Instructions:

Please submit all work products, including documentation, through your open public github account and discuss these during the technical interview process. Please complete as many assessment challenges as you would like. No hard timeline, so submit when you feel ready and have a good high-level understanding of what we do. These exercises are extracted from our working code and are intended to provide realistic examples of what we do to support AFRL R&D ionospheric impact application development. Evaluation criteria are based on thoroughness, attention to detail, problem-solving, and communication in writing and verbal.

Assessment IRI EDP: Create a C-based modeling and simulation program that drive IRI model Fortran code. The code should capture and generate vertical EDP (Electron Density Profile) for a given time and location of interest.

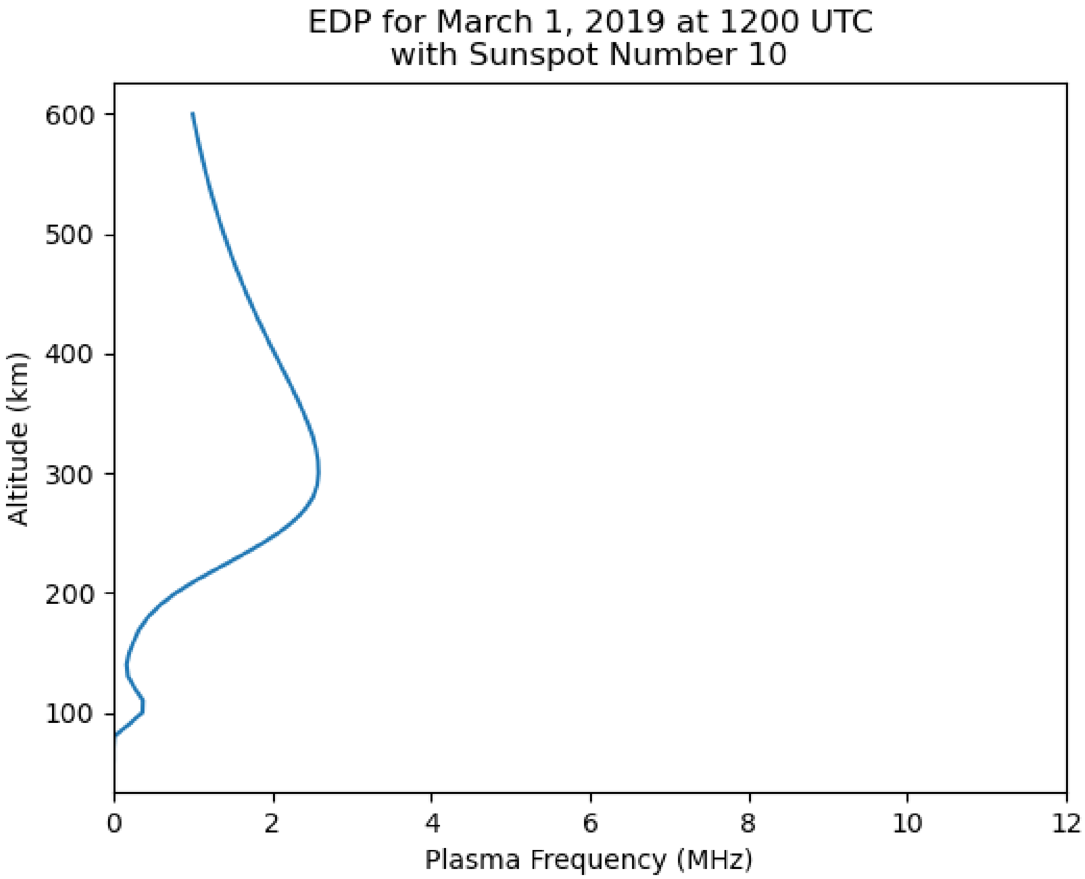
time of interest: Mar 3 2021 UT 11:00:00 and Mar 4, 2021 UT 23:00:00

location o interest: Lat 37.8N and Lon 75.4W

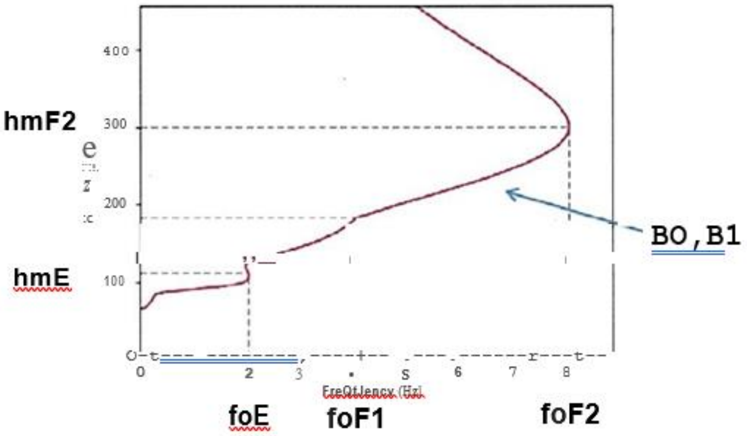
Assessment Criteria:

1) Create a simple Makefile that can compile iri2016 (http://irimodel.org) and generate a shared object/library
2) Write a C-program that links with the shared object created and create all data needed for step 3)
3) Use gnuplot (www.gnuplot.info) or other similar C-based plotting tools to generate plots of EDP parameters using the shared objective created in step 1.
4) alternatively, use F2PY (https://www.numfys.net/howto/F2PY/) and Python to create EDP plots using the shared object created in step 1. (Although C-based plotting is the preferred solution)
5) Furnish instructions/documentation, etc. on how to run the code and lesson/insights learned by doing this exercise.

Example EDP plots are shown next page:



IRI Standard Profile Definition



Assessment Coordinate Transformation: Implement coordinate conversion utilities that can convert radar coordinates (bearing, range) to GIS coordinates (lat, lon) and vice versa in C. The header declaration could be the following or your own design.

```
int GIS2Radar(double *range,
              double *bearing,
              double glonInit,
              double glatInit,
              double glonFinal,
              double glatFinal);

int RtoG (double range,
          double bearing,
          double  glonInit,
          double glatInit,
          double *glonFinal,
          double *glatFinal);
```

You can use the following as the location of interest

Initial: Wallops Islands, lat: 37N, long: 75W)

Final: Puerto Rico, lat: long: 18N, 66W)

Data Collection

You have been assigned to write an app that collects data from an outside source. The source is from the NOAA Space Weather Prediction Center (SWPC) and the data is the Real-Time Solar Wind (RTSW) from the NOAA/DSCOVR satellite. The source is available at:
https://services.swpc.noaa.gov/products/geospace/propagated-solar-wind-1-hour.json

1. Develop an application and/or associated with Application Programming Interfaces (APIs), in Python 3, that will download RTWS data. If you are asked to persist this data, how will you go about achieving this task? Write a persistent layer with extensibility in mind.
    - Be aware that you need to extend it to other different NOAA data sources.
    - Pay attention to cadence, real-time requirements, and network constraints.
2. (Optional) Finally, you are asked to containerize your app, please provide the appropriate necessary files, that will allow running your app as a container, in your favorite container runtime.
3. (Bonus: web-gui) You are now tasked with writing a dashboard that will provide early warning of solar storms, you may use any web-based GUI framework, e., g React, Vue, angular, you wish to accomplish this task.
4. (Bonus: Machine Learning): Write a prediction engine that will try to forecast future space weather based on past data.
    a. What was your preferred approach to this problem?
    b. Test it for some time, how would you go about comparing it to the real results?

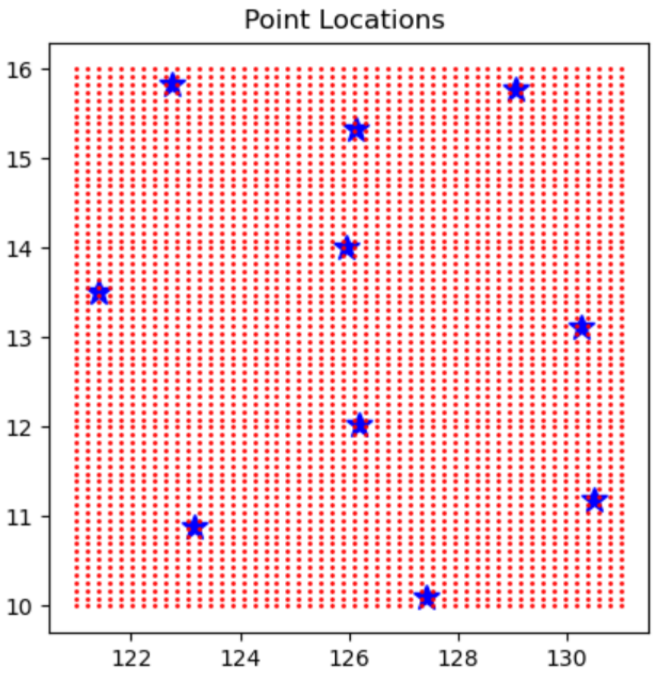
Assessment Javascript: Write a web page with a 400x400 pixel canvas and a single button. Using jquery to listen for the button click event, draw a circle 1/2 the size of the canvas in the upper right-hand corner of the canvas and an ellipse in the lower left-hand corner of the canvas. Fill both the circle and the ellipse.

(optional) Assessment Interpolation: Using any language and plotting packages you like, interpolate the given point location values to a structured grid of longitude and latitude locations. This task will ask the candidate to implement a procedure that interpolates known grid points and values to a structured grid

lon	lat	value
121.39	13.51	1.494
126.19	12.02	1.934
130.27	13.11	2.148
127.42	10.09	9.155
126.14	15.33	2.221
125.96	14	8.1
123.15	10.88	2.039
130.5	11.18	1.916
129.08	15.78	3.729
122.74	15.82	7.137

You are free to choose any interpolation method you like. The spatial grid should have 50 rows and 70 columns with a longitude range [121.0, 131.0] and latitude range [10.0, 16.0]. Provide a plot of your results - including a short summary on the interpolation procedure you implemented.

The plot below just illustrates the known value locations (blue) and the grid point locations (red), to be interpolated.



(optional) Assessment: "convolve_raw_*.dat" (attached) have data arranged in 3 columns.

1) Column 1 and 2 designate x and y coordinates respectively.

2) Column 3 is the main data defined at the corresponding (x, y) points.

3) Compute the 2-d convolution of these data and provide visualization.

4) Use any programming language or library as necessary.