

Projekt 4 - Algorytm Ewolucyjny

Selekcja

```
In[1]:= selection[function_, listofpoints_, number_] := Module[{
  f = function,
  points = listofpoints,
  n = number},
  (*wartości funkcji muszą być nieujemne!*)
  sum = Sum[f[points[[i]]], {i, 1, Length[points]}];
  (*obliczanie prawdopodobieństwa dla każdego punktu*)
  probability = Table[f[points[[i]]]/sum, {i, 1, Length[points]}];
  (*wybór n elementów z podanym prawdopodobieństwem*)
  selected = RandomChoice[probability → points, n];
  Return[selected]
]
f[{x_, y_}] := 20 - (x + y);
selection[f, {{1, 1}, {-2, 3}, {-3, 2}, {0, -4}}, 10]

Out[3]= {{-2, 3}, {0, -4}, {0, -4}, {-2, 3}, {1, 1}, {0, -4}, {0, -4}, {1, 1}, {1, 1}, {0, -4}}
```

Mutacja

```
mutation[ListOfPoints_, ProbabilityOfMutation_, rangex_, rangey_] := Module[{
  pm = ProbabilityOfMutation,
  Points = ListOfPoints,
  dx = rangex,
  dy = rangey},
  temp = 0;
  mutatedTable = {};
  For[i = 1, i ≤ Length[Points], i++,
  If[Random[Real, {0, 1}] ≤ pm || temp == 1,
  new = Points[[i]] +
    {RandomReal[NormalDistribution[0, 1]], RandomReal[NormalDistribution[0, 1]]};
  (*sprawdzenie warunku czy nowy punkt należy do przedziału*)
  If[IntervalMemberQ[Interval[dx], new[[1]]] &&
    IntervalMemberQ[Interval[dy], new[[2]]], AppendTo[mutatedTable, new];
  temp = 0, i-- ;
  temp = 1 ];
  (*by zapobiec policzeniu prawdopodobieństwa mutacji poraz kolejny w sytuacji
  gdy wyjdziemy poza przedział, dodany jest warunek z 'temp'*)
  ];
  ];
  Return[mutatedTable]
]
```

In[89]:= mutation[{{1, 1}, {2, 3}, {3.2, 3.5}}, 0.9, {0, 4}, {0, 4}]

Out[89]= {{1.29677, 1.92742}, {3.18971, 3.67598}}

Krzyżowanie

Sukcesja

Algorytm Ewolucyjny

```
In[8]:= evolution[function_, numberofpopulation_, probabilityC_,
  probabilityM_, numberofbest_, rangex_, rangey_, maxIteration_] := Module[{
  f = function,
  n = numberofpopulation,
  pC = probabilityC,
  pM = probabilityM,
  k = numberofbest,
  dx = rangex,
  dy = rangey,
  maxIt = maxIteration},
  firstPopulation = {};
  (*losowanie początkowej populacji*)
  For[i = 1, i ≤ n, i++,
  AppendTo[firstPopulation, {Random[Real, dx], Random[Real, dy]}]];
  xBest = {};
  average = {};
  (*dodanie najlepszego punktu oraz wartości średniej*)
  AppendTo[xBest, SortBy[firstPopulation, f][[Length[firstPopulation]]]];
  startF := Table[f[firstPopulation[[d]], {d, 1, Length[firstPopulation]}];
  AppendTo[average, Mean[startF]];
  newPopulation = firstPopulation;
  For[r = 1, r ≤ maxIt, r++,
  TT = selection[f, newPopulation, n];
  OT = mutation[TT, pM, dx, dy];
  OTC = crossing[TT, pC, dx, dy];
  ALL = Join[OT, OTC];
  AppendTo[xBest, SortBy[ALL, f][[Length[ALL]]]];
  newF = Table[f[ALL[[d]], {d, 1, Length[ALL]}];
  AppendTo[average, Mean[newF]];
  newPopulation = succession[newPopulation, ALL, f, k];
  ];
  xBestValues = Table[f[xBest[[t]], {t, 1, Length[xBest]}];
  bestOfAllBest = SortBy[xBest, f][[Length[xBest]]];
  Return[{xBest, xBestValues, average, bestOfAllBest}]
```

```
In[9]:= f1[{x_, y_}] := 20 - x - y;
n = 10;
pC = 0.9;
pM = 0.9;
k = 3;
dx = {-3, 3};
dy = {-3, 3};
it = 7;
evolution[f1, n, pC, pM, k, dx, dy, it]
```

```
Out[17]= {{{-2.52923, -0.196956}, {-2.52923, -0.196956},
{-2.57502, -0.949624}, {-2.15737, -2.5797}, {-2.61654, -1.87863},
{-2.62953, -2.87916}, {-2.50949, -2.87675}, {-2.62141, -2.84783}},
{22.7262, 22.7262, 23.5246, 24.7371, 24.4952, 25.5087, 25.3862, 25.4692},
{19.594, 19.8403, 21.2078, 22.4852, 23.1121, 23.4487, 24.2358, 24.4888},
{-2.62953, -2.87916}}
```

Procedura wyświetlania

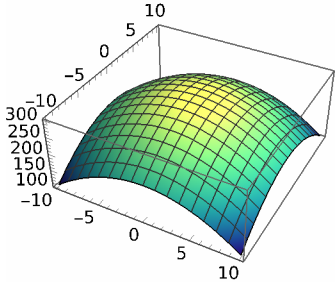
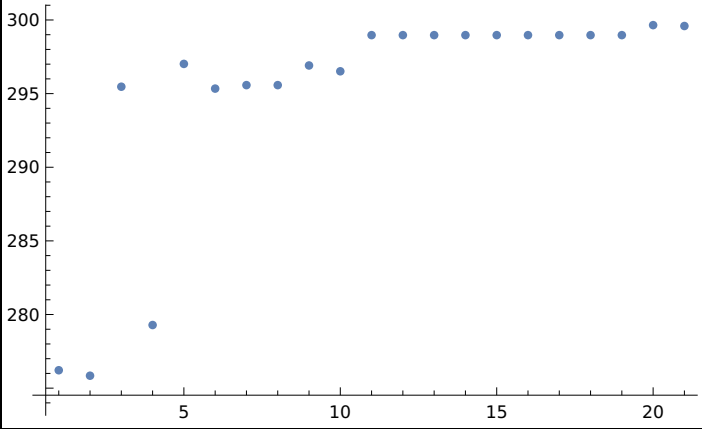
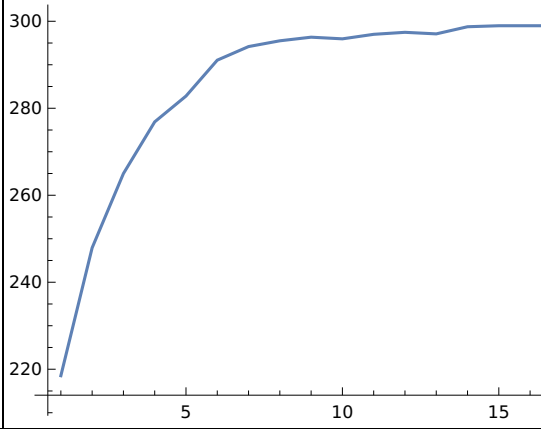
Przykład 1

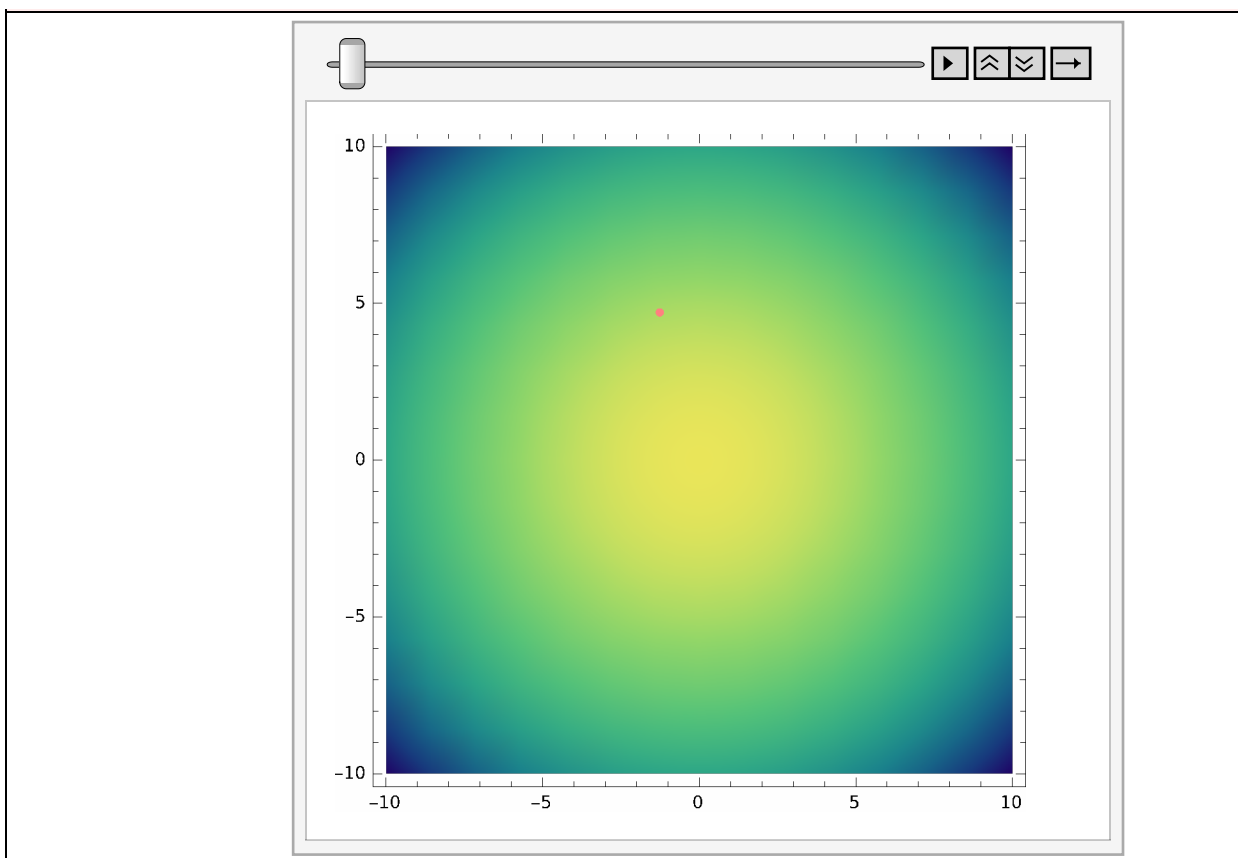
Przykład 2

```
In[134]:= Clear[n, pC, pM, k, dx, dy, it, solution]
SphericalFunction[{x_, y_}] := 300 - (x ^ 2 + y ^ 2);
n = 10;
pC = 0.7;
pM = 0.05;
k = 3;
dx = {-10, 10};
dy = {-10, 10};
it = 20;
solution = evolution[SphericalFunction, n, pC, pM, k, dx, dy, it];
Drawing[SphericalFunction, solution, dx, dy]
```

Out[144]=

ALGORYTM EWOLUCYJNY	
Wzór funkcji	Wykres funkcji

$300 - x^2 - y^2$	
Przebieg wartości maksymalnej w zależności od iteracji	Przebieg wartości średniej w zależności od ite
	
Najlepsza znaleziona wartość	Punkt, w którym znaleziono mak
299.65	$\{-0.576793, 0.131866\}$
Przebieg poszukiwania rozwiązania	



In[264]:=

Przykład 3

Wnioski

Zwiększenie ilości losowanych punktów zwiększa prawdopodobieństwo uzyskiwania coraz to dobrych wyników już w pierwszej.

Procedura selekcji zapewnia większe prawdopodobieństwo na wylosowanie z listy większej ilości najlepszych punktów, ponieważ im większa wartość funkcji tym większe prawdopodobieństwo wyboru danego punktu.

By zapobiec problemom w sukcesji związanymi ze zbyt małą licznością pokolenia potomnego, osobno na danym pokoleniu została wykonana mutacja i osobno krzyżowanie.

Mutacja zwraca tylko wartości zmutowane. Gdyby zwracała punkty zmutowane jak i niezmutowane to istniałaby większa szansa na to, że w kolejnym pokoleniu po zadziałaniu sukcesji ilość pewnego najlepszego punktu (a,b) zwiększałaby się wraz z każdą iteracją.

Jednakże nie jesteśmy w stanie zapobiec temu w 100% ponieważ selekcja może wylosować nam większą liczbę tego samego punktu, które po skrzyżowaniu nadal mogą dać nam ten sam punkt.

By pokolenie potomne skrzyżowane było dostatecznie duże jako rodzice zostały przyjęte pary $(2i+1, 2i+2)$ oraz $(2i+2, 2i+1)$. Krzyżowanie zachodzi z lewej i prawej strony.

Program w każdym przedstawionym przykładzie trafia bardzo blisko maksimum globalnego, co świadczy o tym że algorytm działa bardzo dobrze. Średnia wartość rośnie ponieważ wraz z każdą iteracją pokolenie potomne jest coraz to lepsze i modyfikacje są wykonywane na coraz to lepszych pokoleniach.