Answers

Problem 1.

A fourth-order Runge-Kutta (RK4) integrator with the prototype, def rk4_step(fun, x, y, h), taking one step was constructed to integrate the following function:

$f(x,y) = dy/dx = y(1+x^2)^{-1}$

in the x-domain [-20, 20] with an initial value y(-20)=1 using 200 steps.

Next, another RK4 integrator to achieve a targeted improvement in accuracy with the prototype def rk4_stepd(fun, x, y, h) was written. This modified RK4 integrator would take a step length h, compare it to two steps of length h/2, and use them to cancel out the leading-order error term from RK4.

RK4 is based on the following relationship:

$y(x+h) = y(x)+(k_1 + 2k_2 + 2k_3 +k_4)/6$

where four function calls are made:

$k_1 = hf(x,y)$

$k_2 = hf(x+h/2,y+k_1/2)$

$k_3 = hf(x+h/2,y+k_2/2)$

$k_4 = hf(x+h/2,y+k_3)$

Let the single-step evaluation equal y_1 and the two-step evaluation equal y_2., with error term $O(h^6)$ to be eliminated:

$y(x+h) = y_1+ \phi(2h)^5 + O(h^6) + \ldots = y_1+ 32\phi h^5 + O(h^6) + \ldots$

$y(x+h) = y_2 + 2\phi h^5 + O(h^6) + \ldots$

Solution of the preceding pair of equations reveals the following expression:

$15y(x+h) = 16y_2 - y_1$

$y(x+h) = y_2 + (1/15)[y_2 - y_1]$

The method rk4_step required 4 function evaluations per step, while the rk4_stepd would demand 4x3=12 function evaluations per step. Thus, the number of steps for rk4_stepd was set at 200/3~67. The total number of calls in integration via rk4_step and rk4_stepd equaled 800 and 804, respectively.

The analytical solution to the differential equation was evaluated for comparison purposes:

$y=c_0\exp(\arctan(x))$

Evaluate y(-20) = 1 to solve for c_0.

$1= c_0\exp(\arctan(-20))$

$c_0 = [\exp(\arctan(-20)]^{-1}$

Both integrator results were in close agreement with that of the analytical solution (Figure 1) in the corresponding domain.

The absolute error of each integration method is shown in Figure 2. In addition, the Root Mean Square Error (RMSE) for each integrator were calculated. RMSE was reduced by approximately 40 folds via the rk4_stepd integration method, supporting the efficiency and targeted accuracy improvement for approximately the same number of function evaluations in either integrator.

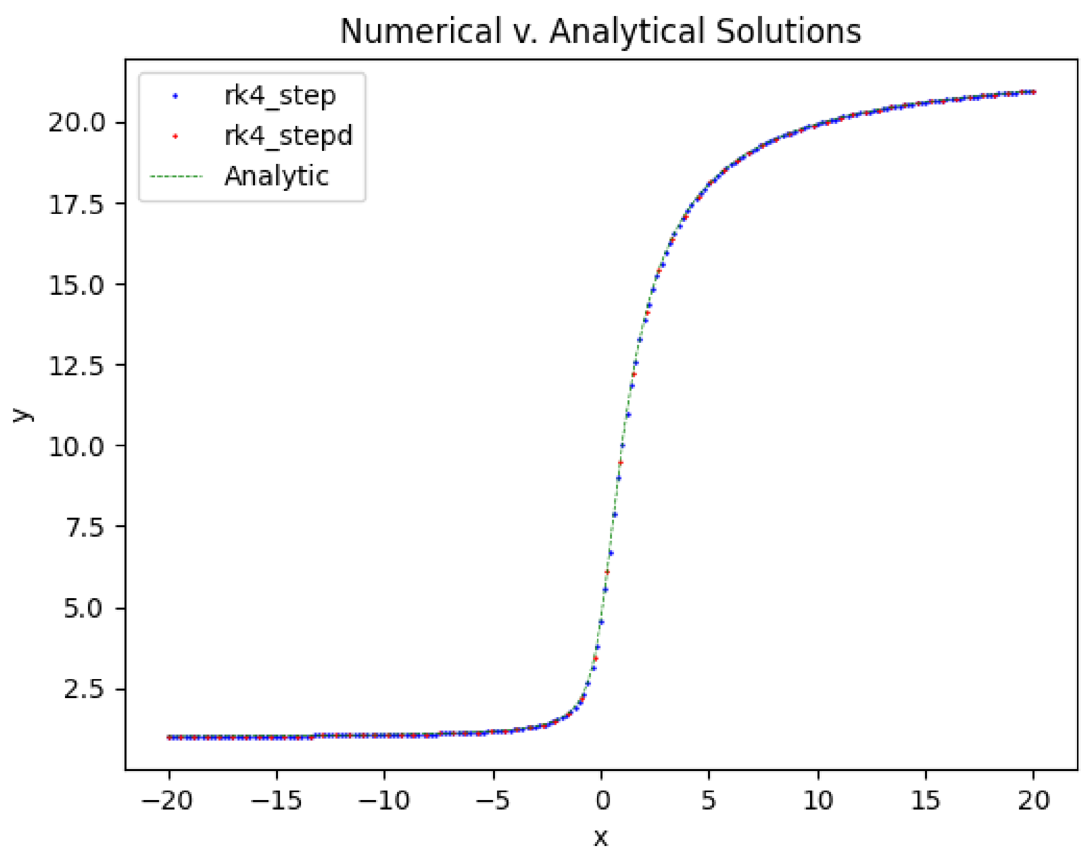

Figure 1. Comparison of numerical solutions to the analytical solution

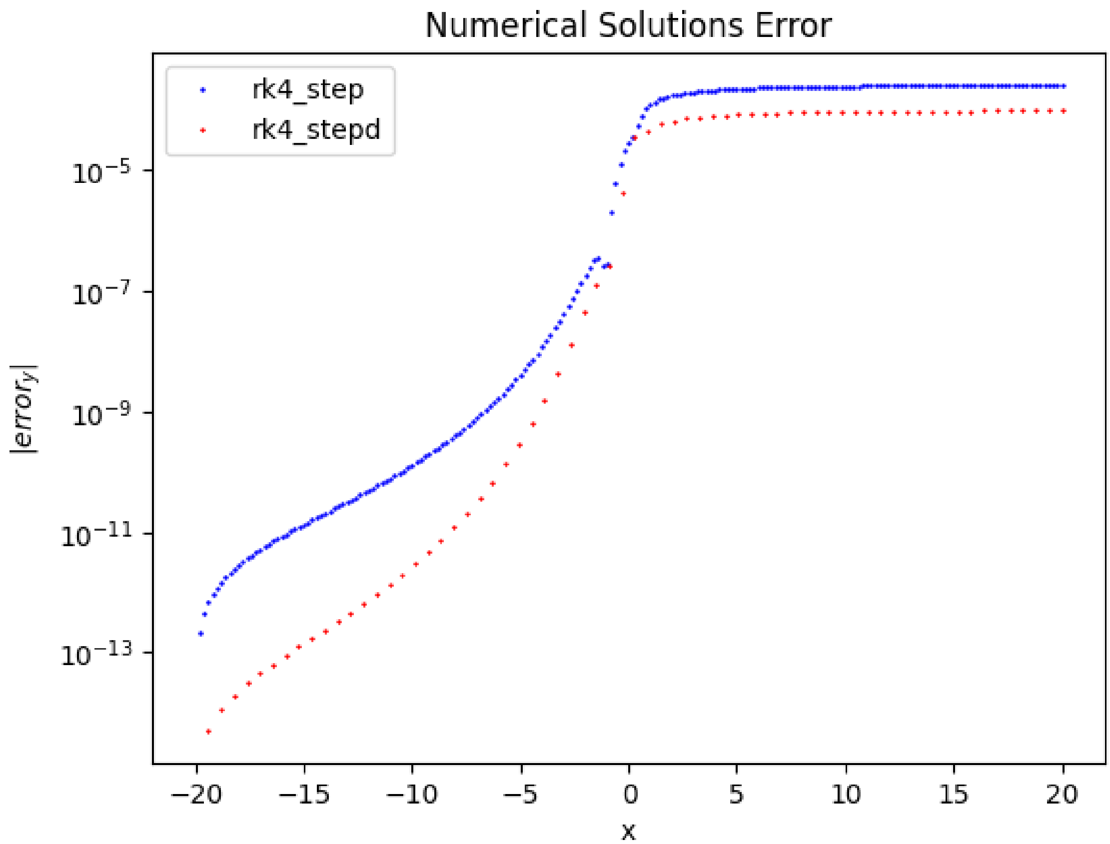

Figure 2. Comparison of numerical solutions error

Problem 2.

a)

A program to solve for the decay products of U238 according to first-order kinetics was written to determine the nuclide population over time for all decay elements in the chain ending with the stable isotope Pb206. A total of 15 nuclides with 14 half-lives $t_{1/2}$ were considered.

The assumption was that pure U238 was the initial specimen, whose population fraction was adopted as 1 (i.e. 100%). Let N(t) represent the population fraction of any nuclide i except for Pb206 at time t, whose decay rate -dN(t)/dt could be modeled after first-order kinetics:

$-dN_i(t)/dt = k_i[N_i(t)]$

where the first order rate constant $k_i = \ln 2/t_{1/2\_i}$

Nuclide fraction N_f of the beginning composition of pure element based on $N_o = 1$ at t=0 for U238 established the initial value to solve the differential equations.

A time domain of [0, $10^3 t_{1/2}$] where the reference half-life was adopted from that of U238. The initial value problem was solved with maximum step size of 10^8. The Python ODE solver *scipy.integrate* with the ***Radau*** method was employed. Given the vast disparity between the orders of magnitude of the rate constants (i.e. μseconds to Gyears), stiffness was inevitable. Whereas explicit Runge-Kutta methods should be useful for non-stiff problems, implicit methods such as *Radau* was appropriate for the U-238 decay chain, which would otherwise demand excessive number of iterations and/or would simply fail to generate a solution. The nuclide fractions over time are shown in Figure 3.

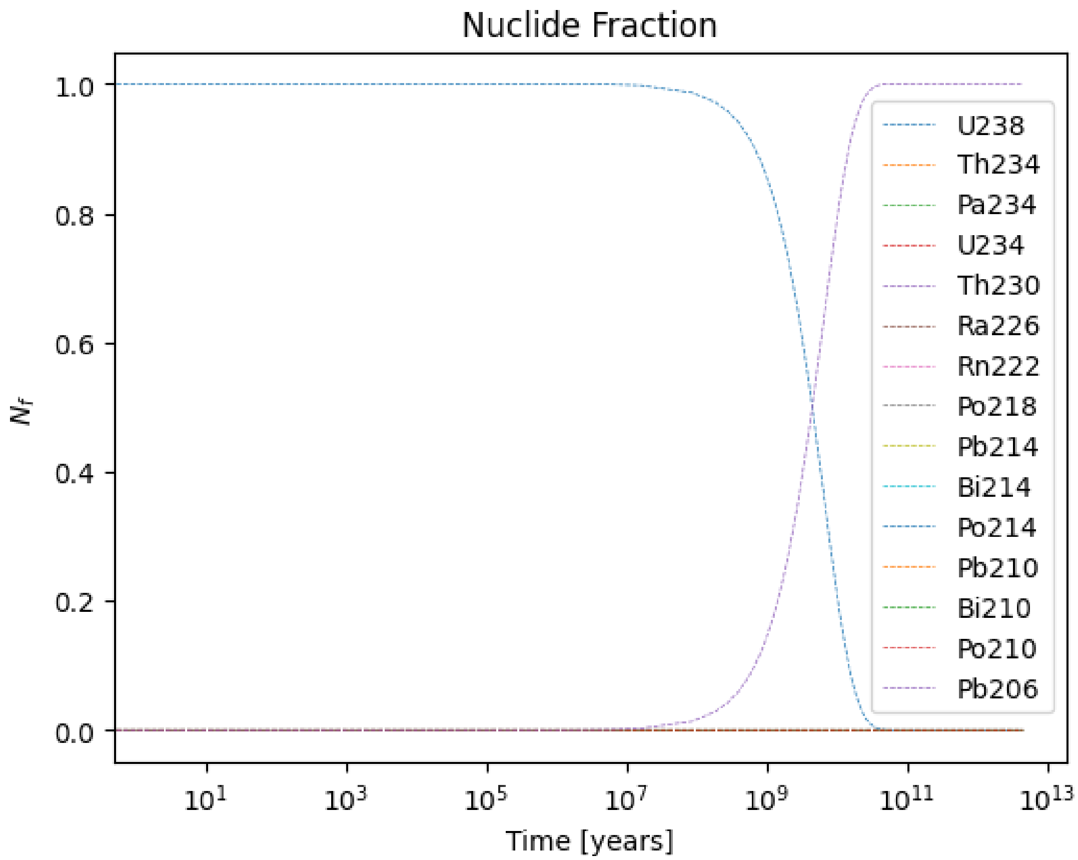

Figure 3. Fraction of each nuclide in the U238 to Pb206 decay chain

b)

The ratio of Pb206 to U238 was plotted as a function of time with a sensible range of fractions over a domain of interest [0, 50 Gyears]. The ratio of nuclides ranged from 0 to 2000 (Figure 4). Since $t_{1/2}$ of the first step in the chain is far greater than those of all other steps, U238 may be approximated as directly decaying into Pb206. The simplified kinetics would be:

$N_{U238} = N_{U238\_0}\exp(-k_{U238}t) = \exp(-k_{U238}t)$

$N_{Pb206} = N_{U238\_0} - N_{U238\_0}\exp(-k_{U238}t) = 1 - \exp(-k_{U238}t)$

The simplified analytic solution is an acceptable model (Figure 4) which is in good agreement with the numerically evaluated model based on a system of several differential equations.

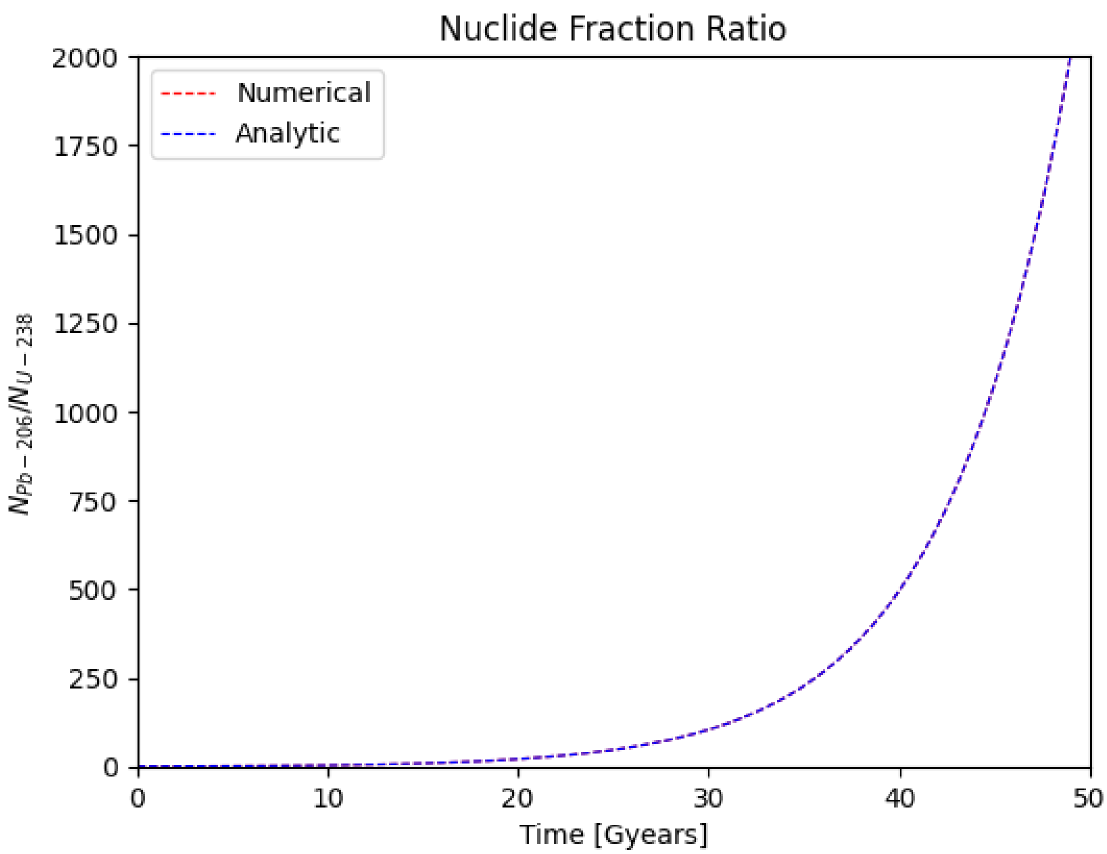

Figure 4. Ratio of Pb206 to U238 populations over 50 billion years

Moreover, the population ratio of Th230 to U234 was computed for the time domain of interest [0, 5 Myears]. The ratio increased from 0 to 0.3 over 1 Myears and reached a steady-state for the remainder of the domain, categorized as a secular equilibrium. U234 has a much longer $t_{1/2}$ than Th230. When the daughter nuclide Th230 has a fast decay tendency, this equilibrium arises because of equal rates of production and consumption of Th230, as evidenced by the horizontal asymptote observed for Th230 to U234 population ratio (Figure 5).

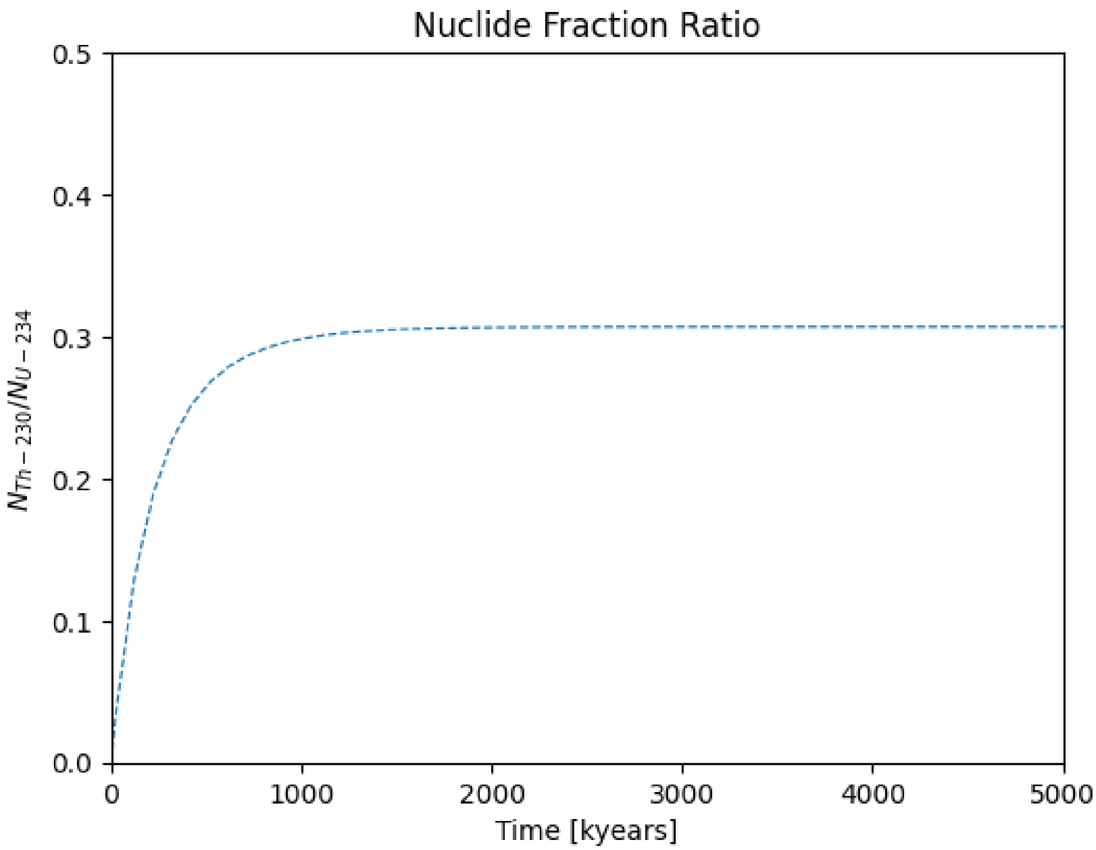

Figure 5. Ratio of Th230 to U234 populations over 5 million years

Problem 3.

A linear least-squares fit was implemented on the variables from the file dishzenith.txt, which contained photogrammetry data for a prototype telescope dish. A reconstruction of the three-dimensional points in the form of a dish was observed. The file provided (x, y, z) positions in [mm] of a few hundred targets placed on the dish. An ideal telescope dish was expected to be rotationally symmetric. This study intended to evaluate the shape of the dish, which would be compared to that of a paraboloid. The raw data plot can be seen in Figure 6.

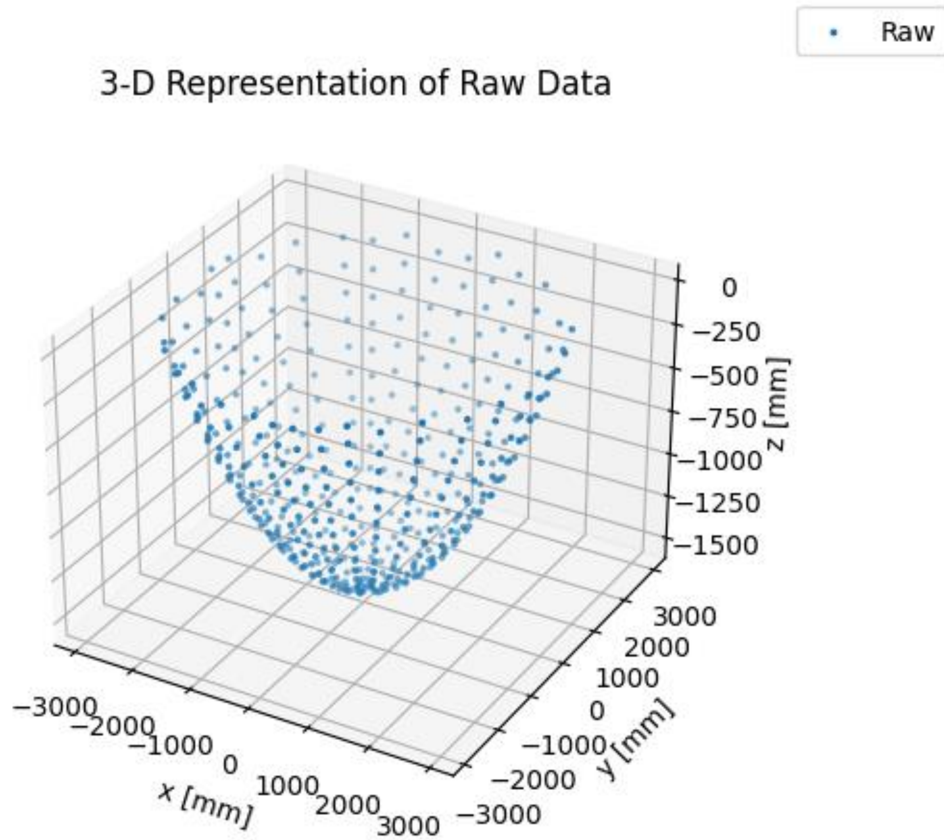

Figure 6. Raw data plot of the satellite dish

a)

Let a rotationally symmetric paraboloid be represented by:

$z - z_0 = a[(x - x_0)^2 + (y - y_0)^2]$

Although the preceding relationship appears to be non-linear, a quick parametrization could convert it to a linear one as follows:

$z = ax^2 - 2axx_0 + ax_0^2 + ay^2 - 2ayy_0 + ay_0^2 + z_0$

$z = a(x^2 + y^2) - 2ax_0x - 2ay_0y + ax_0^2 + ay_0^2 + z_0$

The new set of parameters **$A=a$, $B=-2ax_0$, $C=-2ay_0$, and $D=ax_0^2 + ay_0^2 + z_0$** indicate the following linear relationship:

$z = A(x^2 + y^2) + Bx + Cy + D$

b)

The linear z expression with parameters A, B, C, and D were analyzed via singular value decomposition approach, utilizing the Python *numpy.linalg.svd* function. The resultant fit parameters were reverted to the initial set of parameters of the putative symmetric paraboloid:

$a=A$, $x_0 = -B/(2A)$, $y_0 = -C/(2A)$, $z_0 = D - ax_0^2 - ay_0^2 - z_0$

The best fit parameters were identified as:

$a= 1.6670 \times 10^{-4}$

$x_0= -1.3605$

$y_0= 5.8221 \times 10^{1}$

$z_0 = -1.5129 \times 10^{3}$

The comparison of raw data and the fit data is shown in Figure 7.

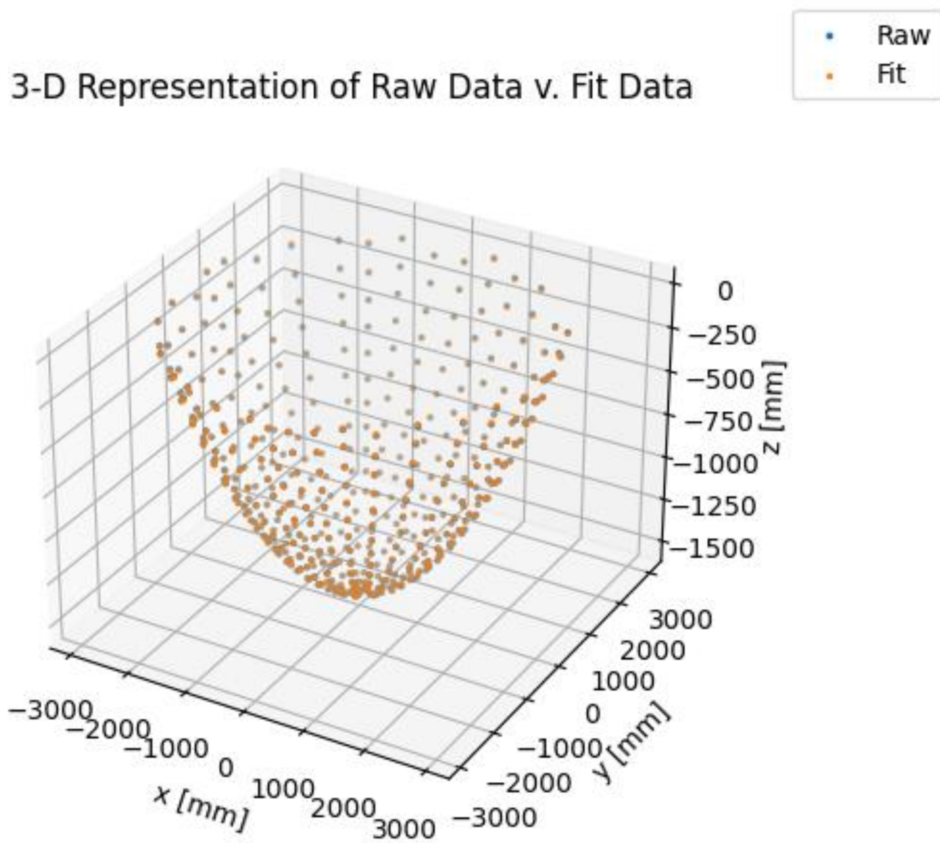

Figure 7. Comparison of raw data and fit data plots of the satellite dish

c)

Given that noise in the data was not provided, an estimate could be carried out based on the mean square of differences between the fit and raw z values – N_s. A diagonal N matrix for noise with common term N_s was utilized. Subsequently, the parameter fit evaluation could be accomplished via an analysis of the covariance matrix.

Square root of the covariance matrix for the parameters $Cov=(M^TN^{-1}M)^{-1}$, where M is the raw data matrix, was examined. The first term in the resultant $Cov^{1/2}$ matrix represented the error Δa in parameter a, whose units were [$meters^{-1}$]. The focal length of the paraboloid f equals$(4a)^{-1}$:

$f(a)=(4a)^{-1}$

A first order Taylor expansion of propagation of error in a, Δa, = $6.4519x10^{-5}$ $meters^{-1}$ facilitated calculation of the error, Δf, in the focal length of the paraboloid.

$\Delta f = \Delta a|f'(a)|$

$\Delta f = \Delta a(4a^2)^{-1}$ where $f'(a)=-(4a^2)^{-1}$

The focal length was computed as **1.4997 m** with an estimated error of **$5.8041x10^{-4}$ m**. The calculated focal length agreed well with the target focal length of 1.5 meters with approximately 0.04% error in its estimation.

d) *BONUS*

A more realistic approach to model the dish was its representation as a circularly non-symmetric surface rotated by some unknown angle θ. Let x' and y' equal the original coordinates. The observed coordinates x and y may be related to x' and y' as follows:

$x= \cos\theta\, x'+\sin\theta y'$

$y= -\sin\theta x'+\cos\theta y'$

Let $x' = x - x_0$, $y' = y - y_0$, and $z' = z - x_0$

$z = ax^2+by^2$

$z = a(\cos\theta\, x'+\sin\theta y')^2+b(-\sin\theta x'+\cos\theta y')^2$

$z = a\cos^2\theta x'^2+a\sin^2\theta y'^2+2a \cos\theta\sin\theta x'y'+ b\sin^2\theta x'^2+b\cos^2\theta y'^2 -2b\cos\theta\sin\theta x'y'$

$z = (a\cos^2\theta+b\sin^2\theta)x'^2+(a\sin^2\theta+b\cos^2\theta)y'^2+(a - b)\sin2\theta x'y'$

Once again, the relationship can be parameterized to achieve a linear configuration:

$z= Ax'^2+ By'^2+ Cx'y'+Dx'+Ey'+F$

where $A= a\cos^2\theta+b\sin^2\theta$, $B= a\sin^2\theta+b\cos^2\theta$, $C=(a - b)\sin(2\theta)$.

Upon fit evaluation and determination of A, B, and C, the original parameter θ, a, and b could be calculated as follows:

$A - B = a\cos^2\theta+b\sin^2\theta - (a\sin^2\theta+b\cos^2\theta)$

$A - B = (a - b)\cos(2\theta)$

$\tan(2\theta)=C/(A - B)$

$a = (A\cos^2\theta - B\sin^2\theta) / (\cos^4\theta - \sin^4\theta)$

$b = (A\sin^2\theta - B\cos^2\theta) / (\sin^4\theta - \cos^4\theta)$

The best fit parameters were identified as:

$A = 1.6573 \times 10^{-4}$

$b = 1.6776 \times 10^{-4}$

$\theta = -6.2000 \times 10^{-1}$

Accordingly, the focal lengths of the two principal axes, f_a and f_b, corresponding to an ellipsoid cross-section were determined as:

f_a = 1.5085 meters

f_b = 1.4902 meters

Although the focal lengths are in the proximity of the reference 1.5 meters of a symmetric paraboloid, the slight eccentricity of the cross-section indicates that the dish should be characterized as **not perfectly circular.**

Appendix A: Python Code

Jupyter notebook with relevant Python code and outputs is available at:

https://github.com/ck22512/comp_phys/tree/main/Assignment3