Autor: Gabriel Majeri

Procesarea Semnalelor
Laboratorul 1
Introducere. Recapitularea unor noțiuni fundamentale

1 Cunoștințe necesare

În prima săptămână, vă veți familiariza cu principalele concepte teoretice și unelte cu care veți lucra în cadrul acestei discipline. Laboratoarele și exercițiile din cadrul lor vă vor ajuta să explorați, să înțelegeți, să vizualizați și să aplicați în practică noțiunile discutate la curs.

Din punct de vedere matematic, ar fi foarte util să vă reamintiți sau să vă consolidați noțiuni de bază de **algebră liniară**, **trigonometrie** și de **numere complexe**. Câteva resurse utile în acest sens:

- Seria de video-uri *Essence of linear algebra* de la 3Blue1Brown;
- Tot de la 3Blue1Brown, un video de 5 minute care explică formula lui Euler ($e^{i\pi} = -1$);
- Khan Academy pentru consolidarea/recapitularea oricăror altor noțiuni de matematică din liceu;

2 Limbajul Python

Python este un limbaj de nivel înalt, multi-paradigmă (cu suport inclusiv pentru programare procedurală, programare orientată pe obiecte și programare funcțională), care se concentrează pe productivitatea programatorului și pe lizibilitatea codului sursă.

Acesta a fost creat de programatorul olandez Guido van Rossum în anii '90, cu scopul de a dezvolta un limbaj ușor de învățat, de folosit și de extins. În anii care au urmat, a fost preluat de comunitatea de calcul științific (apărând astfel biblioteci ca NumPy și SciPy). Python a ajuns să fie foarte popular și folosit și în domeniul învățării automate, mai ales după apariția unor biblioteci de învățare automată precum Tensorflow (în 2015) și PyTorch (în 2016).

Noi la laborator nu vom lucra direct cu fișiere sursă / script-uri de Python, ci printr-o interfață mai facilă și mai potrivită pentru lucrul interactiv și iterativ: Jupyter Notebooks. Acest sistem permite crearea de "caiete de laborator" interactive, care combină codul sursă cu secțiuni de text, conținut multimedia șamd.

2.1 Instalare

Dacă vreți să folosiți calculatorul personal pentru laborator și nu aveți deja instalat Python, puteți urma instrucțiunile de aici. După aceea, va trebui să folosiți managerul de pachete `pip` pentru a instala celelalte dependințe. Alternativ, dacă vreți un pachet software complet, care să includă Python, NumPy, SciPy, Matplotlib și Jupyter pre-instalate, puteți să folosiți Anaconda.

3 Biblioteca NumPy

NumPy este o bibliotecă *open source* care oferă structuri de date și algoritmi de o importanță fundamentală pentru calculul numeric în Python. Este „calul de lucru" folosit în majoritatea aplicațiilor numerice sau științifice în Python.

4 Biblioteca SciPy

SciPy este o bibliotecă care construiește peste structurile de date și funcțiile de bază oferite de NumPy, pentru a oferi o suită completă de funcționalități necesare calculului științific în Python. Printre altele, oferă implementări eficiente de funcții pentru optimizare, integrare, rezolvarea de ecuații diferențiale, prelucrarea semnalelor etc.

5 Exerciții

1. Calculați „de mână” următorul produs matrice-vector (scrieți, pe cât posibil, și calculele intermediare): (1p)

$$A = \begin{pmatrix} 1 & 0 & 3 \\ 2 & -1 & 5 \\ 0 & 7 & 0 \end{pmatrix} \qquad v = \begin{pmatrix} -2 \\ 0 \\ 4 \end{pmatrix}$$

$$Av = ?$$

2. Calculați „de mână” valoarea următoarei integrale: (1p)

$$\int_{-1}^{1} e^x \mathrm{d}x$$

(nu este nevoie să scrieți o valoare numerică aproximativă, răspunsul poate să-l conțină pe e)

3. Calculați „de mână”, arătând pașii intermediari, valoarea următoarei integrale: (1p)

$$\int_{0}^{1} e^{i\pi x} \mathrm{d}x$$

Sfat: puteți să calculați o integrală complexă “spărgând-o” în integrala calculată pentru partea reală, respectiv pentru partea imaginară.

4. Definiți în Python (folosind biblioteca `numpy`) două variabile, care să stocheze matricea A, respectiv vectorul v, de la exercițiul 1. Puteți urma pașii din acest tutorial. Calculați același produs matrice-vector, dar de data aceasta folosind operatorul `@` din Python/`numpy`. (1p)

5. Calculați numeric valoarea integralei de la exercițiul 2, folosind funcția `quad` din modulul `integrate` al SciPy. Dacă nu ați mai lucrat în trecut cu funcții lambda în Python, tot ce aveți nevoie pentru a defini și transmite funcția de integrat la `integrate.quad` este expresia `lambda x: np.exp(x)` (presupunând că aveți importat modulul `numpy` cu alias-ul `np`). (1p)

6. Calculați numeric valoarea integralei complexe de la exercițiul 3, de data aceasta folosind o altă funcție de integrare numerică în afară de `quad` (e.g. puteți încerca `fixed_quad`). În acest caz, funcția de integrat ar veni `lambda x: np.exp(1j * np.pi * x)`. (1p)