Programowanie 1R – Ćwiczenia 9
Biblioteka `numpy`

Mikołaj Marcinkowski

12 maja 2026

Zadania możesz wykonywać jako oddzielne skrypty `.py` lub w notatniku jupytera `.ipynb`, chyba że treść zadania wskazuje inaczej.

Zadania wzorowane na zadaniach ze strony `https://numpy.org/doc/stable/user/quickstart.html` oraz na Serii 9 zadań autorstwa Bartłomieja Zglinickiego (`https://www.fuw.edu.pl/~bzglinicki/teaching/p1r/zadania/seria-9/`), udostępnionej na licencji CC BY 4.0..

Zadanie 1. Wykresy

Korzystając z numpy'owych arrayów, narysuj wykresy funkcji $\sin(x)$, $\cos(x)$, $\cos^2(x) - \sin^2(x)$.

Zadanie 2. Szybkość numpy'a

Stwórz **listę** kolejnych 10.000.000 liczb całkowitych. Stwórz listę dwukrotności tych liczb i zmierz, ile czasu to zajęło.

Powtórz te operacje na arrayach.

Możesz też sprawdzić inne operacje, np. podnoszenie do kwadratu.

Możesz też spróbować na większych liczbach, ale komputery w pracowni mogą nie mieć dość RAMu.

Zadanie 3. Macierze

Napisz funkcję `R(theta)` zwracającą macierz obrotu:

$$\begin{bmatrix} \cos\theta & -\sin\theta \\ \sin\theta & \cos\theta \end{bmatrix} \tag{1}$$

jako numpy'owy array.

Korzystając z `np.allclose` sprawdź numerycznie dla kilku kątów, że

$$R(\theta)R(\phi) = R(\theta + \phi)\,. \tag{2}$$

Stwórz array $2 \times N$ wektorów dwuwymiarowych i obróć je o wybrany kąt korzystając z macierzy $R(\theta)$.

Zadanie 4. Zbiór Mandelbrota

Stwórz następujące arraye:

- `c` : array $N \times M$ zawierający siatkę równomiernie rozstawionych liczb zespolonych z kwadratu $-2 - 2i$ do $2 + 2i$. Do tego celu może przydać się funkcja `np.meshgrid`;
- `z` : array tych samych wymiarów co `c`, zainicjiowany zerami;
- `iteracje` : array tych samych wymiarów co `c`, zainicjiowany nieskończonościami.

Następnie powtarzaj w pętli następujące operacje:

1. $\texttt{z} = \texttt{z}^2 + \texttt{c}$;
2. znajdź te pola w siatce, w których wartość $|\texttt{z}|$ przekroczyła 2 w tej iteracji, po czym nadpisz odpowiadające im pola w arrayu `iteracje` numerem tej iteracji.

Wypróbuj kilka różnych maksymalnych liczb iteracji np. 10, 20, 100.

Po zakończeniu pętli wykonaj rysunek, w którym poszczególne piksele są pokolorowane innymi kolorami w zależności od numeru iteracji w arrayu `iteracje`. Możesz wykorzystać do tego celu bibliotekę `matplotlib` i funkcję `ax.imshow`. Innymi funkcjami dobrymi do takich rysunków są `ax.contour` i `ax.contourf`.

Zadanie 5. Monte Carlo

Wylosuj `N` punktów w kwadracie $[0, 1] \times [0, 1]$. Policz, ile z nich znajduje się w ćwiartce okręgu o promieniu 1 i środku w punkcie $(0, 0)$. Oszacuj na tej podstawie liczbę π.

Powtórz całą procedurę `M` razy. Na podstawie `M` eksperymentów oblicz spodziewaną wartość π i odchylenie standardowe.

Do losowania punktów możesz skorzystać z `np.random.rand`.