Projekt 1 – Perkolacja Szymon Słomkowski

Perkolacja – z łac. *percolare*, czyli przefiltrowywać, przeciekać – termin pierwotnie opisujący przepływ wody na podłoże stałe, powodujący jego wymywanie, obecnie nabrał szerszego znaczenia. Współcześnie termin ten używany jest do opisu wielu zjawisk w dziedzinach fizyki, inżynierii materiałowej, epidemiologii, czy w badaniu układów złożonych.

W tej pracy przedstawię wyniki badania nad perkolacją układów złożonych reprezentowanych przez dwuwymiarowe tablice o rozmiarze 50x50. Zdecydowałem się na taki rozmiar ze względu na ograniczenia – dla większych tablic przy niskich wartościach prawdopodobieństwa program nie działa prawidłowo, nie podając żadnego komunikatu błędu lub informacji o przekroczeniu dostępnych zasobów.

1. Metoda generowania układu złożonego

Do wygenerowania układów złożonych wykorzystałem funkcję choices() z modułu random. Funkcja ta pozwala wylosować jeden z podanych w tablicy elementów z prawdopodobieństwem wylosowania kolejnych elementów podanych jako tablica liczb zmiennoprzecinkowych w argumencie *weights*. W swojej metodzie generowania zdecydowałem się na losowanie pomiędzy 0 a 2 (gdzie 0 oznacza pole otwarte, natomiast 2 – pole zamknięte) ze względów kosmetycznych – liczba 2 podczas tworzenia obrazu macierzy zawsze będzie miała ten sam kolor. Liczbie 2 została przyporządkowana wartość p z zakresu od 0.00 do 1.00, kolejno co 0.01, natomiast liczbie 0 wartość 1-p.

2. Ilość pomiarów

Zdecydowałem się na przeprowadzenie 10 kolejnych prób, dla każdego ze 100 kolejnych prawdopodobieństw generując po 100 macierzy. Sumaryczna liczba wygenerowanych macierzy to 100 000. Wszystkie powstające obrazy macierzy były automatycznie zapisywane w folderze „plots”, w podfolderach oznaczonych wartością prawdopodobieństwa, wymiarem tablicy i numerem próby.

3. Metoda sprawdzania perkolacji systemu

Do sprawdzenia, czy dany system perkoluje, stworzyłem funkcję percolation(), która jako argument przyjmuje tablicę dwuwumiarową utworzoną przez funkcję create_matrix(). Każda z macierzy była kopiowana przy użyciu funkcji deepcopy() z moduły copy do pomocniczej macierzy full_matrix. Następnie, używając rekurencyjnej funkcji _percolation() wzorowanej na funkcji zaproponowanej w pracy Roberta Sedgewicka, Kevina Wayne’a i Roberta Dondero’a omawianej na zajęciach, układ był poddawany ocenie możliwości do perkolacji. Funkcja ta po sprawdzeniu, czy pole w macierzy open_matrix, w którym się znajdujemy, mieści się w zakresie i czy jest otwarte (ma wartość 0), zmieniała wartość korespondującego pola w macierzy full_matrix na 1. Na koniec następowało ponowne wywołanie funkcji _percolation() dla pól sąsiadujących.

Rekurencyjna funkcja była wywoływana dla każdego otwartego pola w pierwszym rzędzie macierzy open_matrix. Po zakończeniu wywoływania tej funkcji następowało sprawdzanie, czy w ostatnim rzędzie macierzy full_matrix znajduje się jakiekolwiek pole z wartością 1. Jeżeli tak, to funkcja zwracała krotkę (True, full_matrix) co oznaczało, że dany system perkoluje.

Jeżeli takiego pola nie było, zwracała (False, full_matrix), sygnalizując tym samym, że perkolacja nie zachodzi.

4. Obliczenia

Po każdorazowym sprawdzeniu, czy system perkoluje do pomocniczej zmiennej *summ* była dodawana liczba 1, gdy perkolacja zachodziła. Następnie, po wygenerowaniu 100 systemów dla danej wartości prawdopodobieństwa, zmienna *summ* była dzielona przez liczbę wygenerowanych systemów i dopisywana do pomocniczej tablicy *prob*. Na zakończenie każdej z 10 prób tablica *prob* była dopisywana do tablicy *list_prob*. W celu uzyskania ostatecznych wyników kolejne wartości z tablic *prob* były sumowane i dzielone przez ilość prób, w celu uzyskania średniej wartości prawdopodobieństwa zachodzenia perkolacji dla 1000 prób z daną wartością prawdopodobieństwa występowania zamkniętego pola w układzie.

5. **Wyniki**

5.1. Prezentacja wygenerowanych systemów i zachodzenia w nich perkolacji, na przykładzie macierzy open_matrix i full_matrix dla prawdopodobieństwa występowania pola zamkniętego w układzie równego 0.41

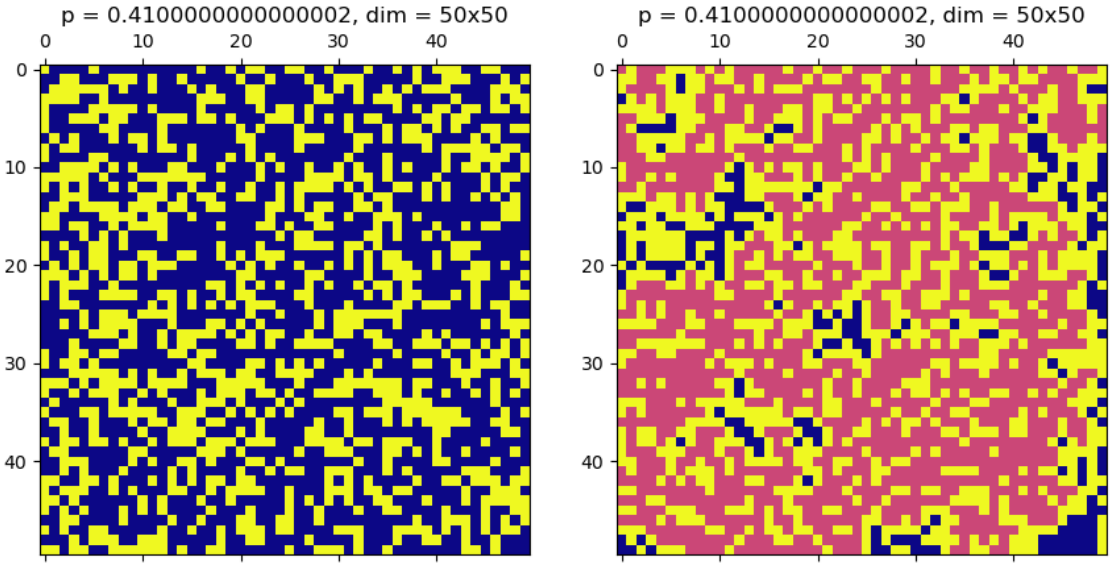


fig. 1. System 50x50 z p = 0.41, w którym perkolacja zachodzi

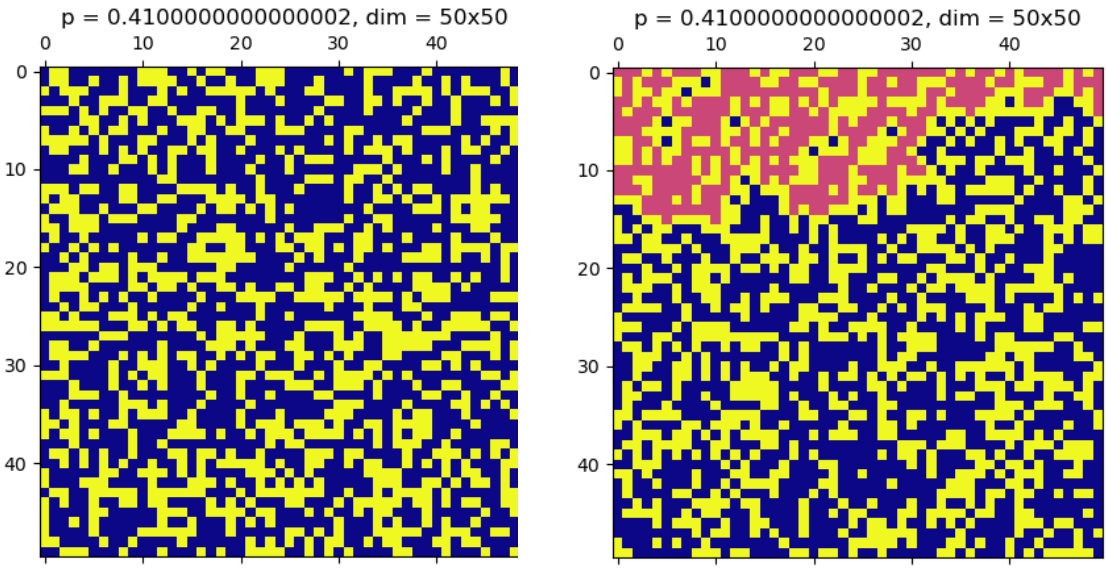
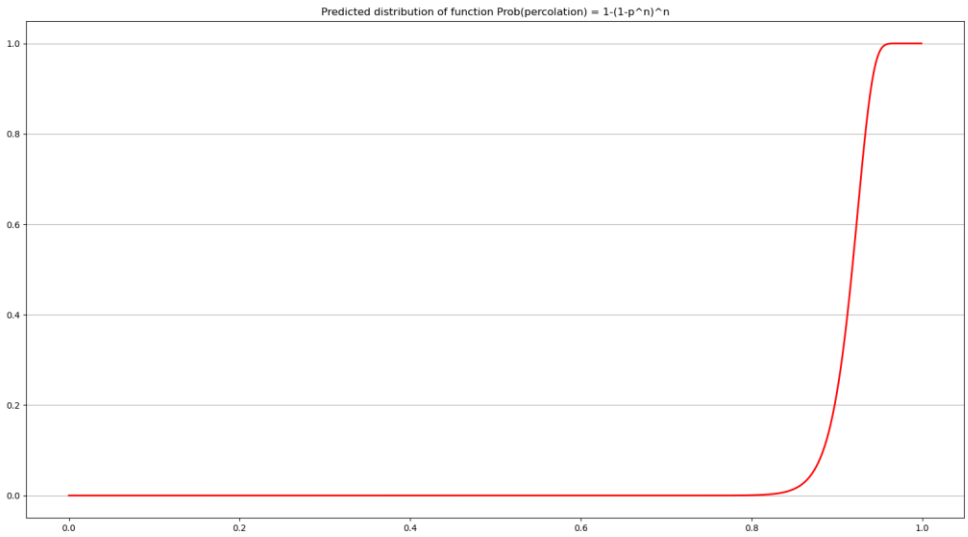


fig. 2. System 50x50 z p = 0.41, w którym perkolacja nie zachodzi

W przypadku, gdy nie wystąpią miejsca, do których funkcja _percolation() nie doszła, macierze nie będą się od siebie różniły niczym – kolor granatowy wartości 0 zostanie przejęty przez wartość 1

5.2. Rozkład funkcji prawdopodobieństwa podanej w zadaniu

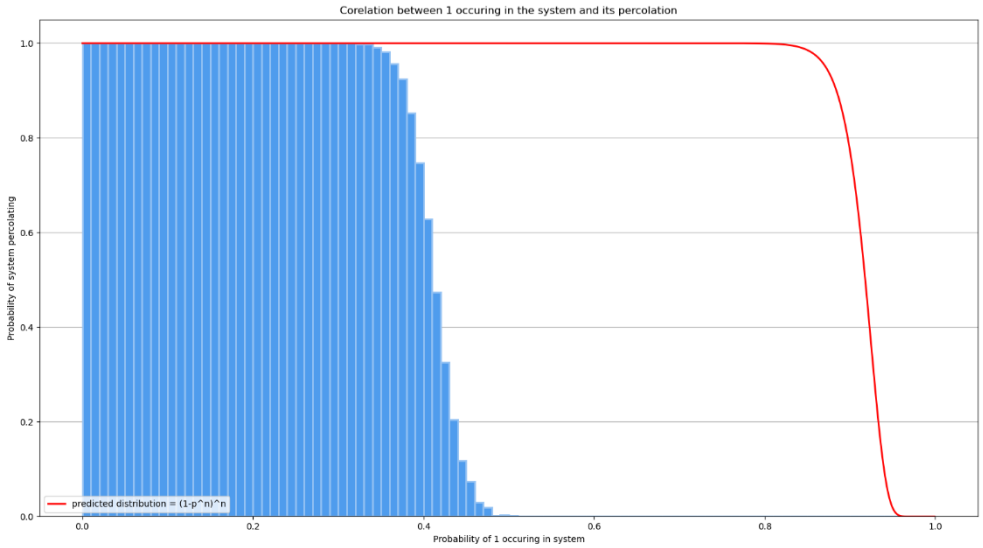
$$\text{prob(percolation)} = 1 - (1 - p^n)^n$$



Na wykresie funkcji widać, że funkcja może obrazować przejście fazowe.

Funkcja ta nie opisuje dobrze otrzymanych przeze mnie wyników. Lepiej opisuje je funkcja prob(percolation) = $(1 - p^n)^n$, będąca lustrzanym odbiciem funkcji podanej w zadaniu.

5.3. Średnie wartości prawdopodobieństwa zachodzenia perkolacji i rozkład funkcji $\mathrm{prob(percolation)} = (1 - \mathrm{p}^n)^n$



Na powyższym wykresie widać, że własność przejścia fazowego jest zachowana, natomiast wykres funkcji jest przesunięty względem wyników o ~0.45 wartości prawdopodobieństwa występowania „1” (pola zamkniętego) w układzie złożonym.

5.4. Średnie wartości prawdopodobieństwa zachodzenia perkolacji dla każdej z 10 prób są załączone w pliku „results.png”, jednak ze względu na ich ograniczoną czytelność i ogromną ilość nie załączam ich w tym dokumencie.

6. Wnioski

Prawdopodobieństwo perkolacji układu złożonego jest zależne od prawdopodobieństwa występowania pola zamkniętego w tym układzie – im mniejsze prawdopodobieństwo wystąpienia pola zamkniętego, tym wyższe prawdopodobieństwo zachodzenia perkolacji układu. W okolicy wartości 0.41 prawdopodobieństwa występowania pola zamkniętego w układzie zachodzi zjawisko przypominające przejście fazowe między wartościami 1.0 i 0.0 prawdopodobieństwa zachodzenia perkolacji układu.

Krzywa, opisana przez funkcję $\mathrm{prob(percolation)} = 1 - (1 - p^n)^n$, nie opisuje dobrze prawdopodobieństwa zachodzenia perkolacji. Lepiej robi to krzywa opisana przez funkcję $\mathrm{prob(percolation)} = (1 - \mathrm{p}^n)^n$.