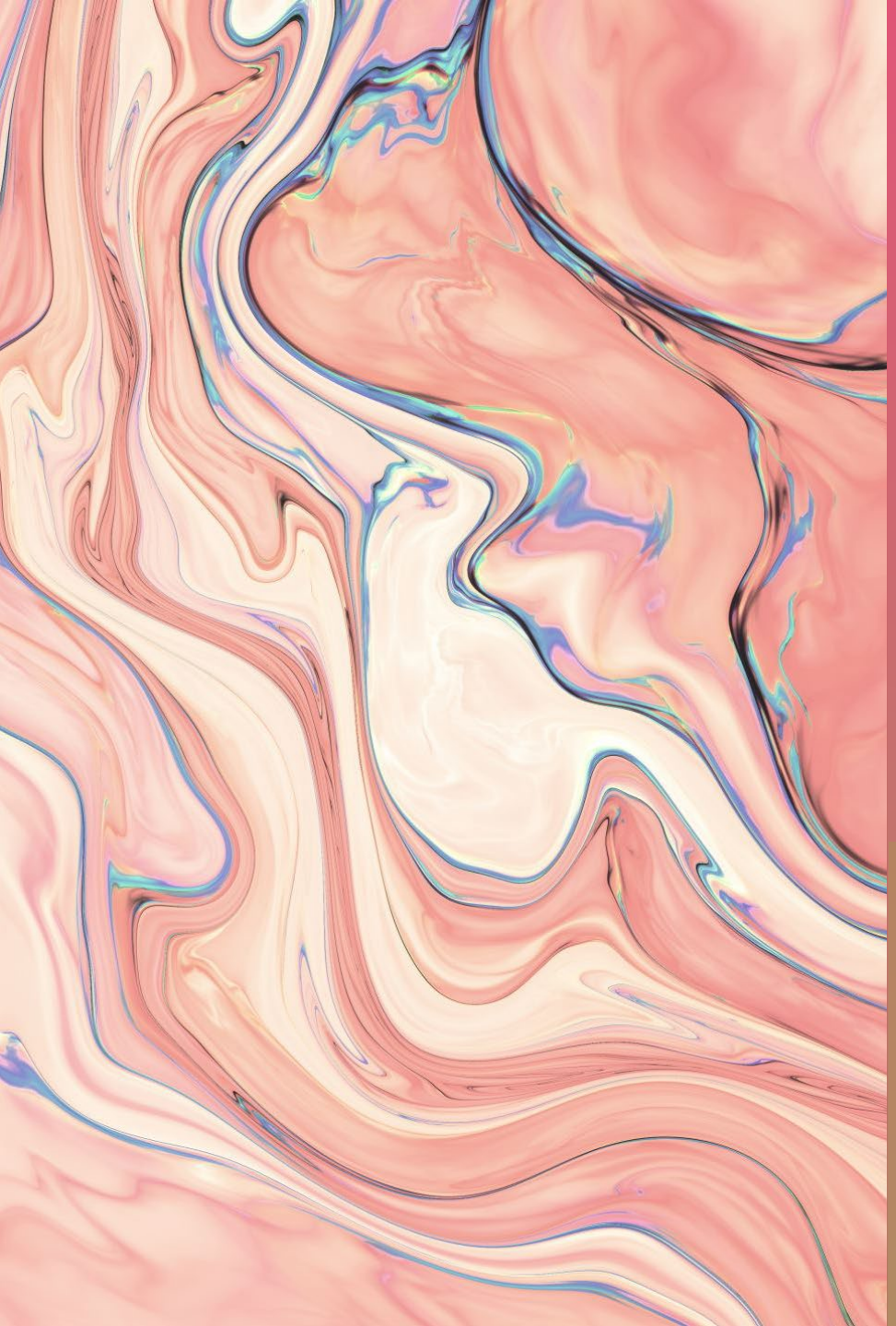


ALGORYTMY EWOLUCYJNE

LABORATORIUM

DR HAB. INŻ. **RAFAŁ POROWSKI**, PROF. UJK

EMAIL: RPOROWSKI@GMAIL.COM

INSTYTUT FIZYKI UJK
(BUD. G, POK. A-109)

ZADANIE 1:

Celem zadania jest zaprojektowanie algorytmu ewolucyjnego, który będzie optymalizował funkcję celu, wykorzystując binarną reprezentację liczb. Studenci mają zastosować dostarczoną funkcję do kodowania rozwiązań w przestrzeni binarnej i przeprowadzić proces selekcji, krzyżowania oraz mutacji.

Treść zadania

Założenia

1. Algorytm będzie optymalizował funkcję celu:

   $$f(x) = x^2$$

   dla x w przedziale $[0, 63]$.

2. Rozwiązania są reprezentowane w postaci binarnej (6-bitowa reprezentacja liczb z przedziału $[0, 63]$).

3. Operacje genetyczne (selekcja, krzyżowanie, mutacja) będą stosowane na poziomie binarnym.

ZADANIE 1:

Kroki do wykonania

1. **Inicjalizacja populacji**:
   - Wygeneruj populację $n = 10$ liczb z przedziału $[0, 63]$.
   - Przekonwertuj liczby na reprezentację binarną.
2. **Obliczenie dopasowania**:
   - Dla każdego osobnika oblicz wartość funkcji celu $f(x) = x^2$.
3. **Selekcja**:
   - Zaimplementuj selekcję ruletkową lub turniejową, aby wybrać osobniki do reprodukcji.
4. **Krzyżowanie**:
   - Wykonaj krzyżowanie jednopunktowe na wybranych parach rodziców.

ZADANIE 1:

5. **Mutacja**:
    - Zastosuj mutację punktową na genotypach potomków z prawdopodobieństwem $p = 0.05$.
6. **Aktualizacja populacji**:
    - Zastąp starą populację nową generacją.
7. **Iteracja**:
    - Powtórz kroki 2–6 przez 20 pokoleń.
8. **Wynik końcowy**:
    - Wyświetl najlepszego osobnika i jego wartość dopasowania po zakończeniu iteracji.

ZADANIE 1:

Instrukcje dla studentów

1. **Uruchomienie kodu**:
   - Skopiuj kod do pliku MATLAB o nazwie `evolutionary_algorithm.m` i uruchom funkcję.
2. **Eksperymenty**:
   - Zmień parametry algorytmu (np. wielkość populacji, liczbę pokoleń, prawdopodobieństwo mutacji) i sprawdź, jak wpływają na wynik.
3. **Analiza wyników**:
   - Obserwuj, jak maksymalne dopasowanie zmienia się w kolejnych pokoleniach.
4. **Pytania do rozważenia**:
   - Jakie są zalety reprezentacji binarnej w algorytmach ewolucyjnych?
   - Czy zmiana liczby pokoleń zwiększa dokładność algorytmu?
   - Jakie wyzwania stwarza losowość w krzyżowaniu i mutacji?

ZADANIE 2:

Zaprojektuj algorytm ewolucyjny w MATLAB, który optymalizuje funkcję celu z wykorzystaniem kodowania Graya. Studenci mają za zadanie zaimplementować algorytm ewolucyjny, uwzględniając konwersję binarnej reprezentacji na kodowanie Graya za pomocą załączonej funkcji bin2gray.

Treść zadania

Założenia

1. Algorytm będzie optymalizował funkcję celu:

   $$f(x) = x^2$$

   dla x w przedziale $[0, 63]$.

2. Rozwiązania będą reprezentowane w **kodzie Graya** dla poprawy stabilności mutacji i lepszej eksploracji przestrzeni rozwiązań.

3. Użyj funkcji `bin2gray` do konwersji reprezentacji binarnej na kod Graya.

ZADANIE 2:

Kroki do wykonania

1. **Inicjalizacja populacji**:
   - Wygeneruj populację $n = 10$ liczb z przedziału $[0, 63]$.
   - Przekonwertuj liczby na reprezentację binarną.
2. **Kodowanie w Grayu**:
   - Dla każdej binarnej reprezentacji wygeneruj kod Graya za pomocą funkcji `bin2gray`.
3. **Obliczenie dopasowania**:
   - Zamień kod Graya z powrotem na postać dziesiętną, aby obliczyć wartość funkcji celu $f(x) = x^2$.
4. **Selekcja**:
   - Wykorzystaj selekcję ruletkową lub turniejową do wyboru osobników do reprodukcji.

ZADANIE 2:

5. **Krzyżowanie**:
    - Wykonaj krzyżowanie jednopunktowe na wybranych parach rodziców w reprezentacji Graya.
6. **Mutacja**:
    - Zastosuj mutację punktową z prawdopodobieństwem $p = 0.05$.
7. **Aktualizacja populacji**:
    - Zastąp starą populację nową generacją.
8. **Iteracja**:
    - Powtórz kroki 2–7 przez 20 pokoleń.
9. **Wynik końcowy**:
    - Wyświetl najlepszego osobnika i jego wartość dopasowania po zakończeniu iteracji.

ZADANIE 2:

Instrukcje dla studentów

1. **Uruchomienie kodu**:
   - Skopiuj kod do pliku MATLAB o nazwie `evolutionary_algorithm_gray.m` i uruchom funkcję.
2. **Eksperymenty**:
   - Zmodyfikuj parametry algorytmu (np. wielkość populacji, prawdopodobieństwo mutacji) i obserwuj wpływ na wynik.
3. **Pytania do rozważenia**:
   - Jak kodowanie Graya wpływa na stabilność algorytmu w porównaniu do kodowania binarnego?
   - Jak różne metody selekcji wpływają na efektywność algorytmu?

ZADANIE 3:

Zaprojektuj algorytm ewolucyjny w MATLAB, który optymalizuje funkcję celu, uwzględniając dekodowanie kodu Graya do reprezentacji binarnej. W zadaniu należy zaimplementować algorytm wykorzystujący załączoną funkcję gray2bin, która konwertuje kod Graya na odpowiadającą mu binarną reprezentację.

Treść zadania

Założenia

1. Algorytm będzie optymalizował funkcję celu:

   $$f(x) = x^2$$

   dla x w przedziale $[0, 63]$.

2. Rozwiązania są reprezentowane w kodzie Graya, a wartości funkcji celu są obliczane po konwersji kodu Graya na binarny.

ZADANIE 3:

Kroki do wykonania

1. **Inicjalizacja populacji**:
   - Wygeneruj populację $n = 10$ liczb z przedziału $[0, 63]$.
   - Przekonwertuj liczby na reprezentację binarną, a następnie na kod Graya.
2. **Dekodowanie kodu Graya**:
   - Skorzystaj z funkcji `gray2bin`, aby konwertować kod Graya na postać binarną.
3. **Obliczenie dopasowania**:
   - Zamień binarną reprezentację na liczbę dziesiętną, aby obliczyć wartość funkcji celu $f(x) = x^2$.
4. **Selekcja**:
   - Wykorzystaj selekcję ruletkową lub turniejową do wyboru osobników do reprodukcji.

ZADANIE 3:

5. **Krzyżowanie**:
    - Wykonaj krzyżowanie jednopunktowe na wybranych parach rodziców w reprezentacji Graya.
6. **Mutacja**:
    - Zastosuj mutację punktową z prawdopodobieństwem $p = 0.05$.
7. **Aktualizacja populacji**:
    - Zastąp starą populację nową generacją.
8. **Iteracja**:
    - Powtórz kroki 2–7 przez 20 pokoleń.
9. **Wynik końcowy**:
    - Wyświetl najlepszego osobnika i jego wartość dopasowania po zakończeniu iteracji.

ZADANIE 4:

Zaprojektuj algorytm ewolucyjny w MATLAB, który optymalizuje funkcję celu w zadanym przedziale liczbowym, wykorzystując kodowanie Graya i przekształcenie kodu Graya na wartość dziesiętną w określonym przedziale [a,b]. W zadaniu należy zaimplementować algorytm z użyciem funkcji gray2decInterval, załączonej w pliku.

Treść zadania

Założenia

1. Algorytm optymalizuje funkcję celu:

   $$f(x) = x^2$$

   dla $x \in [a, b]$, gdzie $a = -4, b = 1$.

2. Liczby są reprezentowane w kodzie Graya, a ich wartości dziesiętne są obliczane w przedziale $[a, b]$ za pomocą funkcji `gray2decInterval`.

3. Długość reprezentacji binarnej to $nbits\_dv = 4$.

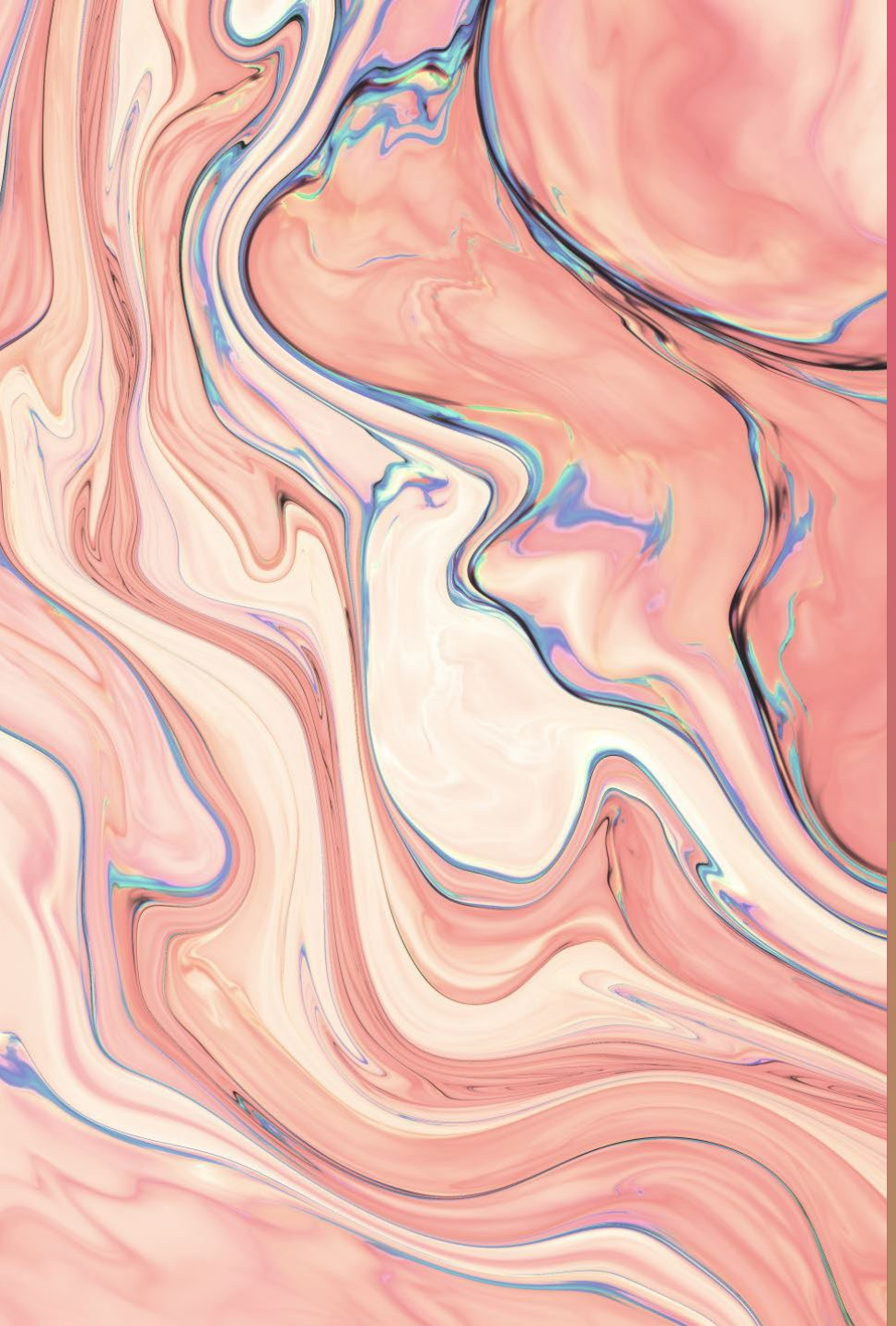
ZADANIE 4:

Kroki do wykonania

1. **Inicjalizacja populacji**:
   - Wygeneruj $n = 10$ losowych liczb w reprezentacji binarnej o długości 4 bitów.
2. **Konwersja na kodowanie Graya**:
   - Przekształć binarne liczby na kodowanie Graya.
3. **Przekształcenie na wartość dziesiętną w przedziale**:
   - Użyj funkcji `gray2decInterval` do obliczenia wartości dziesiętnych w przedziale $[a, b]$.
4. **Obliczenie dopasowania**:
   - Zastosuj funkcję celu $f(x) = x^2$ dla każdej wartości dziesiętnej.
5. **Selekcja**:
   - Wybierz rodziców za pomocą selekcji ruletkowej lub turniejowej.

ZADANIE 4:

6. **Krzyżowanie**:
    - Wykonaj krzyżowanie jednopunktowe na kodzie Graya.
7. **Mutacja**:
    - Wprowadź mutacje punktowe z prawdopodobieństwem $p = 0.05$.
8. **Aktualizacja populacji**:
    - Zastąp starą populację nową generacją.
9. **Iteracja**:
    - Powtórz kroki 2–8 przez 20 pokoleń.
10. **Wynik końcowy**:
    - Wyświetl najlepsze rozwiązanie i jego wartość funkcji celu.



ALGORYTMY EWOLUCYJNE

LABORATORIUM

DR HAB. INŻ. **RAFAŁ POROWSKI**, PROF. UJK

EMAIL: RPOROWSKI@GMAIL.COM

INSTYTUT FIZYKI UJK
(BUD. G, POK. A-109)