Gliwice, 01.01.2025

Mathematical modeling and computer simulations in theory and practice

Documentation of laboratory task no 6

Title: **Simple Local Search**

Author: Aleksandra Stachecka
Field of studies: Informatics sem.V

1 Project Objective

The objective of the project is to find a local minimum of a given function within a specified 2D domain using a simple random search technique. This is done through an iterative process where random points are generated and evaluated.The program visualizes the progression of the search path and the function's contour plot.

2 Description

2.1 Program

The program consists of a function `search` taking the following inputs:

- **f_**: The objective function to minimize. This function takes two arguments: x and y, representing the coordinates in the 2D domain.
- **domain_**: The 2D domain within which the function will be minimized. This is specified as list of pairs of values: $xmin, xmax, ymin, ymax$.
- **radius_**: Value determining the maximum distance a new point can move from the current point in each iteration.
- **iterations_**: The number of iterations to perform in the random search.

The first step is to define the function which will generate random points within the domain and generate the initial random point. Then we initialize a variable *path* to keep track of the search path, starting with the initial point.

```
randomPoint[{xmin_, xmax_}, {ymin_,
   ymax_}] := {RandomReal[{xmin, xmax}], RandomReal[{ymin,
       ymax}]};

currentPoint = randomPoint @@ domain;
path = {currentPoint};
```

After that, we can enter the loop in which we will generate the points and potentially add them to the search path. We generate a new random point near the current point, then we clip the new point to ensure that the newly generated point stays within the specified domain. We check whether the new point improves the function value compared to the current point. If it does, the current point is updated to the new point and it is added to the path. We repeat the loop for the specified number of iterations. After completing the search loop, we assign the current point to $finalPoint$, which is the result of the search process.

```
Do[
  newPoint = currentPoint + RandomReal[{-radius, radius},
     2];
  newPoint = {Clip[newPoint[[1]], domain[[1]]],
    Clip[newPoint[[2]], domain[[2]]]};

  If[f @@ newPoint < f @@ currentPoint, currentPoint =
     newPoint];

  AppendTo[path, currentPoint];, {iterations}];

finalPoint = currentPoint;
```

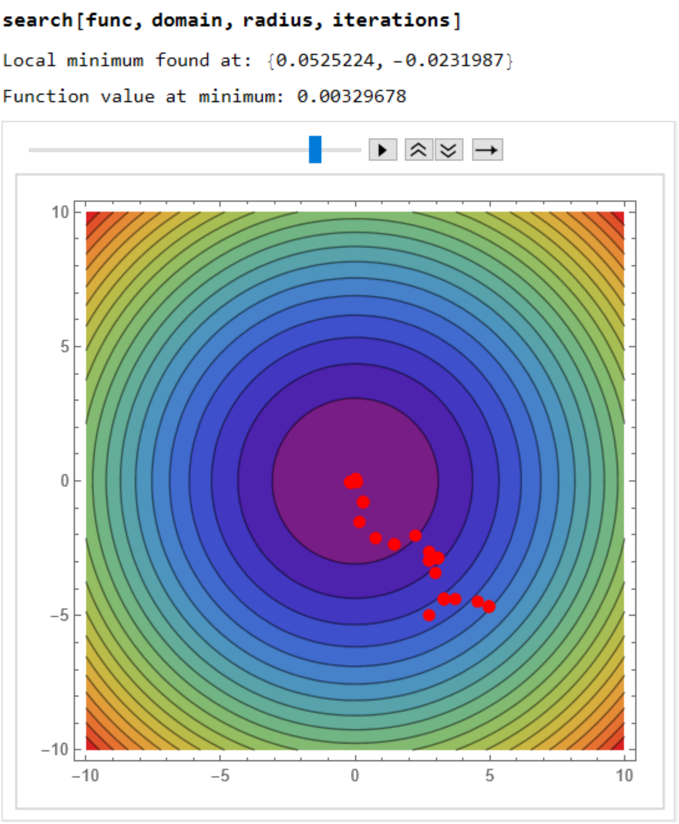
After the search is completed, we can visualize the results. We create a contour plot of the function $f(x, y)$ over the specified domain, then we create a table of frames for the animation, where each frame shows the contour plot with the search path up to the i-th point in the path list. We create an animation from the created table, then we display the animation.

```
cPlot = ContourPlot[
   f[x, y], {x, domain[[1, 1]], domain[[1, 2]]}, {y, domain
       [[2, 1]],
    domain[[2, 2]]},
   Contours -> 20,
   ColorFunction -> "Rainbow",
   PlotLegends -> None,
   PlotRange -> All];

points = Table[
   Show[
    cPlot,
    Graphics[{Red, PointSize[0.02], Point[path[[1 ;; i]]]}],
    PlotRange -> All],
   {i, 1, Length[path]}];

animationFrames = ListAnimate[points];

animationFrames
```

2.2 Example of usage, visualization

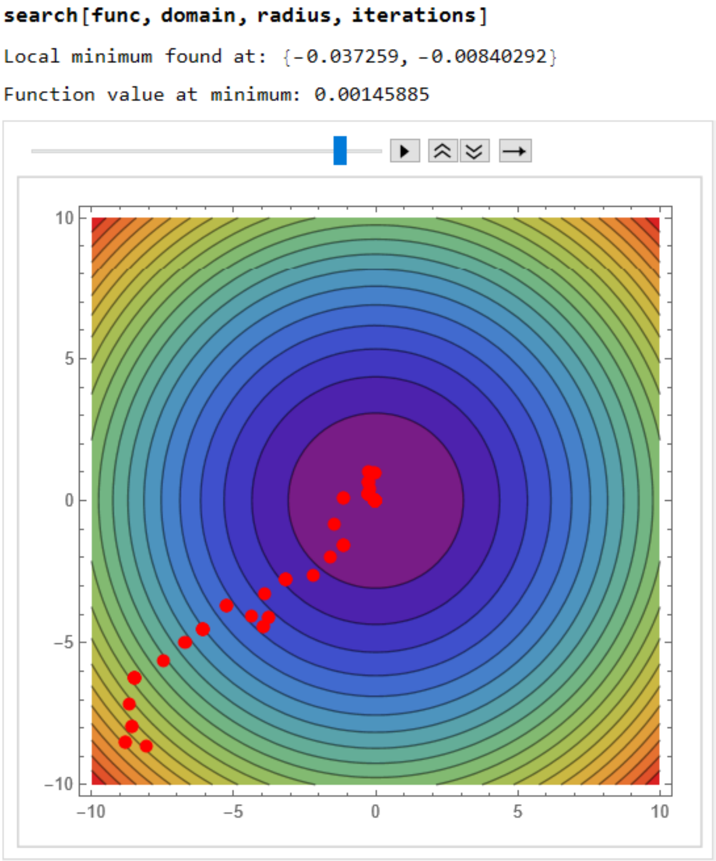
We will test the program using the following values:

```
func[x_, y_] := x^2 + y^2;

domain = {{-10, 10}, {-10, 10}};
radius = 1;
iterations = 100;
```



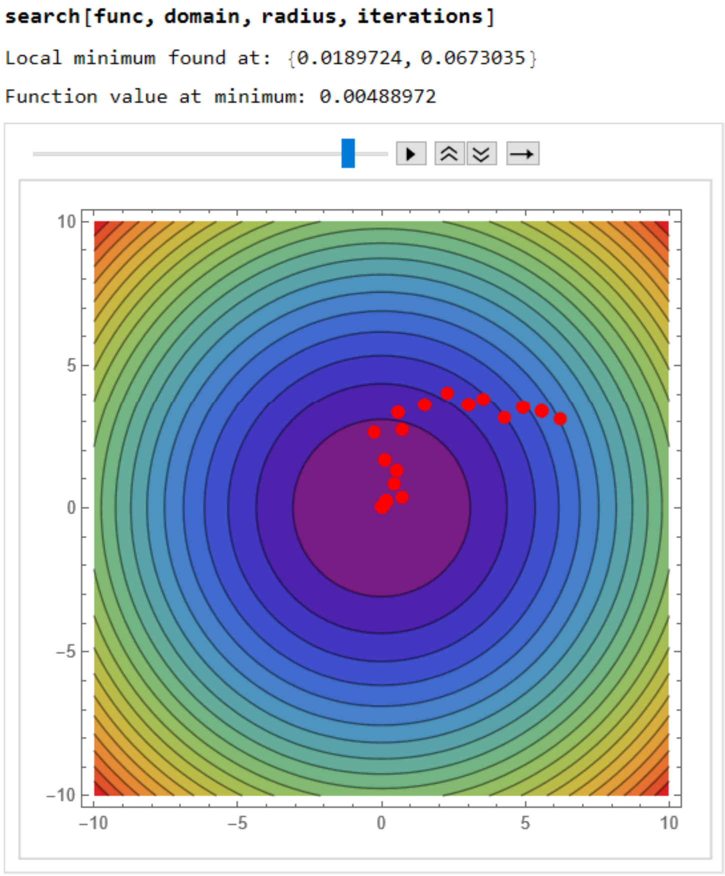
```
search[func, domain, radius, iterations]
Local minimum found at: {0.0525224, -0.0231987}
Function value at minimum: 0.00329678
```

Figure 1: Search 1



```
search[func, domain, radius, iterations]
Local minimum found at: {-0.037259, -0.00840292}
Function value at minimum: 0.00145885
```

Figure 2: Search 2



```
search[func, domain, radius, iterations]
Local minimum found at: {0.0189724, 0.0673035}
Function value at minimum: 0.00488972
```

Figure 3: Search 3

3 Enclosures

”STACHECKA Aleksandra Project 6.nb”