QUESTION 1

CALCULATION PROCESS

Calculating Yearly Averages: The first step involved calculating the yearly average temperatures by averaging daily temperatures for each year. This was important to minimise the computational time required for any numerical algorithm, while computing 50 years of daily temperature data.

Smoothing Data with Univariate Spline: To identify the underlying trends and obtain a smooth temperature curve, cubic spline smoothing was used. This method approximates a curve that best fits the yearly average temperatures while maintaining a certain level of smoothness.

Derivative Estimation: The derivative of the spline provides the rate of change of temperature over time. This rate of change is essential for analysing how quickly temperatures are rising or falling each year.

Forecasting Future Temperatures: To forecast temperatures, the time series was extended for the next 10 years, and the spline was used to predict these future years.

NUMERICAL ALGORITHMS USED

The numerical algorithms and techniques used in the temperature trend analysis include spline interpolation, derivative calculation, and forecasting. Spline interpolation was employed to smooth the yearly average temperature data, derivative calculation to estimate the rate of temperature change over the years and forecasting to predict future temperature trends.

It is important to note that the scipy.interpolate library was included in the initial code, which was to be expanded. This guided and significantly influenced the decision making, in regard to the algorithms and functions that were used.

SPLINE INTERPOLATION

To achieve the interpolation, Cubic Spline was chosen over alternative methods such as Lagrange interpolation, Newton interpolation, and piecewise linear interpolation. This is due to its ability to provide a smooth and continuous curve that accurately represents the underlying temperature trends.

While Lagrange and Newton interpolation methods may produce accurate interpolants, they often result in oscillations and overshooting, particularly when dealing with noisy or irregularly spaced data. If oscillations and overshooting occurred in the temperature trend analysis, it would likely lead to inaccuracies and distortions in the interpretation of the data. Oscillations refer to rapid and repetitive fluctuations in the interpolated curve, while overshooting occurs when the interpolated curve extends beyond the range of the actual data points. This can result from the interpolation method trying to fit the data too closely, particularly in regions where the data is sparse or noisy.

Piecewise linear interpolation, on the other hand, may not capture the subtle variations in temperature trends as effectively as spline interpolation. By using spline interpolation, the analysis can better capture both local and global temperature trends while minimising unwanted fluctuations and ensuring a more reliable representation of the data.

DERIVATIVE CALCULATION

For derivative calculation, the derivative() method of the UnivariateSpline function from scipy.interpolate was used. This method computes the derivative of the spline function, allowing for accurate estimation of the rate of change of temperature over time. For forecasting, spline interpolation was used to extrapolate future temperature trends based on historical data, providing insights into potential future temperature patterns.

FORECASTING

The spline method was selected for forecasting due to its ability to capture both local and global trends in the data while providing smooth extrapolations beyond the observed data range. This method offers flexibility and accuracy in predicting future temperature trends based on historical data.

UNIVARIATESPLINE() FUNCTION

The scipy.interpolate module was crucial for spline interpolation in the temperature trend analysis. Specifically, the UnivariateSpline function facilitated the smoothing of yearly average temperature data, offering a versatile and efficient approach for fitting spline curves.

CUSTOMISATION THROUGH PARAMETERS

The built-in functions, including UnivariateSpline, were parameterised to enhance control over the spline fitting process. Parameters such as the spline degree (k) and smoothing factor (s) were utilised for customisation and fine-tuning of the analysis.

Implications of Varying k Values: The choice of a higher or lower k value influences the flexibility and smoothness of the spline curve. Higher k values increase flexibility, enabling the capture of intricate patterns but raising the risk of overfitting and computational complexity. Conversely, lower k values result in smoother curves, albeit with potential oversimplification of the underlying trend.

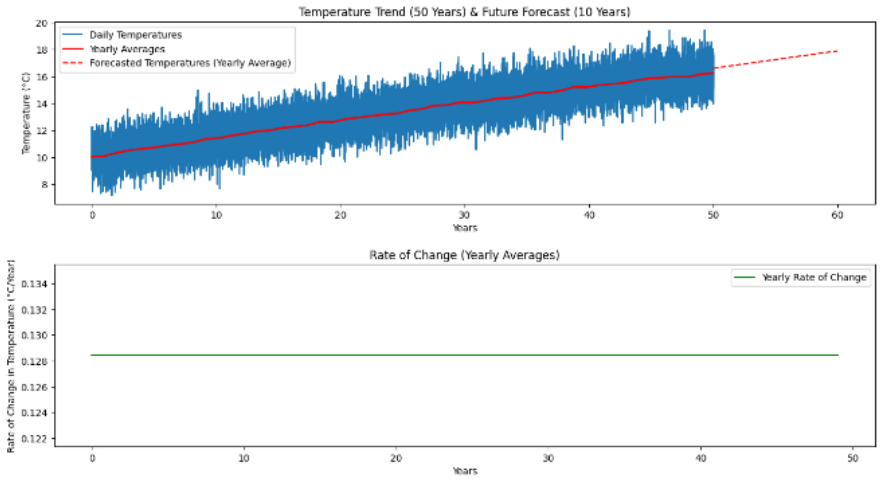
Rationale for k=3: The decision to set k=3 for the UnivariateSpline parameter was based on leveraging cubic splines (k=3 is the default for the function and is the documented setting for a cubic spline). Cubic splines strike a balance between smoothness and flexibility, making them well-suited for capturing both local and global temperature trends while mitigating issues such as oscillations and overfitting.

Implications of a Varying Smoothing Factor: The smoothing factor significantly impacts the smoothness and sensitivity of the spline curve. A higher smoothing factor yields a smoother curve by averaging out fluctuations and noise, albeit with the risk of over smoothing. Conversely, a lower smoothing factor preserves more detail in the data but may amplify sensitivity to noise and fluctuations, resulting in a less smoothed curve.

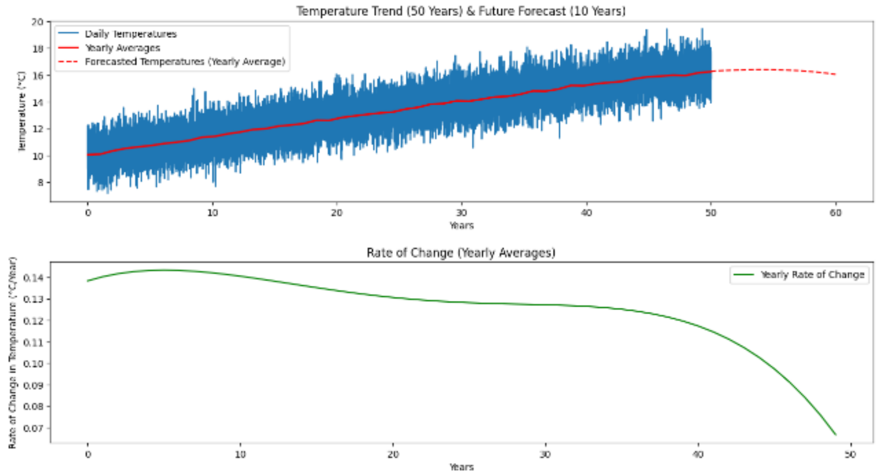
Rationale for s=1: A smoothing factor of s=1 in spline interpolation strikes a balance between noise reduction and preserving important data features. It minimises overfitting by removing high-frequency fluctuations while retaining essential trends. This choice ensures sensitivity to data variations without being overly influenced by outliers. With s=1, the spline maintains flexibility to adapt to the underlying data structure, making it suitable for capturing both local and global trends effectively. Overall, s=1 is considered a good smoothing factor, offering a reliable and interpretable spline fit for temperature trend analysis.

VARIATION RESULTS

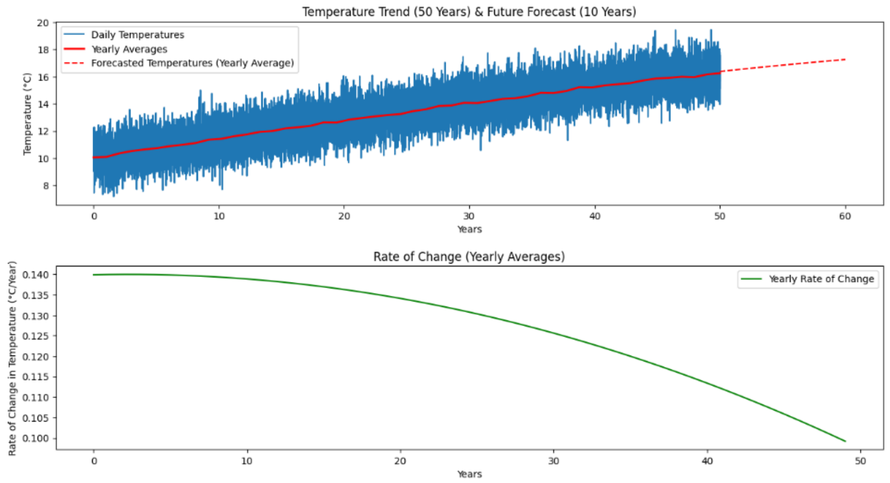
K=1



K = 5

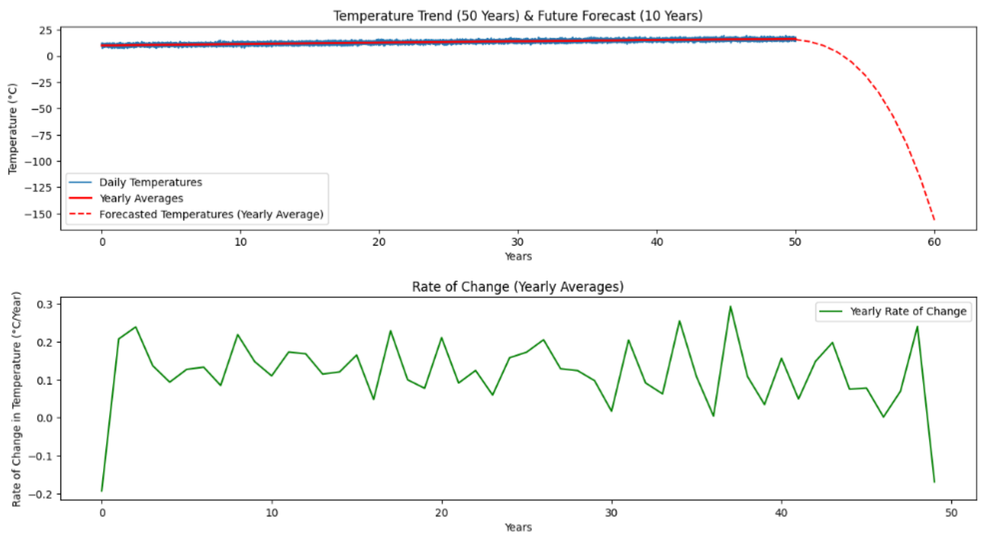


K=3 (OPTIMAL)

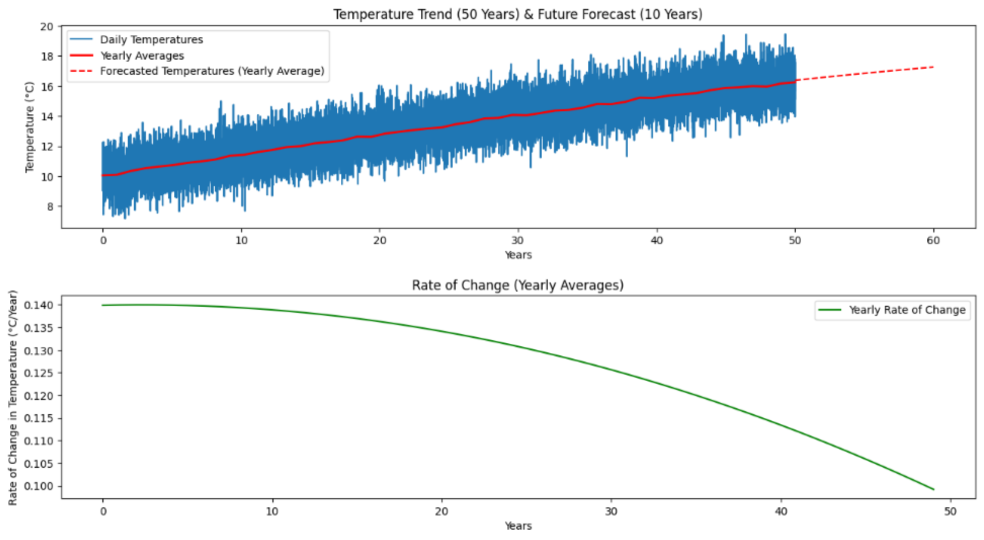


VARIATION RESULTS

S=0



S=1 (OPTIMAL)



DERIVATIVE() METHOD

The derivative() method was chosen for its simplicity and accuracy in computing the derivative of the spline function. This method seamlessly integrates with the spline fitting process and provides a straightforward way to estimate the rate of temperature change over the years.

RESULTS & ANALYSIS

The chosen algorithms demonstrate robustness and reliability, facilitated by parameter tuning and reliance on established libraries. Utilising well-tested numerical techniques from libraries like NumPy and SciPy ensures the stability of spline interpolation, derivative calculation, and forecasting processes.

Through the adjustment of parameters such as spline degree and smoothing factor, the algorithms are tailored to effectively capture temperature trends while mitigating noise and overfitting. Consequently, these algorithmic selections underpin a robust and dependable temperature trend assessment. As such, the following assumptions can be made:

Temperature Trend Over 50 Years: The plot of daily temperatures and yearly averages illustrates a steady increase in temperature over time.

Future Temperature Predictions: The forecasted temperatures show a continued gradual increase in temperatures for the next 10 years. This implies that the warming trend is expected to continue.

MORE CONTENT BELOW

QUESTION 2

CALCULATION PROCESS

Simulating Pollution Data: The process begins by simulating pollution data for each day of the year using mathematical models. This involves incorporating base pollution levels, seasonal variation, and random noise into the simulation process.

Polynomial Regression: Next, polynomial regression is applied to fit a polynomial curve to the simulated pollution data. This step determines the polynomial function that best represents the relationship between time and pollution levels, enabling approximation of seasonal trends.

Numerical Integration: The fitted polynomial curve is then integrated over time using the trapezoidal rule. This numerical integration calculates the total pollution exposure throughout the year by estimating the area under the curve.

NUMERICAL ALGORITHMS USED

The air pollution analysis employed Polynomial Regression for curve fitting and Numerical Integration. This was achieved using the trapezoidal rule for estimating total pollution exposure. Polynomial Regression facilitated the fitting of a curve to the pollution data, enabling the approximation of seasonal trends. Meanwhile, Numerical Integration, specifically utilising the trapezoidal rule, calculated the total pollution exposure by integrating the fitted curve over time.

It is important to note that the numpy.polynomial.polynomial library was included in the initial code, which was to be expanded. This guided and significantly influenced the decision making, in regard to the algorithms and functions that were used.

POLYNOMIAL REGRESSION

Polynomial regression was selected as the preferred method, due to its adaptability in fitting curves of diverse shapes to the data. This makes the method highly suitable for approximating the relationship between time and pollution levels in the air pollution analysis.

Specifically, interpolation primarily aims to estimate unknown data by directly connecting known data points. Whereas curve fitting focuses on determining the optimal curve that fits the data, prioritising the overall fit of the curve rather than ensuring it passes through every individual data point.

Both Lagrangian interpolation and Newton interpolation, while capable of producing accurate interpolants, often introduce oscillations and overshooting, especially when dealing with noisy or irregularly spaced data. Oscillations refer to rapid and repetitive fluctuations in the interpolated curve, while overshooting occurs when the interpolated curve extends beyond the range of the actual data points. This can result from the interpolation method trying to fit the data too closely, particularly in regions where the data is sparse or noisy. These issues can lead to inaccuracies in data interpretation, potentially affecting the reliability of the analysis.

Another option, spline interpolation, while effective in generating smooth and continuous curves, may not capture underlying trends as accurately as polynomial regression. Although spline interpolation offers advantages in certain scenarios, such as minimising unwanted fluctuations, its limitations in accurately representing global trends make it less suitable for the air pollution analysis conducted in this study.

NUMERICAL INTEGRATION

The trapezoidal rule was chosen for numerical integration due to its simplicity, efficiency, and effectiveness in handling irregularly spaced data. While other methods such as Simpson's Rule and Gaussian Quadrature may offer higher accuracy for smoothly varying functions, they require more computational resources and may be less suitable for irregularly spaced or noisy data encountered in air pollution analysis.

The trapezoidal rule provides a good balance between accuracy and computational efficiency, making it well-suited for estimating the total pollution exposure by integrating the fitted curve over time in this analysis. Additionally, the trapezoidal rule is straightforward to implement and widely used in numerical analysis, further supporting its selection for this application.

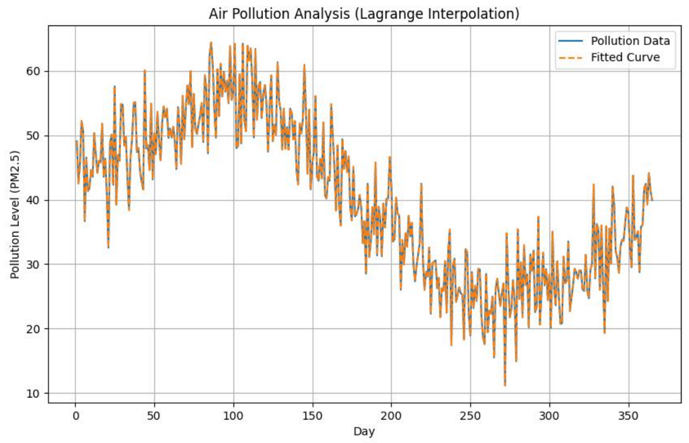
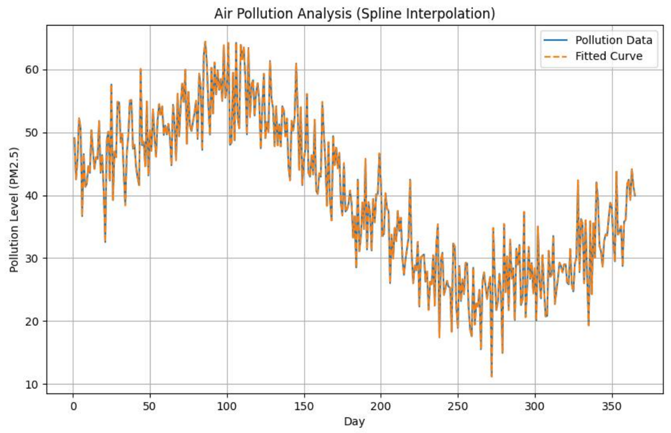
METHOD VARIATION

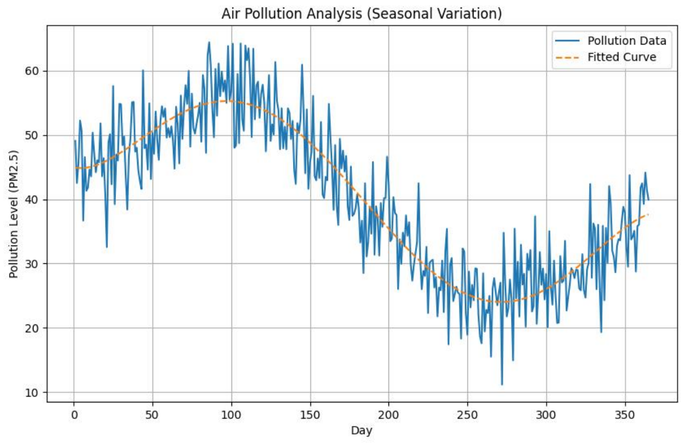
Although the end results for each method closely matched, notable differences emerged in their implementation. The interpolation methods exhibited tendencies towards overfitting, resulting in curves that closely followed the original data points, but potentially missed broader trends.

Additionally, the computational time for the Lagrange method proved significantly longer compared to spline interpolation and polynomial regression. Even though the interpolation methods returned identical results, the polynomial regression figure exhibited a slight variation. This may be due to a higher accuracy, which underscores the significance of selecting the most suitable method based on given circumstances.

MORE CONTENT BELOW

METHOD VARIATION RESULTS

Lagrange Interpolation (14532.715777795085)	Cubic Spline Interpolation (14532.715777795085)
	

Polynomial Regression (14535.954906023784)


RESULTS & ANALYSIS

The reliability and robustness of the program are underpinned by the numerical algorithms and methods employed, notably Polynomial Regression for curve fitting and Numerical Integration via the trapezoidal rule for estimating total pollution exposure. Polynomial Regression, known for its flexibility in capturing complex relationships between variables, ensures a robust approximation of seasonal trends in pollution data. Its adaptability to various data distributions enhances the program's reliability in accurately representing the underlying patterns.

Additionally, the trapezoidal rule, a widely used numerical integration technique, provides a reliable method for estimating total pollution exposure by effectively summing up areas under the fitted curve. Its simplicity and efficiency contribute to the program's robustness in handling large datasets and irregularly spaced data points commonly encountered in air pollution analysis. Together, these numerical algorithms and methods enhance the program's reliability and robustness, ensuring accurate assessments of pollution levels and facilitating informed decision-making processes, as discussed below:

Total Pollution Exposure: The calculated total pollution exposure provides insight into the cumulative impact of pollution over the year, indicating the overall severity of pollution levels. This metric is crucial for assessing potential health implications associated with prolonged exposure to pollutants, as higher exposure levels may increase the risk of respiratory and cardiovascular diseases among the population.

Seasonal Trends: Analysis of the seasonal trends in pollution levels reveal fluctuations across different seasons, with variations influenced by environmental factors such as weather patterns and human activities. Identifying these trends allows for the development of season-specific strategies to address peak pollution periods and minimise health impacts. By understanding these seasonal trends, stakeholders can implement targeted measures to improve air quality and protect public health throughout the year.

MORE CONTENT BELOW

QUESTION 3

CALCULATION PROCESS

Calculate Position Observations: Initially, a set of time-stamped position observations of a satellite over one orbit was generated. This involved defining the number of observation points and the period of the orbit in hours. A circular orbit was simulated for simplicity, with positions calculated based on the given radius and angles computed from the orbit period. To simulate real observations, noise was added to the generated positions. The noise was generated with a mean of zero and a standard deviation of 100 km, representing typical observational uncertainties.

Cubic Spline Interpolation: Cubic spline interpolation was utilised to estimate the satellite's position at times where direct observations were not available. This interpolation method provided a smooth estimate of the satellite's trajectory between the observed points.

Fuel Usage Calculation: A function was defined to calculate the hypothetical fuel usage based on trajectory adjustments. This function quantified the total fuel required by summing the absolute differences between the adjusted positions and the observed positions.

Optimisation: The trajectory was optimised to minimise fuel usage. The Quasi-Newton – BFGS optimisation method was employed to minimise the fuel usage function with respect to the adjusted points. The initial guess for optimisation was set as the time stamps of observations. The optimised points were then used to calculate the optimised positions of the satellite.

NUMERICAL ALGORITHMS USED

In the pursuit of estimating a satellite's trajectory and optimising it for fuel efficiency, key numerical algorithms were employed. These included cubic spline interpolation for trajectory estimation and the Quasi-Newton Method for optimisation. Through these methodologies, the trajectory was iteratively refined to meet both fuel efficiency and orbit accuracy goals.

It is important to note that the scipy.interpolate and scipy.optimize libraries were included in the initial code, which was to be expanded. This guided and significantly influenced the decision making, in regard to the algorithms and functions that were used.

CUBIC SPLINE INTERPOLATION

In selecting cubic spline interpolation over alternative methods like Lagrange Interpolation, Newton Interpolation, and Piecewise Linear Interpolation, several considerations were paramount. Cubic spline interpolation was favoured for its ability to provide a smooth trajectory estimate between observed points, ensuring a continuous and natural representation of the satellite's motion. This method offers higher accuracy compared to linear interpolation approaches, capturing complex variations in the data more effectively.

Moreover, its numerical stability ensures that small changes in input data yield proportional changes in the interpolated trajectory, enhancing reliability. Importantly, cubic spline interpolation mitigates issues such as Runge's phenomenon commonly encountered with higher-degree polynomial interpolations, thus offering a more stable and robust solution for trajectory estimation in this context.

Lagrange Interpolation, while effective for interpolating data points, can suffer from Runge's phenomenon, resulting in oscillations near the edges of the interpolation interval. This instability could lead to inaccuracies and inconsistencies in estimating the satellite's trajectory, particularly in scenarios with noisy or sparse data.

Newton Interpolation, although similar to Lagrange Interpolation in its ability to interpolate data points, also shares the risk of Runge's phenomenon. Additionally, its reliance on divided differences for coefficient calculation may introduce numerical instability, especially when dealing with large datasets or high-degree interpolating polynomials.

Piecewise Linear Interpolation offers simplicity and computational efficiency but lacks the ability to capture the subtle variations and curvature present in the satellite's trajectory. This limitation could lead to a less accurate representation of the satellite's motion, particularly in cases where smoother trajectories are desired for accurate estimation.

QUASI-NEWTON METHOD – BFGS

The Quasi-Newton Method – BFGS was selected for its efficiency and robustness in optimising the trajectory of the satellite. Given the complexity of the optimisation problem and the need to minimise fuel usage while maintaining orbit accuracy, BFGS offers computational efficiency, allowing for timely adjustments to be made to the trajectory. Its robustness ensures that it can handle various optimisation landscapes effectively, providing reliable solutions even in scenarios where the optimisation function may be complex or poorly understood.

BFGS also offers versatility and rapid convergence, making it well-suited for the optimisation task at hand. Additionally, BFGS's memory efficiency is advantageous for large-scale optimisation problems, allowing for the optimisation process to be carried out without excessive memory requirements. Overall, these qualities make the Quasi-Newton Method – BFGS a suitable and effective choice for optimising the satellite's trajectory to achieve fuel efficiency and orbit accuracy goals.

Gradient Descent and Steepest Descent methods are inferior to more advanced optimisation techniques due to their reliance solely on gradient information, lacking the exploitation of curvature information provided by the Hessian matrix. While they are simple to implement and memory-efficient, their convergence can be slow, especially in the presence of narrow valleys or plateaus in the optimisation landscape.

Newton's Method, while utilising both gradient and Hessian information, can also be inferior in certain contexts due to computational overhead and numerical stability concerns. Despite its potential for rapid convergence, particularly for well-conditioned problems, its performance may degrade when faced with ill-conditioned or poorly scaled optimisation landscapes. Calculating and inverting the Hessian matrix can be computationally expensive, especially in high-dimensional optimisation problems, where the memory requirements and computational complexity can become prohibitive.

RESULTS & ANALYSIS

The numerical algorithms employed in the optimisation process play a crucial role in ensuring the robustness and reliability of the trajectory adjustments. The Cubic Spline interpolation method effectively handles noisy observational data, providing smooth estimates of the satellite's position even in the absence of direct observations. This robust interpolation technique contributes to the stability of the optimisation results by mitigating the impact of data fluctuations.

Moreover, the Quasi-Newton optimisation method, specifically the BFGS algorithm, demonstrates robustness by efficiently navigating the optimisation landscape. BFGS adapts well to different scenarios, effectively minimising fuel usage while maintaining orbit accuracy. Its reliable convergence properties ensure consistent performance across multiple optimisation runs, instilling confidence in the optimised trajectory's suitability for real-world satellite navigation applications. Overall, these numerical algorithms exhibit both robustness and reliability, ensuring the effectiveness and trustworthiness of the optimisation process.

MORE CONTENT BELOW

QUESTION 4

CALCULATION PROCESS

Modelling GDP Growth: The initial step involved formulating a mathematical model to represent GDP growth over time. This model, based on the given equation, relates the GDP growth rate to the current GDP level and R&D investment as a percentage of GDP. By defining this relationship, the dynamics of GDP evolution could be simulated over the 30-year period.

Numerical Integration: To compute the GDP values over time, a numerical integration technique, specifically the Runge-Kutta 4th order method, was employed. This method breaks down the calculation into smaller time steps, iteratively estimating the GDP for each year based on the previous values. By systematically integrating the GDP growth equation, accurate GDP values were obtained for the entire duration of the analysis.

Total GDP Growth Estimation: Once the GDP values for each year were determined, the total GDP growth over the 30-year period was estimated by subtracting the initial GDP value from the final GDP value. This provided insight into the overall economic expansion experienced throughout the analysed period, quantifying the cumulative growth achieved.

NUMERICAL ALGORITHMS USED

The exploration of GDP growth dynamics over a 30-year horizon necessitates the employment of robust numerical techniques capable of accurately modelling complex economic interactions. In this specific problem, the Runge-Kutta 4th Order method was chosen.

RUNGE-KUTTA 4TH ORDER

The Runge-Kutta 4th Order (RK4) method was selected for its superior accuracy and stability, especially in managing complex differential equations like those governing economic dynamics. Unlike simpler methods such as Euler's Method and Heun's Method, RK4 employs multiple intermediate steps within each time interval, resulting in more precise approximations.

Its stability ensures reliable solutions, particularly in handling stiff differential equations prone to rapid or erratic behaviour. This safeguards against solution divergence and ensures the integrity of GDP calculations over the 30-year period. In addition, RK4 strikes a balance between accuracy and computational efficiency, making it ideal for tasks where precision and speed are essential. Examples may include economic modelling, and providing timely, reliable insights crucial for decision-making.

Euler's Method, on the other hand, while simple to implement, is often deemed inferior to RK4 due to its lower accuracy and stability. Euler's Method relies on linear approximations within each time step, leading to significant errors, especially when dealing with nonlinear or rapidly changing systems. This lack of accuracy can result in unreliable predictions and inaccurate representations of GDP growth dynamics over the 30-year period. Additionally, Euler's Method tends to exhibit poor stability, particularly in the presence of stiff differential equations.

Heun's Method, while slightly more accurate than Euler's Method, still falls short of RK4 in terms of accuracy and stability. Heun's Method improves upon Euler's approach by incorporating a predictorcorrector scheme, where an initial estimate is refined using a corrected estimate based on a weighted average of two slopes. However, this method still suffers from limitations in accuracy, especially when compared to RK4.

METHOD VARIATION

The estimated GDP growth over the 30-year period obtained using Euler's method, Heun's method, and RK4 differs slightly, with values listed below. This slight variation in results is primarily attributed to differences in the numerical integration techniques employed by each method.

Euler's method, while straightforward, tends to produce less accurate results due to its reliance on linear approximations within each time step, resulting in larger truncation errors. Heun's method, an improvement over Euler's method, incorporates a correction step to mitigate these errors, leading to slightly improved accuracy. On the other hand, RK4 offers superior accuracy and stability compared to Euler's and Heun's methods, thanks to its consideration of multiple intermediate steps within each time interval.

This enables RK4 to provide more precise approximations of the solution, achieving results that more closely align with the true values. Thus, while all three methods yield similar estimates of GDP growth, RK4's superior performance in accuracy and stability ultimately leads to the closest approximation of the true GDP growth over the period.

Estimated GDP Growth (Euler's Method): 61.771654032892

Estimated GDP Growth (Heun's Method): 61.83638040491428

Estimated GDP Growth (RK4): 61.836510538779294

RESULTS & ANALYSIS

The Runge-Kutta 4th Order (RK4) method stands out for its robustness and reliability in numerical integration tasks, particularly evident in modelling GDP growth dynamics over a 30-year period. Its capacity to manage stiff differential equations ensures stability, even amidst rapid changes, or oscillatory behaviour which is inherent in economic systems. This stability lends credibility to the computed GDP values, providing a dependable basis for economic analysis.

Moreover, RK4's higher accuracy, compared to simpler methods, reduces truncation errors, yielding more precise estimates of GDP growth. This enhanced accuracy is vital for capturing subtle nuances in economic trends, empowering decision-makers with reliable insights into long-term economic trajectories. Additionally, RK4's efficiency, striking a balance between accuracy and computational complexity, ensures practical applicability in real-world scenarios.

Overall, RK4's robustness, reliability, and efficiency make it the preferred choice for numerical integration in economic modelling. Its ability to provide accurate and dependable estimates of GDP growth facilitates informed decision-making and enhances the credibility of economic analysis.