Earth Resources Technology, Inc
Scientific Programmer
Coding tests

Kazuyoshi Kikuchi (kazuyosh@hawaii.edu)

Submitted 14 June 2021

1. Assessment IRI EDP

I creates a C-based modeling and simulation program that drives IRI model Fortran code as instructed as follows.

1. I created a simple Makefile that can compile iri2016 and generate a shared object/library. In order to make the shared library, execute the following command

   $ make libiri.so

2. A C-program iriexe_main.c along with irisub_wrapper.f90 let you link with the shared library and run the iri2016. To successfully link the main program with the shared library, make sure that the current directory (.) is included in your LD_LIBRARY_PATH. You can make an executable file by carrying out the following command

   $ make iriexe

   The input data are given by way of Fortran NAMELIST. Modify the file "NMPARA" for your needs. Here is an example of NMPARA

   ```
   &nmiri jmag=0,
          xlat=37.8,
          xlon=284.6,
          iy=2021,
          imd=0304,
          iut=1,
          hour=23.,
          vbeg=60.,
          vend=600.,
          vstp=10.,
    &end
   ```

3. When you run iriexe (i.e., $ ./iriexe), a gnuplot plot showing the vertical electron density profile (EDP) like Fig. 3.1 will come up on you screen. Note that it has been confirmed that results are quite consistent with those obtained through CCMC website (https://ccmc.gsfc.nasa.gov/modelweb/models/iri2016_vitmo.php).

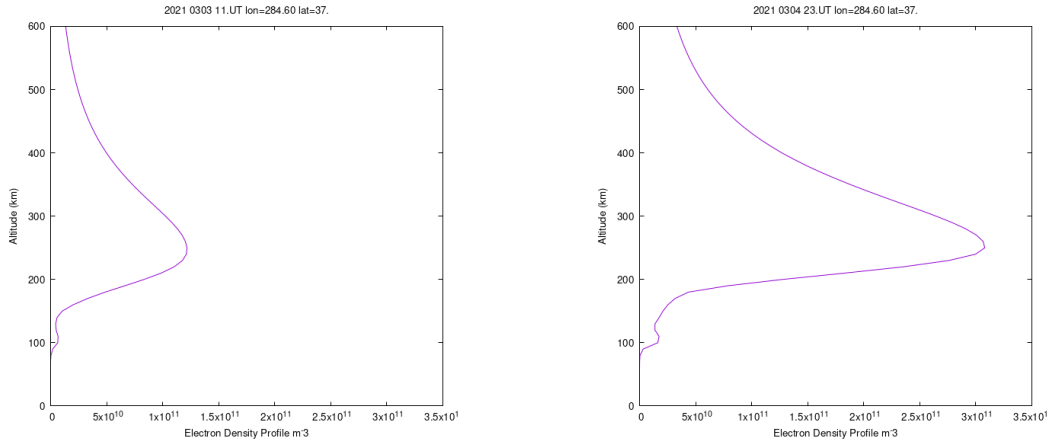

Fig. 1 Vertical electron density profile (EDP) in m^{-3} for (left) 2021 3/3 11UTC and (right) 2021 ¾ 23UTC at (37.8° N, 75.4° W).

4. Lesson/insights: I briefly summarize what I learned by doing this exercise. The EDP varies greatly on diurnal timescale. The two profiles examined here are 12 hours apart and each corresponds to 6 am and 6 pm LT, respectively. Although the overall shapes are similar, the magnitudes of the peaks are very different. Both profiles exhibit two peaks, one is located around ~100 km (height of E layer peak, hmE) and the other ~250 km (height of F2 layer peak, hmF2). It is suggested that both hmE and hmF2 are not much different, while the peak values of EDP are quire different: EDP at hmF2 and hmE at 6 pm LT is greater than that at 60 am LT by a factor of ~3. Qualitatively, this is consistent with our common knowledge that EDP develops during daytime. It is also anticipated that EDP is strongly affected by the seasonal cycle. I am also interested how EDP is influenced by solar activity.

2. Assessment Coordinate Transformation

Let β and r be bearing and range, respectively, R be the radius of the Earth, (λ_1, ϕ_1) and (λ_2, ϕ_2) be the longitude and latitude of the initial and final locations, respectively. A C program, gis_radar_convert.c, contains two functions (GIS2Radar and RtoG) that deal with coordinate conversion between radar and GIS coordinates.

1. GIS2Radar

First consider a case in which the initial and final locations are given and derive the bearing and range along a great circle. The range can be obtained by the harversine formula as follows

$$a = \sin^2(\Delta\phi/2) + \cos(\phi_1) * \cos(\phi_2) * \sin^2(\Delta\lambda/2)$$
$$c = 2 * atan2(\sqrt{a}, \sqrt{1-a})$$
$$r = R * c$$

The bearing can be calculated as follows

$$y = \sin(\Delta\lambda) * \cos(\phi_2)$$
$$x = \cos(\phi_1) * \sin(\phi_2) - \sin(\phi_1) * \cos(\phi_2) * \cos(\Delta\lambda)$$
$$\beta = \text{atan2}(y, x)$$

2. RtoG

Next, let us consider a case in which the initial location, bearing and range are given and derive the final location. The longitude and latitude of the final location can be obtained as follows

$$\phi_2 = \text{asin}(\sin(\phi_1) * \cos(r/R) + \cos(\phi_1) * \sin(r/R) * \cos(\beta))$$
$$\lambda_2 = \lambda_1 + atan2(\sin(\beta) * \sin(r/R) * \cos(\phi_1), \cos(r/R) - \sin(\phi_1) * \sin(\phi_2))$$

3. Example

Consider an example case where the initial and final locations are given as follows:
Initial: Wallops Islands, lat: 37°N, long: 75°W
Final: Puerto Rico, lat: 18°N, long: 66°W
An executable file can be created by

$ make gis_radar_convert

The program calculates

$$\beta = 155°$$

$$r = 2291 \text{ km}$$

Also it has been confirmed that the function RtoG gives the correct longitude and latitude of the final location when the initial location, bearing, range are given.

3. Assessment Interpolation

Python code: interpolate.ipynb

Interpolated distribution of value is obtained by interpolating and extrapolating the given scattered data. I use the so-called radial basis function (RBF) interpolation, an effective method to interpolate high dimensional scattered data. Here seven different radial basis functions are examined (Table 3.1).

Table 3.1 Summary of RBF functions used here

RBF type	Mathematical formula
multiquadric	$\sqrt{(r/\varepsilon)^2+1}$
inverse	$1/\sqrt{(r/\varepsilon)^2+1}$
gaussian	$\exp(-(r/\varepsilon)^2)$
linear	r
cubic	r^3
quintic	r^5
thin plate	$r^2\log(r)$

Figure 3 shows the interpolation results. Overall consistent features are obtained by the RBF functions examined here except for quintic, which employs high order of polynomial resulting in overfitting.

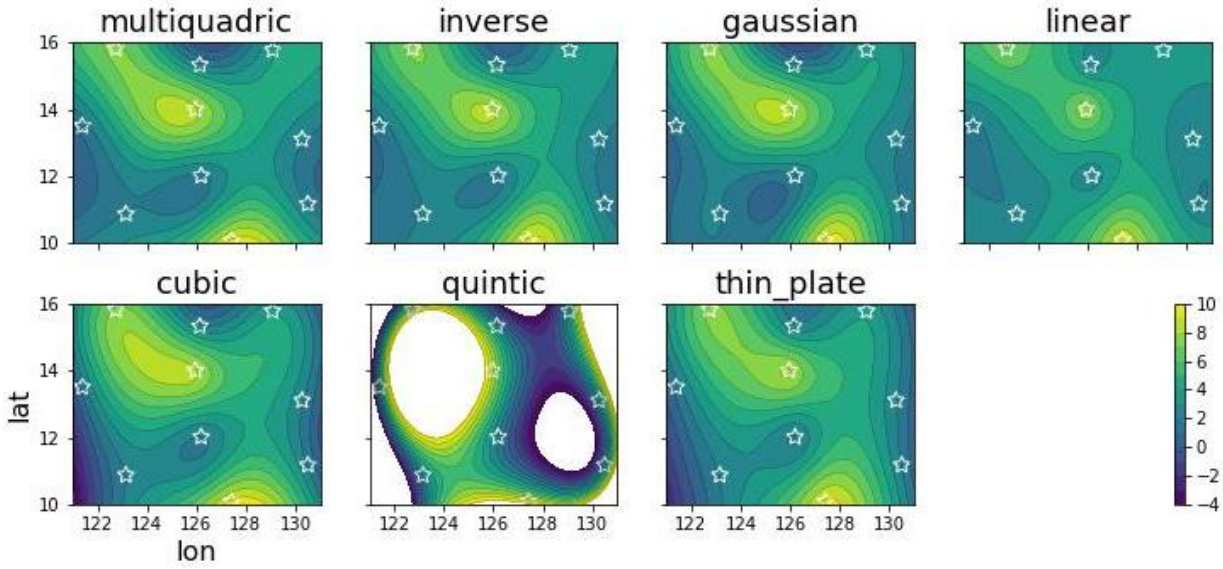

Fig. 3 Interpolated distributions of value using radial basis function (RBF) interpolation based on various functions. The locations of the data given are represented by stars.

4. Assessment Clustering

Python code: cluster.ipynb

I employ K-Means algorithm to cluster the data set "cluster_raw.dat." Before applying the K-Means algorithm, the dataset is normalized so that the mean is 0 and the standard deviation is 1. Shown in Fig. 4.1 is a pair plot matrix of the dataset. It is found that variables x0 and x2 have two obvious peaks in distribution, while other variables have virtually one peak (x4 has a minor peak). Variables x1, x3, x4 have large volumes around 0, although they have large outliers. Some linear relationship may be suggested between some variables (e.g., x1 and x3)..

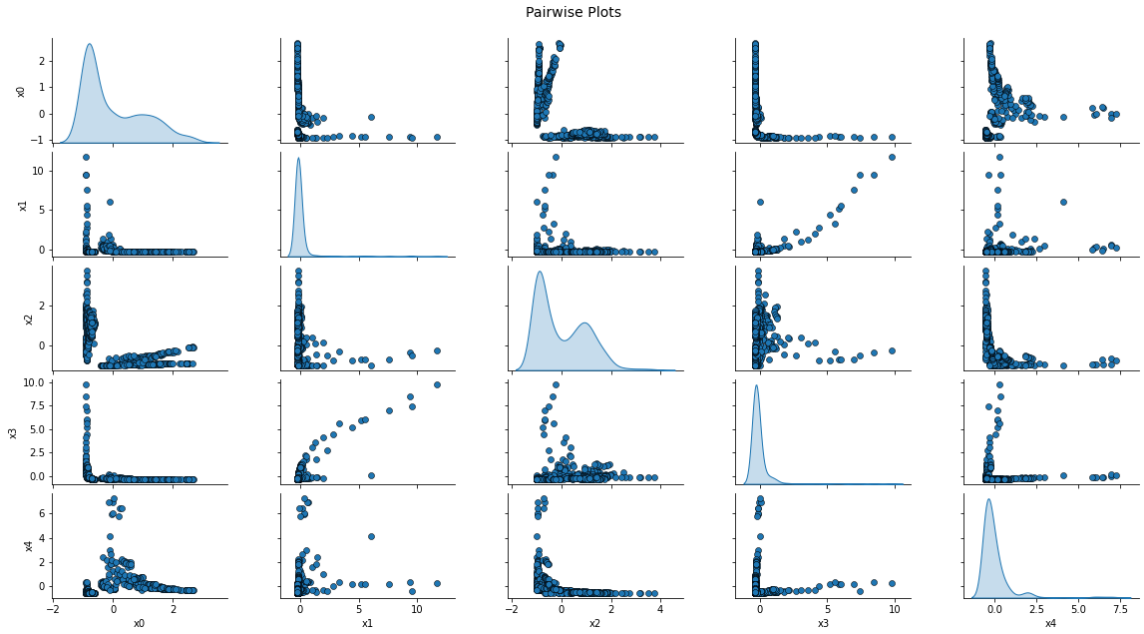

Fig. 4.1 Pair plot matrix of cluster_raw.dat. Each variable is normalized so that they have zero mean and one standard deviation.

To find out an optimal number of clusters, k, inertia (how well a dataset is clustered by K-Means) is calculated for various k values (Fig. 4.2). Inertia decreases with k and it is argued that the point where the decrease in inertia begins to slow is an optimal number of k. In this case, $k = 3$ seems to be the point. The mean feature of each cluster is shown in Fig. 4.3. Clusters 0 and 2 have similar number of members and account for the majority of the data. Cluster 2 has only 9 members and captures outliers of x1 and x3.

Let us try another popular approach, called silhouette score. A silhouette coefficient is defined as $(b - a)/\max(a, b)$, where a is the mean distance to the other instances in the same cluster and b is the mean nearest-cluster distance. Fig. 4.4 shows Silhouette score (the larger the better) as a function of k, suggesting that an optimal number of k is 4. The silhouette score approach provides a more informative visualization, called a silhouette diagram, as shown in Fig. 4.5. It is suggested that in $k = 2 - 4$ would be a reasonable choice.

Finally, let us go back to Fig. 4.1 to see how the clustering works. Shown in Fig. 4.6 is the same pair plot matrix except that different colors are used for different clusters in the case of $k = 3$. It is now clear that the clustering works to a certain degree. For instance, instances are well separated in terms of x0 and x2 and of x2 and x4 (clusters 0 and 2), which is of course in agreement with Fig. 4.3. On the other hand, outlies of x1 and x3 are capture by cluster 1, as mentioned above.

An advantage of K-Means is computational inexpensiveness, it is fast and scalable. A disadvantage of K-Means is that it is not necessarily evident what the optimal number of k is. In addition, K-Means does not behave very well when the clusters have varying sizes, different densities, or nonspherical shapes.

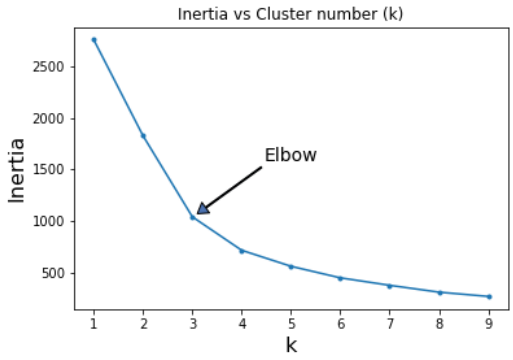

Fig. 4.2 Inertia as a function of K (number of clusters). The point where the decrease in inertia begins to slow is called "Elbow."

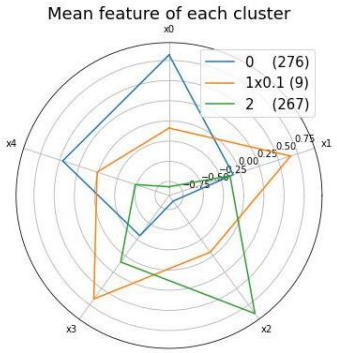

Fig. 4.3 The mean values of each variable for each cluster. Cluster 2 is multiplied by 0.1. Each number in parenthesis is the number of data classified into the corresponding cluster.

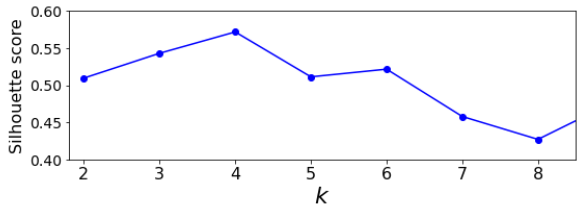

Fig. 4.4 Silhouette score as a function of k.

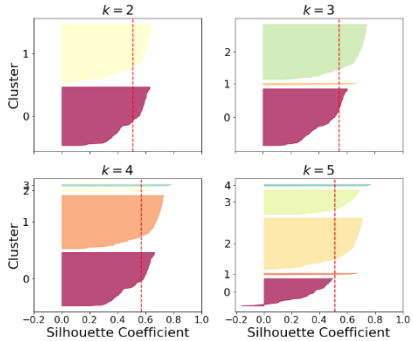

Fig. 4.5 Analyzing the silhouette diagrams for various values of k. The red vertical dashed lines represent the silhouette score for each number of k.

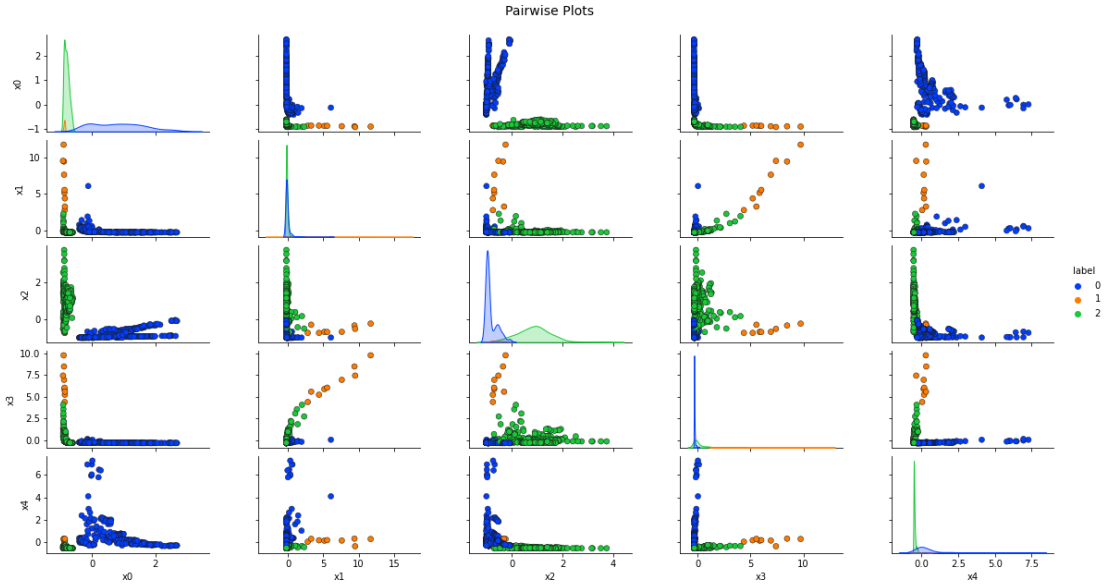

Fig. 4.6 Same as Fig. 4.1 except that different colors are used for different clusters.

5. Assessment Convolve data

Python code: convolve.ipynb

Figure 5.1 shows the horizontal distribution of values of convolve raw data. It is found that values are given at certain locations. In order to carry out the FFT, data should have no missing values. Thus, the first step would be to replace the missing values with some values. Any type of interpolation and extrapolation, however, does not seem to work well in this case, because high value areas are isolatedly present at particular locations. In other words, at a given y, the values may be well represented in terms of gaussian distribution or $\alpha = 2$ gamma distribution. Therefore, it seems to be reasonable to me to just replace the missing values with 0.

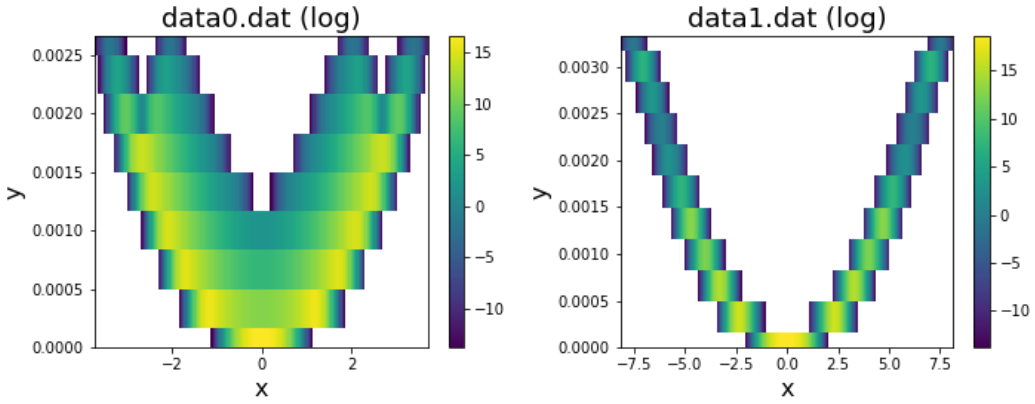

Fig. 5.1 Plot of convolve raw data for (left) data0 and (right) data1. The natural log is applied to the raw data values.

Figure 5.2 shows the results of the FFT applied to the preprocessed data. Prior to the application of the FFT, the mean over the entire domain is subtracted. A strong spectral peak is found at $(k = 0, l = 1)$ in both cases, in particular in data0. This is easily expected from Fig. 5.1, as the value distribution is strongly projected on $l = 1$ component (e.g., at $x = 0$ it has values in the lower half, while it is 0 in the upper half). Next, profound spectral peaks are found along straight lines rising to the left and right that start from $(k = 0, l = 0)$. These spectral peaks correspond to the value distributions rising to the left and right starting from $(x = 0, y = 0)$. If the PSD is normalized by some reasonable red noise, we might be better able to present the spectral peaks.

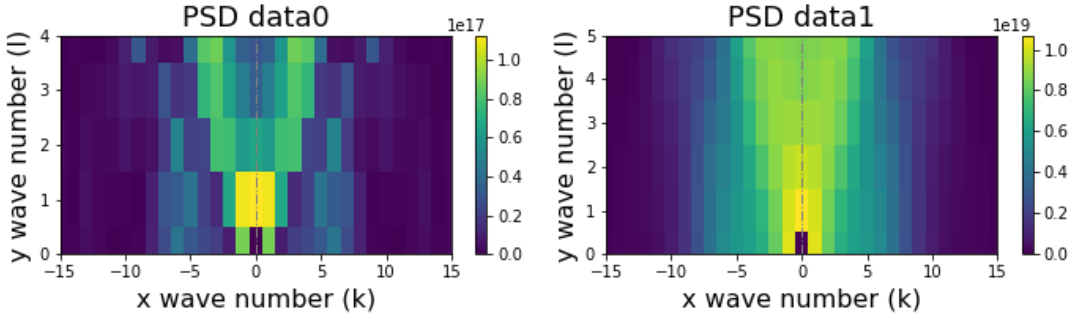

Fig. 5.2 Power spectral density (PSD) of (left) data0 and (right) data1 as a function of x wavenumber and y wavenumber. Since the PSD in the third and four quadrants are the mirror images of those in the first and second quadrants, respectively, the first two quadrants are only shown.