***Note:*** *the following is an excerpt from a currently-ongoing astrophysics research paper, which sheds more light on the inner workings and mechanisms involved in the Python code.*

1. **Equation form:**
2. **Derivation:**
   1. A best fit surface is to be obtained in terms of , and with high correlation.
   2. Method followed is similar to the one followed in Linear Regression (constructing the general equation in the currently assumed form and minimizing the Sum of Squares of Errors and obtain the corresponding values of the equation coefficients):
      1. Let:
      2. The equation for one tile will be
      3. Then the expression for Sum of Squares of Errors is
      4. For best fit conditions, the equation coefficients and should have such a value that is minimum
      5. Implies:
      6. Since these equations are nigh-impossible to solve using conventional methods, we have used an alternate method to “solve” them by running a Python program to find the value of from an assigned range with precision set to 5 decimal places (which is adjusted to include the real solution within it) that comes closest to solving the obtained set of constraint equations:
         1. Upon closer inspection it becomes apparent that solving the first three equations as a set of Linear Equations, we obtain , and in terms of (though the caveat is that we must assign value beforehand before trying to solve it)
         2. In theory, if we substitute these above-obtained relations in place of , and in the fourth expression, it should be equal to zero.
         3. In the context of our adopted method, this implies that the fourth constraint equation turns into an evaluation expression which becomes equal to zero when our best-fit condition is met.
         4. This works well with the idea of implementing a Python program to find the values closest to the true set of values for , , and . We set a range of values for such that the true value lies somewhere in between, and the program loops for each value of in the given range, finding the corresponding values of , and until our evaluation expression is closest to zero.
         5. The corresponding set of values obtained will then be taken as the estimated values of , , and as in the scope of the precision set, it is impossible to find a more suitable solution.
         6. The true value is guessed by checking whether our estimated value of is at the boundary of the range taken. Due to the nature of the method being implemented, if the estimated value coincides with the upper boundary, then it can imply that the true value may be beyond the upper boundary. Similarly, If the estimated value coincides with the lower boundary, then it can imply that the true value may be beyond the lower boundary. The program is rerun in such conditions with an expanded range to see if the estimated value changes.
   3. These estimated values for the equation coefficients are recorded for each possible tile in a separate excel file along with that tile’s minimum and its corresponding and automatically by the same program.