Projekt 2024 - část 1

předmět Zpracování a vizualizace dat v prostředí Python

Změny: 8. 10. 2024: Opraven graf sinusovky (úkol 2)

Cílem projektu je získat, zpracovat a analyzovat data dostupná na internetu. První část projektu se bude týkat ověření vašeho osvojení technik pro efektivní zpracování, vizualizaci a získání dat.

Orientační struktura projektu

- Část 1 (20 b)
  - efektivní numerické výpočty
  - jednoduchá vizualizace
  - stažení a zpracování dat
- Část 2 (20 b)
  - různé pohledy na data
  - pokročilá vizualizace výsledků
  - zpracování závěrů (porozumění datům)
- Celkový projekt (60 b)
  - znázornění dat na mapě, operace nad těmito daty
  - korelace a predikce
  - automatické vytváření částí zpráv
  - spojení do analytické zprávy

Získání a předzpracování dat (20 bodů)

Vytvořte soubor part01.py, který bude implementovat níže uvedené metody. Pro zpracování a vizualizaci dat **není povoleno** použít pokročilých knihoven jako je Pandas či Seaborn. Kromě vestavěných knihoven (os, sys, re, gzip, pickle, csv, zipfile…) byste si měli vystačit s: numpy, matplotlib, BeautifulSoup, requests. Další knihovny je možné použít po schválení opravujícím (např ve fóru IS VUT).

Úkol 1: Numerický výpočet euklidovské vzdálenosti (3 body)

Cílem této funkce je pro dvě pole a jejich všechny body *a* a *b* zjistit jejich eukleidovskou vzdálenost. Platí tedy, že pole by měly mít stejné rozměry (není třeba testovat ve funkci). Pro pole a i b platí tedy, že mají rozměr [N, D], kdy N je počet prvků a D je počet dimenzí. Výsledkem volání funkce je jednorozměrné numpy pole o N prvcích, kdy každý prvek se spočítá následovně

$$d(a_i, b_i) = \sqrt{(a_{i,1} - b_{i,1})^2 + (a_{i,2} - b_{i,2})^2 + \cdots + (a_{i,D} - b_{i,D})^2}$$

Pro získání plného hodnocení je nutné se zaměřit na efektivitu a využít možností knihovny *numpy* - t.j. vyhnout se procházení všech prvků cyklem `for`. Je zakázáno použít knihovní

funkci, která přímo vrací určitý euklidovskou vzdálenost, redukční funkce jsou samozřejmě povolené.

Prototyp funkce:

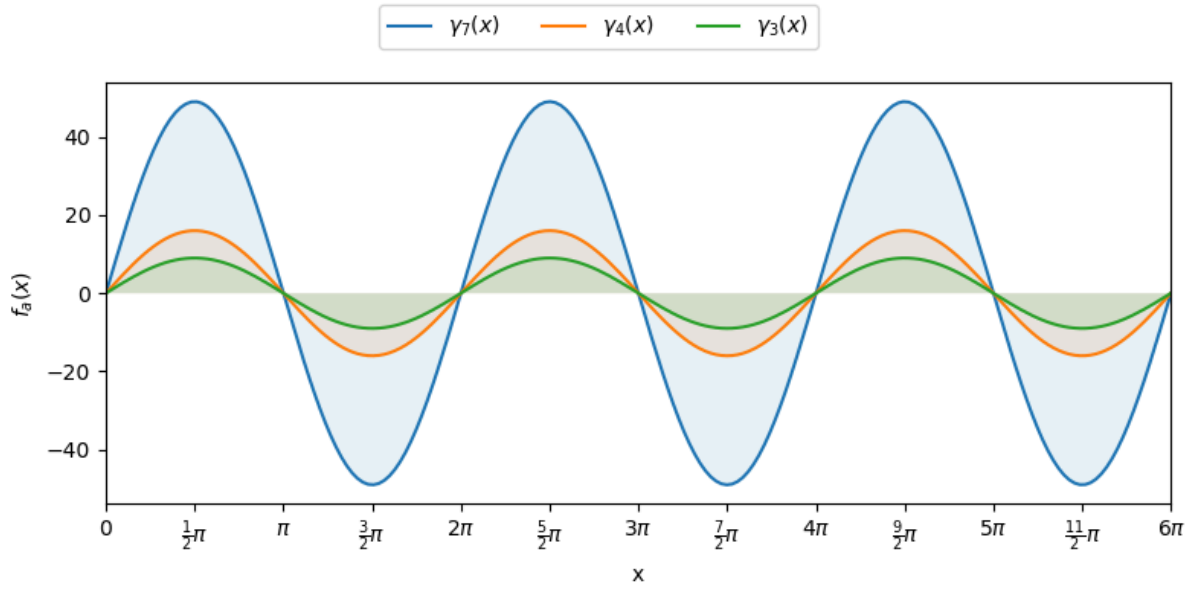
```
def distance(a : np.array, b : np.array) -> np.array:
```

Ukázka volání:

```
distance(
   np.array([[0, 0], [0, 0], [2, 2]]),
   np.array([[1, 1], [2, 2], [5, 6]])
)
```

Úkol 2: Generování grafu s různými koeficienty (6 bodů)

Navrhněte funkci `generate_graph`, která bude vizualizovat funkci $f_a(x) = a^2 * sin(x)$, definovanou na rozsahu <0, 6pi>. Na začátku kódu vygenerujte v jednom kroku výsledky funkce *f* pro všechny hodnoty *a* (t.j. bez cyklů, do dvourozměrné matice) zadané ve vstupním argumentu (reprezentované jako seznam čísel s plovoucí desetinnou čárkou). Je tedy nutné (pro plné hodnocení) využít broadcasting.
Následně tuto matici po jednotlivých řádcích vizualizujte tak, aby výsledný graf vypadal následovně. Pro nastavení rozsahů zobrazení, umístění popisků a podobně můžete počítat s tím, že `a=[7, 4, 3]`. Je nutné dodržet Latex styl sazby popisků os a jednotlivých čar. Zachovejte následující vzhled (včetně popisků a ticků os a podbarvení):



Funkce `generate_graph` má další dva argumenty - boolean hodnotu `show_figure`, která určuje, zda se má graf zobrazit pomocí funkce `show()` a `save_path`, která (pokud je nastavena), určuje, kam se má graf uložit pomocí funkce savefig().

Prototyp funkce

```
def generate_graph(a: List[float],
```

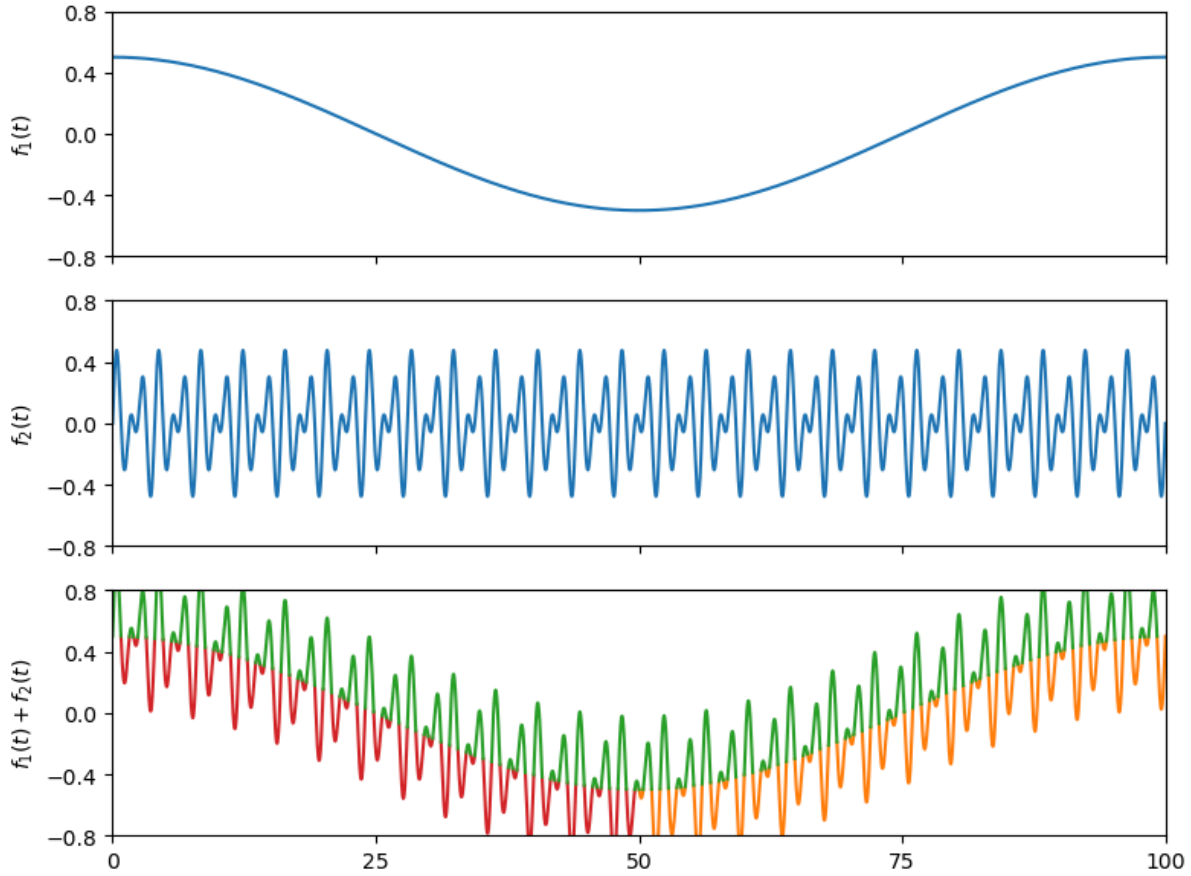
```
                    show_figure: bool = False,
                    save_path: str | None = None):
```

Úkol 3: Pokročilá vizualizace sinusového signálu (5 bodů)

Vytvořte graf se třemi podgrafy, zobrazující funkci f_1, f_2 a součet f_1+f_2 v rozsahu $t \in$ <0,100>. Funkce jsou definovány následovně:

$$f_1(t) = 0.5 \cdot cos(\frac{1}{50}\pi t),\ f_2(t) = 0.25 \cdot (sin(\pi t) + sin(\frac{3}{2}\pi t))$$

V třetím podgrafu bude část, kdy se hodnota součtu obou funkcí dostává nad hodnotu samotné funkce f_1 zeleně, v opačném případě červeně pro t < 50 a oranžově pro t >=50. Je nutné, aby na grafu nevznikly žádné další artefakty (např. nějaké spojovací čáry a podobně). Upravte také body na ose tak, aby vypadaly, jako na tomto vzorovém obrázku a použijte sdílenou osu y. Pro argumenty `show_figure` a `save_path` platí stejné podmínky, jako v druhém úkolu.



Prototyp funkce

```
def generate_sinus(show_figure: bool = False,
                   save_path: str | None = None)
```

Úkol 4: Stažení tabulky (5 bodů)

Ze stránek https://ehw.fit.vutbr.cz/izv/stanice.html stáhněte meterologických stanic. Jedná se o kopii stránech CHMI a je zakázáno se připojovat na oficiální stránky. Url, ze které stahujete, můžete mít uloženou v kódu. Nemusíte pracovat rovnou s touto stránkou (v

prohlížeči např. přes *Network panel* zjistíte, že stránky jsou řešeny poměrně “originálním” stylem). Každý sloupec bude reprezentovaný položkou ve slovníku obsahující seznam všech hodnot ve vhodném datovém typu. Příklad výstupu

```
{'positions': ['Cheb',  'Karlovy Vary' ],
 'lats': [50.0683,  50.2016],
 'longs': [12.3913, 12.9139],
 'heights': [483.0,  603.0]}
```

Výstupem funkce bude slovník (dict) obsahující seznamy (list) pro jednotlivé řádky. Validita výstupního formátu je základním způsobem testována i v přiloženém *unittestu*. Můžete počítat s tím, že struktura stránek se nezmění, vlastní načítání je však povinné dělat ze stránek https://ehw.fit.vutbr.cz/izv a je přísně zakázáno přistupovat na stránky CHMI.

Prototyp funkce

```
def download_data() -> Dict[str, List[Any]]:
```

Testování

K otestování funkčnosti můžete přistoupit dvěma základními způsoby. Buď do části, která se spouští pouze pokud je zavolán celý skript (`if __name__=="__main__"`), vložíte vaši testovací sekvenci, nebo můžete použít předpřipravený unittest v souboru *test_part01.py* a knihovnu pytest. Pokud máte tuto knihovnu nainstalovanou, můžete spustit příkaz `pytest` či `python3 -m pytest`. Pokud projdete testovacím skriptem, je to podmínka nutná (nikoliv dostačující) k dobrému hodnocení.

Dokumentace všech částí (souborů, funkcí) bude přímo v odevzdaném souboru. Snažte se dodržovat konvenci PEP 257 [https://www.python.org/dev/peps/pep-0257] a PEP 8 [https://www.python.org/dev/peps/pep-0008/]. Pozor, Python má přímo definovaný kódovací styl (narozdíl od C), a proto kontrola bude součástí hodnocení (1 bod).

Odevzdávání a hodnocení

Do 3. 11. 2024 odevzdejte jeden soubor *part01.py*.

Hodnotit se bude zejména:

- správnost výsledků
- vizuální dojem z grafů
- kvalita kódu
    - efektivitu implementace a reprezentace (i rychlost v porovnání s ostatními řešeními), využívání efektivních funkcí (např. NumPy)
    - přehlednost kódu
    - dodržení standardů a zvyklostí pro práci s jazykem Python - dokumentační řetězce, splnění PEP8
    - dokumentace funkcí a souborů
    - znovupoužitelnost kódů - správná izolace potřebných částí do funkcí

Celkem za první část můžete získat až 20 bodů, přičemž je k zápočtu nutné získat z této části minimálně 1 bod.

Dotazy a připomínky

Na fóru IS VUT případně na mailu mrazek@fit.vutbr.cz.