Solutions to the Exercises

Houjun Wang

July 14, 2022

1 Ionosphere Modeling and Ray Tracing

A - Quasi Parabolic Model

1. Since $\left(\frac{(r-r_m)r_b}{y_m r}\right)^2 \geqslant 0$, the terms in the square bracket is $\leqslant 1$. Thus, the maximum electron density occurs when $\left(\frac{(r-r_m)r_b}{y_m r}\right)^2 = 0$, or at $r = r_m$.

2. The critical frequency f_c is the maximum plasma frequency, given by

$$f_c = \frac{1}{2\pi}\sqrt{\frac{N_m e^2}{\varepsilon_0 m}} \tag{1}$$

in Hz, where e is the electron charge, ε_0 is the permittivity of free space, and m is the electron mass. When the wave frequency $f > f_c$, the wave will penetrate the layer (for the vertically incident waves).

3. Three parameters, N_m, r_m and r_b, uniquely and completely define this simplified model ionosphere.

4. Figure. 1 shows the quasi-parabolic electron density profiles for five different semi-thickness y_m. The python script plot_profile.py is used for the plot.

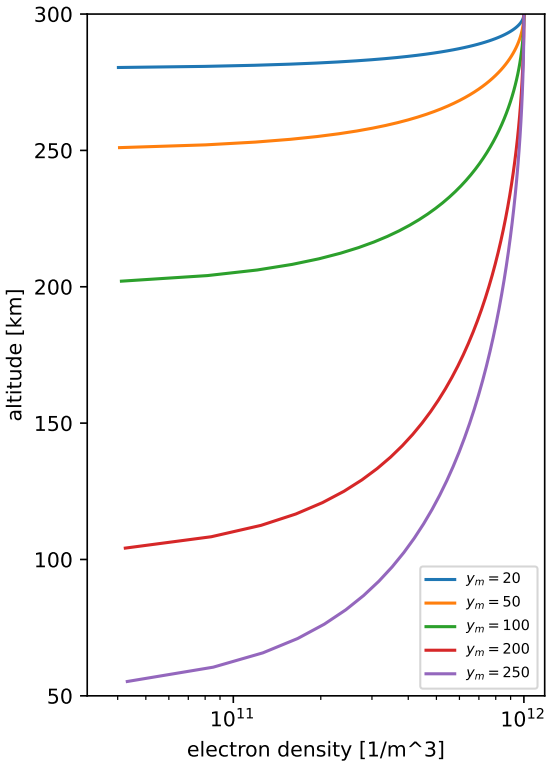


Figure 1: The quasi-parabolic electron density profiles for five different semi-thickness.

For $N_m = 10^{12}$ m^{-3}, the critical frequency is $f_c \approx 8.98$ MHz.

B - Ionosphere Reflection

1. If $f > f_N$, the wave rays will penetrate the layer; otherwise, it will be reflected. In addition, for waves with frequency less than the critical frequency (maximum plasma frequency), the wave rays will reach higher altitudes (but still below the altitude of the electron density peak) before they are reflected.

2. From Snell's law $\sin\theta_i = \mu(h)\sin\theta_h$, we have

$$\sin\theta_i = \sqrt{1 - \frac{f_N^2}{f^2}}\sin\theta_h = \sqrt{1 - k\frac{N(h)}{f^2}}\sin\theta_h \tag{2}$$

   where $k = e^2/(4\pi^2\varepsilon_0 m) \approx 80.6$ [$\mathrm{m^3\,s^{-2}}$]. At reflection, $\theta_h = \frac{\pi}{2}$, or $\sin\theta_h = 1$. Thus,

$$\sin\theta_i = \sqrt{1 - k\frac{N(h)}{f^2}} \tag{3}$$

   or

$$f = \sqrt{kN(h)}\sec\theta_i \tag{4}$$

   This is the so-called secant law (in a slightly different form).

3. The maximum frequency f_{ob} for oblique propagation is given by

$$f_{\mathrm{ob}} = \sqrt{kN_m}\sec\theta_i = 8.98 \times 10^6 \sec\theta_i \tag{5}$$

   in Hz, when $N(h) = N_m = 10^{12}$ $\mathrm{m^{-3}}$. It depends on both the peak electron density N_m and the incidence angle θ_i, and is higher with oblique propagation than with vertical propagation.

4. Using the concept of virtual height, h', we have

$$\tan\theta_i = \frac{D/2}{h'} = \frac{D}{2h'} \tag{6}$$

   from simple geometric consideration, itself a consequence of the theorems of Breit and Tuve and of Martyn. Thus, the range D can be expressed in terms of virtual height h' as

$$D = 2h'\tan\theta_i \tag{7}$$

   The virtual height h' can be calculated from

$$h' = \int_{r_0}^{r_1} \frac{1}{\mu(r)}dr \tag{8}$$

   where r_0 is the earth's radius, and r_1 is the radial distance of the reflection point.

Since the refractive index $\mu(r)$ for the QP ionospheric layer makes the integral quite complicated algebraically, we'll use the refractive index from a simpler parabolic ionospheric layer, defined by

$$N_e(r) = \begin{cases} N_m\Big[1 - \Big(\frac{r-r_m}{y_m}\Big)^2\Big] & r_b < r < r_m + y_m \\ 0 & \text{elsewhere} \end{cases}$$

The QP and parabolic layers are very similar below the electron density peak, and this is what matters concerning the ray reflection problem.

For the parabolic ionosphere, following Budden (1985), introduce a new height variable

$$\zeta = \frac{r - r_m}{y_m} \tag{9}$$

Then we have

$$f_N^2 = f_c^2(1 - \zeta^2) \quad \text{for } |\zeta| \leqslant 1 \tag{10}$$

where f_c is the critical frequency, or the maximum plasma frequency.

The refractive index becomes

$$\mu(r) = [1 - (1 - \zeta^2)f_c^2/f_\perp^2]^{\frac{1}{2}} \tag{11}$$

where $f_\perp = f_\mathrm{o}\cos\theta_i$, f_o is the operating frequency.

Now consider the case when $f_\perp < f_c$, the reflection occurs at a height h_1 below the peak density. The value ζ_1 of ζ at the reflection level is

$$\zeta_1(f_\perp) = -(1 - f_\perp^2/f_c^2)^{\frac{1}{2}} \tag{12}$$

The virtual (equivalent) height of the reflection is then

$$h'(f) = \int_{r_0}^{r_1} \frac{1}{\mu(r)} dr = h_b + \frac{1}{2} y_m \frac{f_\perp}{f_c} \ln\Big(\frac{f_c + f_\perp}{f_c - f_\perp}\Big) \tag{13}$$

Remark 1. In the calculation of h' for the range $D = 2h'\tan\theta_i$ of oblique propagation, the *equivalent vertical incidence frequency*, $f_\perp = f_\mathrm{o}\cos\theta_i$, should be used, instead of f_o.

Remark 2. Since the integral in calculation of h' becomes singular (infinity) when $f_\perp = f_c$, we can choose a slightly smaller values for $f_\perp$ when computing for maximum $D_\mathrm{max} = 2h'_\mathrm{max}\tan\theta_i$, e.g., $f_\perp = (1 - 1.0 \times 10^{-12})f_c$.

5. As remarked above, we assume that the equivalent vertical incidence frequency $f_\perp = (1 - 1.0 \times 10^{-12}) f_c$ in calculation of the maximum virtual height $h'_{\max}$ and the corresponding maximum range:

$$D_{\max} = 2h'_{\max} \tan\theta_i \tag{14}$$

Notice that for fixed $f_\perp$, the operating frequency $f_o = f_\perp \sec\theta_i$ depends only on incidence angle θ_i.

Since the flat earth and ionosphere approximation is more appropriate for small incidence angles, $\theta_i \leqslant 45^\circ$ is used in the following calculations. Table 1 lists computational results of the maximum ranges $D_{\max}$, using five incidence angles θ_i and five different parabolic ionospheric profiles. The five ionospheric profiles have semi-thickness y_m[km] = 20, 50, 100, 200, 250, with the corresponding ionospheric bottom heights h_b[km] = 280, 250, 200, 100, 50. Also listed are the operating frequencies f_o and the maximum virtual height $h'_{\max}$. Python script calc_range.py is used for the computation.

Table 1: The maximum ranges $D_{\max}$ [km] for five incidence angles θ_i [degree] and five ionospheres with different semi-thickness y_m. Also listed are the operating frequencies f_o [MHz] and the maximum virtual height $h'_{\max}$ [km].

θ_i	f_o	I	II	III	IV	V
25	9.91	525.29	893.54	1507.30	2734.82	3348.58
30	10.37	650.38	1106.32	1866.24	3386.06	4145.98
35	10.96	788.77	1341.74	2263.36	4106.60	5028.22
40	11.72	945.23	1607.89	2712.32	4921.18	6025.61
45	12.70	1126.48	1916.21	3232.42	5864.83	7181.04
$h'_{\max}$		563.24	958.10	1616.21	2932.42	3590.52

C - Ray Tracing

1. The range of the ray is given by (Croft and Hoogasian, 1968)

$$\begin{aligned} D &= 2r_0 \int_0^{\theta_t} d\theta \\ &= 2r_0 \int_{r_0}^{r_t} \frac{dr}{\tan\beta} \\ &= 2\int_{r_0}^{r_t} \frac{r_0^2 \cos\beta_0}{r\sqrt{r^2\mu^2 - r_0^2\cos^2\beta_0}} dr \end{aligned} \tag{15}$$

by change of variables using the relation $\tan\beta = \frac{dr}{rd\theta}$ from the first to the second equation, and by using a special form of Snell's law $r\mu\cos\beta = r_0\cos\beta_0$ (Bouger's rule) from the second to the third equation. The range equation applies to any ionosphere concentric with the earth.

2. The evaluation of the integral can be done as follows (Croft and Hoogasian, 1968). The refractive index for a simplified quasi-parabolic (QP) ionosphere can be written in the form

$$\mu^2 = 1 - \frac{80.6N_e}{f^2} = 1 - \frac{1}{F^2} + \Big(\frac{r_m - r}{Fy_m}\Big)^2\Big(\frac{r_b}{r}\Big)^2 \tag{16}$$

where $F = f/f_c$, the ratio of operating frequency to critical frequency. The critical frequency is $f_c^2 = 80.6N_m$ in the MKS units, if we neglect the effects of geomagnetic field and collisions. Thus, the expression under the square root sign in the range equation can be written in the form

$$r^2\mu^2 - r_0^2\cos^2\beta_0 = Ar^2 + Br + C \tag{17}$$

where

$$\begin{aligned} A &= 1 - \frac{1}{F^2} + \Big(\frac{r_b}{Fy_m}\Big)^2 \\ B &= -2r_m\Big(\frac{r_b}{Fy_m}\Big)^2 \\ C &= r_m^2\Big(\frac{r_b}{Fy_m}\Big)^2 - r_0^2\cos^2\beta_0 \end{aligned}$$

The range equation becomes (the refractive index is one in free space)

$$D = 2r_0^2\cos\beta_0\left\{\int_{r_0}^{r_b}\frac{dr}{r\sqrt{r^2 - r_0^2\cos^2\beta_0}} + \int_{r_b}^{r_t}\frac{dr}{r\sqrt{Ar^2 + Br + C}}\right\} \tag{18}$$

Using the standard integral tables, the first integral can be evaluated as follows:

$$\begin{aligned} I_1 = \int_{r_0}^{r_b}\frac{dr}{r\sqrt{r^2 - r_0^2\cos^2\beta_0}} &= \Big[\frac{1}{r_0\cos\beta_0}\sec^{-1}\Big(\frac{r}{r_0\cos\beta_0}\Big)\Big]_{r_0}^{r_b} \\ &= \frac{1}{r_0\cos\beta_0}\Big[\sec^{-1}\Big(\frac{r_b}{r_0\cos\beta_0}\Big) - \beta_0\Big] \end{aligned}$$

Introduce γ, the angle of the ray to the horizontal at the bottom of the ionosphere, using the law of sines, we have

$$\frac{r_0}{\sin(\frac{\pi}{2} - \gamma)} = \frac{r_b}{\sin(\frac{\pi}{2} + \beta_0)} \Longrightarrow \frac{r_0}{\cos\gamma} = \frac{r_b}{\cos\beta_0} \tag{19}$$

or

$$\cos\gamma = \frac{r_0}{r_b}\cos\beta_0 \tag{20}$$

Thus,

$$I_1 = \int_{r_0}^{r_b} \frac{dr}{r\sqrt{r^2 - r_0^2\cos^2\beta_0}} = \frac{1}{r_0\cos\beta_0}(\gamma - \beta_0)$$

The second integral can be evaluated as, assuming $C > 0$,

$$\begin{aligned} I_2 = \int_{r_b}^{r_t} \frac{dr}{r\sqrt{Ar^2 + Br + C}} &= \Big[-\frac{1}{\sqrt{C}}\ln\Big(\frac{2C + Br + 2\sqrt{CX(r)}}{r}\Big)\Big]_{r_b}^{r_t} \\ &= \frac{1}{\sqrt{C}}\ln\left[\frac{r_t(2C + Br_b + 2\sqrt{CX(r_b)})}{r_b(2C + Br_t + 2\sqrt{CX(r_t)})}\right] \end{aligned}$$

where $X(r) \equiv Ar^2 + Br + C$. Therefore, we have an analytical expression for the range

$$D = 2r_0(\gamma - \beta_0) + \frac{2r_0^2\cos\beta_0}{\sqrt{C}}\ln\left[\frac{r_t(2C + Br_b + 2\sqrt{CX(r_b)})}{r_b(2C + Br_t + 2\sqrt{CX(r_t)})}\right] \tag{21}$$

3. The equation for the apogee height is given by (see discussion in Hill, 1979)

$$Ar_t^2 + Br_t + C = 0 \tag{22}$$

with $B^2 \geqslant 4AC$. The maximum apogee height is obtained when $B^2 = 4AC$ as

$$r_{t,\max} = -B/2A \tag{23}$$

In addition, using the equation for the apogee, the range equation simplifies to

$$D = 2r_0(\gamma - \beta_0) + \frac{2r_0^2\cos\beta_0}{\sqrt{C}}\ln\left[\frac{r_t(2C + Br_b + 2\sqrt{CX(r_b)})}{r_b\sqrt{B^2 - 4AC}}\right] \tag{24}$$

4. From $B^2 = 4AC$, and using definition of C and then B, we can find the elevation angle $\beta_{0,\max}$ when the apogee height is maximal as follows:

$$\frac{B^2}{4A} = C = r_m^2\Big(\frac{r_b}{Fy_m}\Big)^2 - r_0^2\cos^2\beta_{0,\max} = -\frac{Br_m}{2} - r_0^2\cos^2\beta_{0,\max}$$

or

$$\beta_{0,\max} = \arccos\left(\sqrt{-\frac{B}{2r_0^2}\Big(r_m + \frac{B}{2A}\Big)}\right) \tag{25}$$

This angle produces the so-called Pedersen ray (e.g., Hill 1979). It may be called the Pedersen angle, and will be denoted by β_p in the next question.

5. A point on the ray can be defined by the coordinates (D_r, r), where D_r is the distance traversed, measured along the earth's surface. As shown above, it is given by

$$D_r = r_0(\gamma - \beta_0) + \frac{2r_0^2 \cos\beta_0}{\sqrt{C}} \ln\left[\frac{r_t(2C + Br_b + \sqrt{CX(r_b)})}{r_b(2C + Br + 2\sqrt{CX(r)})}\right] \tag{26}$$

Also notice that from D_r we can get $\theta = D_r/r_0$.

Table 2: The apogee height r_t [km] for four operating frequencies $f_o = 5, 10, 15, 20$ [MHz], four elevation angles $\beta_0 = 5, 10, 15, 20$ [degrees], and five ionospheres with different semi-thickness y_m. Also listed are the maximum apogee height $r_{t,\max}$ [km] and the corresponding elevation angles, the Pedersen angle β_p [degrees].

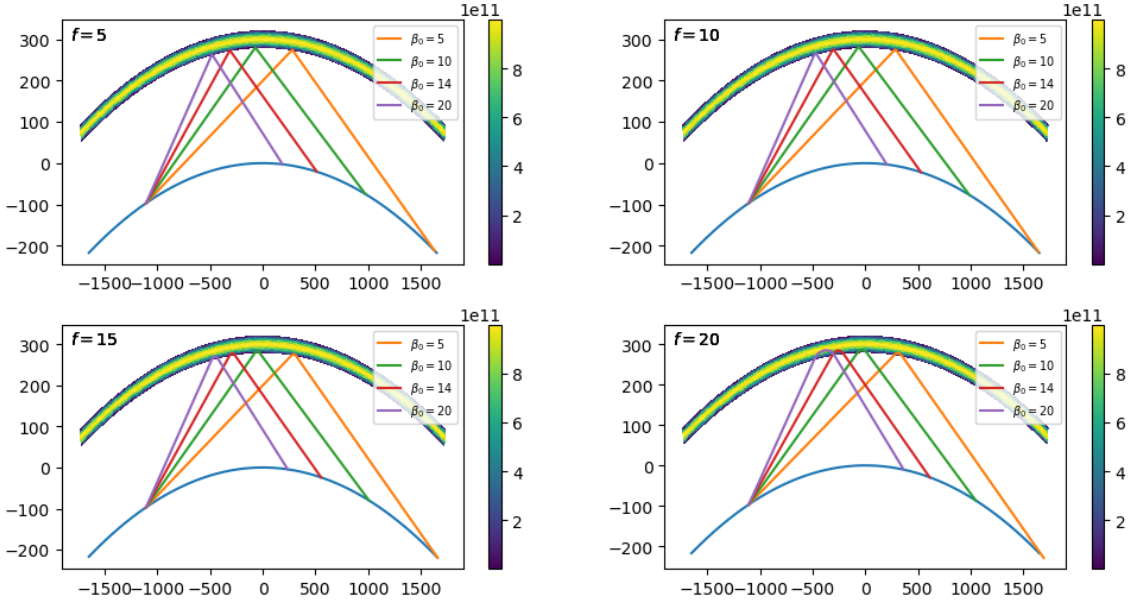
Ionosphere	f_o	$\beta_0 = 5$	$\beta_0 = 10$	$\beta_0 = 15$	$\beta_0 = 20$	$r_{t,\max}$	β_p
I	5	280.28	280.34	280.45	280.6	300.04	nan
	10	281.14	281.42	281.87	282.51	299.99	62.54
	15	282.69	283.38	284.56	286.33	299.89	32.98
	20	285.16	286.65	289.52	296.04	299.76	20.67
II	5	250.63	250.79	251.06	251.43	300.26	nan
	10	252.59	253.28	254.43	256.03	299.91	62.54
	15	256.12	257.84	260.81	265.27	299.32	32.99
	20	261.81	265.53	272.74	289.05	298.49	20.68
III	5	201.03	201.36	201.9	202.64	301.07	nan
	10	204.27	205.66	207.97	211.21	299.63	62.54
	15	210.16	213.62	219.62	228.58	297.23	33.00
	20	219.8	227.28	241.73	274.14	293.88	20.73
IV	5	101.16	101.83	102.92	104.42	304.40	nan
	10	104.81	107.63	112.33	118.91	298.47	62.54
	15	111.59	118.62	130.76	148.91	288.60	33.06
	20	123.02	138.0	166.84	229.21	274.84	20.95
V	5	50.87	51.71	53.09	54.98	306.98	nan
	10	53.63	57.18	63.1	71.38	297.57	62.54
	15	58.79	67.62	82.89	105.7	281.93	33.10
	20	67.6	86.28	122.13	197.86	260.17	21.11

Table 2 lists the apogee height r_t, the maximum apogee height $r_{t,\max}$ and the corresponding elevation angles, the Pedersen angle β_p. This table is produced using python script calc_rt_max_beta_p.py. Figure 2 shows ray tracing and ionosphere heatmap for cases of four frequencies, four elevation angles, and five model ionospheres. These figures, including ray tracing, are produced using python script ray_tracing.py.

Remark 0. The larger the operating frequency and/or the higher the elevation angle, the higher the apogee heights.

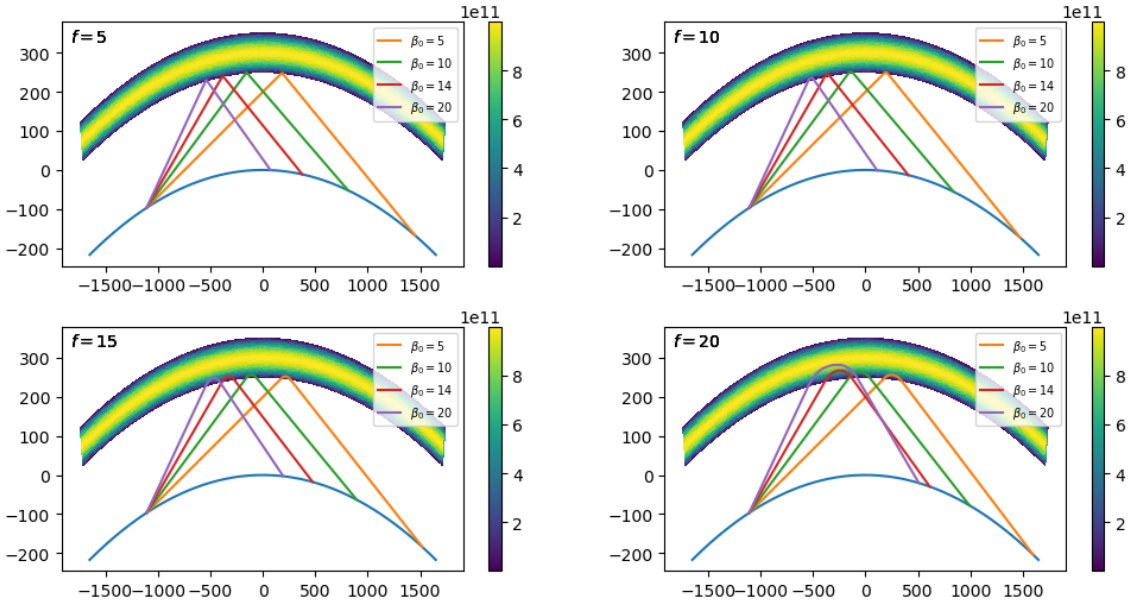
Remark 1. In all the cases considered, $r_t < r_{t,\max}$ and $\beta_0 < \beta_p$. This means that all rays are totally reflected.

Remark 2. For the case $f = 5$ [Mhz], the computed Pedersen angle β_p is 'nan' ('not a number'). This means that, at this low frequency, the radio ray would never be able to penetrate the model ionosphere, whatever the elevation angle may be.

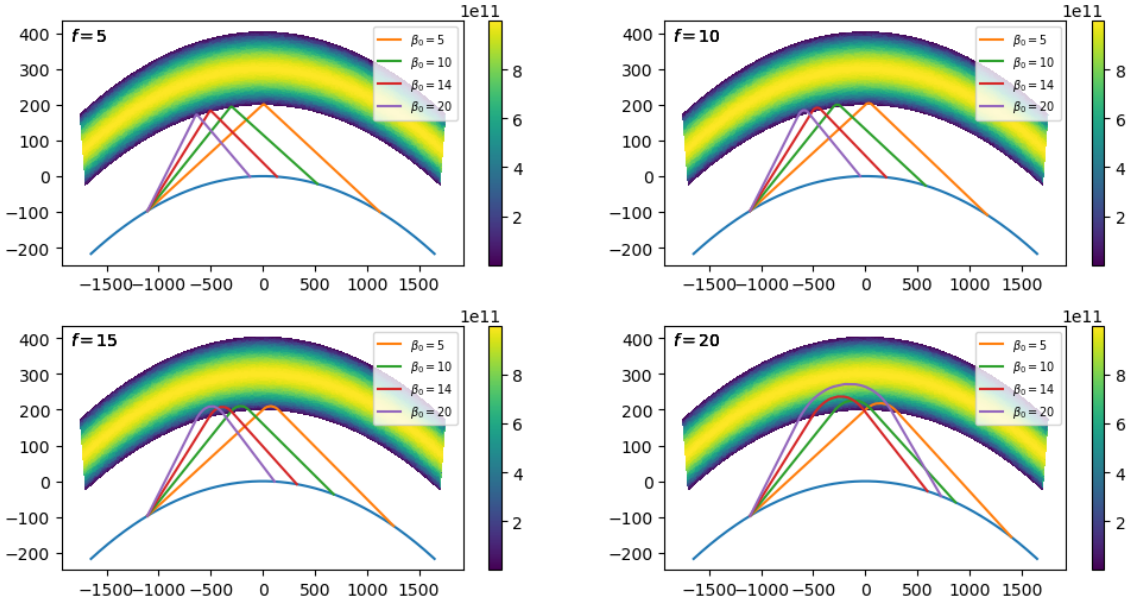


(a) Ray tracing for ionosphere I ($y_m = 20$ km).

Figure 2: Ray tracing for four frequencies, and four elevation angles, and five model ionospheres.

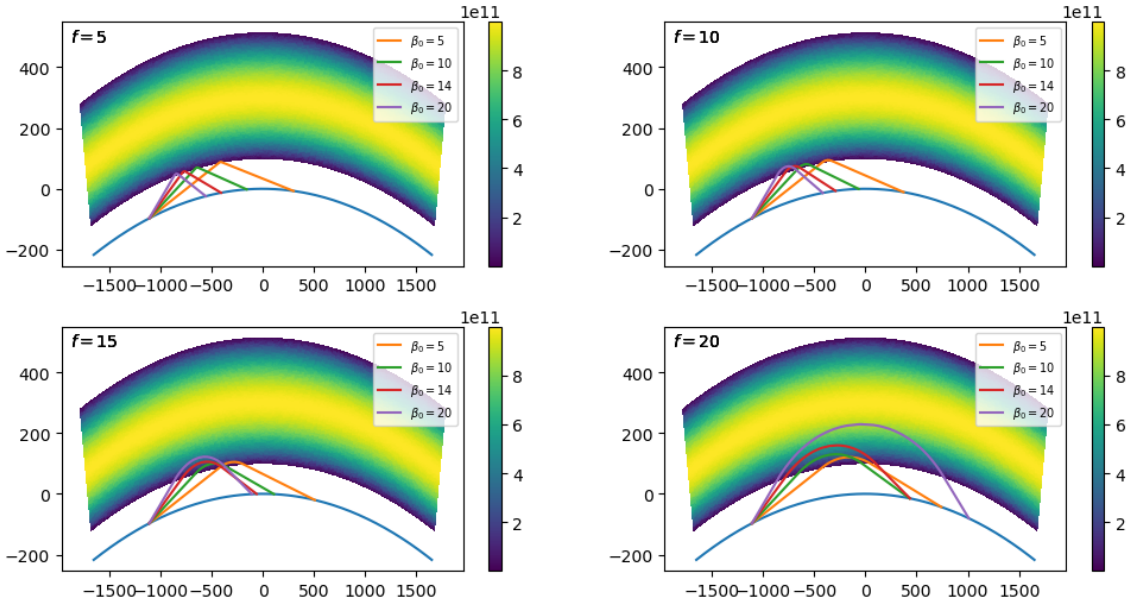


(b) Ray tracing for ionosphere II ($y_m = 50$ km).

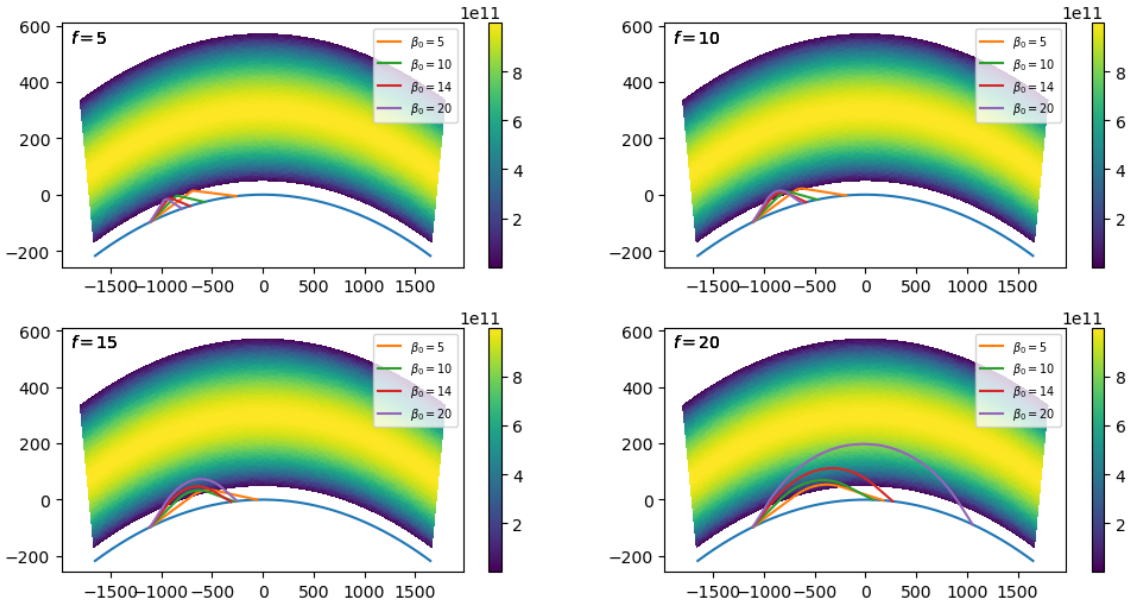


(c) Ray tracing for ionosphere III ($y_m = 100$ km).

Figure 2: Ray tracing for four frequencies, and four elevation angles, and five model ionospheres (cont.).



(d) Ray tracing for ionosphere IV ($y_m = 200$ km).



(e) Ray tracing for ionosphere V ($y_m = 250$ km).

Figure 2: Ray tracing for four frequencies, and four elevation angles, and five model ionospheres (cont.).

2 Cluster Analysis

For any data analysis, it's always a good idea to take a look at the original/raw data. Figure. 3 shows the raw data and "standardized" data. In standardized data, each value of each variable is subtracted from the mean and divided by the standard deviation of each variable. The standardized data are used in cluster analysis.

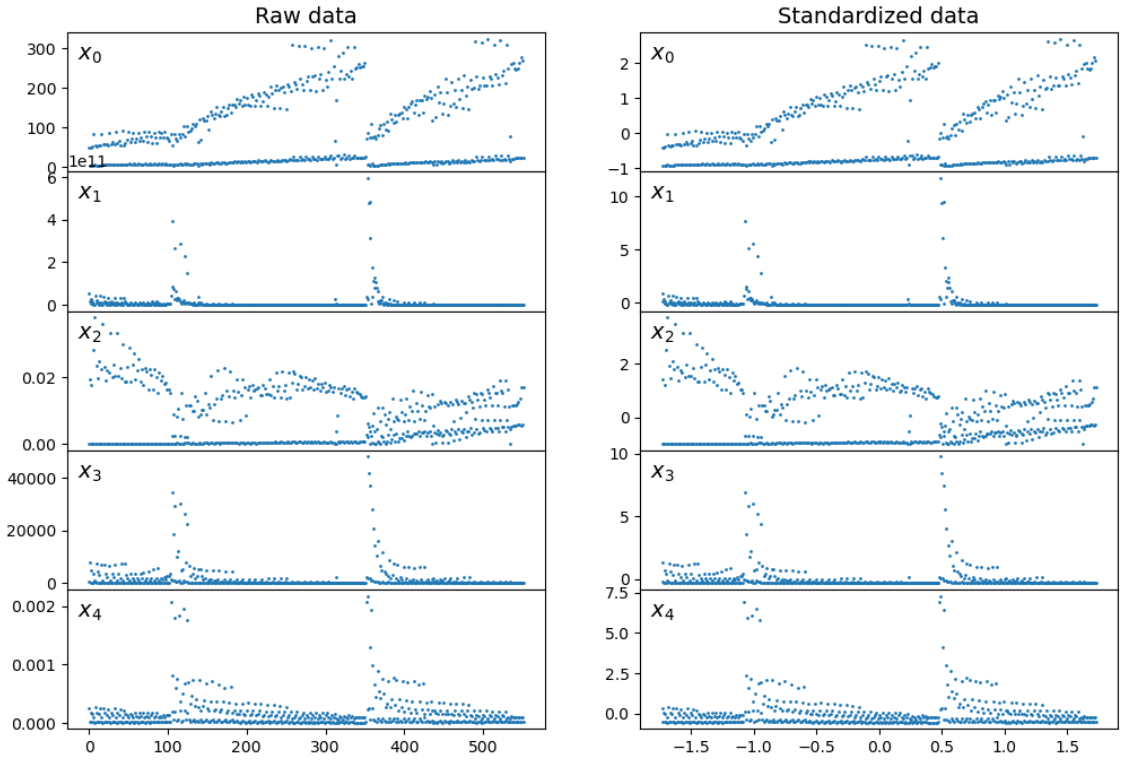


Figure 3: Raw data and standardized data.

Cluster analysis is one type of unsupervised learning techniques for unlabeled datasets. We use the K-Means algorithm in the Scikit-Learn package to perform the cluster analysis. To choose the "optimal" number of clusters, we use the elbow method and the silhouette score (Géron, 2019), see Fig. 4 and Fig. 5 for inertia and silhouette score, respectively. Figure 6 shows the cluster analysis results. Perhaps the cleanest result is the separation into two clusters, based on variables x_0 and x_2 ($x_0|x_2$), and less clearly based on variables x_1 and x_3 ($x_1|x_3$), as well as x_3 and x_4 ($x_3|x_4$). The other analysis is not quite clear, and perhaps other cluster analysis methods can be used to improve the results. Python script cluster_analysis.py is used for cluster analysis and plotting.

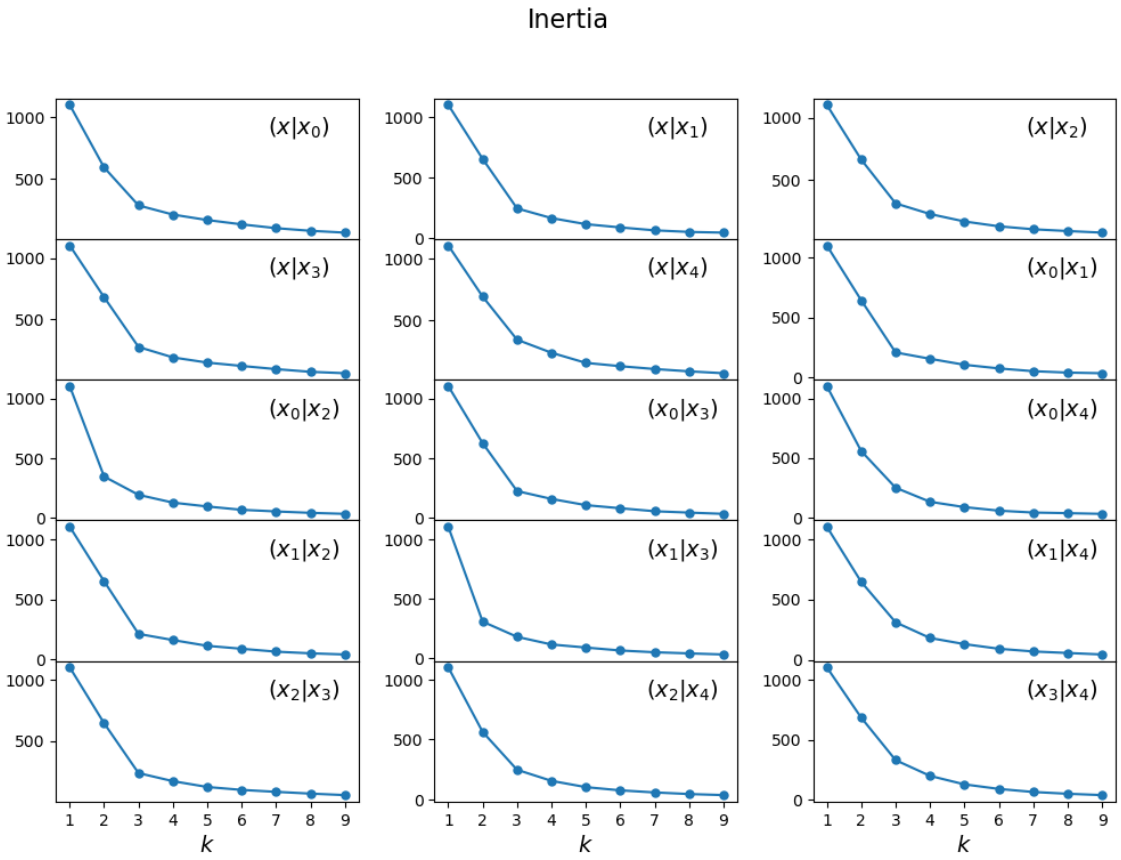


Figure 4: The inertia as a function of the number of clusters k. The "optimal" number of clusters can be chosen to be the inflection point called the "elbow".

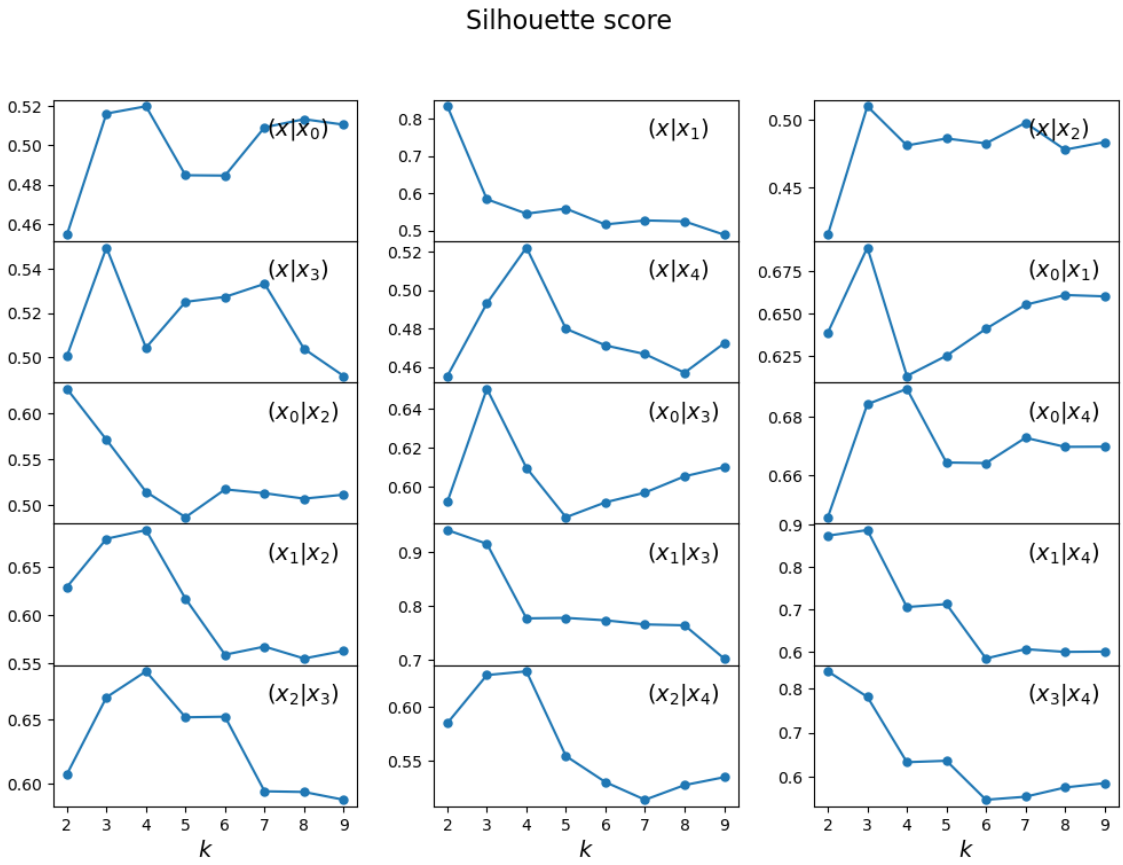


Figure 5: The silhouette score as a function of the number of clusters k. The "optimal" number of clusters can be selected using the silhouette score.

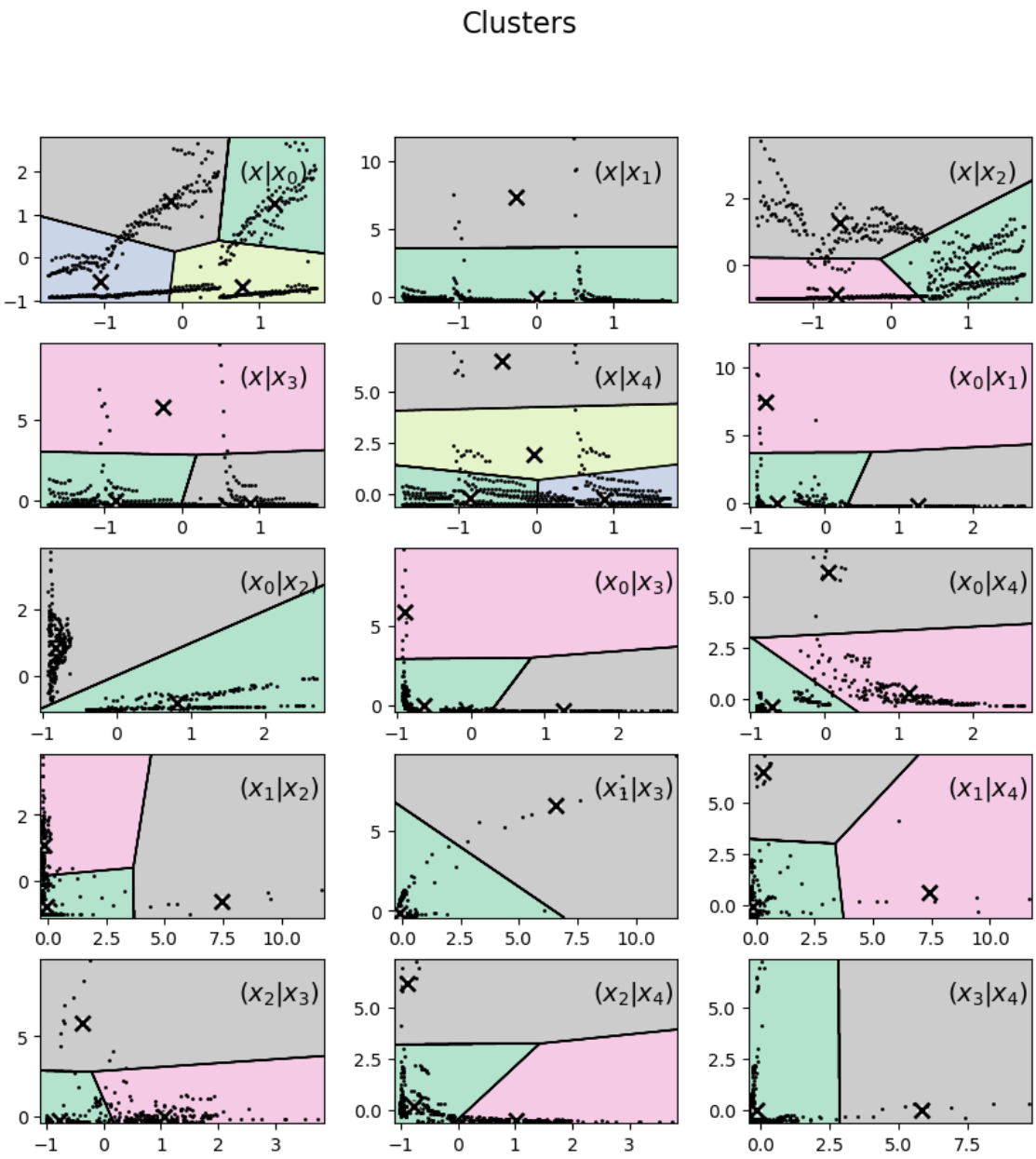


Figure 6: The clusters. The number of clusters are selected using the elbow method and the silhouette score.

3 Convolution

Convolution is a mathematical operation on two functions that produces a third function that expresses how the shape of one is modified by the other. In 2d convolution, a small matrix of weights, called "kernel" operates on ("slides" over) the 2d input data, performing an elementwise multiplication with that part of the input. Different kernels can be used for different purposes. There are many applications of convolution operations, such as convolution filtering in signal processing, and image and text processing in convolutional neural networks (CNN).

In this exercise, we perform the 2d convolution using the Scharr operator. The Scharr operator is a discrete differentiation operator that can be used to compute an approximation of the gradient of the image intensity function. Thus it can be used to detect the edges of objects in an image.

Figure 7 show the computation results of 2d convolution. The first 2 rows are the raw data, plotted using scatter plot and pseudocolor plot on an unstructured triangular grid, respectively. The third row shows interpolated data. The left and right hand sides are for dataset 0 and dataset 1, respectively. The last row shows the absolute gradient the 2d fields, resulting from the 2d convolution of the interpolated data with the complex Scharr operator. It demonstrates that this kind of 2d convolution can be used to detect the edges of the 2d dataset/image very well. Python script convolve2d.py is used for the computation and plotting.

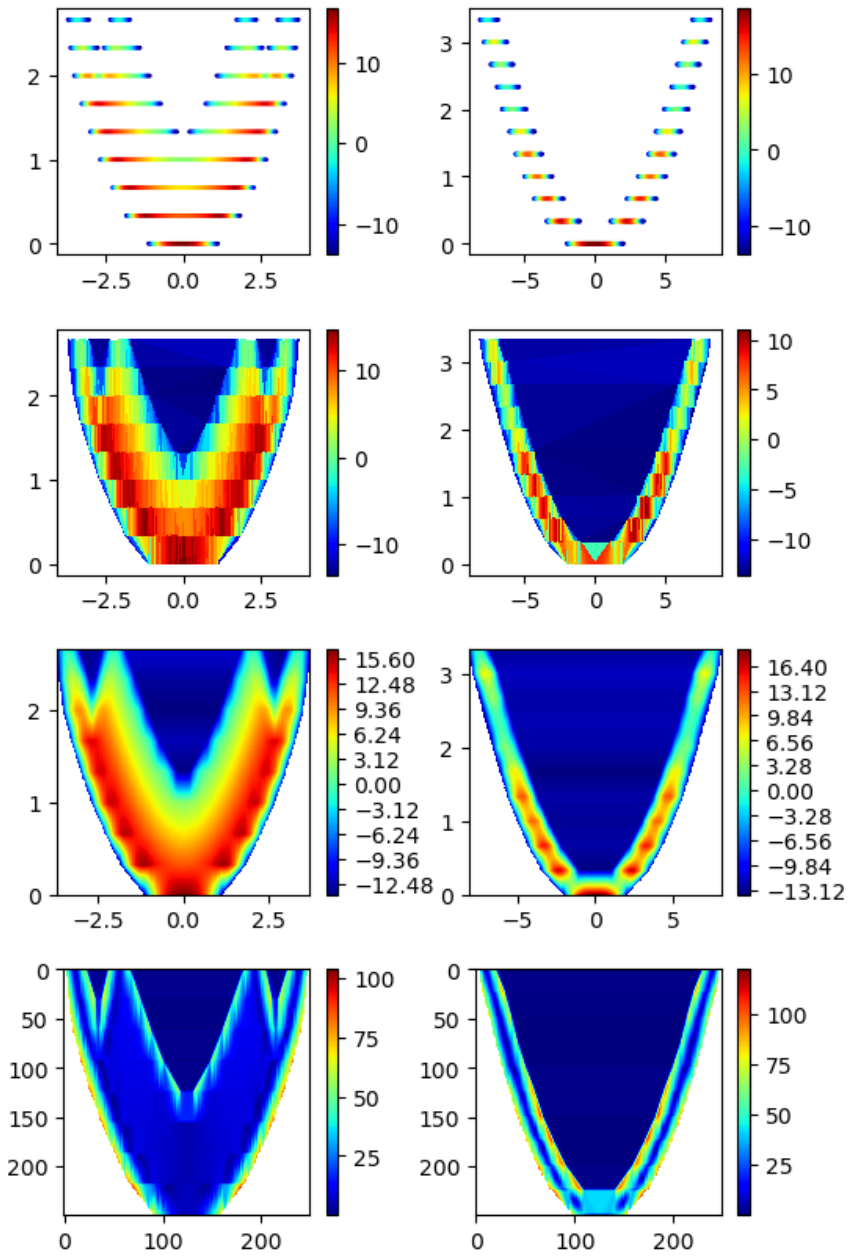


Figure 7: Computation of 2d convolution. The first 2 rows are the raw data, plotted using scatter plot and pseudocolor plot on an unstructured triangular grid, respectively. The third row shows interpolated data. The left and right hand sides are for dataset 0 and dataset 1, respectively. The last row shows the absolute gradient the 2d fields, resulting from the 2d convolution of the interpolated data with the complex Scharr operator.

4 Modeling Fluid Dynamics

The 1D equations for the mass and momentum conservations (without forcing terms) are written in the form

$$\begin{aligned} \frac{\partial \rho}{\partial t} + \frac{\partial \rho u}{\partial x} &= 0 \\ \frac{\partial \rho u}{\partial t} + \frac{\partial \rho u u}{\partial x} &= 0 \end{aligned} \tag{1}$$

Expanding the terms in the momentum equation and making use of the equation of mass conservation, the momentum equation is transformed backed to the simple advection form

$$\frac{\partial u}{\partial t} + u\frac{\partial u}{\partial x} = 0$$

which can be further written in the form

$$\frac{\partial u}{\partial t} + \frac{1}{2}\frac{\partial u^2}{\partial x} = 0 \tag{2}$$

i.e., transport of velocity u by velocity $\frac{1}{2}u$.

This form of the equation of motion appears to be better than the original form for the test cases. The original momentum conservation is written in the form of transport of the momentum density ρu (momentum per unit volume) by flow velocity. To get u from the updated ρu, we need to divide ρu by ρ. This can cause larger numerical errors near the edges of the square wave where large gradients of ρ exist. But this certainly does not mean it is always the case; in many cases the original conservation form is the choice; e.g., when the full set of equations, including conservation of energy and forcing terms, and/or other numerical methods, including using numerical flux limiters, etc. In any case, it would be interesting to compare the simulations using two equivalent forms.

For the 1D tests, a uniform mesh is used with $\Delta x = 1$ m and total 1000 grid cells. Variables are defined at the cell centers. The initial condition for the density [m^{-3}] is a square wave, given by

$$\rho(x) = \begin{cases} 25.0 & 249.5\,\mathrm{m} \leqslant x \leqslant 448.5\,\mathrm{m} \\ 1.0 \times 10^{-5} & \text{elsewhere} \end{cases}$$

Two initial conditions for velocity are used. The first one is a constant:

$$u(x) = 10.0 \qquad \text{(I)}$$

and the second one is a single sine wave:

$$u(x) = 10.0\sin(2\pi x/1000) \qquad \text{(II)}$$

Figure 8 shows simulations using two equivalent forms of equation of motion for test case I. In this case, the initial velocity is a constant. Without forcing, the velocity should remain constant. Indeed, in the simulation using the second form, the velocity remains constant. The simulation using the first form has some small spikes near the edges of the square wave. Since the system is essentially in a linear regime, differences in simulations between two forms are relatively small.

Now if we choose a different initial velocity profile, such as a sine wave (test case II), the system becomes nonlinear. We can see the steepening of the initial wave, as shown in Fig. 9. In this case, simulations can become quite different using two equivalent forms of equation of motion.

The source code in C++ is in src folder, including test1d.cc and Makefile. Python scripts plot_test1d.py and create_animation.py are used for plotting and creating animation. The animations are shown for 10 seconds simulation of test case I and 15 seconds simulation of test case II.

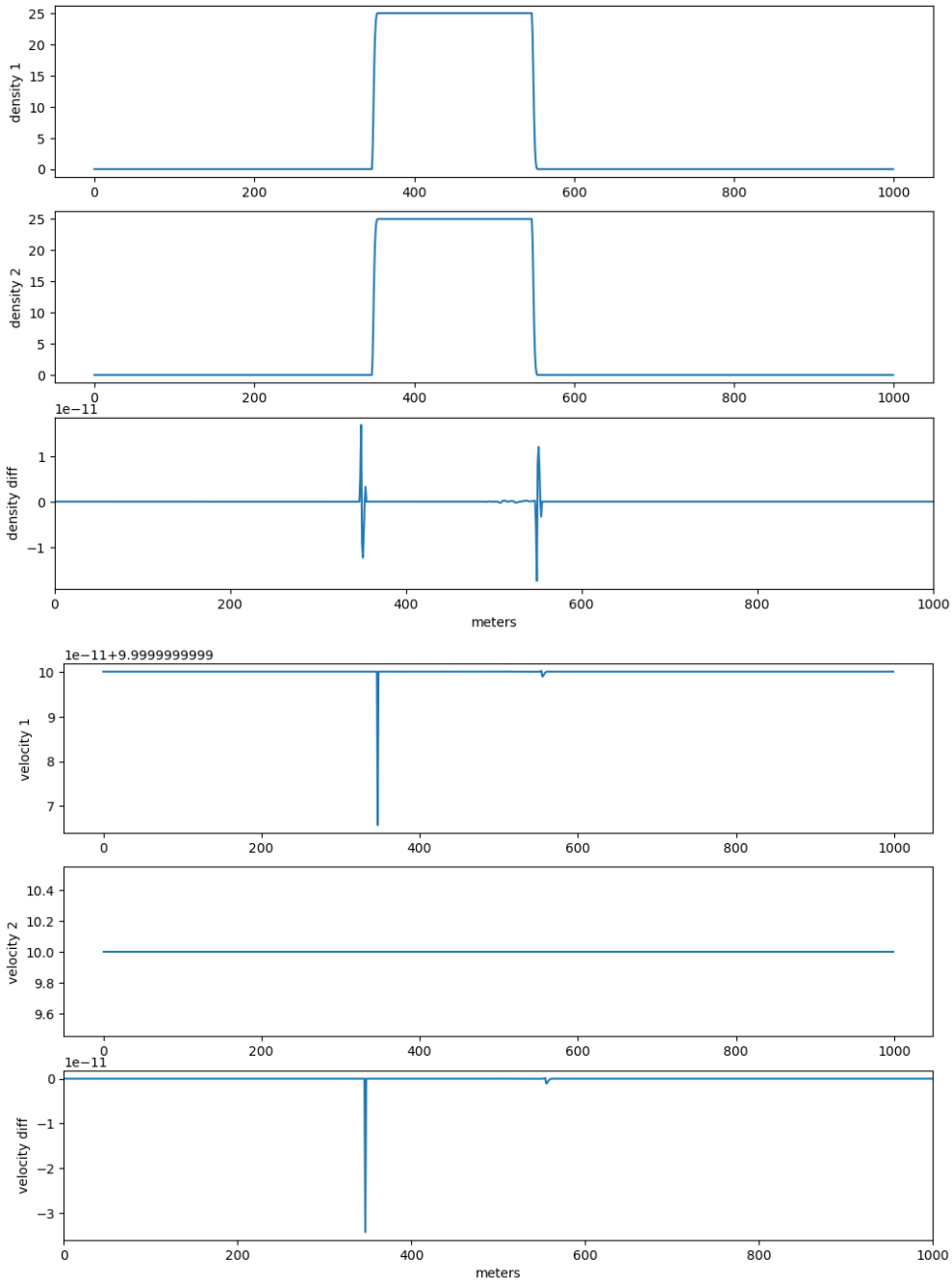


Figure 8: Comparing simulations using two equivalent forms of equation of motion for test case I: density and velocity at 10 second. The top three panels are for density and the lower three panels are for velocity.

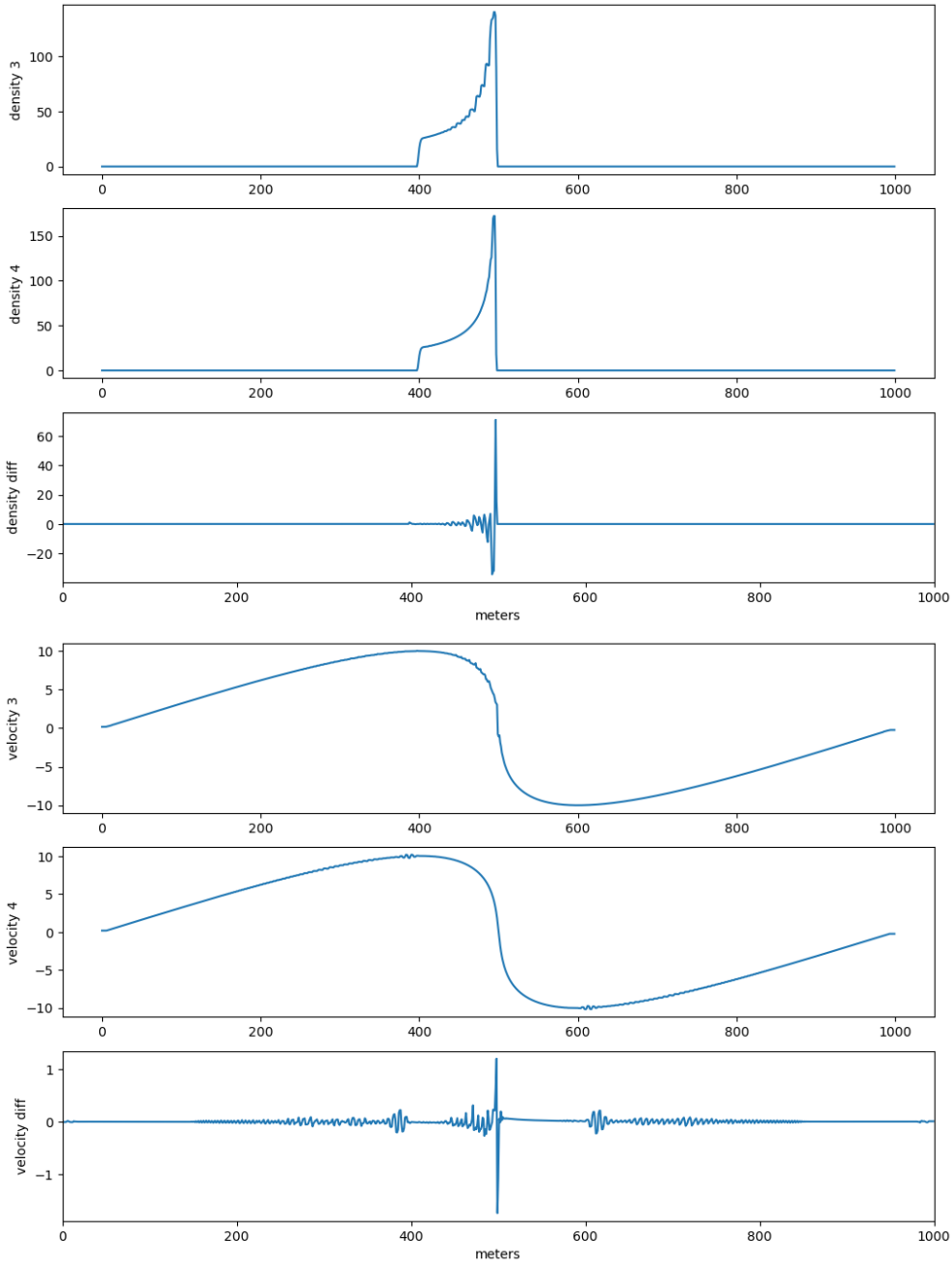


Figure 9: Comparing simulations using two equivalent forms of equation of motion for test case II: density and velocity at 15 second, The top three panels are for density and the lower three panels are for velocity.