Optimization in MATLAB

Seth DeLand
Topics

- Intro
- Using gradient-based solvers
- Optimization in Comp. Finance toolboxes
- Global optimization
- Speeding up your optimizations
Optimization workflow

1. Initial Design Variables
2. Modify Design Variables
3. Model
4. Objectives met?
   - Yes: Optimal Design
   - No: Repeat step 2
Optimization Problems

Maximize Fuel Efficiency

Minimize Risk

Maximize Profits
Demo: Solving an optimization problem

- Simple objective function of 2 variables:

\[
f(x) = \log \left( 1 + 3 \left( x_2 - \left( x_1^3 - x_1 \right) \right)^2 + \left( x_1 - \frac{4}{3} \right)^2 \right)
\]

- Bound constraints:

\[-2.5 \leq x_1 \leq 2.5\]
\[-2.5 \leq x_2 \leq 2.5\]
Portfolio Optimization – Quadratic Constraints

- Maximize returns, constraint is risk
  \[
  \text{minimize} \ - f^T x \quad \text{subject to} \quad \frac{1}{2} x^T C x \leq C_{\text{max}}
  \]

- Use \texttt{fmincon} interior-point algorithm
  - Specify analytic gradients of objective function and constraints
    - \( g = -f \) (gradient of objective function)
    - \( gc = Cx \) (gradient of constraint)
  - Hessian (H) of Lagrangian
    \[
    L = -f^T x + \lambda \left( \frac{1}{2} x^T C x - C_{\text{max}} \right)
    \]
    \[
    H = \nabla^2 L = \lambda C
    \]

This is a second order conic programming problem
Demo: Cash-flow matching

Idea: Buy bonds to cover pension fund obligations

Variables: How many of each bond to buy?

Constraints: Payments from bonds must be greater than or equal to pension fund obligations

Objective: Minimize the size of the investment you make
Extending the problem

\[
\begin{align*}
x: & \quad \begin{bmatrix} n_1 \\ \vdots \\ n_5 \end{bmatrix} \\
\end{align*}
\]

\[
\begin{align*}
n_1 &= 1000y_1 \\
\vdots \\
n_5 &= 1000y_5 \\
\end{align*}
\]

\[
\begin{bmatrix} 1 & 0 & -1000 & \vdots & 0 \\
0 & 1 & 0 & \vdots & -1000 \\
\end{bmatrix}
\begin{bmatrix} n_1 \\ \vdots \\ n_5 \\ y_1 \\ \vdots \\ y_5 \end{bmatrix} = \begin{bmatrix} 0 \\ \vdots \\ 0 \end{bmatrix}
\]

\[
\begin{align*}
[-\text{cashFlows}] & \begin{bmatrix} n_1 \\ \vdots \\ n_5 \end{bmatrix} \leq \{-\text{obligations}\}
\end{align*}
\]

\[
\begin{align*}
[-\text{cashFlows} & \text{zeros}(8,5)]
\begin{bmatrix} n_1 \\ \vdots \\ n_5 \\ y_1 \\ \vdots \\ y_5 \end{bmatrix} \leq \{-\text{obligations}\}
\end{align*}
\]
Optimization Toolbox solvers

Linear and Mixed-Integer
- LINPROG
- INTLINPROG

Quadratic
- QUADPROG

Nonlinear
- FMINCON
- FMINUNC
- FMINBND
- FMINSEARCH
- FSEMINF

Least Squares
- LSQLIN
- LSQNONNEG
- LSQCURVEFIT
- LSQNONLIN

Nonlinear Equation Solving
- FSOLVE
- FZERO

Multiobjective
- FGOALATTAIN
- FMINIMAX

2x₁ - x₂ = e⁻ˣ₁
-x₁ + 2x₂ = e⁻ˣ₂.
Key optimization problems addressed by financial toolboxes

- **Financial Toolbox**
  - Mean-variance portfolio optimization
  - Conditional Value at Risk (CVaR) portfolio optimization

- **Econometrics Toolbox**
  - Parameter estimation of conditional mean/variance models

- **Financial Instruments Toolbox**
  - Hedging portfolios
  - Fitting interest rate curves
  - Bootstrapping
Global optimization algorithms

- MultiStart
- GlobalSearch
- Patternsearch
- Genetic Algorithm
- Simulated Annealing
What is MultiStart?

- Run a local solver from each set of start points
- Option to filter starting points based on feasibility
- Supports parallel computing
MultiStart demo – nonlinear regression

**lsqcurvefit solution**

- Starting Value
  - b(1): 1.0000
  - b(2): 1.0000
  - b(3): 1.0000
  - b(4): 0.2500
  - b(5): 0.2500
  - b(6): 0.2500

- Fitted Value
  - b(1): 0.1836
  - b(2): 0.1849
  - b(3): 0.1833
  - b(4): 0.0171
  - b(5): 0.0171
  - b(6): 0.0171

- MSE = 2.2198e-01

**MultiStart solution**

- Starting Value
  - b(1): 3.8968
  - b(2): -4.2832
  - b(3): -7.0852
  - b(4): 0.0845
  - b(5): 0.3014
  - b(6): 0.3720

- Fitted Value
  - b(1): 0.3453
  - b(2): 1.7269
  - b(3): -4.3074
  - b(4): 0.0146
  - b(5): 0.1031
  - b(6): 0.3025

- MSE = 1.4548e-05
What is Pattern Search?

- An approach that uses a pattern of search directions around the existing points
- Expands/contracts around the current point when a solution is not found
- Does not rely on gradients: works on smooth and nonsmooth problems
Pattern Search overview – Iteration 1

*Run from specified x0*
Pattern Search overview – Iteration 1

Apply pattern vector, poll new points for improvement

Mesh size = 1
Pattern vectors = [1,0], [0,1], [-1,0], [0,-1]

\[ P_{\text{new}} = \text{mesh\_size} \times \text{pattern\_vector} + x_0 \]

\[ = 1 \times [-1,0] + x_0 \]

First poll successful

Complete Poll (not default)
Pattern Search overview – Iteration 2

Mesh size = 2
Pattern vectors = [1,0], [0,1], [-1,0], [0,-1]
Pattern Search overview – Iteration 3

Mesh size = 4
Pattern vectors = [1,0], [0,1], [-1,0], [0,-1]
Pattern Search overview – Iteration 4

Mesh size = 4*0.5 = 2
Pattern vectors = [1,0], [0,1], [-1,0], [0,-1]
Pattern Search overview – Iteration N

Continue expansion/contraction until convergence…

![Graph showing pattern search iteration](image)
Patternsearch demo – stochastic function

- Stochastic objective function

```
>> Objfcn([1,1])
an =
  6.2937
```

```
>> Objfcn([1,1])
an =
  0.1312
```

```
>> Objfcn([1,1])
an =
  4.8071
```
What is a Genetic Algorithm?

- Uses concepts from *evolutionary biology*

- Start with an initial generation of candidate solutions that are tested against the objective function

- Subsequent generations evolve from the 1st through *selection*, *crossover* and *mutation*
How evolution works – binary case

- **Selection**
  - *Retain* the best performing bit strings from one generation to the next. *Favor these for reproduction*
  - parent1 = \[1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\]
  - parent2 = \[1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\]

- **Crossover**
  - parent1 = \[1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\]
  - parent2 = \[1\ 0\ 0\ 1\ 0\ 0\ 1\ 0\ 1\ 0\]
  - child = \[1\ 0\ 0\ 0\ 0\ 1\ 1\ 0\ 1\ 0\]

- **Mutation**
  - parent = \[1\ 0\ 1\ 0\ 0\ 1\ 1\ 0\ 0\ 0\]
  - child = \[0\ 1\ 0\ 1\ 0\ 1\ 0\ 0\ 0\ 1\]
Genetic Algorithm – Iteration 1

Evaluate initial population
Genetic Algorithm – Iteration 1

Select a few good solutions for reproduction
Genetic Algorithm – Iteration 2

*Generate new population and evaluate*
Genetic Algorithm – Iteration 2

![Genetic Algorithm Diagram](Image)
Genetic Algorithm – Iteration 3
Genetic Algorithm – Iteration 3
Genetic Algorithm – Iteration N
Continue process until stopping criteria are met
Genetic Algorithm demo - Multiobjective

- Use `gamultiobj` to find Pareto front

- Two competing objectives:

  \[ f_1(x) = \sum_{i=1}^{2} \left[ -10 \exp \left( -0.2 \sqrt{x_i^2 + x_{i+1}^2} \right) \right] \]

  \[ f_2(x) = \sum_{i=1}^{3} \left[ |x_i|^{0.8} + 5 \sin \left( x_i^3 \right) \right] \]

- Second objective has sinusoidal component – results in discontinuous front
Global Optimization

- Solvers designed to explore the solution space and find global solutions

- Problems can be stochastic, or nonsmooth

- Solvers in Global Optimization Toolbox
  - MultiStart
  - GlobalSearch
  - simulannealbnd
  - patternsearch
  - ga (single- and multi-objective)

- Parallel Computing with ga, patternsearch, MultiStart
Speeding up with Parallel Computing Toolbox

- **Global Optimization Toolbox:**
  - `ga`: Members of population evaluated in parallel at each generation
  - `patternsearch`: Pattern evaluated in parallel (CompletePoll == on)
  - `MultiStart`: Start points evaluated in parallel

- **Optimization Toolbox:**
  - `fmincon`: parallel evaluation of objective function for finite differences
  - `fminimax`, `fgoalattain`: same as `fmincon`

- **In the objective function**
  - `parfor`
Key takeaways

- Solvers for a wide variety of problems
  - New Mixed-Integer Linear Programming Solver

- Supply gradient and Hessian if possible

- Global solvers for multiple minimum and nonsmooth problems

- Speed up with parallel computing
Learn more about optimization with MATLAB

- **MATLAB Digest**: Using Symbolic Gradients for Optimization

- **MATLAB Digest**: Improving Optimization Performance with Parallel Computing

- **Recorded webinar**: Tips and Tricks – Getting Started Using Optimization with MATLAB

- **Recorded webinar**: Global Optimization with MATLAB Products