Programowanie i metody numeryczne

zestaw zadań 12

Równania różniczkowe: ciąg dalszy

Równania jednej zmiennej wyższych rzędów

Równania cząstkowe

Opracowanie *Jędrzej Wardyn*

27 maja 2024

Polecam: kody C++, jak ktoś potrzebuje sobie C++ powtórzyć: Rozwiązania na githubie   Kurs na youtubie

1 Równania jednej zmiennej

1.0.1 Metoda z uwikłaniem (implicit)

Mając inne proste równanie:

$$\frac{dy}{dx} = -2xy,$$

gdzie: $y_0 = 1.0$, $x_0 = 0.0$, $x_n = 2.0$.

napisz funkcję z templatką(możesz dodać do niej więcej argumentów funkcji):

```
template < typename T>
vector<T> METODA(T (*f) (T,T), T y0, T x0, T h, T xn)
```

gdzie `T` oznacza nazwę zastosowanego typu (na przykład `double`, tylko nie macie go wpisywać w templatkę, a jedynie w funkcji main używacie konkretnego typu na wartości np a, b i wtedy kod przyjmie te wartości jako `T`.)

Rozwiąż równanie używając metody Eulera "do tyłu"(backward Euler Method) Wymaga to rozwiązania równania uwikłanego. Zajrzyj na Wykład z ODE po wskazówki lub wiki. Wypisz wyniki do plików z rozszerzeniem `.csv` i wyrysuj błędy używając dostarczonego skryptu pythona.

1.1 Metoda Bogackiego-Shampine'a ze zmiennym krokiem

Dla równania:

$$dy/dx = y - x^2 + 1,$$

gdzie $y(x = 0) = 0.5$, a rozwiązanie analityczne to

$$y(x) = (x + 1)^2 - 0.5e^x$$

Zaimplementuj metodę ze zmiennym krokiem Bogackiego-Shampine'a.

Porównaj wynik analityczny z wynikiem numerycznym gdzie zastosujesz inne kroki początkowe

$$h = 0.1, 0.01, 0.001$$

.

1.2 Równania wyższego rzędu: oscylator harmoniczny

Mając dany klasyczny jednowymiarowy oscylator harmoniczny:

$$\frac{d^2x}{dt^2} + kx = 0$$

gdzie: $k = 1.0$, $m = 1.0$, $x_0 = 1$,$v_0 = 0$ oraz $t_{end} = 10.0$.
Rozwiąż równanie używając metod (sprawdź wykład strona 19 i 21):

- Eulera
- Leap-frog
- Velocity Verlet

oraz policz odpowiadające im lokalne błędy (LTE). Zmodyfikuj otrzymany skrypt pythona by wyrysować średni błąd w zależności od długości kroku h w danej metodzie.

jeszcze o leapfrog

2 Metoda różnic skończonych (Finite difference method) do równań różniczowych cząstkowych

2.1 Równanie falowe

Mając równanie falowe:

$$\frac{\partial^2 u}{\partial t^2} = c^2 \frac{\partial^2 u}{\partial x^2},$$

gdzie mamy skończoną przestrzeń długości $L = 2.0$ i skończonego czasu $T = 4.0$ Przyjmij prędkość fali jako $c = 1$, a kolejne kroki jako d$x = 0.1$, d$t = 0.05$.
Pomocny link: kliknij w źródło, opis jak to zrobić

W jednym wymiarze przestrzennym (jak i w trzech wymiarach przestrzennych) nasze rozwiązanie powinno z czasem zanikać do zera, niezależnie od przyjętego warunku początkowego Zasada Huygensa.

Zobacz w rozwiązaniu numerycznym, czy zasada działa.

2.2 bonus: [nie zadanie]

Dla ciekawskich: biblioteka do obliczania czasoprzestrzeni dla napędu typu Warp: możliwego do zbudowania, niepozwalającego przekroczyć prędkości światła

3 Rozkład LU

: Autor: Z.B.

Napisz szablon klasy

```
template <typename T>
class Vector
```

reprezentującej wektor n-wymiarowy, którego skladowe są liczbami reprezentowanymi przez typ T. W klasie tej zaimplementuj:

- konstruktor jednoargumentowy, ustalający wartość n równą liczbie przekazanej mu jako argument
- metodę Length zwracającą długość wektora (liczbę reprezentowaną przez typ T),
- operator== porównywania wektorów,
- jednoargumentowy operator zmiany znaku-,
- operator +dodawania wektorów,
- operator * mnozenia wektora przez liczbę (reprezentowaną przez typ T),
- operator* iloczynu skalarnego wektorów (wynik powinien być typu T)
- operator«, wypisujący do strumienia typu ostream wektor reprezentowany przez obiekt, oraz operator
- >> wczytujący odpowiedni wektor ze strumienia typu istream; przyjmij dowolny, wygodny dla Ciebie sposób tekstowego reprezentowania wektora,

Napisz szablon klasy

```
template <typename T>
class Matrix
```

reprezentującej macierz nxn, której składowe są liczbami reprezentowanymi przez typ T. W klasie tej zaimplementuj:

- konstruktor jednoargumentowy, ustalający wartość n równą liczbie przekazanej mu jako argument
- metodę Length zwracającą długość wektora (liczbę reprezentowaną przez typ T),
- operator `==` porównywania macierzy,
- jednoargumentowy operator zmiany znaku `-` ,
- operator `+` dodawania macierzy,
- operator `*` mnozenia macierzy przez liczbę (reprezentowaną przez typ T),
- operator `*` iloczynu skalarnego macierzy (wynik powinien być typu T)
- operator `<<`, wypisujący do strumienia typu ostream macierz reprezentowaną przez obiekt, oraz operator `>>` wczytujący odpowiedni wektor ze strumienia typu istream; przyjmij dowolny, wygodny dla Ciebie sposób tekstowego reprezentowania macierzy,

Metody:

1. metodę do LU decomposition

   ```
   pair<Matrix<T>, Matrix<T>> DecomposeLU(const Matrix<T> \&C)
   ```

   zwracającą parę macierzy złożoną z macierzy górnotrójkątnej i dolnotrójkątnej, których iloczyn jest równy macierzy przekazanej jako argument metody,

2. ```
   template <typename T>
   Vector<T> SolveU(const Matrix<T><<U,const Vector<T><<y)
   ```

   zwracającej wektor x stanowiący rozwiązanie równania $Ux = y$ przy założeniu, że macierz U jest dolnotrójkątna

3. ```
   template <typename T>
   Vector<T> SolveL(const Matrix<T> \&L, const Vector<T> \&y)
   ```

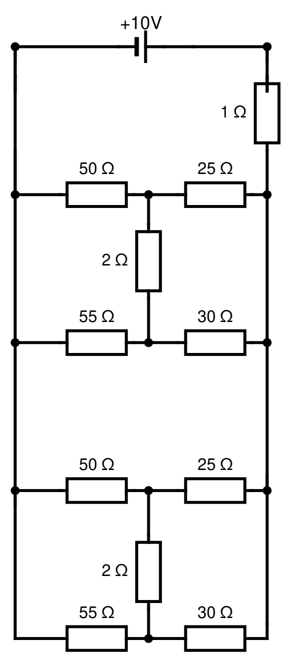
   zwracającej wektor x stanowiący rozwiązanie równania $Lx = y$ przy założeniu, ze macierz L jest górnotrójkątna,

4. ```
   template <typename T>
   Vector<T> Solve(const Matrix<T> &C, const Vector<T> &y)
   ```

   zwracającej wektor x stanowiący rozwiązanie równania $Cx = y$ dla dowolnej macierzy C; funkcja ta powinna wykorzystywać rozkład LU macierzy C.

Korzystając z tych szablonów, napisz program eqsolver, który wczytuje ze standardowego wejścia macierz C oraz wektor y, a następnie wypisuje na standardowe wyjście wektor x stanowiący rozwiązanie równania $Cx = y$.

3.1 Rozwiązywanie układu oporników, ale nie ręcznie: programem



Rysunek 1: Układ rezystorów, do którego podłączone jest stałe źródło napięcia +10V

Napisz program rozwiązujący i zastosuj go do powyższego układu rezystorów (patrz Rysunek 3.1).
   ```

Wystarczy, że zapiszesz 6 równań dla 6 oczek (pętli/loopów), które są widoczne [czyli nie trzeba równania dla kilku naraz]. Na przykład pierwsze oczko to:

$$10\text{V} = (1\Omega)I_1 + (25\Omega)(I_1 - I_2) + (50\Omega)(I_1 - I_3)$$

(Nie trzeba zapisywać równań dla węzłów, wystarczą oczka.) Mając zapisaną macierz rezystancji oraz wektor napięć dokonaj:

1. Rozkładu **LU** wykorzystując metodę Dolittle
2. Policz wyznacznik znając macierz **U**
3. Na macierzy **U** dokonaj eliminacji Gaussa i znajdź wektor natężeń prądu

Jak wykonać rozkład **LU** i eliminację gaussa? Linki:
LU metodą Dolittle
Eliminacja Gaussa