Numerical Methods (DS288): Assignment 2

Name: Subhasis Biswas
Serial Number: 23-1-22571

Question 1 :

Computing the Mean Anomaly of the Moon (M):

This process involves computing how much angle (in radian) the Moon has covered since it was at the periapsis. Here, the time is being described in the following format:
DD-MM-YYYY HH:MM:SS

The time of the launch T_0: 14-07-2023 14:40:00
The time of orbital insertion T_1: 05-08-2023 19:00:00

Now, since the Moon's orbit around the Earth is almost circular ($e \approx 0.0549$), we can assume that the Moon orbits its host planet with almost uniform angular velocity, i.e. it sweeps equal amount of angle in unit time at any given position on its orbit. Let the uniform angular velocity be $\omega = \dfrac{2\pi}{T}$, T being the orbital period of the Moon; $T = 27.32$ days.
Therefore, the computation for M boils down to: $M = \dfrac{2\pi(T_1 - T_0)}{T} = \omega(\Delta t)$, since at T_0 the Moon was at the periapsis.

One way we can compute $(T_1 - T_0)$ is by hand. However, it's better to build a function that takes this kind of time format as inputs and returns the "Unix Time", say $unix(T_i)$.

<u>Definition of the Unix Time</u>: Number of seconds that have elapsed since 00:00:00 UTC on 1 January 1970, the Unix epoch, without adjustments made due to leap seconds.
Source: `https://en.wikipedia.org/wiki/Unix_time`

As it turns out: $M = \dfrac{2\pi(unix(T_1) - unix(T_0))}{24 \times 3600T} = 5.10119109706249$ rad. [This implementation has been carried out in the code]

Checking the Applicability of FPI:

We can write the Keplar's equation as: $E = M + e\sin(E) = \phi(E)$.
Now, $\phi'(E) = e\cos(E)$, and $|\phi'(E)| = |e\cos(E)| \leq |e| < 1$, $\forall E \in \mathbb{R}$. Therefore, for any initial approximation E_0, the fixed point iteration scheme $E_{n+1} = \phi(E_n)$ converges to some root E_* of the Keplar's Equation. So yes, the Fixed Point Iteration method is indeed applicable here.

Algorithm for the Fixed Point Iteration Method:

Let the function under the application of FPI method be $f(x)$, and let $f(x) = 0 \implies \phi(x) = x$, with $|\phi'(x)| < 1$ on $x \in I$, which is an interval containing the root of $f(x) = 0$. Let the relative tolerance be ϵ.

Relative error $\epsilon_n = \dfrac{|x_{n+1} - x_n|}{|x_{n+1}|}, \quad n = 0, 1, ...$

1. Take any initial approximation $x_0 \in I$.
2. $x_{n+1} = \phi(x_n), \quad n = 0, 1, ...$
3. Check if $(f(x_{n+1}) = 0$ or $\epsilon_n \leq \epsilon)$. If true, then exit the loop, else go to step 2.

[Note: The parenthesis in the last line means the following: (P or Q) means if either statement P or Q holds, then the statement: (P or Q) is True.]

Proof that there is only one possible root:

Consider $f(E) = E - M - e\sin(E)$. We have $f'(E) = 1 - e\cos(E)$, which has no roots on the real line, since: $f'(E) = 0 \implies \cos(E) = \frac{1}{e} > 1$, an impossibility. Thereby, applying the Rolle's theorem, we can say that $f(E) = 0$ can have at most one root. Now, observe that $f(-\infty) = -\infty$ and $f(+\infty) = +\infty$. So there must exist a root, and it is the only one.
Therefore, we can say conclusively that for any initial approximation to the root, the iteration scheme converges to that particular root, say E_*.

Computational Outcome of FPI:

$\phi(E) = M + e\sin(E)$; $E_0 = M$; Relative Tolerance=10^{-8}

$\mathbf{E_* = 5.04937899}$ (rounded off to 8 decimal places)
iteration count=5

Position with respect to the Earth:
$(\mathbf{x_*, y_*}) = (\mathbf{105997.49400736km, -362212.85668586km})$ (using the rounded-off value of E_*)

using $x = a(\cos E - e),\ y = b\sin E$, with the Earth at the origin $(0, 0)$.

[Since Keplar's laws state that the host celestial body must be located at one of the foci of the elliptical orbit, and the origin turns out to be location of the Earth, by construction of the governing equation]

Question 2 :
a)
Order of Convergence of the Newton-Raphson Method:

Firstly, let us proceed with α being a simple root of $f(x)$. Let at the nth stage, $e_n = x_n - \alpha$, i.e the difference between the nth approximation to the root and the actual root.
Therefore, the Newton's method $x_{n+1} = x_n - \dfrac{f(x_n)}{f'(x_n)}$ can be put as $e_{n+1} + \alpha = (e_n + \alpha) - \dfrac{f(e_n+\alpha)}{f'(e_n+\alpha)} \implies e_{n+1} = e_n - \dfrac{f(e_n+\alpha)}{f'(e_n+\alpha)}$.
Expanding $f(e_n+\alpha) = f(\alpha) + e_n f'(\alpha) + \dfrac{{e_n}^2}{2!} f''(\zeta_n)$, where $\zeta_n \in \mathbb{R}$, (We're approximating by first order Taylor Polynomial, with the last term as the Lagrange's form of remainder).
Similarly, $f'(e_n+\alpha) = f'(\alpha) + e_n f''(\eta_n)$
Thus, we get $e_{n+1} = e_n - \dfrac{e_n f'(\alpha) + \frac{{e_n}^2}{2} f''(\zeta_n)}{f'(\alpha) + e_n f''(\eta_n)} = \dfrac{{e_n}^2 f''(\eta_n) - \frac{{e_n}^2}{2} f''(\zeta_n)}{f'(\alpha) + e_n f''(\eta_n)}$. Now, taking $\sup |f''(\eta_n) - \dfrac{f''(\zeta_n)}{2}| = M$ and $\inf |f'(\alpha) + e_n f''(\eta_n)| = N (\neq 0)$ within a suitable neighbourhood of α we get $|e_{n+1}| \leq \dfrac{M{e_n}^2}{N}$, which implies that the convergence is quadratic.

With the observation that $f'(\alpha)$ is a fixed constant and the additional constraint that $N \neq 0$ within that neighbourhood, if such a neighbourhood exists in the first place. One can say that since we cannot completely free the denominator without making some sort of assumption regarding the behaviour of the function in a certain neighbourhood, and additionally the existence of such a neighbourhood at all, we cannot guarantee the order of convergence of the method (or even the convergence), even though the root is simple. Even in the proofs that conclude quadratic convergence as $|e_{n+1}| \leq \dfrac{|f''(\xi_n)|}{2|f'(x_n)|}{e_n}^2$, a hidden assumption is made that $f'(x)$ is well behaved at the approximations x_i's.

For a root α of multiplicity $m > 1$, we'll expand $f(\alpha+e_n) = f(\alpha) + e_n f^{(1)}(\alpha) + \dfrac{{e_n}^2}{2!} f^{(2)}(\alpha) + \ldots + \dfrac{{e_n}^m}{m!} f^{(m)}(\zeta_n)$ and all the terms $f(\alpha) = f^{(1)}(\alpha) = \ldots = f^{(m-1)}(\alpha) = 0$ but $f^{(m)}(\alpha) \neq 0$. [Here ζ_n and η_n are of different values than the previous section, but the idea is the same]
Again, $f'(\alpha + e_n) = f^{(1)}(\alpha) + e_n f^{(2)}(\alpha) + \ldots + \dfrac{{e_n}^{m-1}}{(m-1)!} f^{(m)}(\eta_n)$.
Therefore, $e_{n+1} = e_n - \dfrac{f(e_n+\alpha)}{f'(e_n+\alpha)} = e_n - \dfrac{\frac{{e_n}^m}{m!} f^{(m)}(\zeta_n)}{\frac{{e_n}^{m-1}}{(m-1)!} f^{(m)}(\eta_n)} = e_n - e_n \dfrac{f^{(m)}(\zeta_n)}{m f^{(m)}(\eta_n)}$. Thus, we get $|e_{n+1}| = |e_n| \left| 1 - \dfrac{f^{(m)}(\zeta_n)}{m f^{(m)}(\eta_n)} \right| \leq K|e_n|$, where $K = \sup \left| 1 - \dfrac{f^{(m)}(\zeta_n)}{m f^{(m)}(\eta_n)} \right|$, provided that $f^{(m)}(\eta_n)$ is non-zero in a suitable neighbourhood. Thus, linear order of convergence is established when the multiplicity is more than 1.

In case a root α of $f(x)$ has multiplicity $m > 1$, we can modify the method by taking $\phi(x) = \dfrac{f(x)}{f'(x)}$ and applying the ordinary Newton's method on ϕ to get a quadratic order

of convergence while computing the same root α of f. It can be shown as follows.
We can write $f(x) = (x-\alpha)^m g(x)$ and by assumption $g(\alpha) \neq 0$, otherwise α would have a greater multiplicity than m. Differentiating both sides w.r.t x, we obtain $f'(x) = (x-\alpha)^m g'(x) + m(x-\alpha)^{m-1}g(x)$.
Now, $\phi(x) = \dfrac{(x-\alpha)^m g(x)}{(x-\alpha)^m g'(x) + m(x-\alpha)^{m-1}g(x)} = \dfrac{(x-\alpha)g(x)}{(x-\alpha)g'(x) + mg(x)}$.
Therefore, $\phi(\alpha) = 0$ with the additional property that α is a simple root of $\phi(x)$, since $\phi(x) = 0 \implies (x-\alpha)g(x) = 0$; but $g(\alpha) \neq 0$. Therefore, applying Newton's method on ϕ would give us a quadratic order of convergence, while approximating the same root α.

b)
Order of Convergence of the Secant Method:

The iteration scheme for the secant method is given by:
$x_{n+2} = \dfrac{f_{n+1}x_n - f_n x_{n+1}}{f_{n+1} - f_n}$, where $f_i = f(x_i)$.
Writing $e_i = x_i - \alpha$, where α is the root of $f(x)$ we intend to find, we can write $f(x_i) = f(e_i + \alpha) = f(\alpha) + e_i f'(\zeta_i) = e_i f'(\zeta_i)$, where $\zeta_i \in \mathbb{R}$, using Taylor series expansion with Lagrange's form of remainder, along with α being a root of f.
Thus, we can write the formula for iteration as:

$$e_{n+2} + \alpha = \frac{(e_n+\alpha)f_{n+1} - (e_{n+1}+\alpha)f_n}{f_{n+1} - f_n} = \frac{\alpha(f_{n+1} - f_n) + (e_n f_{n+1} - e_{n+1} f_n)}{f_{n+1} - f_n}$$

$$\implies e_{n+2} = \frac{e_n e_{n+1} f'(\zeta_{n+1}) - e_{n+1} e_n f'(\zeta_n)}{f_{n+1} - f_n}$$

Thus, we get $|e_{n+2}| \leq |e_n||e_{n+1}|K$, where $K = \sup \dfrac{f'(\zeta_{n+1}) - f'(\zeta_n)}{f_{n+1} - f_n} < \infty$ in some neighbourhood of α. (Provided that the denominator does not become zero at some pair $\{x_i, x_{i+1}\}$).
Now, letting $|e_{i+1}| \propto |e_i|^p$, we can see that $|e_{i+2}| \propto |e_{i+1}|^p \propto (|e_i|^p)^p = |e_i|^{p^2}$.
Therefore, we end up with $|e_i|^{p^2} \propto |e_i||e_i|^p = |e_i|^{p+1}$, which can hold only when $p^2 = p + 1 \implies p^2 - p - 1 = 0$.
The roots of the above quadratic are $\dfrac{1 \pm \sqrt{5}}{2}$. Since the order of convergence being negative is absurd, we choose root $\dfrac{1+\sqrt{5}}{2} \approx 1.618$, which is the Golden Ratio.

Algorithms for Secant and Müller's Method:
Secant method: Take a starting interval containing the root $[x_0, x_1]$.
1. Check if $f(x_{n+1}) = f(x_n)$. If yes, Secant method has failed, else proceed to the next step.
2. Set $x_{n+2} = x_{n+1} - f(x_{n+1})\dfrac{x_{n+1} - x_n}{f(x_{n+1}) - f(x_n)}$
3. If $f(x_{n+2}) = 0$, we've found the exact root, else continue the loop until approximation is within tolerance.
Relative tolerance: $\dfrac{|x_{n+1} - x_n|}{|x_{n+1}|}$, Prescribed error: 10^{-6}.

Müller's Method: Start with the triplet of initial approximations (x_0, x_1, x_2)

1) Set $a = \dfrac{(x_1 - x_2)[f(x_0) - f(x_2)] - (x_0 - x_2)[f(x_1) - f(x_2)]}{(x_0 - x_2)(x_1 - x_2)(x_0 - x_1)}$

$b = \dfrac{(x_0 - x_2)^2[f(x_1) - f(x_2)] - (x_1 - x_2)^2[f(x_0) - f(x_2)]}{(x_0 - x_2)(x_1 - x_2)(x_0 - x_1)}$

$c = f(x_2)$

2) Check if $b^2 - 4ac \geq 0$. If yes, proceed to the next step, else stop. (Since we're interested in real roots).

3) $x_3 = x_2 - \dfrac{2c}{b + \text{sgn}(b)\sqrt{b^2 - 4ac}}$ where $\text{sgn}(b) = 1$ or -1 according as $b > 0$ or $b < 0$. If $b = 0$, $\text{sgn}(b) = 0$.

4) Check if $f(x_3) = 0$. If yes, we've found the exact root, else replace (x_0, x_1, x_2) with (x_1, x_2, x_3) and continue until tolerance is reached.

Relative tolerance $\dfrac{|x_{n+2} - x_{n+1}|}{|x_{n+2}|}$ for the triplet (x_n, x_{n+1}, x_{n+2}). Prescribed error: 10^{-6}.

Computational Outcome:

Secant Method: 0.517757 Iteration Count: 7

Müller's Method: 0.517757 Iteration Count: 4

(Both rounded off to 6 decimal places.)

Question 3 :

Analyzing the Nature of $f(x) = x^2 + \cos(30e \cdot x)$:

First, we discuss upon the periodicity of the function $\cos(k \cdot x)$, where $x \in \mathbb{R}$ is the variable and k is any positive constant.
The consine function is periodic with a period of 2π (in real numbers), therefore, $\cos(k \cdot x) = \cos(k \cdot x + 2\pi) = \cos(k \cdot (x + T))$, where T is such that $k \cdot (x + T) = k \cdot x + 2\pi$ (i.e, the minimum step required so that cosine repeats the values, which is the period of the function $\cos(k \cdot x)$.)
So, we get $T = \dfrac{2\pi}{k}$. Evidently, the period is inversely proportional to the positive constant.

Now, we show that no zero of $f(x)$ can lie beyond 1. (since we're concerned only with the positive roots)
For $f(x) = x^2+\cos(30e\cdot x) = 0$, we have $-x^2 = \cos(30e\cdot x)$. We know that $|\cos(z)| \leq 1$, for any $z \in \mathbb{R}$. Since, for any $x > 1$, we have $|-x^2| > 1$, therefore, $|-x^2| = |\cos(30e \cdot x)| > 1$ is an impossibility.
So, all the positive roots of $f(x)$ lie in $(0, 1]$.
Direct computation shows that $f(0) = 1 \neq 0$, thus if we search for roots of f in the interval $[0, 1]$, the extra 0 will not interfere in any way with our search.

The polynomial component of $f(x)$, i.e, x^2 ($= g(x)$, say) is strictly monotone increasing for $x > 0$, since $g'(x) = 2x > 0$ for all $x > 0$; and the only zero $g(x)$ attains is at 0.
Thus, it's evident that the contributing factor to the sign change of $f(x)$ in $[0, 1]$ is purely due to the oscillatory nature of $\cos(30e \cdot x)$.

How farther apart are the roots?

From the above discussion, it's clear that the periodicity of the cosine function comes into play. Within $[0, 2\pi]$, $\cos(z)$ attains two zeros, one at $\dfrac{\pi}{2}$ and the other at $\dfrac{3\pi}{2}$, which are respectively in the interval $[0, \pi]$ and in $[\pi, 2\pi]$. Now that we have "scaled" the variable by a $+ve$ factor k (meaning kz), so it would still attain two zeros, however, they will lie in the interval $[0, T]$, where $T = \dfrac{2\pi}{k}$; with one of the roots in $\left[0, \dfrac{T}{2}\right]$ and the other in $\left[\dfrac{T}{2}, T\right]$. As apparent from this, we need to divide the entire $[0, 1]$ into steps of $\dfrac{T}{2}$ starting from 0, where $T = \dfrac{2\pi}{30e \cdot x}$, and look for any sign changes at those endpoints, because that's an upper bound on the step-length we need to cover all the roots, and still not miss out on any change in sign, given the strictly increasing and positive nature of x^2 and the periodicity and the magnitude of cosine component of f. This ensures that each of those small intervals of length $\frac{T}{2}$ contains at most one root of f.
Taking smaller steps than this will result in an unnecessary increase in computation time.

Algorithm:

a) Searching for intervals containing roots:

1. Break $[0, 1]$ into $\left[0, \frac{T}{2}\right]$, $\left[\frac{T}{2}, 2\frac{T}{2}\right]$, $\left[2\frac{T}{2}, 3\frac{T}{2}\right]$, ..., $\left[(N-1)\frac{T}{2}, 1\right]$

2. Start looking for sign changes at the endpoints of each interval. If the $i-$th interval shows a sign change, add $(i-1)\frac{T}{2}$ to a list A. If at any endpoint f is zero, add it to the list of roots B, and move on to the next interval.

3. After checking the last interval, move to step (b).

b) Selecting the endpoints:

1. Read an element t from A.
2. Select the endpoints as $\left[t, t + \frac{T}{2}\right]$
3. Go to step (c)

c) Computing roots the fastest way possible:

1) First apply the Newton Raphson Method with a random initial approximation in the selected subinterval, as it is the fastest method. If we reach the exact root/desired precision, add the outcome to the list of roots B and go to step (b) again.
If somehow $f(x_n) = 0$, or $x_n \notin \left[t, t + \frac{T}{2}\right]$, stop the method and move on to the Secant Method.

2) If the Newton's method has failed, apply the Muller's Method. If somehow the discriminant of the quadratic $b^2 - 4ac < 0$, or $x_n \notin \left[t, t + \frac{T}{2}\right]$, move to the Secant method. If the method succeeds, add the root to list B and go back to step (b).

3) Select the endpoints of the interval as the initial approximation. If iteration reaches desired precision/exact root, terminate the method and add the last approximation to B and go to step (b).
If however $f(x_n) - f(x_{n-1}) = 0$ (that is when a division by zero is about to occur) or $x_n \notin \left[t, t + \frac{T}{2}\right]$, we terminate this process and move on to the next best method, i.e. the Regula Falsi Method.

4) Select the endpoints as the initial approximation. If desired outcome is reached, append it to B and go to step (b).
Else, if $f(x_n) - f(x_{n-1}) = 0$, we stop the method and move to the Bisection Method as our last resort.

5)Select the endpoints as the starting interval, and keep applying the Bisection Method until it reaches the desired tolerance/root, afterwards add the root to the list B and go to step (b). This method is guaranteed to converge, given the continuity of f on the entire real line.
Repeat until the list of right-endpoints A is exhausted.

Choice of stopping criterion:

One more thing to note here is that, we prescribe the stopping criterion for each individual method in this context as $\left(\dfrac{|x_{n+1} - x_n|}{|x_{n+1}|} \leq \epsilon \text{ and } |f(x_{n+1})| \leq \epsilon\right)$, since the intervals are of small size and the relative tolerance might be small by default, even before the iteration starts. Here $\epsilon = 10^{-6}$.

Computational Outcome:

List of +ve Roots (rounded off to 6 decimal places):
[0.019267, 0.057745, 0.096425, 0.134613, 0.173729, 0.211335, 0.251182, 0.287914, 0.328784, 0.364348, 0.406541, 0.440633, 0.484458, 0.516762, 0.56255, 0.592724, 0.64084, 0.668493, 0.719368, 0.744029, 0.798218, 0.819249, 0.877575, 0.893966, 0.958104, 0.967514]

Count of +ve Roots:
26

Time Taken for Root Evaluation (including locating the intervals): ≈ **0.0007s** (average of 1000 iterations of the main computation loop). The code is given below the main code for this problem in the file.
When the midpoint is taken as the initial approximation for the Newton's Method, the time taken is: ≈ 0.0003s. (Given in the last section of the code file)

Computation times have been reported by running the codes through terminal in individual .py files, i.e "Q3_randomized.py" and "Q3_midpoint.py" without any background applications running. Speed might significantly improve on a better workstation.

Question 4 :

Algorithm for Lagrange's Interpolation:

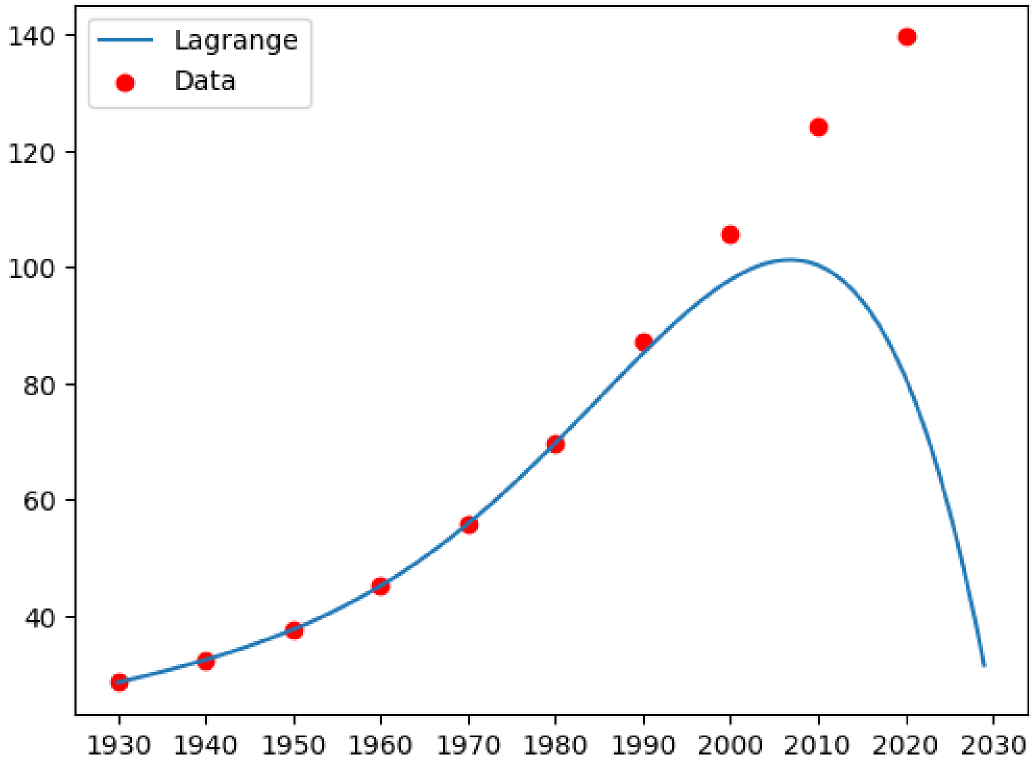
Let the Lagrange's interpolating polynomial be $P(x)$, which interpolates the dataset $\{(x_0, f(x_0)), (x_1, f(x_1)), ..., (x_n, f(x_n))\}$. To find the value of the polynomial at x, we follow the algorithm below:

1. Initialize $P(x) = 0$ and set a counter $i = 0$.
2. Compute the product $l_1(x) = \dfrac{(x - x_1)...(x - x_n)}{(x_0 - x_1)...(x_0 - x_n)} f(x_0)$
3. $P(x) = P(x) + l_1(x)$.
4. Increment i, and go to step 2, thus forming a loop, till $i = n$

For the general ith step, we compute $l_i(x) = \dfrac{(x - x_0)...(x - x_{i-1})(x - x_{i+1})...(x - x_n)}{(x_i - x_0)...(x_i - x_{i-1})(x_i - x_{i+1})...(x_i - x_n)} f(x_i)$ for step (2) and follow the loop till $i = n$.

Plot obtained for the 5th degree Lagrange's Polynomial:

The data points are from 1930 to 1980 for figuring out $P(x)$.



Denoting y_i as the true values and $\hat{y}_i$ as the corresponding estimates given by the model, we get: $(y_i, \hat{y}_i)$ as (87.1,85.1),(105.7,97.8),(124.1,100.3),(139.6,81) for $i = 1, 2, 3, 4$, respectively for the years $1990, 2000, 2010, 2020$.

MSE: 1016.7025, RMSE: 31.88577 for estimates. As we can see, the MSE is very high. However, we can conclude the ineffectiveness of our model in a more formal way.

Is this a good model to predict future population count?

R^2 Value of our Model: -1.624
Definition of R^2 value: `https://en.wikipedia.org/wiki/Coefficient_of_determination`

Where $R^2 = 1 - \frac{SS_{res}}{SS_{tot}}$
$SS_{res} = \sum_{i=1}^{4}(y_i - \hat{y}_i)^2$ and $SS_{tot} = \sum_{i=1}^{4}(y_i - \overline{y})$ with $\overline{y}$ being the mean of true values.

"A negative R-squared or negative adjusted R-squared value means that the reported predictive power of the model is less accurate that the average value of the data set over time. It also means that the model is predicting worse than the mean of the data set The implication is that the explanatory variables do not predict the specific human behaviour or social norm being estimated."
(The acceptable R-square in empirical modelling for social science research: Ozili, Peterson K, 2023) `https://mpra.ub.uni-muenchen.de/115769/1/MPRA_paper_115769.pdf`

Abiding by the convention, the answer is: **No, this is not a good predictor of future population of India.**