The Simplex Method in Global Search

1 Introduction

The Simplex method, also known as the Nelder-Mead method, is a popular technique used in global search optimization. It is particularly effective for multidimensional unconstrained optimization problems where the objective function is continuous but not necessarily differentiable.

2 How the Simplex Method Works

The Simplex method operates by maintaining a simplex, which is a geometric figure consisting of $n+1$ vertices in n dimensions. For a two-dimensional problem, the simplex is a triangle. The algorithm iteratively updates this simplex to move towards the optimum.

3 Mathematical Formulation

Let x_i be the i-th vertex of the simplex, and $f(x_i)$ be the objective function value at x_i. The vertices are ordered such that $f(x_1) \leq f(x_2) \leq ... \leq f(x_{n+1})$.

The centroid $\bar{x}$ of all points except the worst is calculated as:

$$\bar{x} = \frac{1}{n} \sum_{i=1}^{n} x_i$$

The reflection point x_r is given by:

$$x_r = \bar{x} + \alpha(\bar{x} - x_{n+1})$$

where α is the reflection coefficient (typically 1).
The expansion point x_e is:

$$x_e = \bar{x} + \gamma(x_r - \bar{x})$$

where γ is the expansion coefficient (typically 2).
The contraction point x_c is:

$$x_c = \bar{x} + \rho(x_{n+1} - \bar{x})$$

where ρ is the contraction coefficient (typically 0.5).
For shrinkage, all points are updated as:

$$x_i = x_1 + \sigma(x_i - x_1)$$

where σ is the shrinkage coefficient (typically 0.5).
The conditions for applying these operations are:

- Reflection: Always performed first.
- Expansion: If $f(x_r) < f(x_1)$, i.e., if the reflected point is better than the best point.
- Outside Contraction: If $f(x_n) \leq f(x_r) < f(x_{n+1})$, i.e., if the reflected point is between the second worst and worst point.
- Inside Contraction: If $f(x_r) \geq f(x_{n+1})$, i.e., if the reflected point is worse than the worst point.
- Shrinkage: If both outside and inside contractions fail to improve the worst point.

4 Example Implementation

Consider the following optimization problem:

$$\begin{aligned} \text{minimize} \quad & f(x) = (x_2 - x_1)^4 + 12x_1x_2 - x_1 + x_2 - 3 \\ \text{subject to} \quad & -1 \leq x_1, x_2 \leq 1 \end{aligned}$$

The MATLAB code implementing this optimization using the Simplex method is as follows:

```
% Objective function
f = @(x) (x(2) - x(1))^4 + 12 * x(1) * x(2) - x(1) + x(2) - 3;

% Initial simplex points
initial_points = [0.55, 0.7; -0.9, -0.5];

% Define the domain
lb = [-1, -1];
ub = [1, 1];

% Create a figure for plotting
figure;
hold on;
% Create a mesh grid and calculate function values for contour plot
[X1, X2] = meshgrid(linspace(-1, 1, 100), linspace(-1, 1, 100));
F = arrayfun(@(x1, x2) f([x1, x2]), X1, X2);
contour(X1, X2, F, 50); % Plot contour

colors = ['r', 'b']; % Different colors for different starting points

for i = 1:2
    % Starting point
    x0 = initial_points(i, :);

    % Initialize the points array for plotting the path
    points = x0;

    % Optimization using fminsearch (Nelder-Mead algorithm)
    options = optimset('Display', 'iter', 'OutputFcn', @(x, optimValues, state) outfunc(
    x, state, colors(i)));
    [x_opt, fval] = fminsearch(@(x) constrained_function(x, f, lb, ub), x0, options);

    fprintf('Starting point: [%f, %f]\n', x0(1), x0(2));
    fprintf('Optimized point: [%f, %f] with function value %f\n', x_opt(1), x_opt(2),
    fval);
end

hold off;

% Constrained function to enforce domain constraints
function y = constrained_function(x, f, lb, ub)
    x = max(min(x, ub), lb);
    y = f(x);
end

% Output function to capture and plot points
function stop = outfunc(x, state, color)
    stop = false;
    persistent points;

    if isequal(state, 'init')
        points = x; % Initialize points with the starting point
    elseif isequal(state, 'iter')
        points(end+1, :) = x; % Add new point to points array
        plot(points(:,1), points(:,2), [color, '-o']); % Plot points with lines
        drawnow;
    elseif isequal(state, 'done')
        % Final plot for the complete path
        plot(points(:,1), points(:,2), [color, '-o'], 'LineWidth', 2);
    end
end
```

5 Results

The optimization was run from two different starting points. Here are the results:

5.1 Starting Point 1: [0.550000, 0.700000]

```
Optimized point: [0.650421, -0.650392] with function value -6.513905
```

5.2 Starting Point 2: [-0.900000, -0.500000]

```
Optimized point: [0.650423, -0.650413] with function value -6.513905
```

Both starting points converged to essentially the same solution, indicating a likely global minimum.

6 Conclusion

The Simplex method successfully found the minimum of the given function within the specified constraints. The method's ability to converge to the same solution from different starting points demonstrates its robustness in finding global optima for this particular problem.

The contour plot generated by the code provides a visual representation of the function landscape and the optimization paths, which can be valuable for understanding the behavior of the algorithm and the nature of the objective function.

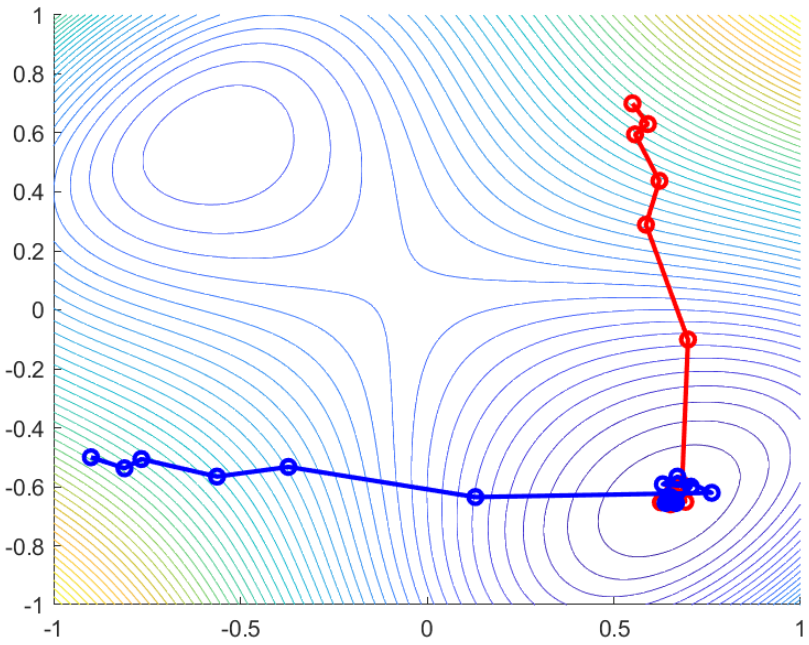

Figure 1: Simplex search method applied to minimization of a function.