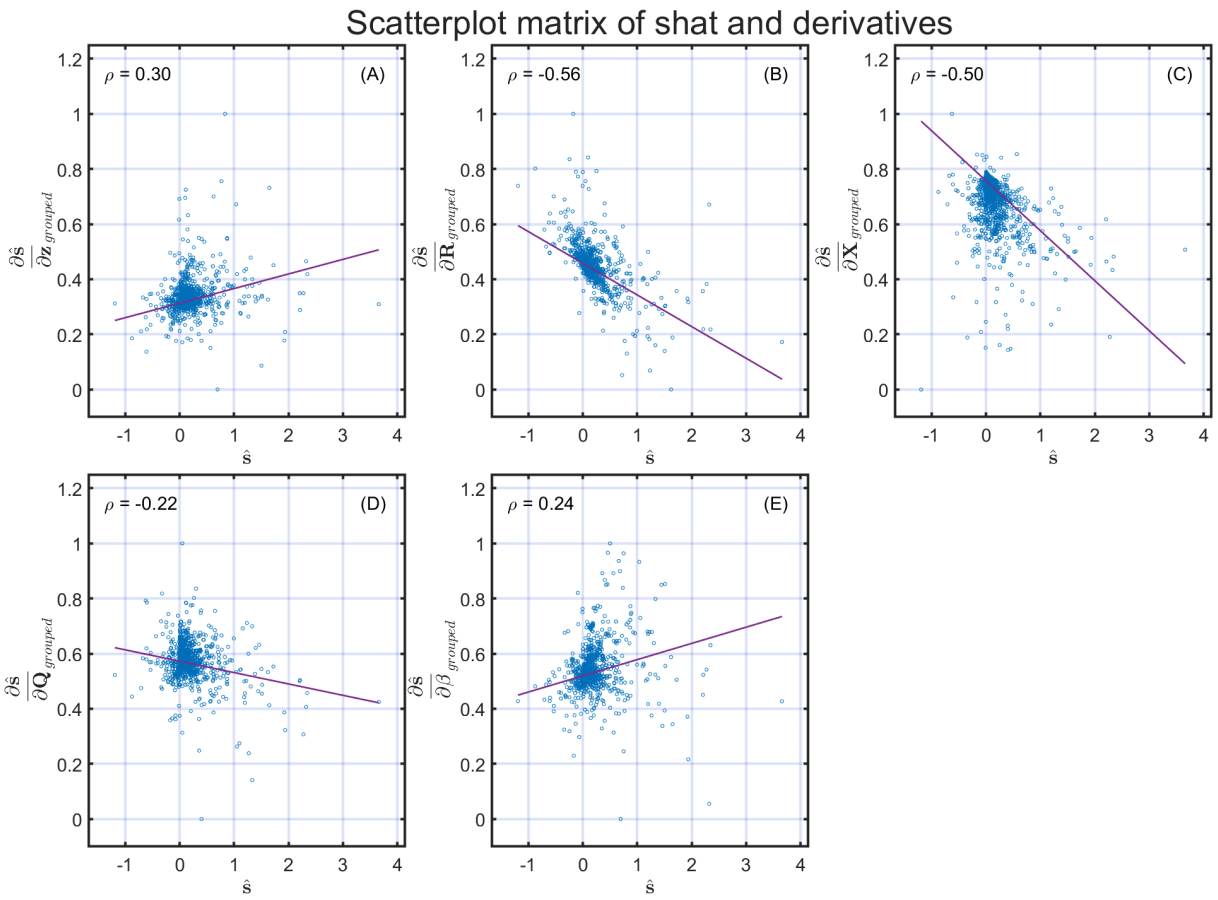
Plot Scatter and Best Fit Line for overall importance

```
close all
% Empty display to create Gap between Figures and Tables of Matrices
disp(' ');
```

```
disp(' ');
```

```
disp(' ');
```

```
%fcnCorrMatrixPlot([shat independent_variables], ...
% correlation_names(2:end),'Scatterplot matrix of shat and derivatives');
figure_last_scatter(shat, independent_variables,...
```




```
    'Scatterplot matrix of shat and derivatives');
```

Create Grid Maps for Plotting

```
mapMostImpObs=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapLeastImpObs=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapScaledZ=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapScaledR=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapScaledX=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapScaledQ=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapScaledB=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapgridMost=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapgridHMost=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
mapgridHLeast=ones(size(gridlat1,1),size(gridlon1,2))*NaN;
% because we will plot for onse time period
H1=H';
for i = 1: fluxD
    mapgridHMost(index(i,1),index(i,2))=H1(i,weights_delsdelz(1,1)) +...
        H1(i+fluxD,weights_delsdelz(1,1));
    mapgridHLeast(index(i,1),index(i,2))=H1(i,weights_delsdelz(end,1))+...
        H1(i+fluxD,weights_delsdelz(end,1));
    mapMostImpObs(index(i,1),index(i,2))=delsG_delZ(i,weights_delsdelz(1,1));
    mapLeastImpObs(index(i,1),index(i,2))=delsG_delZ(i,weights_delsdelz(end,1));
    mapScaledZ(index(i,1),index(i,2))=delsdelZ_scaled(i+1826);
```

```
    mapScaledR(index(i,1),index(i,2))=delsdelR_scaled(i+1826);
    mapScaledX(index(i,1),index(i,2))=delsdelX_scaled(i+1826);
    mapScaledQ(index(i,1),index(i,2))=delsdelQ_scaled(i+1826);
    mapScaledB(index(i,1),index(i,2))=delsdelB_scaled(i+1826);
end
mapScaledZ=real(log(mapScaledZ));
mapScaledR=real(log(mapScaledR));
mapScaledX=real(log(mapScaledX));
mapScaledQ=real(log(mapScaledQ));
mapScaledB=real(log(mapScaledB));
```

Plot Most and Least Important Observation

```
% Empty display to create Gap between Figures and Tables of Matrices
disp(' ');
```

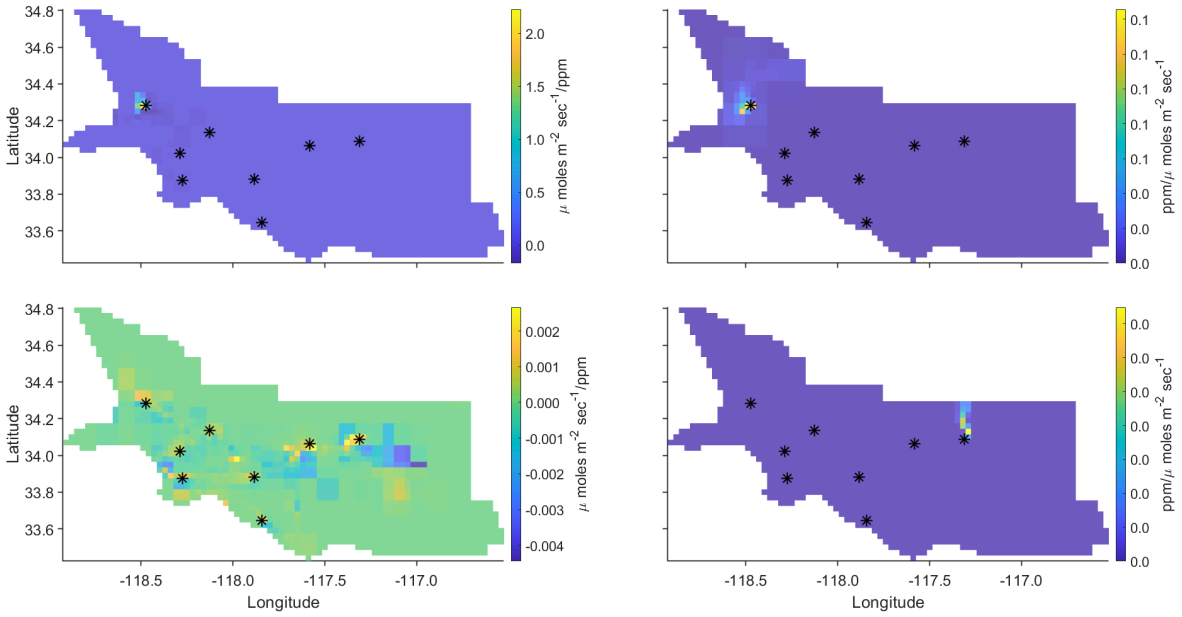
```
disp(' ');
```

```
disp(' ');
```

```
close all
figure('Renderer', 'painters', 'Position', [12 12  2000  1000])
t = tiledlayout(2,2,'TileSpacing','Compact','Padding','Compact');
nexttile
% z & shat
h=pcolor(gridlon1,gridlat1, (mapMostImpObs));
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
%text(-117,34.7, ['z = ',num2str(round(z(weights_delsdelz(1,1)),2)) ' ppm'])
set(gca,'Fontsize',12); colormap("default");
h.FaceAlpha=0.75;% Get Tick Values
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
% caxis([min(min(mapMostImpObs)) max(max(mapMostImpObs))])
% caxis([0 1])

ylabel(m, '\mu moles m^{-2} sec^{-1}/ppm',...
    'Fontname','Arial','FontSize', 18)
ylabel('Latitude','FontSize', 12);
%xlabel('Longitude','FontSize', 12)
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
```

```
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
%legend('Enhancement',[num2str(round(z(weights_delsdelz(1,1)),2)) ' ppm']) ;
set(gca,'fontsize',18,'xticklabel',{[]},'TickDir','out');
set(gca,'yticklabel',num2str(get(gca,'ytick')','%.1f'))
hn=gca;
hn.LineWidth=1.3;
box off
hold off

nexttile
% z & shat
h=pcolor(gridlon1,gridlat1, mapgridHMost);
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
h.FaceAlpha=0.75;% Get Tick Values
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
% caxis([min(min(mapgridHMost)) max(max(mapgridHMost))])
ylabel(m, 'ppm/\mu moles m^{-2} sec^{-1}',...
    'Fontname','Arial','FontSize', 18)
%ylabel('Latitude');
%xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
%legend('z',[num2str(round(z(weights_delsdelz(1,1)),2)), 'ppm']) ;
set(gca,'Fontsize',18,'TickDir','out'); colormap("default");
set(gca,'xticklabel',{[]})
set(gca,'yticklabel',{[]})
hn=gca;
hn.LineWidth=1.3;
box off
hold off

nexttile
% z & shat
h=pcolor(gridlon1,gridlat1, real((mapLeastImpObs)));
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
h.FaceAlpha=0.75;% Get Tick Values
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.3f',tix);
% caxis([min(min(mapLeastImpObs)) max(max(mapLeastImpObs))])
%caxis([min(min(mapMostImpObs)) max(max(mapMostImpObs))])
```

```
%caxis([0 1])
ylabel(m, '\mu moles m^{-2} sec^{-1}/ppm',...
    'Fontname','Arial','FontSize', 18)
ylabel('Latitude'); xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
%legend('Enhancement (z; ppm)',num2str(round(z(weights_delsdelz(end,1)),2))) ;
set(gca,'Fontsize',18,'TickDir','out');
set(gca,'yticklabel',num2str(get(gca,'ytick)','%.1f'))
set(gca,'xticklabel',num2str(get(gca,'xtick)','%.1f'))
hn=gca;
hn.LineWidth=1.3;
box off
hold off

nexttile
% z & shat
h=pcolor(gridlon1,gridlat1, mapgridHLeast);
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',12);
h.FaceAlpha=0.75;% Get Tick Values
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
% caxis([min(min(mapgridHLeast)) max(max(mapgridHLeast))])
ylabel(m, 'ppm/\mu moles m^{-2} sec^{-1}',...
    'Fontname','Arial','FontSize', 18)
%set(m,'Fontsize',12);
%ylabel('Latitude');
xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
%legend('Enhancement (z; ppm)', num2str(round(z(weights_delsdelz(end,1)),2))) ;
set(gca,'Fontsize',18,'TickDir','out');
set(gca,'yticklabel',{[]})
set(gca,'xticklabel',num2str(get(gca,'xtick)','%.1f'))
hn=gca;
hn.LineWidth=1.3;
box off
hold off
print('mostleast_vineet_2015.png','-dpng','-r500')
```



Plot Scaled Derivatives

```
% Empty display to create Gap between Figures and Tables of Matrices
disp(' ');
```

```
disp(' ');
```

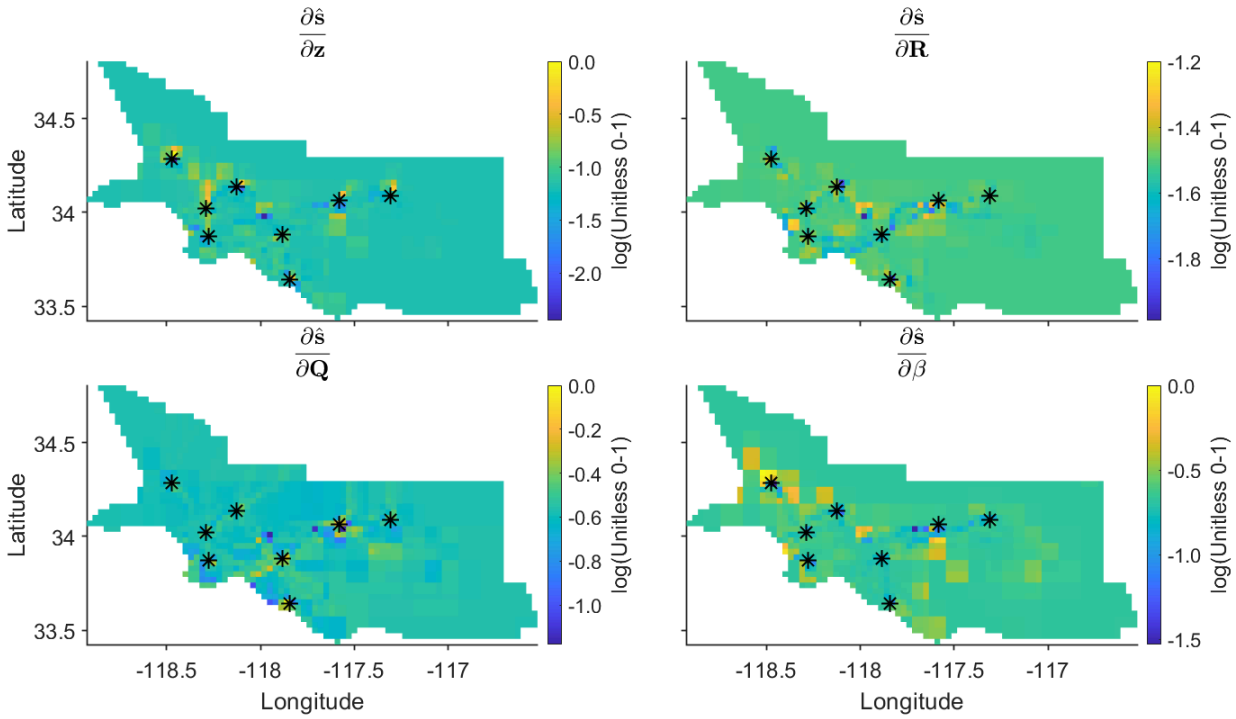
```
disp(' ');
```

```
figure('Renderer', 'painters', 'Position', [12 12  1400  800])
% f = figure();
% f.WindowState = 'maximized';
t = tiledlayout(2,2,'TileSpacing','Compact','Padding','Compact');
%subplot(3,2,1)
nexttile
% z & shat
h=pcolor(gridlon1,gridlat1, mapScaledZ);
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',12); colormap("default");
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
%caxis([min(min(mapScaledZ)) max(max(mapScaledZ))])
```

```
%caxis([0 1])
ylabel(m, 'log(Unitless 0-1)',...
    'Fontname','Arial','FontSize', 18)
ylabel('Latitude');
%xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
title('$$\frac{\partial \hat{\mathbf{s}} }{\partial \mathbf{z}}$$','interpreter',...
    'latex','FontSize',18)
set(gca,'fontsize',18,'TickDir','out','xticklabel',{[]}); colormap("default");
hn=gca;
hn.LineWidth=1.3;
box off
hold off

%subplot(3,2,2)
nexttile
% z & shat
h=pcolor(gridlon1,gridlat1, mapScaledR);
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
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
%caxis([min(min(mapScaledR)) max(max(mapScaledR))])
%caxis([0 1])
ylabel(m, 'log(Unitless 0-1)',...
    'Fontname','Arial','FontSize', 18)
%ylabel('Latitude'); %xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
title('$$\frac{\partial \hat{\mathbf{s}} }{\partial \mathbf{R}}$$',...
    'interpreter','latex','FontSize',18)
set(gca,'Fontsize',18,'TickDir','out','xticklabel',{[]}, ...
    'yticklabel',{[]}); colormap("default");
hn=gca;
hn.LineWidth=1.3;
box off
hold off

% %subplot(3,2,3)
% nexttile
% % z & shat
% h=pcolor(gridlon1,gridlat1, mapScaledX);
% 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
% m=colorbar;
% tix = m.Ticks;                                          % Get Tick Values
% m.TickLabels = compose('%1.1f',tix);
% %caxis([min(min(mapScaledX)) max(max(mapScaledX))])
% %caxis([0 1])
% ylabel(m, 'log(Unitless 0-1)',...
%     'Fontname','Arial','FontSize', 18)
% ylabel('Latitude'); %xlabel('Longitude')
% hold on
% plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
%     'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% % text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
% %     'top','FontSize', 12,'Fontname','Arial','Color','y')
% title('$$\frac{\partial \hat{\mathbf{s}} }{\partial \mathbf{X}}$$','interpreter',...
% 'latex','FontSize',18)
% set(gca,'fontsize',18,'TickDir','out','xticklabel',{[]}); colormap("default");
% hn=gca;
% hn.LineWidth=1.3;
% box off
% hold off

%subplot(3,2,4)
nexttile
% z & shat
h=pcolor(gridlon1,gridlat1,  mapScaledQ);
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',12);
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
%caxis([min(min( mapScaledQ)) max(max( mapScaledQ))])
%caxis([0 1])
ylabel(m, 'log(Unitless 0-1)',...
    'Fontname','Arial','FontSize', 18)
%set(m,'Fontsize',12);
ylabel('Latitude');
xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
title(['$$\frac{\partial \hat{\mathbf{s}} }{\partial ...' ...
    '\mathbf{Q}}$$'],'interpreter','latex','FontSize',18)
set(gca,'Fontsize',18,'TickDir','out');
hn=gca;
```

```
hn.LineWidth=1.3;
box off
hold off

%subplot(3,2,5)
nexttile
% z & shat
h=pcolor(gridlon1,gridlat1,  mapScaledB);
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',12);
m=colorbar;
tix = m.Ticks;                                          % Get Tick Values
m.TickLabels = compose('%1.1f',tix);
%caxis([min(min( mapScaledQ)) max(max( mapScaledQ))])
%caxis([0 1])
ylabel(m, 'log(Unitless 0-1)',...
    'Fontname','Arial','FontSize', 18)
%set(m,'Fontsize',12);
%ylabel('Latitude');
xlabel('Longitude')
hold on
plot(towerCoord(:,2), towerCoord(:,1),'*','MarkerEdgeColor',[0 0 0],...
    'MarkerFaceColor',[0 0 0] ,'MarkerSize', 14,'LineWidth',1.5);
% text(towerCoord(:,2),towerCoord(:,1),towerNames,'VerticalAlignment',...
%     'top','FontSize', 12,'Fontname','Arial','Color','y')
title(['$$\frac{\partial \hat{\mathbf{s}} }{\partial ...', ...
    '\mathbf{\beta}}$$'],'interpreter','latex','FontSize',12)
set(gca,'Fontsize',18,'TickDir','out','yticklabel',{[]});
hn=gca;
hn.LineWidth=1.3;
box off
hold off
```




```
%First formulate variance in diag
varRQ=diag(inv(fisher));
varRQ=diag([varRQ(1:end-1); varRQ(end)]);
% Keep contributions in a matrix (#rows=#fluxes,#columns=#parameters)
contribs=zeros(length(parameters),1);
vartheta=diag(varRQ);
delsdelRQ=[delsG_delalpha delsG_delGamma];
totalFluxes=size(delsdelRQ,1);

for i=1:totalFluxes
    for j=1:size(parameters,1)
        contribs(i,j)= delsdelRQ(i,j).^2 * vartheta(j);
    end
end

% Obtain relative contribution of the Q and R parameters ...
% onto posterior uncertainty
G=sum(contribs);
disp(['Relative contribution of flux sensitivities w.r.t R ...' ...
    'and Q parameters on posterior variance'])
```

```
Relative contribution of flux sensitivities w.r.t R ...and Q parameters on posterior variance
```

```
G./sum(G)
```

```
ans = 1×9
    0.9998    0.0000    0.0000    0.0000    0.0000    0.0001    0.0000    0.0001 ···
```

```
clear A amap* ans Area Fit F f H_* z_* part* ipsi iomega nfluxes nFluxes ...
    noTowers obs* psi Qderiv Rderiv XQR rtowers row omega ...
```

```
dataPath begin* betaHat* col cumtowersize derivative_scaled ...
derivative_summed endMeas i derivative titles mapgrid K m uniqueLat ...
uniqueLon timePeriods towerSize parameters mask h M dates_
```