TP n° 2: Building a geomagnetic field model

Raphael Chedozeau, Preshtibye Raggoo, Diana Saqui

November 9, 2023

1 Introduction

The aim of this TP is to build our own field model using the Swarm data. This will allow us to examine the model by examining different cases whereby we change the variables. These include examining the model at different radii, truncating the field model at different spherical harmonic degrees, observing the evolution of the inverted field model over the Swarm era, and comparing it with maps from the CHAOS-7 field model. This could aid us in our understanding of how the model behaves under different conditions and hence allow us to get more information about the Earth's geomagnetic field based on observational data.

2 Evolution of field model with radius, r

For this step we modify the variable r in the code such that r varies between the satellite altitude (radius r=6861 km) and the core surface (radius r=3485 km). In order to observe how the field model evolves with r, we take the values of r = 3485.0 km (on the core's surface), r= 6381. km on the earth's surface and r = 6861 km (satellite altitude).

2.1 Field Model at core surface

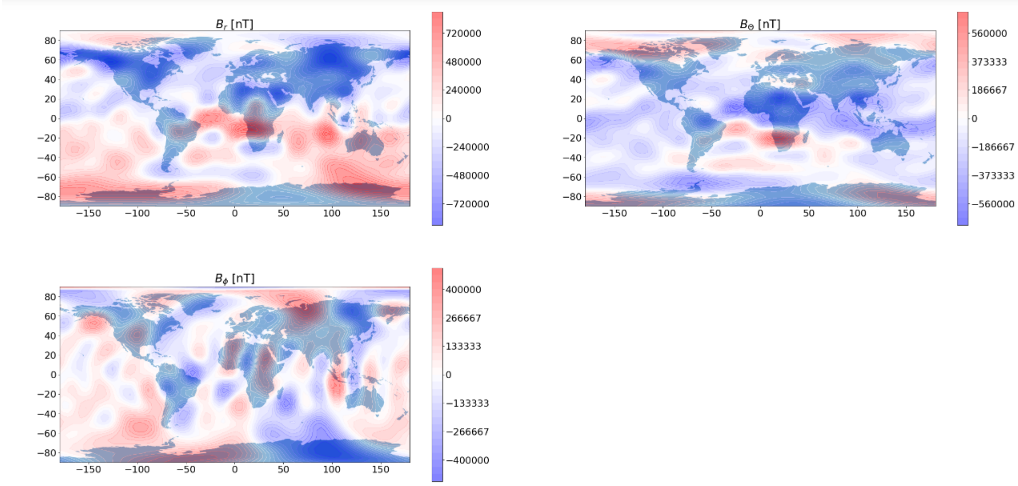

Figure 2: Field Model at core surface

We can observe that the geomagnetic field is very prominent on field model on the core's surface. The core comprises of conductive fluid (namely, nickel and iron) which generates electric currents. This causes the production of the magnetic field that is illustrated on Figure 1 . The generation of electric currents and the rotation of the earth sustains the magnetic field of the core.

2.2 Field Model at satellite altitude

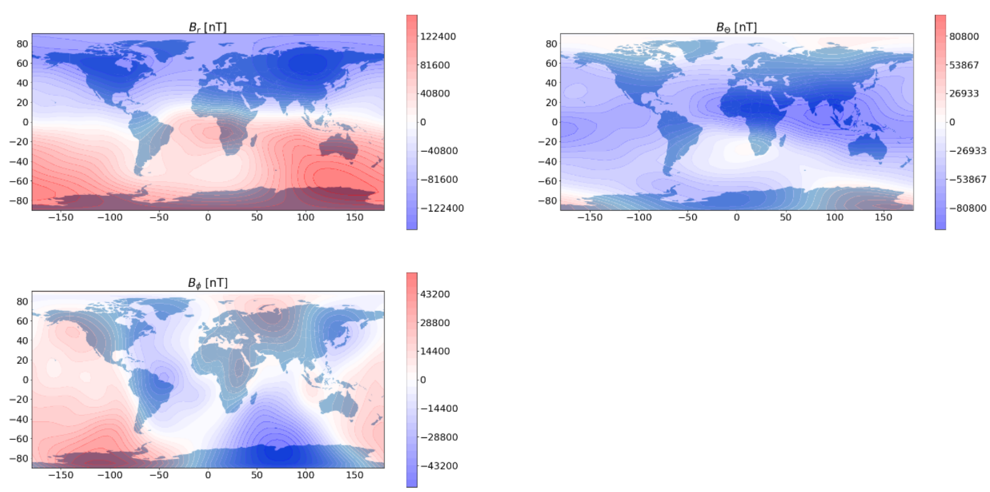

Figure 2: Field Model at satellite altitude

Here, we observe the field model at r = 6861 km. There is a relatively uniform distribution of the magnetic field, varying mainly depending on the latitude and longitude. Note : At the equation the magnetic field tends towards 0. This can be seen by the lighter colour gradient near the equator. This is because at the equator, the magnetic field lines are parallel to the earth.

2.3 Comparison of geomagnetic field at different r values

At the core's surface, we can see that there are more complex patterns and stronger variations in the magnetic field strength.

$$\mathbf{B}(r,\theta,\phi) = \sum_{n=1}^{N} \left(\frac{a}{r}\right)^{n+2} \sum_{m=0}^{n} \left(g_n^m \Pi_n^{mc}(\theta,\phi) + h_n^m \Pi_n^{ms}(\theta,\phi)\right) \tag{1}$$

At the core's surface, we observe more complex patterns and stronger variations in the magnetic field strength. At the satellite's altitude, we can see that the field lines are further apart and the geomagnetic field B becomes weaker. From equation 1 above, we can see that as r increases, the sum

$$\sum_{m=0}^{n} \left(g_n^m \Pi_n^{mc}(\theta,\phi) + h_n^m \Pi_n^{ms}(\theta,\phi)\right)$$

becomes less significant. We hence no longer observe the smaller structures on our map. We can say that:

$$\mathbf{B}(r,\theta,\phi) \propto \frac{1}{r^{n+2}}$$

3 Truncating the field model at varying spherical harmonic degree

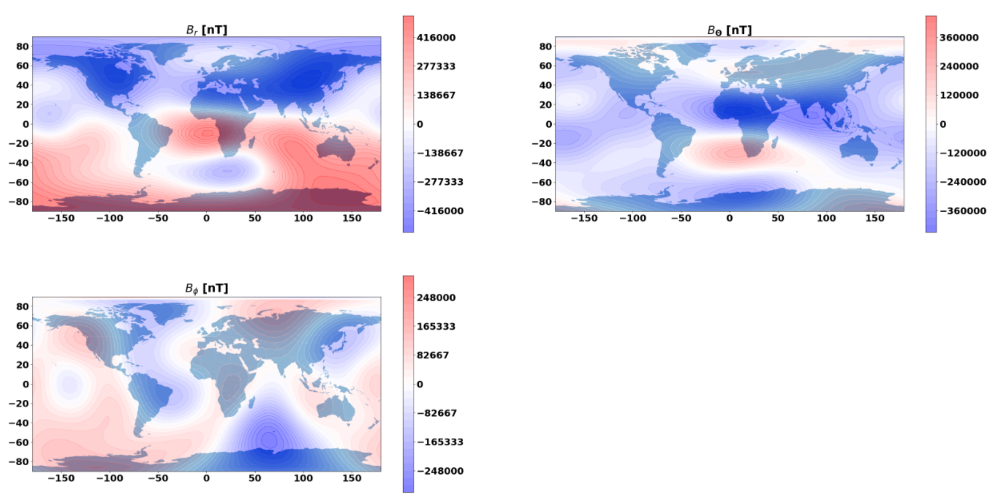

Figure 3: Field Model at L_max = 5

At L_max = 5, we can see on the figure 3 that we don't see a lot of details.

$$\Pi_{ni}^{mc}(\theta,\phi) = (n+1)P_n^m(\cos\theta)\cos(m\phi)\mathbf{e}_r - \frac{dP_n^m(cos\theta)}{d\theta}\cos(m\phi)\mathbf{e}_\theta + \frac{m}{\sin\theta}P_n^m(\cos\theta)\sin(m\phi)\mathbf{e}_\phi \tag{2}$$

$$\Pi_{ni}^{ms}(\theta,\phi) = (n+1)P_n^m(\cos\theta)sin(m\phi)\mathbf{e}_r - \frac{dP_n^m(\cos\theta)}{d\theta}\sin(m\phi)\mathbf{e}_\theta + \frac{m}{\sin\theta}P_n^m(\cos\theta)\cos(m\phi)\mathbf{e}_\phi \tag{3}$$

From equations 2 and 3, as L_max increases, the spherical harmonic degree increases, the functions become more oscillating. From equation 1, we also see that when the right sum increases, B also increases. Hence, when L_max is large, we can see much more details as seen in Figure 4 below.

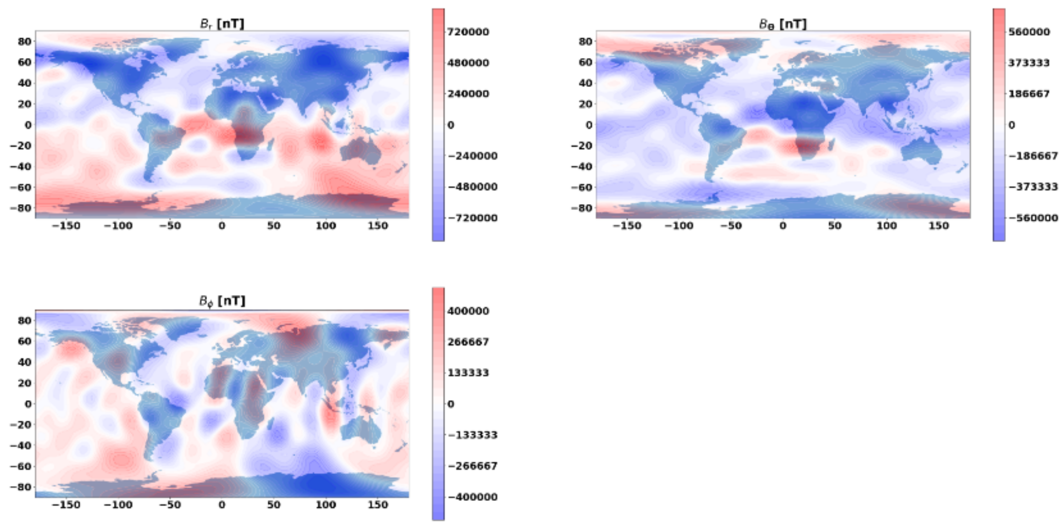

Figure 4: Field Model at L_max = 13

4 Evolution of the inverted field model evolve through time over the Swarm era

Changing the date of the observed data, we see that the field model evolve through time over the Swarm era. Indeed, the values of magnetic currents fluctuate regarding their localisation. Magnetic field is evolving through the time series and its intensity too.
During the Earth's history we can see, through the orientation of certain minerals or for example fired pottery, that it changes. Minerals in fact hold the memory of the magnetic field,i.e we can determine the nature of the magnetic field when examining them as they hold information on the state of the magnetic field at the moment that they solidify. We can determine the orientation of the magnetic poles.

Those variations are due to earth-internal origins (during long time series variation : pole inversion) but also, in our case, due to external origins (small time duration). These external sources are for example the solar cycle or usually solar wind storms.

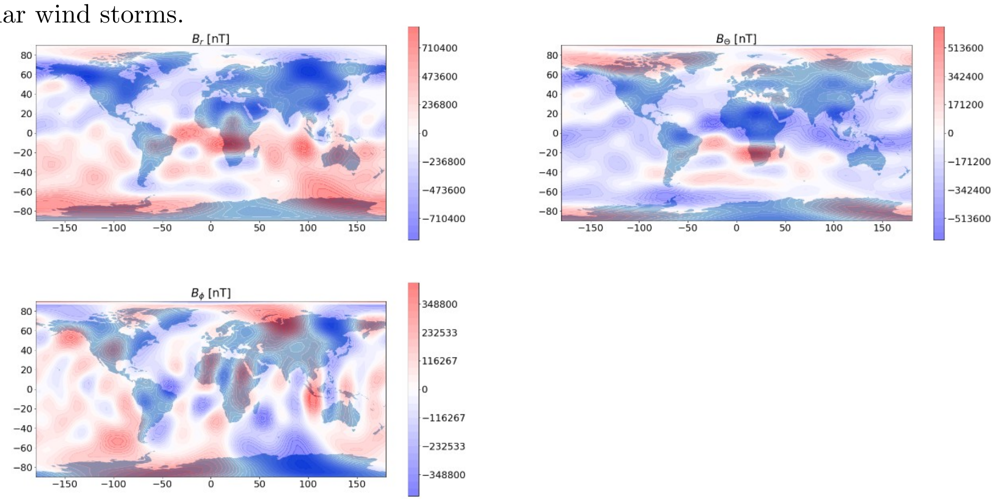

Figure 5: Field model for 2014-05

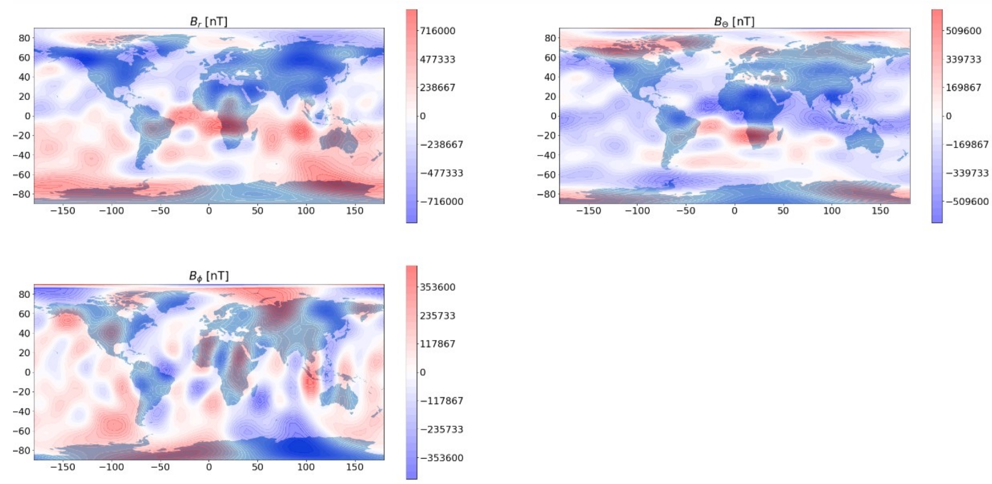

Figure 6: Field model for 2020-05

5 Comparison with maps from the CHAOS-7 field model

There are no distinctive differences between the two models. Observing the data used to plot those maps, we can see that those data are not totally identical but these differences are not discernible on the graphs.
In order to see the differences between the two models, these small differences

can be computed between the two models, with : B_diff = B - B_chaos

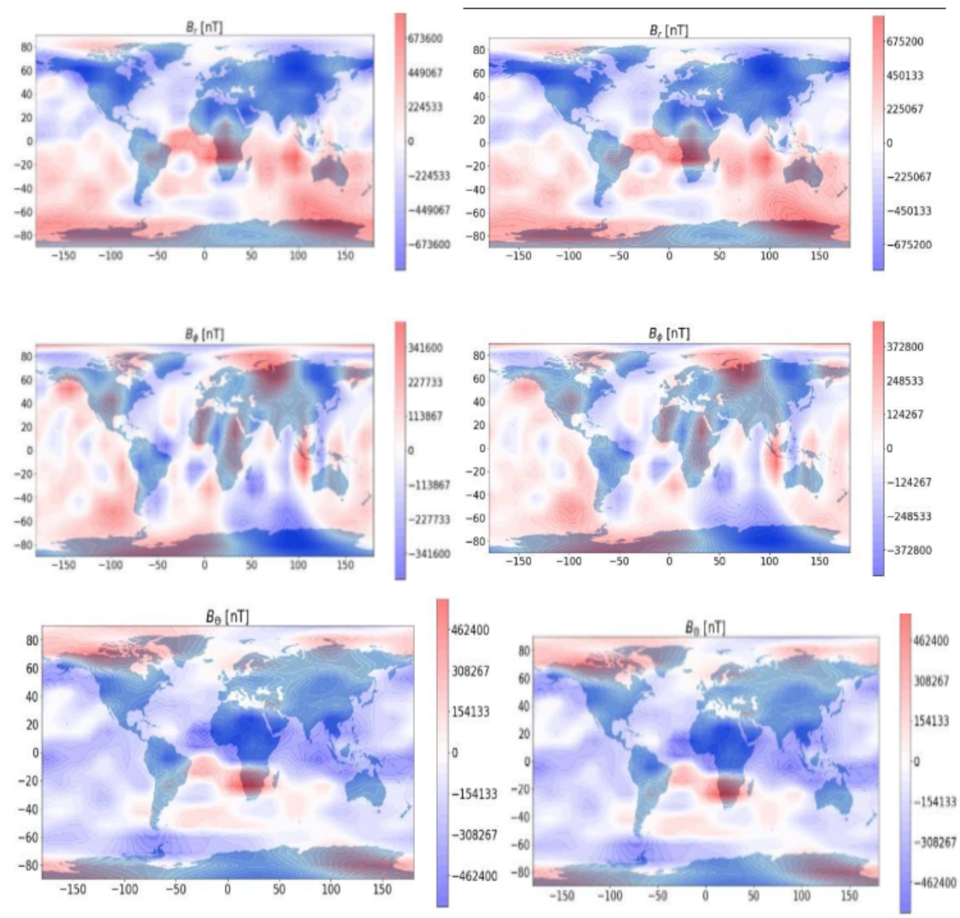

Figure 7: B-chaos and our model

Then the plot of the graphs of the differences in the models can be made as such :

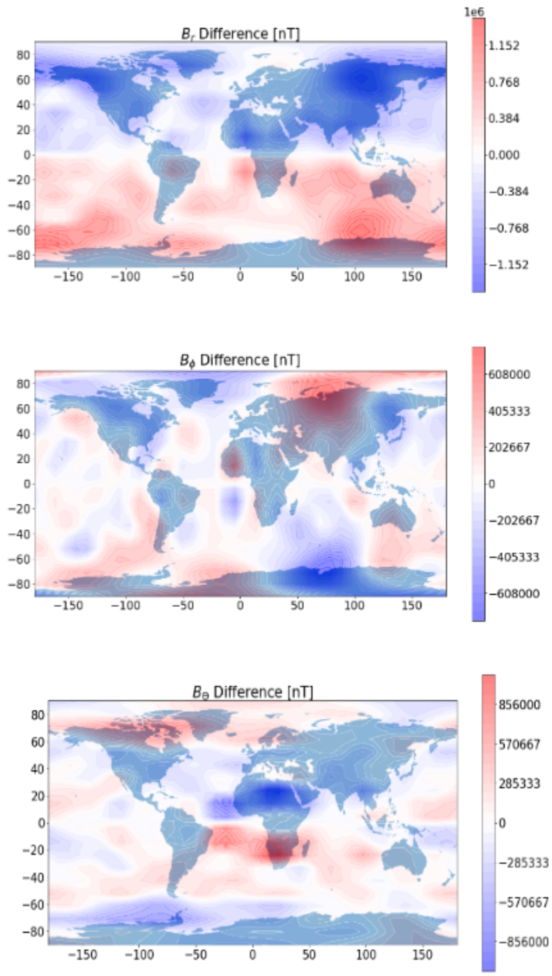

Figure 7: Difference between B-chaos and our model

Regarding the scale of the Br graph we see that this is the scale of the difference and it is a very small difference between both models.

6 Extra

```python
def compute_gauss_coefs_vector_and_cov_matrix(file, date_chosen, max_degree, N_VO, measure_MF, measure_sigma, cm_prior):
    cdf_read = cdflib.CDF(file)
    info = cdf_read.cdf_info()
    zvars = info.zVariables
    alldata = {name: cdf_read.varget(name) for name in zvars}
    times = cdf_times_to_np_date(alldata['Timestamp'])
    unique_times = compute_unique_times_list(times)
    list_sigma_obs = compute_obs_vector_at_t(alldata, date_chosen, N_VO, measure_sigma, unique_times)
    obs_vec_at_t = compute_obs_vector_at_t(alldata, date_chosen, N_VO, measure_MF, unique_times)
    r, th, ph, list_coords_obs = compute_list_coords_obs_at_t(date_chosen, alldata, unique_times, N_VO)
    obs_vec_at_t, list_coords_obs, list_sigma_obs = delete_nan_values(obs_vec_at_t, list_coords_obs,
                                                                      list_sigma_obs)
    cm_obs = compute_cov_matrix_obs_at_t(list_sigma_obs)
    H = compute_direct_obs_operator(list_coords_obs, max_degree)
    mean_prior = np.zeros(max_degree * (max_degree + 2))
    sigma_obs_inv = np.linalg.inv(cm_obs)
    sigma_prior_inv = np.linalg.inv(cm_prior)
    n_max = max_degree*(max_degree+2)
    # Kalman gain
    Kalman_gain = np.linalg.inv(H.T @ sigma_obs_inv @ H + sigma_prior_inv[:n_max, :n_max])
    cov_matrix_inv = sigma_prior_inv[:n_max, :n_max] - Kalman_gain @ H.T @ sigma_obs_inv @ H @ Kalman_gain.T
    # Calculate the posterior covariance matrix
    posterior_cov_matrix_inv = np.linalg.inv(cov_matrix_inv)
    return posterior_cov_matrix_inv

cm_prior = np.loadtxt('./donnees/prior_cov_matrix.txt')
cdf_dir = './donnees/cdf_files_basic_sync_functions_201'
```

```python
post_cov_matrix = compute_gauss_coefs_vector_and_cov_matrix(swarm_file, '2018-05', 13, 300, 'B_CF', 'sigma_CF', cm_prior)
print(post_cov_matrix)
print(len(post_cov_matrix))
```

```
[[-3.43151657e+02  5.37656263e+01  1.75696752e+01 ... -2.61848080e+00
   2.55350588e+00  7.89224887e-01]
 [ 5.37656263e+01 -3.37316523e+02  9.94281552e+00 ...  1.01383239e+00
  -9.16729203e-01  1.58267469e-01]
 [ 1.75696752e+01  9.94281552e+00 -3.17182707e+02 ...  3.84105365e-01
  -9.83697685e-02 -1.19544811e-01]
 ...
 [-2.61848080e+00  1.01383239e+00  3.84105365e-01 ...  3.06918222e-01
   5.00439835e-02  4.09282165e-02]
 [ 2.55350588e+00 -9.16729203e-01 -9.83697685e-02 ...  5.00439835e-02
   4.17958316e-01 -2.99880770e-04]
 [ 7.89224887e-01  1.58267469e-01 -1.19544811e-01 ...  4.09282165e-02
  -2.99880770e-04  6.21050785e-01]]
195
```

```python
# Extract diagonal elements for uncertainties
coeff_uncertainties = np.diag(post_cov_matrix)
print("Posterior uncertainties on Gauss coefficients:", coeff_uncertainties)
print(len(coeff_uncertainties))
```

```
Posterior uncertainties on Gauss coefficients: [-3.43151657e+02 -3.37316523e+02 -3.17182707e+02 -3.07394739e+02
 -4.11877098e+02 -3.64164367e+02 -5.06313722e+02 -4.78983236e+02
 -3.02422229e+02 -4.49802714e+02 -4.80812544e+02 -4.88703476e+02
 -5.33982071e+02 -6.28064606e+02 -5.69026453e+02 -3.78939031e+02
 -4.16284200e+02 -4.65258024e+02 -5.80186781e+02 -5.88788555e+02
 -6.31742512e+02 -5.53913061e+02 -6.50594770e+02 -7.13173912e+02
 -6.43210952e+02 -4.39323163e+02 -8.12950842e+02 -4.79383913e+02
 -4.63781690e+02 -5.94675355e+02 -6.08794883e+02 -6.30633473e+02
 -6.62938910e+02 -6.48577819e+02 -7.80123116e+02 -4.07633728e+02
 -2.23762385e+02 -2.38357611e+02 -4.58648255e+02 -3.87942895e+02
 -6.00007470e+02 -6.54111275e+02 -6.90556884e+02 -9.37906502e+02
 -6.17629457e+02 -7.91957719e+02 -9.05627148e+02 -8.67188671e+02
 -4.31431308e+02 -3.26790860e+02 -7.08693925e+02 -1.76842207e+02
 -4.14980106e+02 -1.40878805e+02 -9.33841739e+02 -1.28181148e+03
 -8.48519515e+02 -1.31860390e+03 -2.21995177e+02 -8.28458580e+02
 -1.46999490e+03 -7.18290384e+02 -1.49554307e+03 -1.59015987e+01
  1.50948413e+01  1.47659551e+02  1.21517783e+01  1.25252600e+02
 -4.16357538e+02  1.16534965e+02  1.05548362e+02  6.27702073e+01
  5.47611896e+01  2.79544372e+02 -1.81787655e+00 -1.32419944e+01
  1.08522552e+03 -1.03025104e+02  5.39357040e+02 -2.31506605e+02
 -2.63343621e+01  8.96218757e+00  1.20329905e+01  2.96867384e+01
  1.73720716e+00  4.15567434e+01  2.50858775e+01  7.84535337e+00
  2.48304692e+01  3.76424034e+01  5.50395853e+01 -4.58517360e+01
  1.15998168e+01  1.39903675e+01  3.15543086e+01  1.36141580e+02
  1.22045049e+02  1.07170185e+02  1.24468198e+02  5.83327475e+00
 -7.10362295e+00  3.53491431e+00  2.81937471e+00  7.02493704e+00
  5.61536564e+00  1.24073910e+01  9.64835451e+00  9.40223618e+00
 -3.03984963e+00  1.69257685e+01  9.18751288e+00  4.72668809e+00
  1.06694368e+01  1.43933142e+01  1.29303975e+01  1.41152223e+01
  1.91747997e+01  2.27581884e+01  2.64539199e+01  2.21938604e+01
  2.46228791e+00  2.09223840e+00  9.15032045e-01  1.85505819e+00
  1.26284958e+00  1.04875998e+00  2.79891929e+00  2.27273487e+00
  2.30603018e+00  2.22878895e+00  2.38832970e-01  1.76299726e+00
  2.45990267e+00  1.90784720e+00  1.26430568e+00  6.67133099e+00
  5.23803918e+00  3.42658447e+00  4.24183002e+00  3.70538119e+00
  9.96963437e+00  5.14715504e+00  6.18250279e+00  5.87225397e-01
  5.27155791e-01  4.32543709e-01  6.30511425e-01  4.17291447e-01
  1.57941101e-01  6.47570577e-01  3.60313019e-01  5.05938854e-01
  8.14910382e-01  7.28443306e-01  6.53843611e-01  9.68139885e-01
  1.24471102e+00  5.92300942e-01  5.77911581e-01  6.69150494e-01
  1.48908963e+00  8.00342963e-01  6.24326476e-01  8.72603457e-01
  1.31253429e+00  1.57526047e+00  1.93388037e+00  1.73760955e+00
  9.66117576e-02  1.53843197e-01  1.04503639e-01  1.58085626e-01
  1.08433684e-01  4.72375776e-02  1.07360869e-01  1.73356854e-01
  3.02062228e-01  2.86978513e-02  2.35544971e-01  1.62297015e-01
  1.39307949e-01  1.37379457e-01  2.11714588e-01  1.45001992e-01
  4.36702065e-01  1.43424732e-01  1.21387465e-01  4.14583740e-01
  3.71473746e-01  2.84902730e-01  2.32858278e-01  4.24039395e-01
  3.06918222e-01  4.17958316e-01  6.21050785e-01]
```