Computational Approach to Function Minimization and Optimization with Constraints

Rohan S. Shrama
Iowa State University of Science and Technology
Symposium on Undergraduate Research & Creative Expression, April 15 2014, Ames, IA.
Presentation Overview

1. Introduction
2. Testing Functions
3. MATLAB Optimization Tools
4. Tested Functions
5. \( f(x) = -5x_1 - 4x_2 - 6x_3 \)
6. \( f(x) = \frac{1}{2}x_1^2 + x_2^2 - x_1x_2 - 2x_1 - 6x_2 \)
7. Ackley’s Function
8. Cross-in-tray Function
9. Conclusion
10. Future Plans
11. Acknowledgements
Introduction

- Various methods of optimization one can employ in MATLAB.
- Multiple functions of various types are selected and each optimization process is rigorously tested.
- To determine the best optimizer to be used for a set type of function.
  - MATLAB Environment
  - Processes tested for accuracy and computation time
- These processes are then used to test the function minimization process of typical optimization testing functions.
Common Benchmarks:

1. **Unimodal, convex, multidimensional, and continuous.**
   - Can cause slow or poor convergence to a single global extremum.

2. **Multimodal, two-dimensional with a small number of local extremes, and continuous.**
   - Functions of this type are employed to test the quality of an optimization tool or process.
   - Typically have few local extremes with a single global extreme.

3. **Multimodal, two-dimensional with a huge number of local extremes, and continuous.**
   - Higher number of local extremes.
   - Further test the quality of the tool used.

4. **Multimodal, multi-dimensional with a large number of local extremes, and continuous.**
   - Typical functions to appear in actual practice.
MATLAB Optimization Tools

- quadprog – quadratic programming
- lsqcurvefit – solved nonlinear curve-fitting (data-fitting) problems in least-squares sense
- lsqnonlin – Solve nonlinear least-squares (nonlinear data-fitting) problems
- fminsearch – Find minimum of unconstrained multivariable function using derivative-free method
- fminunc – Find minimum of unconstrained multivariable function
- linprog – Solve linear programming problems
- lsqlin – Solve constrained linear least-squares problems
- lsqnonlin – Solve nonlinear least-squares (nonlinear data-fitting) problems
- lsqnonneg – Solve nonnegative least-squares constraint problem
- fminbnd – Find minimum of single-variable function on fixed interval
- fmincon – Find minimum of constrained nonlinear multivariable function
- fseminf – Find minimum of semi-infinitely constrained multivariable nonlinear function
- bintprog – Solve binary integer programming problems
Tested Functions

1. \( f(x) = -5x_1 - 4x_2 - 6x_3 \)

2. \( f(x) = \frac{1}{2} x_1^2 + x_2^2 - x_1x_2 - 2x_1 - 6x_2 \)

3. \( \sum_{k=1}^{10} (2 + 2k - e^{kx_1} - e^{kx_2})^2 \) starting at the point \( x = (0.3, 0.4) \)

4. \( f(x) = 3x_1^2 + 2x_1x_2 + x_2^2 \)

5. \( f(x) = -x_1x_2x_3 \) starting at \( x = (10, 10, 10) \)

6. Ackley’s Function

7. Bukin Function No. 6

8. Three-hump camel function

9. Easom function

10. Holder table function

11. Cross-in-tray function
\( f(x) = -5x_1 - 4x_2 - 6x_3 \)

Subject to the following:

- \( x_1 - x_2 + x_3 \leq 20 \)
- \( 3x_1 + 2x_2 \leq 30 \)
- \( 0 \leq x_2 \)
- \( 3x_1 + 2x_2 + 4x_3 \leq 42 \)
- \( 0 \leq x_1 \)
- \( 0 \leq x_3 \)
\[ f(x) = -5x_1 - 4x_2 - 6x_3 \]

Utilization of linprog:

```matlab
Optimization terminated. 
Elapsed time is 0.014166 seconds.

x =

  0.0000
  15.0000
  3.0000

ans =

  0.0000
  1.5000
  0.5000

ans =

  1.0000
  0.0000
  0.0000

f_x >>
```
\[ f(x) = -5x_1 - 4x_2 - 6x_3 \]

Utilization of lsqnonlin:

```matlab
>> leastsquaresnonlin
Warning: Trust-region-reflective algorithm requires at least as many equations as variables; using Levenberg-Marquardt algorithm instead.
> In lsqnonlin at 56
  In lsqnonlin at 237
  In leastsquaresnonlin at 4

Local minimum found.

Optimization completed because the size of the gradient is less than the default value of the function tolerance.

<stopping criteria details>

Elapsed time is 0.008820 seconds.
>> x

x =

-5.0649     10.9481    -3.0779

f_x >>
```
Utilization of fminsearch:

\[ f(x) = -5x_1 - 4x_2 - 6x_3 \]
\[ f(x) = -5x_1 - 4x_2 - 6x_3 \]

Utilization of fminunc:

```
>> minunc
Warning: Gradient must be provided for trust-region algorithm; using line-search algorithm instead.
> In fminunc at 383
In minunc at 3

Problem appears unbounded.

fminunc stopped because the objective function value is less than or equal to the default value of the objective function limit.

<stopping criteria details>

Elapsed time is 0.024999 seconds.
>> x
x =

1.0e+19 *

1.1398    0.9118    1.3677
```

\[ f(x) \]
Utilization of fmincon:

\[ f(x) = -5x_1 - 4x_2 - 6x_3 \]

```matlab
>> mincon
Warning: The default trust-region-reflective algorithm does not solve problems with the constraints you have specified. FMINCON will use the active-set algorithm instead. For information on applicable algorithms, see Choosing the Algorithm in the documentation.
> In fmincon at 501
  In mincon at 5
Warning: Your current settings will run a different algorithm (interior-point) in a future release.
> In fmincon at 506
  In mincon at 5
Local minimum found that satisfies the constraints.

Optimization completed because the objective function is non-decreasing in feasible directions, to within the default value of the function tolerance, and constraints are satisfied to within the default value of the constraint tolerance.

<stopping criteria details>

Active inequalities (to within options.TolCon = 1e-06):
  lower       upper       ineqlin       ineqnonlin
        2
        3
        4

Elapsed time is 0.021024 seconds.
>> x

x =

0.0000    15.0000    3.0000

f(x) >> |
\[ f(x) = -5x_1 - 4x_2 - 6x_3 \]

Evaluation:

- Linear equation
- `linprog` optimization tool yielded the accurate result in the least time possible:

\[ x_1 = 0 \quad x_2 = 15 \quad x_3 = 3 \]

Time Elapsed = 0.014165 seconds
Subject to the following:

- \( x_1 + x_2 \leq 2 \)
- \( -x_1 + 2x_2 \leq 2 \)
- \( 2x_1 + x_2 \leq 3 \)
- \( 0 \leq x_1 \)
- \( 0 \leq x_2 \)

- Polynomial function with quadratic roots.
Method 1 solution:

- The algorithm ‘active-set’ was used to optimize.
- The ‘active-set’ can take large steps, which decreases the computational time of the program.
- The algorithm is effective on some problems with non-smooth constraints. It is not a large-scale algorithm.

\[
f(x) = \frac{1}{2} x_1^2 + x_2^2 - x_1x_2 - 2x_1 - 6x_2
\]
Method 2 solution:

- The algorithm 'interior-point-convex' was used to compare its operation and effectiveness with the previously used algorithm 'active-set'. 'interior-point-convex' handles large, sparse, complex problems as well as small dense problems. The algorithm satisfies bounds at all iterations and can recover from NaN or Inf results.

\[
f(x) = \frac{1}{2}x_1^2 + x_2^2 - x_1x_2 - 2x_1 - 6x_2
\]

<table>
<thead>
<tr>
<th>Iter</th>
<th>(f(x))</th>
<th>Feasibility</th>
<th>optimality</th>
</tr>
</thead>
<tbody>
<tr>
<td>0</td>
<td>-2.000000e+00</td>
<td>1.000e+01</td>
<td>4.500e+00</td>
</tr>
<tr>
<td>1</td>
<td>-2.630486e+01</td>
<td>0.000e+00</td>
<td>9.465e-01</td>
</tr>
<tr>
<td>2</td>
<td>-2.639877e+01</td>
<td>0.000e+00</td>
<td>3.914e-01</td>
</tr>
<tr>
<td>3</td>
<td>-2.639881e+01</td>
<td>0.000e+00</td>
<td>3.069e-12</td>
</tr>
</tbody>
</table>

Minimum found that satisfies the constraints.

Optimization completed because the objective function has feasible directions, to within the default value, and constraints are satisfied to within the default value.

<stopping criteria details>

Elapsed time is 0.010300 seconds.

fval =

-26.3988

eflag =

1
Utilization of lsqnonlin to optimize:

\[ f(x) = \frac{1}{2} x_1^2 + x_2^2 - x_1x_2 - 2x_1 - 6x_2 \]

```matlab
>> leastsquarenonlin
Warning: Trust-region-reflective algorithm requires at least as many equations as variables; using
Levenberg-Marquardt algorithm instead.
> In leastsquarenonlin at 56
  In lsqnonlin at 237
  In leastsquarenonlin at 4

Local minimum found.

Optimization completed because the size of the gradient is less than
the default value of the function tolerance.

<stopping criteria details>

Elapsed time is 0.024857 seconds.

x =
  -0.0662   0.0228

f_x >>
```
Utilization of fminsearch to optimize:

\[ f(x) = \frac{1}{2} x_1^2 + x_2^2 - x_1 x_2 - 2x_1 - 6x_2 \]
Utilization of fminunc to optimize:

```matlab
>> minunc
Warning: Gradient must be provided for trust-region algorithm;
    using line-search algorithm instead.
> In fminunc at 383
  In minunc at 3

Local minimum found.

Optimization completed because the size of the gradient is less than the default value of the function tolerance.

<stopping criteria details>

x =

  10.0000  8.0000

Elapsed time is 0.018594 seconds.
fopt
```
Utilization of fmincon to optimize:

\[ f(x) = \frac{1}{2} x_1^2 + x_2^2 - x_1x_2 - 2x_1 - 6x_2 \]
\[ f(x) = \frac{1}{2} x_1^2 + x_2^2 - x_1 x_2 - 2x_1 - 6x_2 \]

Evaluation:
- Polynomial function with quadratic roots.
- quadprog optimization toolset yielded the most accurate result in the shortest time possible.

- \( x_1 = 0.6667 \quad x_2 = 1.333 \)
- \( Eigenvalues \text{ of matrix } H = 0.3820 \quad \& \quad 2.6180 \)
- Time Elapsed = 0.005229 seconds
Ackley’s Function

N-dimensional multimodal function which has a large number of local minima but only has one global minimum.

1. Set a relevant domain
2. Converging on the minimum value

\[
f(x, y) = -20 \exp \left( -0.2\sqrt{0.5(x^2 + y^2)} \right) - \exp \left( 0.5(Cos(2\pi x) + Cos(2\pi y)) \right) + 20 + e
\]

- Subject to the following: \(-5 \leq x, y \leq 5\)
Ackley’s Function

Solution (0,0)
Ackley’s Function

Changing domain to [-1,1] we see the following convergence on the global minimum.
Cross-in-tray Function

\[ f(x, y) = -0.0001 \left( \left| \sin(x) \sin(y) \exp \left( \left| 100 - \frac{\sqrt{x^2 + y^2}}{\pi} \right| \right) \right| + 1 \right)^{0.1}. \]

- Has multiple global minima but four distinct global minima.
- Function is typically evaluated on the square \( x \in [-10, 10] \) and \( y \in [-10, 10] \).
- The global minima are as follows:
  1. \((1.3491, -1.3491)\)
  2. \((1.3491, 1.3491)\)
  3. \((-1.3491, 1.3491)\)
  4. \((-1.3491, -1.3491)\)
Cross-in-tray Function

Subject to the following $-10 \leq x, y \leq 10$
Cross-in-tray Function

Subject to the following $-2 \leq x, y \leq 2$
Conclusion

Judging from the computation time and the accuracy of each optimization algorithm a table has been prepared depicting the type of optimization to be employed given a particular type of constraint and function.

<table>
<thead>
<tr>
<th>Constraint Type</th>
<th>Objective Type</th>
<th>Linear</th>
<th>Quadratic</th>
<th>Least Squares</th>
<th>Smooth nonlinear</th>
<th>Non-smooth</th>
</tr>
</thead>
<tbody>
<tr>
<td>None</td>
<td>N/A</td>
<td>quadprog</td>
<td>lsqcurvefit, lsqnonlin</td>
<td>fminsearch, fminunc</td>
<td>fminsearch</td>
<td></td>
</tr>
<tr>
<td>Bound</td>
<td>linprog</td>
<td>quadprog</td>
<td>lsqcurvefit, lsqlin, lsqnonlin, lsqnoneg</td>
<td>fminbnd, fmincon, fseminf</td>
<td>fminbnd</td>
<td></td>
</tr>
<tr>
<td>Linear</td>
<td>linprog</td>
<td>quadprog</td>
<td>lsqlin</td>
<td>fmincon, fseminf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>General Smooth</td>
<td>fmincon</td>
<td>fmincon</td>
<td>fmincon</td>
<td>fmincon, fseminf</td>
<td></td>
<td></td>
</tr>
<tr>
<td>Discrete</td>
<td>bintprog</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td>N/A</td>
<td></td>
</tr>
</tbody>
</table>
Future Plans

- Research submitted to the Journal of Optimization
- Continue research modifying developed optimization toolsets
- Currently working on a nozzle optimization tool.
- Develop a Wing Planform Exploration Tool (PET)
  - Aerodynamic Analysis
  - Structural Analysis
  - Cost
  - Ease of manufacturing
Acknowledgements

- Professor Ambar Mitra
- Iowa State University, Aerospace Engineering Department
- MATLAB – The Language of Technical Computing Developers