UNIVERZITA J. E. PURKYNĚ V ÚSTÍ NAD LABEM
Přírodovědecká fakulta

Matematický software

RNDr. Zbyšek Posel, Ph.D.

Ústí nad Labem 2020

Kurz: Matematický software

Obor: Aplikovaná informatika

Klíčová slova:

Anotace: Kurz Matematický software se zaměřuje na využití matematických modulů jazyka `Python` pro řešení úloh z matematického modelování, zejména úloh z numerické matematiky. Jde zejména o využití modulů `math`, `NumPy`, `SymPy`, `PyPlot`. Důraz je kladen na práci s jednoduchými poli a efektivní implementaci numerických metod. Kurz je určen pro studenty, kteří úspěšně prošli úvodním kurzem `Pythonu` (*Algoritmizace a programování I*) a základním kurzem matematiky (*Matematika I*).

Jazyková korektura nebyla provedena, za jazykovou stránku odpovídá autor.

Autor: RNDr. Zbyšek posel, Ph.D.

Obsah

Úvodní slovo

Opora k předmětu *Matematický software* je zaměřena zejména na využití matematicky orientovaných nástrojů jazyka `Python`, kterými jsou knihovny `NumPy`, `SciPy` a `SymPy`. Tyto knihovny jsou aktivně využívány v dalších předmětech, a proto je jejich dobrá znalost nutná. Dále je pozornost v tomto předmětu zaměřena na představení a využití základních metod numerické matematiky zejména v oblasti lineární algebry, diferenciálního a integrálního počtu funkce jedné proměnné. Jednotlivé úlohy jsou procvičovány formou psaní vlastních řešení za pomoci symbolické matematiky (`SymPy`) a numerické matematiky, a výstupy jsou porovnávány s vestavěnými funkcemi knihoven `NumPy` a `SciPy`. V první části opor je pozornost zaměřena na základní fakta o reprezentaci čísel v jazyku Python a na jejich vizualizaci pomocí základní knihovny matplotlib. Poté následuje jádro kurzu, kdy jsou postupně představeny a řešeny úlohy z lineární algebry, diferenciálního a integrálního počtu funkce jedné proměnné. Tyto úlohy jsou vhodně proloženy typickými úlohami numerické matematiky, jakými jsou interpolace a aproximace, hledání kořene rovnice $f(x) = 0$ a představením metody Monte Carlo, které předchází diskuze o náhodných číslech. Po každé ucelené partii přichází zadání dané části seminární práce, které shrnuje dosud nabyté znalosti. Tematicky je výklad zakončen obyčejnými diferenciálními rovnicemi a metodami jejich řešení. V průběhu kurzu budou zadány dvě seminární práce, jejichž možnou náplň lze nalézt v další kapitole. Všechny výše zmíněné kapitoly jsou doplněny o implementace v jazyku Python, které lze prostudovat po přihlášení do e-learningového systému MOODLE.

Příklad seminární práce

V průběhu kurzu budou zadány dvě seminární práce, na kterých budou moci studenti pracovat v průběhu speciálně vyčleněných seminářů. Součástí seminární práce je protokol o vypracování a programové kódy. Při plnění těchto prací mohou studenti spolupracovat na tvorbě programů, ale protokol odevzdává každý sám za sebe. Seminární práce může obsahovat například tato témata.

Laplaceův rozvoj pro výpočet determinantu matice

Dle Laplaceova rozvoje lze získat determinant matice $A(n, n)$ součtem determinantů n matic $A(n-1, n-1)$ jako

$$\begin{aligned} |A| &= \sum_{j=1}^{n} (-1)^{i+j} a_{ij} M_{ij} \\ M_{ij} &= \left|S_{ij}^{A}\right| \end{aligned}$$

kde, S_{ij}^{A} je submatice k matici A, která vznikne vynecháním $i-$tého řádku a $j-$tého sloupce matice A. M_{ij} je tzv. minor matice A a je roven hodnotě determinantu S_{ij}^{A}, a_{ij} je prvek matice A na $i-$tém řádku a v $j-$tém sloupci. Postup je platný i pro případ, kdy děláme rozvoj přes $i-$tý sloupec a $j-$tý řádek.

- Náhodně nagenerujte matici $A(n, n)$.
- Interval celých čísel zvolte v rozmezí $\langle -6, 6 \rangle$.
- Ověřte, že je matice čtvercová.
- Zvolte, zda uděláte rozvoj dle řádku nebo dle sloupce.

Výstupem bude:

- Graf časové náročnosti výpočtu Laplaceova rozvoje v závislosti na velikosti matice $n \in (5, 100)$.
- Porovnání časové náročnost Laplaceova rozvoje a vestavěné funkce `NumPy` nebo `SciPy` pro výpočet determinantu.

Numerická integrace

Vypočítejte následující integrál

$$\int_4^{20} \frac{\sin(x)}{x} + 3\mathrm{d}x$$

pomocí

- Symbolických funkcí v `SymPy`.
- Vlastním programem na složené Simpsonovo pravidlo v jazyku `Python`.
- Vlastním programem v jazyku `Python` na metodou Monte Carlo (viz rovnice (1)).

Výstupem bude:

- Hodnota integrálu pro jednotlivé metody.
- Graf konvergence složeného Simpsonova pravidla k přesné hodnotě integrálu v závislosti na dělení intervalu $\langle 0, 1\rangle$.
- Graf konvergence metody Monte Carlo v závislosti na počtu pokusů N.

Metoda Monte Carlo pro výpočet integrálu

Mějme funkci $f(x)$ na intervalu $\langle a, b\rangle$. Věta o střední hodnotě

$$I = \int_a^b f(x)\mathrm{d}x = (b-a)f(\xi)$$

nám říká, že na existuje bod $f\xi \in \langle a, b\rangle$ takový, že hodnota integrálu I je rovna obsahu obdélníka $(b-a)f(\xi)$.
Metodou Monte Carlo pro výpočet střední hodnoty se nyní budeme snažit aproximovat funkci $f(\xi)$ aritemtickým průměrem $\overline{f(x)}$. Takže

$$\int_a^b f(x)\mathrm{d}x = (b-a)\overline{f(x)} = \frac{(b-a)}{N}\sum_{i=1}^{N} f(x_i) \tag{1}$$

Body $x_i \in (a, b)$ volíme na intervalu náhodně a přitom požadujeme co nejrovnoměrnější vzorkování. Volba bodu x_i probíhá následovně:

- Nagenerujte náhodné číslo $\gamma \in (0, 1)$.
- Transformací čísla γ z intervalu $(0, 1)$ na interval (a, b) získejte číslo x_i.

Šíření nemoci - SIR model

Pomocí vlastního programu vyřešte následující soustavu rovnic, která modeluje vývoj nemoci v uzavřené populaci o $N = 50$ členech.

- Uvažujte SIR model s následujícími rovnicemi a podmínkami.

$$\begin{aligned}\frac{\mathrm{d}S(t)}{\mathrm{d}t} &= -\beta S(t)I(t)\\ \frac{\mathrm{d}I(t)}{\mathrm{d}t} &= \beta S(t)I(t) - \gamma I(t)\\ \frac{\mathrm{d}R(t)}{\mathrm{d}t} &= \gamma I(t)\end{aligned}$$

  a podmínkami

$$\begin{aligned}S(t) + I(t) + R(t) &= N\\ \frac{\mathrm{d}S(t)}{\mathrm{d}t} + \frac{\mathrm{d}I(t)}{\mathrm{d}t} + \frac{\mathrm{d}R(t)}{\mathrm{d}t} &= 0\end{aligned}$$

- Ukažte vliv počátečních podmínek na časový vývoj jednotlivých skupin.
- Ukažte vliv koeficientů β a γ na časový vývoj jednotlivých skupin.
- Ukažte vliv metody řešení soustavy rovnic na přesnost řešení. Za metody zvolte Eulerovu metodu a metodu Rungeho-Kutty 4. řádu.

Výstupem bude:

- Grafické zobrazení průběhu nemoci, alespoň pro 3 skupiny parametrů. Např. pro případ malého nebo velkého počtu nakažených na počátku, při různých velikostech koeficientů β a γ.
- Porovnání stability a rychlosti jednotlivých řešení (Eulerova metoda a metoda Rungeho-Kutty 4.řádu). Jde například o graf časové náročnosti v závislosti na počtu členů populace, čase řešení aj.

1 Základní matematické operace v Pythonu

CÍLE KAPITOLY

Cílem této úvodní kapitoly je poskytnout přehled základních datových typů a matematických funkcí, které se budou v průběhu kurzu intenzivně využívat. Jedná se zejména o samotnou reprezentaci čísel, převody datových typů, základní operace s čísly a s tím spojené operátory, zlomky, vestavěné funkce, operace nad poli, seznamy, aj.

V této kapitole se dozvíte

- jak se v `Pythonu` reprezentují různé datové typy (celočíselné, reálné, komplexní aj.),
- jaké základní operace lze s čísly provádět pomocí balíku `math`, `fractions` aj.,
- jak lze reprezentovat sadu čísel pomocí pole, seznamu, n-tice a jaké funkce lze nad těmito poli vykonávat.

KLÍČOVA SLOVA

`math`, `fractions`, `decimal`, seznam, pole, n-tice, slovníky
Náplň látky je popsána v následujících zdrojích

- Základní informace o datových typech, operacích a funkcích lze naleznete např. v [1] v kapitole 2.
- Nápovědu pro jednotlivé balíky jazyka python lze nalézt např. zde [2].

Jednoduché příklady na datové typy, jejich deklarace, stejně tak jako na základní operace nad jednotlivými poli lze dále nalézt v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Umět deklarovat požadovaný datový typ a rozumět výsledkům operací těchto typů.
- Umět pracovat s funkcemi zejména balíků `math`, `fractions`, `decimal` aj.
- Umět deklarovat pole čísel a rozumět funkcím, které nad těmito poli operují.

OTÁZKY

1. Jaké datové typy (číselné) umožňuje python deklarovat?
2. Jak lze datové typy transformovat mezi sebou?
3. Jak lze efektivně reprezentovat vektor?

OTÁZKY K ZAMYŠLENÍ

1. Jaké jsou rozsahy datových typů?
2. Jak lze definovat přesnost matematických operací na předem definovaný počet desetinných míst?

ÚKOLY

1. Procvičte si základní operace mezi jednotlivými datovými typy a identifikujte, které operace mohou být ”potenciálně”rizikové a proč.
2. Napište program pro řešení kvadratické rovnice a ošetřete všechny možné vstupní podmínky tak, abyste dostali správné řešení (pokud existuje).

MÍSTO PRO VAŠE POZNÁMKY

2 Matematické knihovny a moduly

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled nejpoužívanějších knihoven a modulů jazyka `Python`, které jsou zaměřené na matematické výpočty. Jedná se zejména o populární balíky `NumPy` a `SciPy`, ale i o jejich nadstavby jako je `SageMath` nebo o balík využívající symbolické manipulace `SymPy`.

V této kapitole se dozvíte

- jaká jsou nejrozšířenější využití populárních knihoven a modulů jazyka `Python`,
- jaké jsou jejich možnosti a kam směřuje jejich další vývoj,
- jaké jsou nadstavby nad těmito knihovnami a moduly,
- jak se provádí základní matematické výpočty v těchto knihovnách a modulech.

KLÍČOVA SLOVA

`NumPy`, `SciPy`, `ndarray`, `SymPy`, `SageMath`

Náplň látky je popsána v následujících zdrojích

- Základní informace o `NumPy` lze nalézt zde [3].
- Základní informace o `SciPy` lze nalézt zde [4].
- Základní informace o `Sage` lze nalézt zde [5].
- základní informace o `SymPy` lze nalézt zde [6].

Příklady práce s datovými typy ve výše zmíněných balících lze nalézt dále nalézt v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Umět používat knihovny a moduly jazyka `Python` `NumPy` a `SciPy` pro matematické operace.
- Mít povědomí o nadstavbách těchto knihoven a modulů a jejich použití.

- Umět pracovat se symbolickými proměnnými a provádět základní symbolické manipulace.

OTÁZKY

1. Jaký je rozdíl mezi balíkem `NumPy` a `SciPy`?
2. Na jakém principu jsou založeny symbolické manipulace?
3. Jak lze vytvořit `NumPy` pole a naplnit ho sekvencí celých čísel?

OTÁZKY K ZAMYŠLENÍ

1. Jsou symbolické manipluace rychlejší nebo pomalejší než manipulace s `NumPy` poli?
2. Jaké operace je výhodnější dělat pomocí `SciPy` než `NumPy`?

ÚKOLY

1. Deklarujte `NumPy` pole a naplňte ho sekvencí celých čísel.
2. Převeďte seznam reálných čísel na `NumPy` pole a ošetřete přesnost čísel.
3. Proveďte základní symbolické manipulace v modulu `SymPy` zaměřené na úlohy aritmetiky.
4. Porovnejte rychlost výpočtů symbolické matematiky a `NumPy` nebo `SciPy` implementace.

MÍSTO PRO VAŠE POZNÁMKY

3 Vizualizace dat v Pythonu

CÍLE KAPITOLY

Cílem této kapitoly je podat přehled o využívaných knihovnách a modulech jazyka `Python` pro vizualizaci dat. Základním modulem je zde `matplotlib` a jeho funkce `pyplot`. Pro pokročilejší vizualizace s grafickým rozhraním je využito modulů `PyQt` a `PySide`.

V této kapitole se dozvíte

- jak rychle a jednoduše vizualizovat data pomocí `matplotlib`,
- jak nastavit atributy grafu a vizualizací pomocí funkce `pyplot`,
- jak definovat pokročilejší grafické rozhraní pomocí `matplotlib` a `PyQt`,
- jak definovat okenní aplikaci pomocí `PyQt` a `PySide`.

KLÍČOVA SLOVA

`matplotlib`, `pyplot`, `PyQt`, grafické rozhraní, okenní aplikace, `PySide`
Náplň látky je popsána v následujících zdrojích

- Základní funkce knihovny `matplotlib` je popsána například zde [7].
- Popis tvorby a používání grafického rozhraní pomocí `QtGui` a `QtWidgets` je popsán v `PyQt` například zde [9] a zde [10].
- Popis tvorby a používání grafického rozhraní pomocí `PySide` je popsán například zde [11].

Příklady vizualizací dat a definice jednoduchých grafických rozhraní lze dále najít v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Umět pomocí funkce `pyplot` knihovny `matplotlib` definovat, vykreslit a exportovat jednoduchý dvojrozměrný graf.
- Umět definovat jednoduché grafické rozhraní pomocí knihovny `PyQT` nebo `PySide`.
- Umět propojit `matplotlib` a `PyQt` pro vizualizaci dat.

OTÁZKY

1. Jak lze rychle a efektivně pomocí knihovny `matplotlib` vizualizovat data?
2. Lze pro vizualizaci vícerozměrných dat a ploch využít v `matplotlib` OPenGL podporu?
3. Jaký je rozdíl mezi grafickým rozhraním vytvořeným pomocí `matplotlib` a `PyQT`?

OTÁZKY K ZAMYŠLENÍ

1. Kdy není vhodné využívat `PyQT` a kdy naopak `matplotlib`?

ÚKOLY

1. Pomocí vlastní nebo vestavěné funkce `Pythonu` zobrazte data pomocí `pyplot`.
2. V rámci knihovny `matplotlib` definujte základní interaktivní prvky grafického rozhraní jako je posuvník aj.
3. Pomocí knihovny `PyQt` definujte vlastní grafické rozhraní na vizualizaci jednoduchých dat a definujte atributy grafu.
4. Pomocí knihovny `matplotlib` vizualizujte data v prostředí knihovny `PyQt` .
5. Pomocí knihovny `PyQt` vytvořte analogii programu kalkulačka.

MÍSTO PRO VAŠE POZNÁMKY

4 Úvod do lineární algebry

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled základních metod a přístupů pro řešení úloh z oblasti lineární algebry za pomoci knihoven a modulů jazyka `Python` a pomocí algoritmů numerické matematiky. Úlohy jsou zaměřeny na využití matic a vektorů a to zejména při řešení soustav lineárních rovnic pomocí symbolické matematiky, přímých a iteračních metod numerické matematiky a vestavěných funkcí `Pythonu`.

V této kapitole se dozvíte

- o základních numerických metodách pro řešení soustav lineárních rovnic (přímé i iterační metody),
- o funkcích knihovny `NumPy` a `SciPy` pro řešení úloh lineární algebry (např. `scipy.linalg`),
- o funkcích knihovny `SymPy` pro řešení úloh z lineární algebry pomocí symbolické matematiky (např. funkce `LUsolve`)

KLÍČOVA SLOVA

matice, determinant, Gaussova eliminace, Gaussova-Seidelova metoda, solver

Náplň látky je popsána v následujících zdrojích

- Matematické definice lze nalézt například zde [15] nebo zde [12].
- Metody numerické matematiky pro výpočet determinantu a pro řešení soustav lineárních rovnice lze nalézt například zde [17].
- Využití funkcí symbolické matematiky pomocí knihovny `SymPy` lze nalézt například zde [6].
- Využití funkcí knihovny `SciPy` nebo `NumPy` lze nalézt například zde [3] a zde [4].

Implementace vybraných metod numerické a symbolické matematiky pro řešení úloh z oblasti lineární algebry lze najít dále v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Přehled o metodách řešení soustav lineárních rovnic a jiných úloh z lineární algebry pomocí numerické a symbolické matematiky.
- Povědomí o funkcích knihoven `NumPy` a `SciPy`, které lze pro úlohy z lineární algebry použít.
- Schopnost implementovat základní metody numerické matematiky (přímé a iterační).
- Schopnost implementovat funkce symbolické matematiky pro řešení úloh menšího rozsahu.

OTÁZKY

1. Jaký je rozdíl mezi přímými a iteračními metodami pro řešení soustav lineárních rovnic?
2. Jaké funkce knihoven `NumPy` a `SciPy` lze pro řešení úloh lineární algebry využít?
3. Do jakého počtu lineárních rovnic je možné efektivně použít symbolickou matematiku?

OTÁZKY K ZAMYŠLENÍ

1. Jaké další metody pro řešení soustav lineárních rovnic jsou v knihovnách `NumPy` a `SciPy` implementovány?

ÚKOLY

1. Napište program pro výpočet determinantu pomocí Laplaceova rozvoje.
2. Napište program pro řešení soustav lineárních rovnic pomocí Cramerova pravidla, Gaussovy eleiminace a jedné z iteračních metod.
3. Porovnejte Váš program s funkcemi knihoven `NumPy` a `Scipy`.

MÍSTO PRO VAŠE POZNÁMKY

5 Interpolace a aproximace funkce jedné proměnné

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled základních metod interpolace a aproximace jednorozměrných dat pomocí metod numerické matematiky, funkcí knihoven `NumPy` a `SciPy` a pomocí a knihovny symbolické matematiky `SymPy`.

V této kapitole se dozvíte

- co je interpolace a aproximace dat a jaký je mezi nimi rozdíl,
- jaké jsou základní metody interpolace a aproximace jednorozměrných dat pomocí metod numerické matematiky,
- jaké metody interpolace a aproximace jsou obsaženy v knihovnách `NumPy` a `SciPy`,
- jak lze interpolaci a aproximaci provést pomocí knihovny `SymPy`.

KLÍČOVA SLOVA

interpolace, aproximace, lineární regrese, polynomy, splajny, lineární regrese, metoda nejmenších čtverců

Náplň látky této kapitoly je popsána v následujících zdrojích.

- Přehled základních metod interpolace a aproximace je uveden např. zde [17] nebo zde [18].
- Přehled funkcí implementovaných v knihovnách `NumPy` a `SciPy` lze nalézt zde [3] a zde [4].
- Přehled metod symbolické matematiky lze nalézt zde [6].

Příklady interpolace a aproximace jednorozměrných dat lze dále nalézt v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Mít povědomí o základních numerických metodách interpolace a aproximace jednorozměrných dat.
- Umět implementovat tyto základní metody v prostředí `Pythonu`.
- Využít vestavěné funkce knihoven `NumPy` a `SciPy` pro aproximaci a interpolaci jednorozměrných dat.
- Provést jednoduchou interpolaci pomocí funkcí knihovny `SymPy`.

OTÁZKY

1. Jaké jsou základní metody interpolace jednorozměrných dat?
2. Jaké jsou rozdíly mezi interpolací a aproximací dat?
3. Jaké jsou základní metody aproximace jednorozměrných dat?
4. Jaké jsou limity interpolace a kdy je naopak výhodné ji použít?

OTÁZKY K ZAMYŠLENÍ

1. Jak se zvyšuje náročnost a složitost interpolace a aproximace při zvyšování dimenze dat?

ÚKOLY

1. Pomocí interpolace po částech, lineární interpolace a interpolace pomocí kubických splajnů interpolujte náhodnou sadu dat.
2. Pomocí `SciPy` funkce optimize aproximujte stejnou sadu dat a výsledky porovnejte.
3. Pomocí `SymPy` funkce interpolate proveďte interpolaci jednoduché sady dat s malým rozptylem.

MÍSTO PRO VAŠE POZNÁMKY

6 Hledání kořenů rovnice

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled základních metod pro hledání kořenů algebraických rovnic ve tvaru $f(x) = 0$ a $P_n(x) = 0$ implementovaných v knihovnách `Pythonu` `NumPy` a `SciPy`, jednak v knihovně symbolické matematiky `SymPy`.

V této kapitole se dozvíte

- jaké jsou základní metody pro hledání kořenů algebraických rovnic,
- jak jsou tyto metody implementovány v jazyku `Python`,
- jak lze najít kořeny vybraných rovnic pomocí symbolické matematiky

KLÍČOVA SLOVA

bisekce, prostá iterace, metoda regula falsi, Newtonova metoda, Laguerrova metoda

Náplň látky je popsána v následujících zdrojích.

- Přehled základních metod pro hledání kořene algebraických rovnic lze nalézt v [17].
- Implementace vybraných metod v knihovnách jazyka Python `NumPy` a `SciPy` lze nalézt zde [3] a zde [4].
- Hledání kořenů rovnic pomocí `SymPy` lze nalézt zde [6].

Příklady implementace jednotlivých metod lze dále nalézt v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Přehled o základních metodách pro hledání kořenů algebraických rovnic ve tvaru $f(x) = 0$ a $P_n(x) = 0$.
- Přehled o implementaci těchto metod v knihovnách `Pythonu` `NumPy` a `SciPy`.
- Přehled o možnosti řešit tento typ úlohy pomocí symbolické matematiky a knihovny `SymPy`.

OTÁZKY

1. Jaké rovnice jsou pro tento typ úlohy vhodné?
2. Jaké jsou základní metody pro hledání kořenů algebraických rovnic?
3. Jakým způsobem lze implementovat tyto metody pomocí `Pythonu`?

OTÁZKY K ZAMYŠLENÍ

1. Jaké jsou nevýhody základních metod z hlediska stability a použitelnosti?
2. Lze pro hledání kořenů algebraických rovnic použít i jiné, než iterační metody?

ÚKOLY

1. Pomocí metody bisekce, prosté iterace a Newtonovy metody najděte kořeny algebraické rovnice $x + \ln|x| = 0$
2. Porovnejte konvergence jednotlivých metod.
3. Pomocí Laguerrovy metody najděte kořeny kubické rovnice.

MÍSTO PRO VAŠE POZNÁMKY

7 Generování náhodných čísel a testování generátorů

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled o generátorech náhodných čísel, jejich rozdělení dle typu, vlastnostech, kvalitě a způsobu jejich testování.

V této kapitole se dozvíte

- jaký je princip generování náhodných čísel na počítači,
- jaké jsou generátory a jaké principy pro generování pseudonáhodných čísel využívají,
- jak se navržené generátory testují.

KLÍČOVA SLOVA

pseudonáhodné číslo, lineární kongruenční generátor, násada, DIEHARD, TESTUo1, Mersene Twister, Marsaglia

Náplň látky je popsána v následujících zdrojích.

- Přehled základních generátorů lze nalézt například zde [19] nebo zde [20].
- Přehled testů DIEHARD a TESTUo1 lze nalézt například zde [21] a zde [22].
- Popis náhodného generátoru Mersenne Twister implementovaného v `Pythonu` lze nalézt zde [23].

Implementace vybraných generátorů náhodných čísel a porovnání s vestavěným generátorem `Pythonu` lze nalézt dále v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Mít povědomí o principech generování náhodných čísel na počítači.
- Mít povědomí o testech generátorů náhodných čísel a tím o možnostech jejich použití.
- Umět implementovat vybrané generátory a testy do jazyka Python.

OTÁZKY

1. Jaký je rozdíl mezi náhodným a pseudonáhodným číslem?
2. Jaké znáte dobré a špatné generátory čísel?
3. Jaké jsou nejznámější testy generátorů a na co kladou důraz?

OTÁZKY K ZAMYŠLENÍ

1. Proč se v dnešní době již neuplatňují generátory náhodných čísel, které jsou založené na fyzikálních principech?
2. Jak fungoval první generátor pseudonáhodných čísel?

ÚKOLY

1. Implementujte dva rozdílné typy lineárních kongruenčních generátorů a otestujte je pomocí vybrané sady testů ze sady DIEHARD a TESTU01.
2. Otestujte vestavěný generátor `Pythonu` a při různých vstupních parametrech.

MÍSTO PRO VAŠE POZNÁMKY

8 Metoda Monte Carlo

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout úvod do metody Monte Carlo, jedné ze základních metod počítačového modelování a analýzy. Dále je cílem této kapitoly představit základní aplikace této metody včetně počítání obsahů nebo objemů tvarově komplexních těles a modelování Brownova pohybu.

V této kapitole se dozvíte

- jaký je základní princip metody Monte Carlo,
- kde všude se tato metoda aplikuje a čeho s ní lze dosáhnout,
- jak lze tuto metodu efektivně implementovat v prostředí jazyka `Python`.

KLÍČOVA SLOVA

Monte Carlo, náhodná veličina, rozdělení, vzorkování, Markovovy řetězce, náhodná procházka

Náplň látky je popsána v následujících zdrojích.

- Základní princip metody Monte Carlo lze nalézt například zde [19] nebo zde [24].
- Aplikace metody Monte Carlo ve vědě a technice lze nalézt například zde [25] a zde [26] a například ve finanční oblasti zde [27].

Vybrané příklady aplikace metody Monte Carlo lze dále nalézt v e-learningovám systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Povědomí o principu metody Monte Carlo.
- Povědomí o vzorkování fázového prostoru pomocí generování a transformaci náhodných čísel a o snižování chyby metody.
- Umět implementovat metody Monte Carlo na vybrané problémy z oblasti výpočtu obsahu nebo objemu těles s komplexním tvarem.
- Mít povědomí o generování Markovových řetězců a výběru reprezentativních členů.

- Umět implementovat metody Monte Carlo pro řešení soustavy rovnic a pro modelování Brownova pohybu.

OTÁZKY

1. Na čem je metoda Mone Carlo založena?
2. Co je náhodná veličina a jak probíhá její generování a transformace?
3. Co je fázový nebo stavový prostor a proč je pro nás důležité jeho vzorkování?
4. V jaké aplikaci uplatníte Markovovy řetězce?

OTÁZKY K ZAMYŠLENÍ

1. Jak lze vzorkovat fázový prostor, který má úzká hrdla?

ÚKOLY

1. Pomocí metody Monte Carlo spočtěte obsah jednoduchého (kruh) a složitého (toroid) obrazce. Uvažujte implementaci pomocí `NumPy` polí.
2. Pomocí Metody Monte Carlo (náhodná procházka) modelujte náhodný Brownův pohyb ve dvou dimenzích. Uvažujte implementaci pomocí `NumPy` polí.
3. Na výpočtu čísla π ukažte, jak rychle metoda konverguje k přesnému řešení a výsledek porovnejte například s výpočtem čísla π pomocí Leibnizovy řady.

MÍSTO PRO VAŠE POZNÁMKY

9 Derivace funkce jedné proměnné

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled o základních metodách pro výpočet derivace a to zejména pomocí metod numerické a symbolické matematiky implementovaných v jazyku `Python`.

V této kapitole se dozvíte

- co jsou konečné diference a jak se odhaduje chyba metody,
- co jsou symbolické diference.

KLÍČOVA SLOVA

konečná diference, limita, zaokrouhlovací chyba, Taylorův rozvoj, dopředná a zpětná diference

Náplň látky je popsána v následujících zdrojích.

- Definici derivace a Taylorova rozvoje lze nalézt například v[12] nebo v [15]
- Princip konečné diference a metody pro výpočet numerické derivace lze nalézt například v[17].
- Implementace metod konečných diferencí v jazyku `Python` lze nalézt například v [3] nebo v [4].
- Implementace metod symbolické diference lze nalézt v [6].

Příklady na výpočet derivace funkce jedné proměnné pomocí konečných a symbolických diferencí lze dále nalézt v e-lerarningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste měli mít následující dovednosti a znalosti.

- Znát rozdíl mezi derivací a diferencí.
- Umět popsat konstrukci konečných diferencí například pomocí Taylorova rozvoje.
- Implementovat konečné diference do prostředí jazyka `Python`.

- Využít vestavěné nástroje knihoven `NumPy` a `SciPy` pro derivaci funkce jedné proměnné.
- Využít vestavěné nástroje knihovny `SymPy` pro symbolickou derivaci funkce jedné proměnné.
- Popsat vliv chyby metod konečných diferencí a její minimalizace.

OTÁZKY

1. Jaký je rozdíl mezi derivací a diferencí?
2. Jak lze zmenšit chybu metody založené na konečné diferenci?
3. Co je to zaokrouhlovací chyba a proč je důležitá?
4. Jak se provádí symbolická derivace?

OTÁZKY K ZAMYŠLENÍ

1. Pro jaké funkce je vhodná metoda dopředné diference a pro jaké ne?
2. Jak bude vypadat metoda konečné diference založená na více než třech bodech?

ÚKOLY

1. Pomocí konečných diferencí proveďte odhad derivace monotónní funkce a periodické funkce.
2. Studujte konvergenci odhadu k přesnému řešení v závislosti na chybě metody a zaokrouhlovací chybě.
3. Porovnejte Vaše řešení z hlediska přesnosti a rychlosti s vestavěnými funkcemi pro odhad derivace knihoven `NumPy` a `SciPy`.
4. Porovnejte Vaše řešení s výstupy symbolické derivace pomocí knihovny `SymPy`.

MÍSTO PRO VAŠE POZNÁMKY

10 Integrální počet funkce jedné proměnné

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled o základních počítačových metodách pro výpočet neurčitého a určitého integrálu funkce jedné proměnné. Postupně budou představeny metody pro symbolickou integraci (neurčitý i určitý integrál), numerickou integraci včetně vestavěných nástrojů knihoven jazyka Python a pro integraci ostatními metodami jako je například Monte Carlo.

V této kapitole se dozvíte

- jaké jsou metody numerické matematiky pro výpočet integrálu funkce jedné proměnné,
- jak lze vypočítat integrál pomocí metody Monte Carlo,
- jaké numerické metody jsou implementovány v knihovnách jazyka `Python`.

KLÍČOVA SLOVA

Netwonovy-Cotesovy vzorce, Gaussova integrace, lichoběžníková metoda, metoda střední hodnoty

Náplň látky je popsána v následujících zdrojích.

- Definice integrálu a metody jeho výpočtu jsou uvedeny například zde [15] nebo zde [13].
- Přehled metod pro výpočet neurčitého a určitého integrálu pomocí symbolických manipulací knihovny `SymPy` lze nalézt zde [6].
- Numerické metody pro výpočet integrálu funkce jedné proměnné lze nalézt například zde [17].
- Přehled metod Monte Carlo pro výpočet integrálu funkce jedné proměnné lze nalézt například zde [19].
- Implementace numerických metod v knihovnách jazyku Python `NumPy` a `ScipPy` lze nalézt zde [3] a zde [4].

Implementace vybraných numerických metod integrace funkce jedné proměnné lze dále nalézt v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste mít následující dovednosti a znalosti.

- Mít povědomí o počítačových metodách integrace funkce jedné proměnné.
- Znát základní metody numerické matematiky (obdélníková, lichoběžníková, Simpsonova).
- Znát základní metody Monte Carlo integrace.
- Umět vypočítat neurčitý a určitý integrál pomocí metod symbolické matematiky.
- Umět vypočítat integrál funkce jedné proměnné pomocí knihoven `NumPy` a `SciPy` a pomocí vlastních programů v `Pythonu`.

OTÁZKY TODO

1. Jaké znáte počítačové metody pro výpočet integrálu funkce jedné proměnné?
2. Jak se od sebe liší Newtonovy-Cotesovy vzorce a Gaussova kvadratura?
3. Je pro integrál funkce jedné proměnné lepší využít numerickou metodu nebo metodu Monte Carlo?
4. Jaké metody jsou implementovány v knihovnách `NumPy` a `Scipy` jazyka `Python`?

OTÁZKY K ZAMYŠLENÍ

1. Jak lze vylepšit přesnost numerických metod založených na Newtonových-Cotesových vzorcích?
2. Je v případě numerické integrace nutné uvažovat chybu metody?

ÚKOLY

1. Pomocí symbolické matematiky vypočítejte neurčité integrály funkce jedné proměnné. Výsledky zkuste vyčíslit pomocí funkce `subs`. Uvažujte i nespojitou funkci.
2. Pomocí vybrané metody numerické matematiky vypočítejte jeden z integrálu z předchozího bodu a porovnejte výsledky a složitost výpočtu.
3. Pomocí metody střední hodnoty Monte Carlo vypočítejte integrál funkce jedné proměnné a výsledky porovnejte s předchozími metodami.
4. Výsledky porovnejte z hlediska složitosti a rychlosti s vestavěnými funkcemi knihoven `NumPy` a `SciPy`.

MÍSTO PRO VAŠE POZNÁMKY

11 Obyčejné diferenciální rovnice

CÍLE KAPITOLY

Cílem této kapitoly je poskytnout přehled o základních počítačových metodách pro výpočet obyčejných diferenciálních rovnic (ODR), které obsahují jednu neznámou funkci jedné proměnné a její derivace (rovnice prvního a druhého řádu). Postupně budou představeny metody založené na symbolické matematice, numerické matematice včetně vestavěných nástrojů knihoven jazyka Python a ostatní metody založené na příklad na Monte Carlo integraci. Nakonec budou představeny aplikace ODR a jejich soustav do rozličných oblastí (například biologie, fyzika, finance, výroba aj.).

V této kapitole se dozvíte

- jaké jsou metody numerické matematiky pro řešení ODR prvního a druhého řádu,
- jak lze vypočítat ODR s využitím integrace metodou Monte Carlo,
- jaké numerické metody jsou implementovány v knihovnách jazyka `Python`,
- kde nacházejí ODR uplatnění.

KLÍČOVA SLOVA

Eulerova metoda a její modifikace, metody Rungeho-Kutty, jednokrokové a vícekrokové metody, počáteční podmínky, linearizace

Náplň látky je popsána v následujících zdrojích.

- Definice ODR prvního a druhého řádu a metody jejich výpočtu jsou uvedeny například zde [16] nebo zde [14].
- Přehled metod pro výpočet ODR pomocí symbolických manipulací (zejména `dsolve`) knihovny `SymPy` lze nalézt zde [6].
- Numerické metody pro výpočet ODR prvního řádu lze nalézt například zde [17].
- Přehled metod Monte Carlo pro výpočet ODR lze nalézt například zde [19] nebo zde [28].
- Implementace numerických metod v knihovnách jazyku Python `NumPy` a `ScipPy` lze nalézt zde [3] a zde [4].
- Příklady implementace ODR do rozličných vědních oblastí lze nalézt například zde [29].

Výpočet vybraných ODR prvního a druhého řádu pomocí metod numerické matematiky lze dále nalézt v e-learningovém systému MOODLE.

SHRNUTÍ

Po prostudování byste mít následující dovednosti a znalosti.

- Mít povědomí o počítačových metodách pro řešení ODR a jejich soustav.
- Znát základní metody numerické matematiky (Eulerova metoda a její modifikace, metody Rungeho-Kutty).
- Mít povědomí o metodách Monte Carlo integrace vhodných pro řešení ODR.
- Umět vyřešit ODR pomocí metod symbolické matematiky.
- Umět vyřešit ODR pomocí knihoven `NumPy` a `SciPy` a pomocí vlastních programů v `Pythonu`.

OTÁZKY

1. Jaké jsou základní metody pro řešení ODR prvního řádu?
2. Jak lze ODR vyřešit pomocí metody Monte Carlo?
3. Jak lze vyřešit ODR vyšších řádů?

OTÁZKY K ZAMYŠLENÍ

1. Je při řešení ODR prvního řádu výhodnější uvažovat Monte Carlo integraci nebo například Eulerovu metodu?
2. Jaký je vliv kroku na přesnost výsledku?

ÚKOLY

1. Pomocí symbolické matematiky vyřešte neurčité ODR prvního řádu. Výsledek vizualizujte.
2. Pomocí vybrané metody numerické matematiky vyřešte ODR jeden z předchozího bodu a porovnejte výsledky a složitost výpočtu.
3. Pomocí metody Monte Carlo vyřešte ODR a výsledky porovnejte s předchozími metodami.
4. Výsledky porovnejte z hlediska složitosti a rychlosti s vestavěnými funkcemi knihoven `NumPy` a `SciPy`.

MÍSTO PRO VAŠE POZNÁMKY

Literatura

[1] PILGRIM, Mark. *Ponořme se do Python(u) 3*. CZ.NIC, 2010. ISBN 978-80-904248-2-1.

[2] python [online]. python software foundation: ©2001 - 2020 [cit. 04.06.2020], Dostupné z: https://www.python.org/

[3] NumPy [online]. NumPy developers: ©2020 [cit. 04.06.2020], Dostupné z: https://numpy.org/

[4] SciPy [online]. SciPy developers: ©2020 [cit. 04.06.2020], Dostupné z: https://www.scipy.org/

[5] Sage [online]. ©2020 [cit. 04.06.2020], Dostupné z: https://www.sagemath.org/

[6] SymPy [online]. SymPy Development Team ©2020 [cit. 04.06.2020], Dostupné z: https://www.sympy.org/cs/index.html

[7] matplotlib [online]. The Matplotlib development team ©2012-2018 [cit. 04.06.2020], Dostupné z: https://matplotlib.org/

[8] Python [online]. Pyvec, Jan Javorek, KRAXNET ©2020 [cit. 04.06.2020], Dostupné z: https://python.cz/

[9] LearnPyQt [online]. Learn PyQt ©2019-2020 [cit. 04.06.2020], Dostupné z: https://www.learnpyqt.com/

[10] PyQtGraph [online]. ©2020 [cit. 04.06.2020], Dostupné z: http://www.pyqtgraph.org/

[11] Qt fro Python [online], [cit. 04. 06. 2020], Dostupné z: https://wiki.qt.io/Qt_for_Python

[12] CHARVÁT, Jura; BUDÍNSKÝ, Bruno. *Matematika I-část 1.*, ČVUT Praha, 2000, ISBN 80-01-02175-0.

[13] CHARVÁT, Jura; BUDÍNSKÝ, Bruno. *Matematika I-část 2.*, ČVUT Praha, 2000, ISBN 80-01-02174-2.

[14] CHARVÁT, Jura; BUDÍNSKÝ, Bruno. *Matematika II*, ČVUT Praha, 1996, ISBN 80-01-01092-9.

[15] REKTORYS, Karel a kol. *Přehled užité matematiky-svazek 1.* SNTL Praha, 1988. ISBN 04-022-88-01.

[16] REKTORYS, Karel a kol. *Přehled užité matematiky-svazek 2.* SNTL Praha, 1988. ISBN 04-022-88-02.

[17] VICHER, Miroslav. *Numerická matematika.* UJEP Ústí nad Labem, 2003. ISBN 80-7044-516-5.

[18] RALSTON, Anthony. *Základy numerické matematiky*. Academia, 1978 ISBN 80- 105-1256-4.

[19] HRACH, Rudolf. *Počítačová fyzika I*. UJEP, 2003. ISBN 80-7044-521-1

[20] RANDOM.ORG [online]. RANDOM.ORG ©1998-2020 [cit. 04.06.2020], Dostupné z: https://www.learnpyqt.com/

[21] Robert G. Brown's General Tools Page [online]. Robert G. Brown ©2020 [cit. 04.06.2020], Dostupné z: https://webhome.phy.duke.edu/∼rgb/General/dieharder.php

[22] TestU01: A C library for empirical testing of random number generators [online]. ACM ©2020 [cit. 04.06.2020], Dostupné z: https://dl.acm.org/doi/10.1145/1268776.1268777

[23] MATSUMOTO, Makoto; NISHIMURA, Takuji. Mersenne twister: a 623-dimensionally equidistributed uniform pseudo-random number generator. *ACM* 1998, **8**(1), 3-30. ISSN 1049-3301.

[24] VIRIUS, Miroslav. *Metoda Monte Carlo*. ČVUT Praha, 2010, ISBN 978-80-01-04595-4.

[25] FABIAN, František; KLUIBER, Zdeněk.. *Metoda Monte Carlo a možnosti jejího uplatnění*. Prospektrum, 1998, ISBN 80-7175-058-1.

[26] BUSLENKO, Nikolai A.; GOLENKO, Denis; SOBOL, Ilya M.; SRAGOVIC, Vladimir G. *The Monte Carlo Method: The Method of Statistical Trials*. ed. SCHREIDER, Julius A., Pergamon, 1966, ISBN 9781483155579

[27] GLASSERMAN, Paul. *Monte Carlo Methods in Financial Engineering*. Springer, 2004, ISBN 978-0387004518.

[28] KROESE, Dirk P.; TAIMRE, Thomas, BOTEV, Zdravko I. *Handbook of Monte Carlo Methods*. Wiley & Sons, 2011, ISBN 978-0470177938.

[29] ANG, Whye T.; PARK Y. S.; *Ordinary Differential Equations: Methods and Applications*. Universal-Publishers, 2008, ISBN 9781599429755.