Supplementary File: Appendix MATLAB GUI for Gridded map of India

%%%%%%%%%%%%%%%%

Excel Files DS_H and DS_Z includes H and Z compnents from all stations from 01 January 2015

%%%%%%%%%%%%%%%%%%%%%%%%%%%

Note: This code is executable on MATLAB VERSION 2023b

%%
%%%%%%%

```
classdef app2 < matlab.apps.AppBase

    % Properties that correspond to app components
    properties (Access = public)
        UIFigure                       matlab.ui.Figure
        GridLayout                     matlab.ui.container.GridLayout
        LeftPanel                      matlab.ui.container.Panel
        ExportFigButton                matlab.ui.control.Button
        ExportRMSEButton               matlab.ui.control.Button
        Plot_HButton                   matlab.ui.control.Button
        NoofGridpointsEditField        matlab.ui.control.NumericEditField
        NoofGridpointsEditFieldLabel   matlab.ui.control.Label
        ContourIntervalEditField       matlab.ui.control.NumericEditField
        ContourIntervalEditFieldLabel  matlab.ui.control.Label
        UITable5                       matlab.ui.control.Table
        Plot_ZButton                   matlab.ui.control.Button
        RMSEButton                     matlab.ui.control.Button
        ExportGridButton               matlab.ui.control.Button
        GriddataButton                 matlab.ui.control.Button
        AverageButton                  matlab.ui.control.Button
        end_hrEditField                matlab.ui.control.NumericEditField
        end_hrEditFieldLabel           matlab.ui.control.Label
        end_dayEditField               matlab.ui.control.NumericEditField
```

```
    end_dayEditFieldLabel          matlab.ui.control.Label
    end_monthEditField             matlab.ui.control.NumericEditField
    end_monthEditFieldLabel        matlab.ui.control.Label
    end_yearEditField              matlab.ui.control.NumericEditField
    end_yearEditFieldLabel         matlab.ui.control.Label
    start_hrEditField              matlab.ui.control.NumericEditField
    start_hrEditFieldLabel         matlab.ui.control.Label
    start_dayEditField             matlab.ui.control.NumericEditField
    start_monthEditField           matlab.ui.control.NumericEditField
    start_monthEditFieldLabel      matlab.ui.control.Label
    start_dayEditFieldLabel        matlab.ui.control.Label
    start_yearEditField            matlab.ui.control.NumericEditField
    start_yearEditFieldLabel       matlab.ui.control.Label
    Load_LocationButton            matlab.ui.control.Button
    Load_dataButton                matlab.ui.control.Button
    UITable4                       matlab.ui.control.Table
    UITable3                       matlab.ui.control.Table
    UITable2                       matlab.ui.control.Table
    UITable                        matlab.ui.control.Table
    RightPanel                     matlab.ui.container.Panel
    UIAxes                         matlab.ui.control.UIAxes
end

% Properties that correspond to apps with auto-reflow
properties (Access = private)
    onePanelWidth = 576;
end

% Callbacks that handle component events
methods (Access = private)
```

```
% Button pushed function: Load_dataButton
function Load_dataButtonPushed(app, event)
     [file, path] = uigetfile('*.xlsx');
    if isequal(file,0)
       msgbox('Please input an Excel file')
    else
        t1 = readtable(fullfile(path, file));
        app.UITable.Data = t1;
    end
end

% Button pushed function: Load_LocationButton
function Load_LocationButtonPushed(app, event)
     [file, path] = uigetfile('*.xlsx');
    if isequal(file,0)
       msgbox('Please input an Excel file')
    else
        t2 = readtable(fullfile(path, file));
        app.UITable2.Data = t2;
    end
end

% Button pushed function: AverageButton
function AverageButtonPushed(app, event)
     f1=get(app.UITable,'Data'); %%%%%%%%%% data %%%%%%%%%%%%%
    f2=get(app.UITable2,'Data'); %%%%%%%%  location %%%%%%%%%%%

    num=table2array(f1);
    num1=table2array(f2);
```

```
        %%%%%%%%%% start year%%%%%%%%%%%%

        st1=(app.start_yearEditField.Value);
        st2=(app.start_monthEditField.Value);
        st3=(app.start_dayEditField.Value);
        st4=(app.start_hrEditField.Value);
%           st5=(app.start_minEditField.Value);

         %%%%%%%%%% end year%%%%%%%%%%%%

        et1=(app.end_yearEditField.Value);
        et2=(app.end_monthEditField.Value);
        et3=(app.end_dayEditField.Value);
        et4=(app.end_hrEditField.Value);
%           et5=(app.end_minEditField.Value);
%%%%%%%%% hours data
        st=[st1 st2 st3 st4];
        et=[et1 et2 et3 et4];


        %%%%%%%%%%%% min data
%           st=[st1 st2 st3 st4 st5];
%           et=[et1 et2 et3 et4 et5];

        %%%%%%%%%%%% avetage part%%%%%%%%%%%%%%

        %%%%%%%%%%%%% for hours datasets

        i=find(st(1)==num(:,1) & st(2)==num(:,2) & st(3)==num(:,3) &
st(4)==num(:,4)) ;
```

```
        j=find(et(1)==num(:,1) & et(2)==num(:,2) & et(3)==num(:,3) &
et(4)==num(:,4)) ;
        new_numZ=num(i:j,5:15);   %%%%%% 5:16 for PBR including
        new_numH=num(i:j,16:26);
        NNZ=mean(new_numZ,'omitnan')';
        NNH=mean(new_numH,'omitnan')';
%


        %%%%%%%%%%%%% for min datasets


%           i=find(st(1)==num(:,1) & st(2)==num(:,2) & st(3)==num(:,3) &
st(4)==num(:,4) & st(5)==num(:,5));
%           j=find(et(1)==num(:,1) & et(2)==num(:,2) & et(3)==num(:,3) &
et(4)==num(:,4) & et(5)==num(:,5));
%           ind=[i,j];
%           new_num=num(i:j,7:18);
%           NN=mean(new_num)';


            %%%55
        avg_data=[num1,NNZ,NNH];


        app.UITable3.Data=avg_data;
%


    end


    % Button pushed function: GriddataButton
    function GriddataButtonPushed(app, event)
        f3=get(app.UITable3,'Data'); %%%%%%%%%% data %%%%%%%%%%%%%
        dpZ=f3(:,1:3);              %%% Z DATA POINTS
        dpH=f3(:,[1:2,4]);          %%% H DATA POINTS
```

```
%%% NAN ROW REMOVAL

nanrows_Z=any(isnan(dpZ),2);
dpZ1=dpZ(~nanrows_Z,:);

nanrows_H=any(isnan(dpH),2);
dpH1=dpH(~nanrows_H,:);

grid_point=(app.NoofGridpointsEditField.Value);
xi = linspace(min([dpZ(:,1)]), max([dpZ(:,1)]), grid_point);
yi = linspace(min([dpZ(:,2)]), max([dpZ(:,2)]), grid_point);

[X,Y]=meshgrid(xi,yi);
idp=[X(:),Y(:)];            %%% Interpolation location points

 nd=size(dpZ1,1);
 nid=size(idp,1);
 intZ=zeros(nid,1);        %%%  interpolated Z values
 intH=zeros(nid,1);        %%%  interpolated H values
 p=4; %%%%% POWER

 for i1=1:nid
    sumweightsZ=0;sumweightsH=0;
    weightedsumZ=0; weightedsumH=0;
    for j1=1:nd
       distanceZ=sqrt ( ( idp(i1,1) - dpZ1(j1,1) )^2 + ( idp(i1,2) - dpZ1(j1,2) )^2
);
       distanceH=sqrt ( ( idp(i1,1) - dpH1(j1,1) )^2 + ( idp(i1,2) - dpH1(j1,2)
)^2 );
       weightZ=1/distanceZ^p;
```

```
            weightH=1/distanceH^p;
            sumweightsZ=sumweightsZ+weightZ;
            sumweightsH=sumweightsH+weightH;
            weightedsumZ=weightedsumZ+weightZ*dpZ1(j1,3);
            weightedsumH=weightedsumH+weightH*dpH1(j1,3);
            intZ(i1)=weightedsumZ/sumweightsZ;
            intH(i1)=weightedsumH/sumweightsH;
        end
    end
    msm=[idp,intZ,intH];       %%%%% iNTERPOLATED( long, lat, Z, H)
    app.UITable4.Data=msm;

end

% Button pushed function: ExportGridButton
function ExportGridButtonPushed(app, event)
    % Get the table data from the UI table
   tableData = app.UITable4.Data;

 % Ask the user for the file name and location
    [fileName, filePath] = uiputfile('*.xlsx', 'Save Table as Excel');

  % Check if the user canceled the operation
    if isequal(fileName, 0) || isequal(filePath, 0)
        return; % User canceled
    end

      % Construct the full file path
        fullFilePath = fullfile(filePath, fileName);
```

```
            % Write the table data to an Excel file
                writematrix(tableData, fullFilePath);

             % Display a message to the user
            msgbox(['Table saved as Excel file: ' fullFilePath], 'File Saved');
        end

        % Button pushed function: RMSEButton
        function RMSEButtonPushed(app, event)
            f5=get(app.UITable3,'Data'); %%%%%%%%% data
%%%%%%%%%%%%
            dpZ=f5(:,1:3);              %%% Z DATA POINTS
            dpH=f5(:,[1:2,4]);          %%% H DATA POINTS

            f6=get(app.UITable4,'Data'); %%%%%%%%% data %%%%%%%%%%%%
            msmZ=f6(:,1:3);             %%% Z DATA POINTS
            msmH=f6(:,[1:2,4]);         %%% H DATA POINTS
            intZ=msmZ(:,3);
            intH=msmH(:,3);

           % Initialize arrays to store matched indices
            matched_indices_dpZ = [];
            matched_indices_dpH = [];
            matched_indices_msmZ = [];
            matched_indices_msmH = [];


  % Loop through each point in dataset
            for i = 1:11
   % Extract latitude and longitude from dataset1
                lat1 = dpZ(i, 2);
```

```
            lon1 = dpZ(i, 1);


    % Find the nearest neighbors
            indicesZ = find(abs(msmZ(:, 1) - lon1) <= 1.57 & abs(msmZ(:, 2) - lat1) <=
1.77);
            indicesH = find(abs(msmH(:, 1) - lon1) <= 1.57 & abs(msmH(:, 2) - lat1)
<= 1.77);


    % Store the matched indices
            matched_indices_dpZ = [matched_indices_dpZ; repmat(i,
numel(indicesZ), 1)];
            matched_indices_msmZ = [matched_indices_msmZ; indicesZ];
            matched_indices_dpH = [matched_indices_dpH; repmat(i,
numel(indicesH), 1)];
            matched_indices_msmH = [matched_indices_msmH; indicesH];
          end


        long=dpZ(:,1); lat=dpZ(:,2);
%%%% RMSE CALCULATION
        PPZ=long(matched_indices_dpZ); PPH=long(matched_indices_dpH);
        QQZ=lat(matched_indices_dpZ);  QQH=lat(matched_indices_dpH);
        RRZ=intZ(matched_indices_msmZ); RRH=intH(matched_indices_msmH);
        SSZ=[PPZ QQZ RRZ]; SSH=[PPH QQH RRH];


        locations = [long,lat];
        rmse_per_obsZ = zeros(size(long,1), 1);
        rmse_per_obsH = zeros(size(long,1), 1);


         for jkl = 1:11
    % Extract latitude and longitude for the current location
            lat_lon = locations(jkl, :);
```

```
    % Find actual and predicted data for the current location
            actual_dataZ = dpZ(jkl, 3);
            actual_dataH = dpH(jkl, 3);
            predicted_dataZ = SSZ(ismember(SSZ(:, 1:2), lat_lon, 'rows'), 3);
            predicted_dataH = SSH(ismember(SSH(:, 1:2), lat_lon, 'rows'), 3);

    % Calculate RMSE for the current location
            rmse_per_obsZ(jkl) = sqrt(mean((actual_dataZ - predicted_dataZ).^2));
            rmse_per_obsH(jkl) = sqrt(mean((actual_dataH - predicted_dataH).^2));
          end

            rmse_ZH=[long,lat,rmse_per_obsZ,rmse_per_obsH];
            app.UITable5.Data=rmse_ZH;

        end

        % Button pushed function: Plot_ZButton
        function Plot_ZButtonPushed(app, event)
            [filename,pathname]=uigetfile('*.shp','Select Shapefile');
            if isequal(filename,0)
               msgbox('Please input a shapefile')
            else
               ind_boundary=shaperead(fullfile(pathname,filename));

            f7=get(app.UITable3,'Data'); %%%%%%%%%% data
%%%%%%%%%%%%%
            dpZ=f7(:,1:3);              %%% Z DATA POIN

            % nanrows1=any(isnan(dpZ),2);
            % dpZ1=dpZ(~nanrows1,:);
```

```
    f8=get(app.UITable4,'Data'); %%%%%%%%% data %%%%%%%%%%%%
    msmZ=f8(:,1:3);              %%% Z DATA POINT

    int_x=dpZ(:,1);
    int_y=dpZ(:,2);
    int_z=msmZ(:,3);

    grid_point=(app.NoofGridpointsEditField.Value);

    xi=linspace(min(int_x),max(int_x),grid_point);
    yi=linspace(min(int_y),max(int_y),grid_point);

    [X,Y]=meshgrid(xi,yi);
    ZZ=reshape(int_z,grid_point,grid_point);

    poly=polyshape(ind_boundary.X,ind_boundary.Y);
    in_bound=inpolygon(X,Y,poly.Vertices(:,1),poly.Vertices(:,2));
    ZZ(~in_bound)=NaN;
    cnt_int=(app.ContourIntervalEditField.Value);
    levels=0:cnt_int:50000 ;%%%% Contour Interval

    contour(app.UIAxes,X,Y,ZZ,levels,'ShowText','on')
    hold on
    geoshow(app.UIAxes,ind_boundary,'FaceColor','none','EdgeColor','black');
    title(app.UIAxes,'Year')
    hold off


   end
end
```

```
    % Changes arrangement of the app based on UIFigure width
    function updateAppLayout(app, event)
        currentFigureWidth = app.UIFigure.Position(3);
        if(currentFigureWidth <= app.onePanelWidth)
            % Change to a 2x1 grid
            app.GridLayout.RowHeight = {480, 480};
            app.GridLayout.ColumnWidth = {'1x'};
            app.RightPanel.Layout.Row = 2;
            app.RightPanel.Layout.Column = 1;
        else
            % Change to a 1x2 grid
            app.GridLayout.RowHeight = {'1x'};
            app.GridLayout.ColumnWidth = {279, '1x'};
            app.RightPanel.Layout.Row = 1;
            app.RightPanel.Layout.Column = 2;
        end
    end
end

% Component initialization
methods (Access = private)

    % Create UIFigure and components
    function createComponents(app)

        % Create UIFigure and hide until all components are created
        app.UIFigure = uifigure('Visible', 'off');
        app.UIFigure.AutoResizeChildren = 'off';
        app.UIFigure.Position = [100 100 640 480];
```

```
        app.UIFigure.Name = 'MATLAB App';
        app.UIFigure.SizeChangedFcn = createCallbackFcn(app,
@updateAppLayout, true);

        % Create GridLayout
        app.GridLayout = uigridlayout(app.UIFigure);
        app.GridLayout.ColumnWidth = {279, '1x'};
        app.GridLayout.RowHeight = {'1x'};
        app.GridLayout.ColumnSpacing = 0;
        app.GridLayout.RowSpacing = 0;
        app.GridLayout.Padding = [0 0 0 0];
        app.GridLayout.Scrollable = 'on';

        % Create LeftPanel
        app.LeftPanel = uipanel(app.GridLayout);
        app.LeftPanel.Layout.Row = 1;
        app.LeftPanel.Layout.Column = 1;

        % Create UITable
        app.UITable = uitable(app.LeftPanel);
        app.UITable.ColumnName = {'Year'; 'Month'; 'Days'; 'Hrs'; 'ALI_Z'; 'ALA_Z';
'GUL_Z'; 'JAI_Z'; 'NAG_Z'; 'PON_Z'; 'RKT_Z'; 'SHL_Z'; 'SIL_Z'; 'TIR_Z'; 'VSK_Z';
'ALI_H'; 'ALA_H'; 'GUL_H'; 'JAI_H'; 'NAG_H'; 'PON_H'; 'RKT_H'; 'SHL_H'; 'SIL_H';
'TIR_H'; 'VSK_H'};
        app.UITable.RowName = {};
        app.UITable.Position = [18 402 111 68];

        % Create UITable2
        app.UITable2 = uitable(app.LeftPanel);
        app.UITable2.ColumnName = {'Longitude'; 'Latitude'};
        app.UITable2.RowName = {};
        app.UITable2.Position = [141 402 111 68];
```

```
        % Create UITable3
        app.UITable3 = uitable(app.LeftPanel);
        app.UITable3.ColumnName = {'Longitude'; 'latitude'; 'averaged Z_data'; 'averaged H_data'};
        app.UITable3.RowName = {};
        app.UITable3.Position = [12 180 103 68];

        % Create UITable4
        app.UITable4 = uitable(app.LeftPanel);
        app.UITable4.ColumnName = {'Longitude'; 'Latitude'; 'Interpolated Z Value'; 'Interpolated H Value'};
        app.UITable4.RowName = {};
        app.UITable4.Position = [8 89 104 82];

        % Create Load_dataButton
        app.Load_dataButton = uibutton(app.LeftPanel, 'push');
        app.Load_dataButton.ButtonPushedFcn = createCallbackFcn(app, @Load_dataButtonPushed, true);
        app.Load_dataButton.Position = [35 371 77 23];
        app.Load_dataButton.Text = 'Load_data';

        % Create Load_LocationButton
        app.Load_LocationButton = uibutton(app.LeftPanel, 'push');
        app.Load_LocationButton.ButtonPushedFcn = createCallbackFcn(app, @Load_LocationButtonPushed, true);
        app.Load_LocationButton.Position = [150 370 94 23];
        app.Load_LocationButton.Text = 'Load_Location';

        % Create start_yearEditFieldLabel
        app.start_yearEditFieldLabel = uilabel(app.LeftPanel);
        app.start_yearEditFieldLabel.HorizontalAlignment = 'right';
```

```
app.start_yearEditFieldLabel.Position = [8 341 59 24];
app.start_yearEditFieldLabel.Text = 'start_year';

% Create start_yearEditField
app.start_yearEditField = uieditfield(app.LeftPanel, 'numeric');
app.start_yearEditField.Position = [76 343 37 22];

% Create start_dayEditFieldLabel
app.start_dayEditFieldLabel = uilabel(app.LeftPanel);
app.start_dayEditFieldLabel.HorizontalAlignment = 'right';
app.start_dayEditFieldLabel.Position = [13 285 55 22];
app.start_dayEditFieldLabel.Text = 'start_day';

% Create start_monthEditFieldLabel
app.start_monthEditFieldLabel = uilabel(app.LeftPanel);
app.start_monthEditFieldLabel.HorizontalAlignment = 'right';
app.start_monthEditFieldLabel.Position = [6 315 69 22];
app.start_monthEditFieldLabel.Text = 'start_month';

% Create start_monthEditField
app.start_monthEditField = uieditfield(app.LeftPanel, 'numeric');
app.start_monthEditField.Position = [76 315 39 22];

% Create start_dayEditField
app.start_dayEditField = uieditfield(app.LeftPanel, 'numeric');
app.start_dayEditField.Position = [76 285 37 22];

% Create start_hrEditFieldLabel
app.start_hrEditFieldLabel = uilabel(app.LeftPanel);
app.start_hrEditFieldLabel.HorizontalAlignment = 'right';
```

```
app.start_hrEditFieldLabel.Position = [6 260 61 22];
app.start_hrEditFieldLabel.Text = 'start_hr';

% Create start_hrEditField
app.start_hrEditField = uieditfield(app.LeftPanel, 'numeric');
app.start_hrEditField.Position = [76 260 37 21];

% Create end_yearEditFieldLabel
app.end_yearEditFieldLabel = uilabel(app.LeftPanel);
app.end_yearEditFieldLabel.HorizontalAlignment = 'right';
app.end_yearEditFieldLabel.Position = [134 341 56 22];
app.end_yearEditFieldLabel.Text = 'end_year';

% Create end_yearEditField
app.end_yearEditField = uieditfield(app.LeftPanel, 'numeric');
app.end_yearEditField.Position = [203 341 37 22];

% Create end_monthEditFieldLabel
app.end_monthEditFieldLabel = uilabel(app.LeftPanel);
app.end_monthEditFieldLabel.HorizontalAlignment = 'right';
app.end_monthEditFieldLabel.Position = [131 315 66 22];
app.end_monthEditFieldLabel.Text = 'end_month';

% Create end_monthEditField
app.end_monthEditField = uieditfield(app.LeftPanel, 'numeric');
app.end_monthEditField.Position = [203 315 37 22];

% Create end_dayEditFieldLabel
app.end_dayEditFieldLabel = uilabel(app.LeftPanel);
app.end_dayEditFieldLabel.HorizontalAlignment = 'right';
```

```
            app.end_dayEditFieldLabel.Position = [138 285 55 22];
            app.end_dayEditFieldLabel.Text = 'end_day';

            % Create end_dayEditField
            app.end_dayEditField = uieditfield(app.LeftPanel, 'numeric');
            app.end_dayEditField.Position = [204 285 36 22];

            % Create end_hrEditFieldLabel
            app.end_hrEditFieldLabel = uilabel(app.LeftPanel);
            app.end_hrEditFieldLabel.HorizontalAlignment = 'right';
            app.end_hrEditFieldLabel.Position = [143 258 42 22];
            app.end_hrEditFieldLabel.Text = 'end_hr';

            % Create end_hrEditField
            app.end_hrEditField = uieditfield(app.LeftPanel, 'numeric');
            app.end_hrEditField.Position = [204 258 36 22];

            % Create AverageButton
            app.AverageButton = uibutton(app.LeftPanel, 'push');
            app.AverageButton.ButtonPushedFcn = createCallbackFcn(app,
@AverageButtonPushed, true);
            app.AverageButton.Position = [186 222 59 26];
            app.AverageButton.Text = 'Average';

            % Create GriddataButton
            app.GriddataButton = uibutton(app.LeftPanel, 'push');
            app.GriddataButton.ButtonPushedFcn = createCallbackFcn(app,
@GriddataButtonPushed, true);
            app.GriddataButton.Position = [128 148 59 23];
            app.GriddataButton.Text = 'Grid data';
```

```
        % Create ExportGridButton
        app.ExportGridButton = uibutton(app.LeftPanel, 'push');
        app.ExportGridButton.ButtonPushedFcn = createCallbackFcn(app,
@ExportGridButtonPushed, true);
        app.ExportGridButton.Position = [192 148 72 23];
        app.ExportGridButton.Text = 'Export Grid';

        % Create RMSEButton
        app.RMSEButton = uibutton(app.LeftPanel, 'push');
        app.RMSEButton.ButtonPushedFcn = createCallbackFcn(app,
@RMSEButtonPushed, true);
        app.RMSEButton.Position = [123 114 51 23];
        app.RMSEButton.Text = 'RMSE';

        % Create Plot_ZButton
        app.Plot_ZButton = uibutton(app.LeftPanel, 'push');
        app.Plot_ZButton.ButtonPushedFcn = createCallbackFcn(app,
@Plot_ZButtonPushed, true);
        app.Plot_ZButton.Position = [133 33 60 23];
        app.Plot_ZButton.Text = 'Plot_Z';

        % Create UITable5
        app.UITable5 = uitable(app.LeftPanel);
        app.UITable5.ColumnName = {'Longitude'; 'Latitde'; 'RMSE_Z'; 'RMSE_H'};
        app.UITable5.RowName = {};
        app.UITable5.Position = [8 10 105 70];

        % Create ContourIntervalEditFieldLabel
        app.ContourIntervalEditFieldLabel = uilabel(app.LeftPanel);
        app.ContourIntervalEditFieldLabel.HorizontalAlignment = 'right';
        app.ContourIntervalEditFieldLabel.Position = [135 67 90 22];
```

```
app.ContourIntervalEditFieldLabel.Text = 'Contour Interval';

% Create ContourIntervalEditField
app.ContourIntervalEditField = uieditfield(app.LeftPanel, 'numeric');
app.ContourIntervalEditField.Position = [227 67 44 22];

% Create NoofGridpointsEditFieldLabel
app.NoofGridpointsEditFieldLabel = uilabel(app.LeftPanel);
app.NoofGridpointsEditFieldLabel.HorizontalAlignment = 'right';
app.NoofGridpointsEditFieldLabel.Position = [126 190 98 22];
app.NoofGridpointsEditFieldLabel.Text = 'No. of Grid points';

% Create NoofGridpointsEditField
app.NoofGridpointsEditField = uieditfield(app.LeftPanel, 'numeric');
app.NoofGridpointsEditField.Position = [230 190 30 22];

% Create Plot_HButton
app.Plot_HButton = uibutton(app.LeftPanel, 'push');
app.Plot_HButton.Position = [200 33 60 23];
app.Plot_HButton.Text = 'Plot_H';

% Create ExportRMSEButton
app.ExportRMSEButton = uibutton(app.LeftPanel, 'push');
app.ExportRMSEButton.Position = [184 114 88 23];
app.ExportRMSEButton.Text = 'Export RMSE';

% Create ExportFigButton
app.ExportFigButton = uibutton(app.LeftPanel, 'push');
app.ExportFigButton.Position = [144 4 88 23];
app.ExportFigButton.Text = 'Export Fig';
```

```
        % Create RightPanel
        app.RightPanel = uipanel(app.GridLayout);
        app.RightPanel.Layout.Row = 1;
        app.RightPanel.Layout.Column = 2;

        % Create UIAxes
        app.UIAxes = uiaxes(app.RightPanel);
        title(app.UIAxes, 'Title')
        xlabel(app.UIAxes, 'X')
        ylabel(app.UIAxes, 'Y')
        zlabel(app.UIAxes, 'Z')
        app.UIAxes.Position = [7 92 349 310];

        % Show the figure after all components are created
        app.UIFigure.Visible = 'on';
    end
end

% App creation and deletion
methods (Access = public)

    % Construct app
    function app = app2

        % Create UIFigure and components
        createComponents(app)

        % Register the app with App Designer
        registerApp(app, app.UIFigure)
```

```
            if nargout == 0
                clear app
            end
        end

        % Code that executes before app deletion
        function delete(app)

            % Delete UIFigure when app is deleted
            delete(app.UIFigure)
        end
    end
end
```

%%
%%%%%%%%%%%%%%

Supporting Figure S1

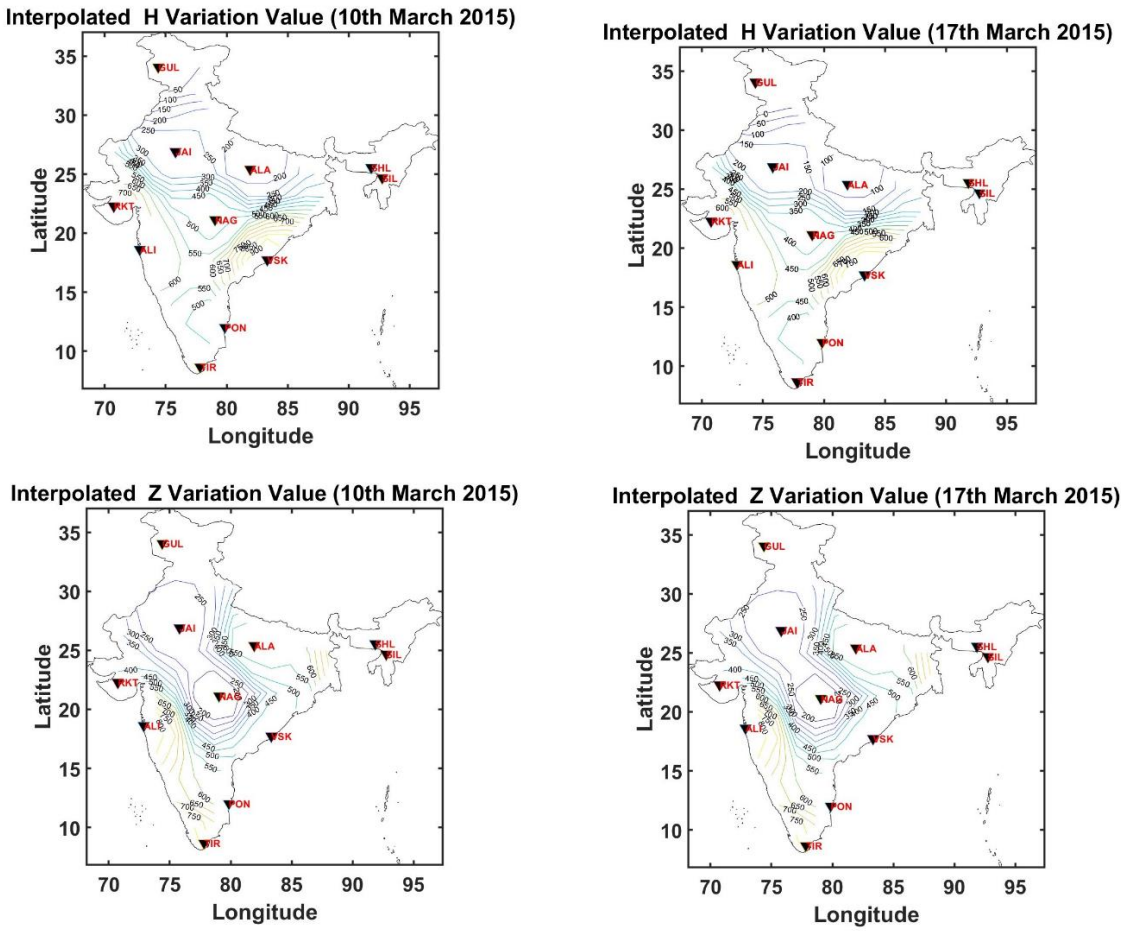

Figure S1: Interpolated gridded H and Z field plots on 10 March 2015 (International Quiet day) and H and Z gridded map for 17 March 2015 (St. Patrick's day geomagnetic storm)

IAGA Code	Data Gaps/Jumps
TIR	
PND	2011, Mar-Dec 2019
VSK	Oct 2014-Feb 2015
ABG	
NGP	2020
RKT	2011 2012 2019 2020
SIL	2011 2012
ALH	Jan 2011-Apr 2013 Nov 2015-Mar 2016 2020
SHL	Jan-Mar, 2015
JAI	2018 2019
GUL	2014-2015 and 2018

Table S1: Data availability in different stations during 2011-2020

Interpolation methods	Advantage	Disadvantage
Kriging method (KRG)	• It has the capability of giving unbiased predictions with minimum variance. • It utilizes rigorous statistical framework which produce optimal and unbiased predictions minimizing the RMSE.	• The disadvantages of this method is that it is more time costlier and computationally intensive than the other methods.
Nearest Neighbour method (NAN)	• Mathematically simpler and effective on small datasets, • It needs no assumptions about data distribution	• In case of larger datasets, the search for the nearest neighbors becomes time-consuming.
Natural Neighbour method (NEN)	• It considers the distribution of data set and allow it to capture local variations providing the data heterogeneity.	• Inaccurate estimates can arise from sparse or clustered data.
Inverse Distance Weighting (IDW)	• Dense evenly space points may also well interpolated (such as flat areas with cliffs). • It does not produce values outside the measurements.	• It has the tendency to generate "bull's-eye" patterns around the data points having of large number of contours. • It does not extrapolate values beyond the range of data.
Radial Basis method (RBF)	• It can generate precise interpolants for high-dimensional datasets with poorly distributed data points.	• It is capable to predict values above the maximum and below the minimum measured values.
Polynomial Regression (POR)	• This method is used for trend analysis where underlying large-scale trends and patterns are analyzed.	• Sometimes the resulting grid may lose local information in the data.
Triangulation with Linear Interpolation (TLP)	• It works best with data that is evenly distributed over the grid area. • This method is fast and does not extrapolate above and below the maximum and minimum value.	• Different triangular facets appear on the map when data sets with sparse areas are used.
Modified Shepard's Method (MSM)	• It does not tend to generate "bull's eye" patterns as it reduce the expressive outliers which cause “bull-eyeing” effects. • It can be adapted to various types of datasets, including 2D and 3D, and can be used with different distance metrics, which can improve results in specific scenarios.	• Using iterative process, this method may reduce numerical errors, improving the precision of the solution. • The flexibility to adjust weights can lead to over-smoothing, where finer details of the data are lost.
Minimum Curvature Method (MNC)	• It is especially effective in creating smooth contour maps, free from sharp breaks or sudden changes, which is crucial for topographical data.	• Minimum Curvature is not an exact Interpolator.
Moving Average (MOA)	• It can extracts information about intermediate-scale trends and variations from large and noisy data.	• The averaging process may blur significant variations in the data, causing the loss of finer details or sharp feature.
Local Polynomial Method (LCP)	• This method is effective in modeling complex and non-linear relationships.	• The edge effect and computational complexity are among the disadvantages associated with this method.

Table S2: Advantage and disadvantages of different interpolation techniques