PHYS 250 Homework 2

Sophie von Coelln

9 February 2025

1 World Population

1 i

We linearly interpolate between populations in 1980 and 2000 to estimate the population in 1990:
Interpolated value: 5.310 billion people
Difference: 0.018
Fractional error: 0.341%

1 ii

We extrapolate that linear polynomial to estimate populations in 1960 and 2020.

For 1960, we get:
Interpolated value: 2.724 billion people
Difference: 0.292
Fractional error: 9.682%

For 2020, we get:
Interpolated value: 7.896 billion people
Difference: 0.009
Fractional error: 0.112%

1 iii

We then use that same polynomial to extrapolate populations in 2050 and 2100.

For 2050, we get 10.482 billion people, and for 2100, we get 14.792 billion people.

1 iv

We repeat parts i-iii, this time using a quadratic interpolation calculated from 1970, 1980, and 2000.

For 1990, we get:
Interpolated value: 5.273 billion people
Difference: 0.055
Fractional error: 1.023

For 1960, we get:
Interpolated value: 3.015 billion people
Difference: 0.001
Fractional error: 0.032

For 2020, we get:
Interpolated value: 8.187 billion people
Difference: 0.300

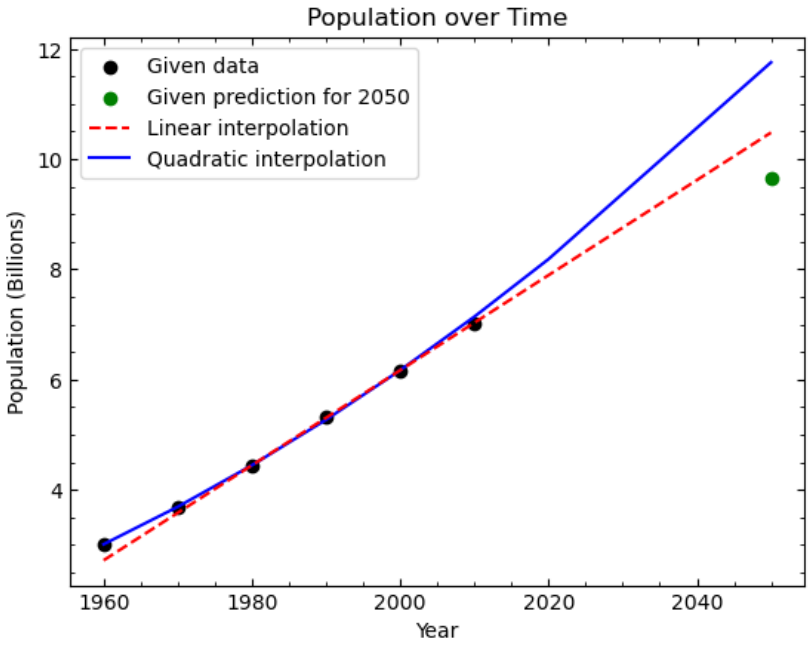
Fractional error: 3.801

For 2050, we get:
Interpolated value: 11.755 billion people

For 2100, we get:
Interpolated value: 19.157 billion people

1 v

Purely based on comparing both polynomials' extrapolated values for 2050 to the expected value given in the table, we expect a linear polynomial to give a better estimate in 2050 and 2100. Though the data grows somewhat quadratically in the mid-20th century, it takes an almost logistic shape when we consider the given estimation of population in 2050. The continued quadratic growth predicted by the quadratic interpolation diverges from this more than the linear interpolation does. We can see this when we plot the data:



At large inputs, the quadratic produces very large outputs, which is not the way we expect a population to scale. The linear interpolation does not produce such extreme outputs at large inputs, so it is a better function to extrapolate with in this circumstance.

2 Satellite Orbiting Planet

2 i

I used the `scipy.interpolate.make_interp_spline()` function to create a cubic spline interpolant for the times and distances given. This yielded -0.746" for day 5 and 2.748" for day 6.

2 ii

Since the angular distance between the planet and satellite are only measured once a day, we do not know how close to the measurement the satellite became occluded. To find the minimum distance, we assume it became occluded moments before the measurement would have been taken on day 5, and became visible again moments after it would have been taken on day 6. In the limit that the time between the satellite passing into/out of the planet's shadow and the measurement being made goes

to zero, we can find the minimum distance by using our spline interpolant to subtract the distance when the measurement on day 5 would have been made from the distance when the measurement on day 6 would have been made:

$$\text{Minimum distance:} \quad |\text{spline}(5) - \text{spline}(6)| = 3.49\text{"}$$

We follow much the same process to find the maximum distance. We assume it passed out of visibility immediately after the measurement on day 4 was taken, and passed back into visibility immediately before the measurement on day 7 was taken. In the limit that the times elapsed between those events goes to zero, we get can simply use the values measured on days 4 and 7:

$$\text{Maximum distance:} \quad |-3.9787359 - 5.3127362| \approx 9.29\text{"}$$

2 iii

To find the period of the satellite's orbit, we double the time elapsed between its minimum and maximum values. We turn to root finding to do this. We use Brent's method on the derivative, which can be easily calculated from our existing cubic spline. Brent's method requires us not only to input the function we want to optimize, but also to bracket the root. We do this by "eyeballing" bounds on the extrema.

2 iv

We estimate bounds of 2 and 3 for the local minimum, and 7 and 8 for the local maximum. We use Brent's method for root finding on the derivative of our spline, once with the first set of bounds for the local minimum, and once with the second set for the local maximum. We find roots at approximately 2.61 days and 7.80 days. Then,

$$2 \times (7.80 - 2.61) = 2 \times 5.19 = 10.38 \text{ days.}$$