Cholesky

Klasa CholeskySolver służy do rozwiązywania układów równań liniowych poprzez dekompozycję Cholesky'ego.

1. Opis

Klasa ta implementuje algorytm dekompozycji Cholesky'ego oraz rozwiązania układu równań liniowych przy użyciu tej dekompozycji. Algorytm ten jest skuteczny w przypadku macierzy symetrycznych i dodatnio określany.

2. Atrybuty

- `matrix_size` (int): Rozmiar macierzy kwadratowej A.

- `A` (numpy.ndarray): Macierz kwadratowa A o wymiarach (matrix_size, matrix_size), generowana losowo i przekształcana na macierz dodatnio określoną.

- `L` (numpy.ndarray): Dolna trójkątna macierz Cholesky'ego o wymiarach (matrix_size, matrix_size).

3. Metody

1. `__init__(self, matrix_size)`: Inicjalizuje obiekt klasy CholeskySolver.

   - `matrix_size` (int): Rozmiar macierzy kwadratowej A.

2. `cholesky_decomposition(self)`: Dokonuje dekompozycji Cholesky'ego na macierzy A.

3. `forward_substitution(self, b)`: Rozwiązuje układ równań liniowych Ly = b.

   - `b` (numpy.ndarray): Wektor prawych stron układu równań.

   - Returns:

     numpy.ndarray: Wektor y.

4. `backward_substitution(self, y)`: Rozwiązuje układ równań liniowych L^T x = y.

   - `y` (numpy.ndarray): Wektor y z poprzedniego kroku.

   - Returns:

     numpy.ndarray: Wektor x - rozwiązanie układu równań.

5. `solve(self, b)`: Wywołuje sekwencyjnie metody dekompozycji Cholesky'ego, forward substitution

   i backward substitution, a następnie wyświetla wyniki.

   - `b` (numpy.ndarray): Wektor prawych stron układu równań.

Deviation

Klasa StandardDeviationCalculator służy do obliczania odchylenia standardowego dla podanych danych.

1. Opis

Klasa ta pozwala użytkownikowi wprowadzić dane, a następnie oblicza i wyświetla ich średnią arytmetyczną oraz odchylenie standardowe.

2. Atrybuty

- `data` (list): Lista zawierająca wprowadzone dane.

3. Metody

1. `__init__(self)`: Inicjalizuje obiekt klasy StandardDeviationCalculator.

2. `get_user_input(self)`: Pozwala użytkownikowi wprowadzić liczbę danych oraz same dane.

3. `calculate_mean(self)`: Oblicza średnią arytmetyczną dla wprowadzonych danych.

- Returns:

float: Średnia arytmetyczna danych.

4. `calculate_standard_deviation(self)`: Oblicza odchylenie standardowe dla wprowadzonych danych.

- Returns:

float: Odchylenie standardowe danych.

5. `calculate_and_display_steps(self)`: Wywołuje sekwencyjnie metody get_user_input, calculate_mean

i calculate_standard_deviation, a następnie wyświetla średnią arytmetyczną i odchylenie standardowe.

Detrivative

Klasa DerivativeCalculator jest przeznaczona do obliczania pochodnych funkcji matematycznych z wykorzystaniem modułu SymPy.

1. Opis

Klasa ta umożliwia użytkownikowi wprowadzenie funkcji matematycznej, obliczenie jej pochodnej względem zmiennej x oraz wyświetlenie zarówno funkcji, jak i jej pochodnej.

2. Atrybuty

- `x` (sympy.Symbol): Symbol zmiennej x.

- `function` (sympy.Expr): Obiekt reprezentujący funkcję matematyczną.

- `derivative` (sympy.Expr): Obiekt reprezentujący pochodną funkcji matematycznej.

3. Metody

1. `__init__(self)`: Inicjalizuje obiekt klasy DerivativeCalculator.

2. `enter_function(self)`: Pozwala użytkownikowi wprowadzić wzór funkcji matematycznej.

3. `calculate_derivative(self)`: Oblicza pochodną funkcji matematycznej.

4. `display_function(self)`: Wyświetla wprowadzoną funkcję matematyczną.

5. `display_derivative(self)`: Wyświetla obliczoną pochodną funkcji matematycznej.

6. `solve(self)`: Wywołuje sekwencyjnie metody enter_function, calculate_derivative i display_derivative, umożliwiając użytkownikowi pełne obliczenia.

Gauss

Klasa GaussianEliminationSolver służy do rozwiązywania układów równań liniowych metodą eliminacji Gaussa.

1. Opis

Klasa ta implementuje klasyczną metodę eliminacji Gaussa do rozwiązywania układów równań liniowych. Przyjmuje na wejściu rozszerzoną macierz (A|b) i przekształca ją do postaci schodkowej, a następnie wykonuje proces substytucji wstecznej w celu znalezienia rozwiązania układu równań.

2. Atrybuty

- `matrix_size` (int): Rozmiar macierzy kwadratowej A (liczba niewiadomych).

- `A` (numpy.ndarray): Rozszerzona macierz (A|b) układu równań.

- `x` (numpy.ndarray): Wektor rozwiązań układu równań.

3. Metody

1. `__init__(self, matrix_size)`: Inicjalizuje obiekt klasy GaussianEliminationSolver.

   - `matrix_size` (int): Rozmiar macierzy kwadratowej A (liczba niewiadomych).

2. `gaussian_elimination(self)`: Przeprowadza eliminację Gaussa na rozszerzonej macierzy (A|b).

3. `back_substitution(self)`: Wykonuje proces substitucji wstecznej w celu znalezienia rozwiązania układu równań.

4. `solve(self)`: Wywołuje sekwencyjnie metody gaussian_elimination i back_substitution, a następnie wyświetla wyniki.

Gauss-Jordan

Klasa GaussJordanSolver służy do rozwiązywania układów równań liniowych

metodą eliminacji Gaussa-Jordana.

1. Opis

Klasa ta implementuje metodę eliminacji Gaussa-Jordana do rozwiązania układu

równań liniowych. Przyjmuje na wejściu rozszerzoną macierz (A|b) i przekształca

ją do postaci kanonicznej, a następnie wyznacza wektor rozwiązań układu równań.

2. Atrybuty

- `matrix_size` (int): Rozmiar macierzy kwadratowej A (liczba niewiadomych).

- `A` (numpy.ndarray): Rozszerzona macierz (A|b) układu równań.

- `x` (numpy.ndarray): Wektor rozwiązań układu równań.

3. Metody

1. `__init__(self, matrix_size)`: Inicjalizuje obiekt klasy GaussJordanSolver.

- `matrix_size` (int): Rozmiar macierzy kwadratowej A (liczba niewiadomych).

2. `gauss_jordan_elimination(self)`: Przeprowadza eliminację Gaussa-Jordana na rozszerzonej macierzy (A|b).

3. `solve(self)`: Wywołuje metodę gauss_jordan_elimination, a następnie wyświetla wyniki.

Integral

Klasa Integral służy do obliczania wartości całki oznaczonej danej funkcji matematycznej

na zadanym przedziale.

1. Opis

Klasa ta umożliwia użytkownikowi wprowadzenie funkcji matematycznej, określenie granic

całkowania oraz obliczenie wartości całki oznaczonej na zadanym przedziale.

2. Atrybuty

- `expression` (sympy.Expr): Wyrażenie matematyczne reprezentujące funkcję do całkowania.

- `lower_limit` (float): Dolna granica całkowania.

- `upper_limit` (float): Górna granica całkowania.

3. Metody

1. `__init__(self)`: Inicjalizuje obiekt klasy Integral.

2. `get_user_input(self)`: Pozwala użytkownikowi wprowadzić funkcję matematyczną oraz granice całkowania.

3. `calculate_integral(self)`: Oblicza wartość całki oznaczonej na zadanym przedziale.

- Returns: sympy.Expr: Wynik całkowania.

4. `display_steps(self)`: Wyświetla kroki postępowania, tj. wprowadzoną funkcję oraz granice całkowania.

5. `solve_integral(self)`: Wywołuje sekwencyjnie metody get_user_input, display_steps i calculate_integral, a następnie wyświetla wynik całkowania.

Polynomials

Klasa PolynomialSolver służy do rozwiązywania wielomianów oraz generowania ich wykresów.

1. Opis

Klasa ta pozwala użytkownikowi wprowadzić stopień wielomianu oraz jego współczynniki,

a następnie używa metody Newtona-Raphsona do znajdowania pierwiastków wielomianu.

Dodatkowo umożliwia generowanie wykresu wielomianu.

2. Atrybuty

- `degree` (int): Stopień wielomianu.

- `coefficients` (list): Lista współczynników wielomianu od najwyższej potęgi do wyrazu wolnego.

3. Metody

1. `__init__(self)`: Inicjalizuje obiekt klasy PolynomialSolver.

2. `get_user_input(self)`: Pozwala użytkownikowi wprowadzić stopień wielomianu oraz jego współczynniki.

3. `evaluate_polynomial(self, x)`: Oblicza wartość wielomianu dla danej wartości x.

- Returns:

float: Wartość wielomianu.

4. `calculate_derivative(self)`: Oblicza współczynniki pochodnej wielomianu.

- Returns:

list: Lista współczynników pochodnej wielomianu.

5. `newton_raphson_method(self, initial_guess, tolerance=1e-6, max_iterations=100)`: Metoda Newtona-Raphsona

do znajdowania pierwiastków wielomianu.

- Parameters:

  - `initial_guess` (float): Przybliżona wartość początkowa dla jednego z pierwiastków.

  - `tolerance` (float, optional): Tolerancja błędu, domyślnie 1e-6.

  - `max_iterations` (int, optional): Maksymalna liczba iteracji, domyślnie 100.

- Returns:

  float: Znaleziony pierwiastek wielomianu.

6. `solve_polynomial(self)`: Rozwiązuje wielomian, znajdując wszystkie jego pierwiastki.

7. `evaluate_polynomial_derivative(self, x)`: Oblicza wartość pochodnej wielomianu dla danej wartości x.

- Returns:

  float: Wartość pochodnej wielomianu.

8. `plot_polynomial(self)`: Generuje wykres wielomianu wraz z oznaczeniem pierwiastków.

9. `solve_and_plot(self)`: Wywołuje sekwencyjnie metody solve_polynomial i plot_polynomial.

Quadratic function

Klasa QuadraticEquationSolver służy do rozwiązywania funkcji kwadratowej oraz generowania jej wykresu.

1. Opis

Klasa ta pozwala użytkownikowi wprowadzić współczynniki funkcji kwadratowej (a, b, c) oraz używa metod matematycznych do obliczenia miejsc zerowych (pierwiastków) i wierzchołka funkcji kwadratowej. Dodatkowo umożliwia generowanie wykresu funkcji kwadratowej.

2. Atrybuty

- `coefficients` (list): Lista zawierająca współczynniki funkcji kwadratowej a, b, c.

3. Metody

1. `__init__(self)`: Inicjalizuje obiekt klasy QuadraticEquationSolver.

2. `get_user_input(self)`: Pozwala użytkownikowi wprowadzić współczynniki funkcji kwadratowej.

3. `calculate_discriminant(self)`: Oblicza deltę (discriminantę) funkcji kwadratowej.

- Returns: float: Wartość delty.

4. `calculate_roots(self)`: Oblicza miejsca zerowe funkcji kwadratowej.

- Returns: tuple: Tuple zawierający pierwiastki funkcji kwadratowej.

5. `display_steps(self)`: Wyświetla kroki postępowania, tj. miejsca zerowe funkcji kwadratowej.

6. `plot_function(self)`: Generuje wykres funkcji kwadratowej.

7. `calculate_vertex(self)`: Oblicza współrzędne wierzchołka funkcji kwadratowej.

8. `solve_quadratic_equation(self)`: Wywołuje sekwencyjnie metody get_user_input, display_steps, calculate_vertex i plot_function, a następnie wyświetla wynik.

Main

Program zawiera zestaw klas i funkcji, które umożliwiają użytkownikowi skorzystanie z różnych metod obliczeniowych.

1. Klasy i metody

a. `GaussianEliminationSolver` (gauss.py)

- Klasa umożliwia rozwiązanie układu równań liniowych za pomocą metody eliminacji Gaussa.

- Metoda `solve` wykorzystuje tę klasę do rozwiązania układu równań.

b. `CholeskySolver` (cholesky.py)

- Klasa służy do rozwiązania układu równań liniowych za pomocą metody Cholesky'ego.

- Metoda `solve` przyjmuje wektor prawej strony układu równań i wykorzystuje klasę do rozwiązania układu.

c. `GaussJordanSolver` (gaussJordan.py)

- Klasa umożliwia rozwiązanie układu równań liniowych za pomocą metody Gaussa-Jordana.

- Metoda `solve` wykorzystuje tę klasę do rozwiązania układu równań.

d. `StandardDeviationCalculator` (deviation.py)

- Klasa służy do obliczania odchylenia standardowego na podstawie danych wprowadzonych przez użytkownika.

- Metoda `calculate_and_display_steps` wykorzystuje tę klasę do obliczeń i wyświetlenia wyników.

e. `QuadraticEquationSolver` (quadratic.py)

- Klasa umożliwia rozwiązanie funkcji kwadratowej, znajdowanie miejsc zerowych i generowanie wykresu.

- Metoda `solve_quadratic_equation` wykorzystuje tę klasę do rozwiązania funkcji kwadratowej.

f. `PolynomialSolver` (polynomials.py)

- Klasa obsługuje rozwiązanie wielomianu, znajdowanie miejsc zerowych i generowanie wykresu.

- Metoda `solve_and_plot` wykorzystuje tę klasę do rozwiązania wielomianu.

g. `DerivativeCalculator` (derivative.py)

- Klasa umożliwia obliczanie pochodnej funkcji wprowadzonej przez użytkownika.

- Metoda `solve` wykorzystuje tę klasę do obliczenia pochodnej.

h. `Integral` (integral.py)

- Klasa służy do obliczania całki oznaczonej funkcji na podstawie danych wprowadzonych przez użytkownika.

- Metoda `solve_integral` wykorzystuje tę klasę do obliczeń całkowania.

2. Funkcja `main`

- Funkcja główna programu, w której użytkownik wybiera metodę obliczeniową.

- Wykorzystuje obiekty klas do wykonania odpowiednich obliczeń w zależności od wyboru użytkownika.

3. Uruchomienie programu

- Po uruchomieniu programu, użytkownik wybiera numer metody obliczeniowej.

- Następnie wprowadza niezbędne dane (np. współczynniki macierzy, funkcji, itp.).

- Program wykonuje odpowiednie obliczenia i wyświetla wyniki.

4. Obsługa błędów

- Program obsługuje błędy, w przypadku nieprawidłowego wyboru metody obliczeniowej lub błędnych danych wejściowych.

- W przypadku błędu, program informuje użytkownika o problemie.