Optimization Toolbox™ 4

Solve standard and large-scale optimization problems

Optimization Toolbox™ extends the MATLAB® technical computing environment with tools and widely used algorithms for standard and large-scale optimization. These algorithms solve constrained and unconstrained continuous and discrete problems. The toolbox includes functions for linear programming, quadratic programming, nonlinear optimization, nonlinear least squares, solving systems of nonlinear equations, multi-objective optimization, and binary integer programming.

MATLAB and Optimization Toolbox software let you easily define models, gather data, manage model formulations, and analyze results. They give engineers and scientists the tools needed to find optimal solutions, perform tradeoff analysis, balance multiple design alternatives, and quickly incorporate optimization methods in their algorithms and models.

Toolbox functions, which are accessible using the MATLAB command-line or through a graphical user interface (GUI), are mostly written in the open MATLAB language. This means that you can inspect most algorithms, modify the source code, and create your own custom functions.

### Defining, Solving, and Assessing Optimization Problems

Optimization Toolbox includes the most widely used methods for performing minimization and maximization. The toolbox implements both standard and large-scale algorithms, enabling you to solve problems by exploiting their sparsity or structure. You can access toolbox functions and solver options through the command-line interface or through the GUI.

KEY FEATURES

- Command line functions and GUI tools for defining, solving, and assessing optimization problems

- Solvers for nonlinear optimization and multi-objective optimization

- Solvers for nonlinear least-squares, data fitting, and nonlinear equations

- Methods for solving quadratic and linear programming problems

- Methods for solving binary integer programming problems

- Parallel computing support in selected constrained nonlinear solvers
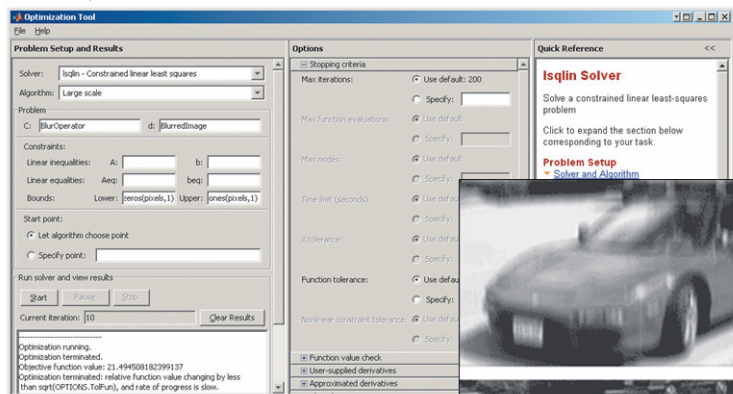
A blurred image is recovered using the large-scale linear least-squares algorithm in the Optimization Toolbox™.
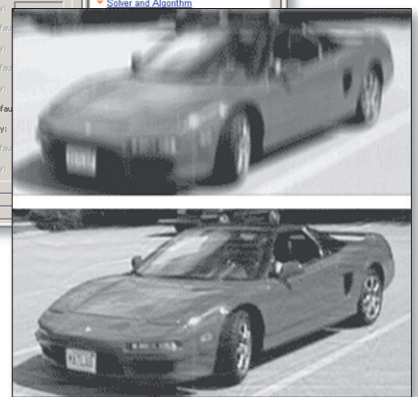
Optimization Tool is a graphical user interface that simplifies common optimization tasks. From the GUI, you can

- Select a solver and define your optimization problem
- Set and inspect optimization options and their default values for the selected solver
- Run problems and visualize intermediate and final results
- View solver-specific documentation in the optional quick reference window
- Import and export your problem definitions, algorithm options, and results between the MATLAB workspace and Optimization Tool
- Automatically generate M-code to capture your work and automate tasks
- Access all Genetic Algorithm and Direct Search Toolbox™ solvers (separate license required)

You can further manipulate and diagnose your optimization using the diagnostic outputs from the optimization methods. Using an output function, you can also write results to files, create your own stopping criteria, and write your own graphical user interfaces to interact with the toolbox solvers.

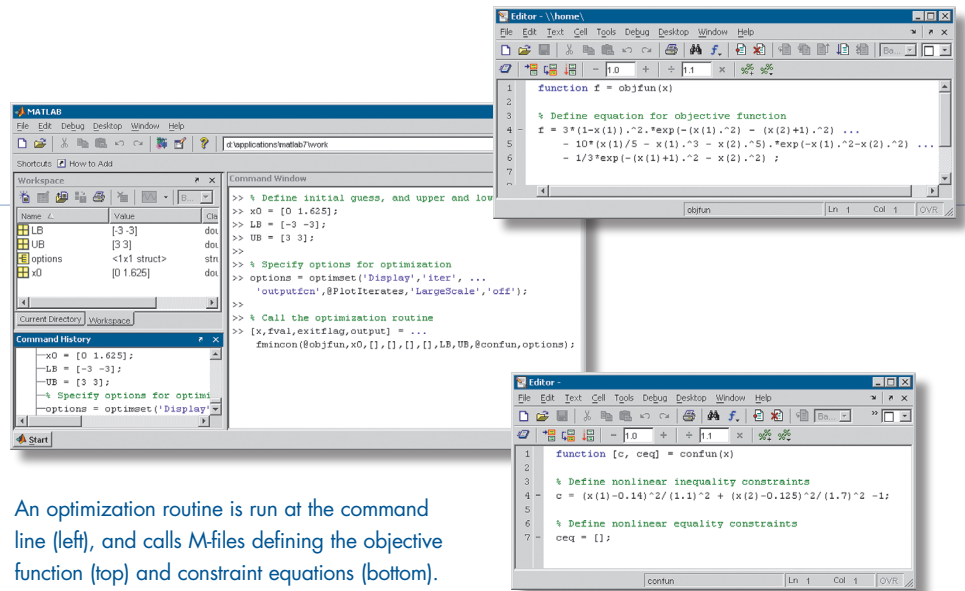## Nonlinear Optimization and Multi-objective Optimization
Unconstrained Nonlinear Optimization

Optimization Toolbox uses three methods to solve unconstrained nonlinear minimization problems: Quasi-Newton, Nelder-Mead, and trust-region.

The **Quasi-Newton** method uses a mixed quadratic and cubic line search procedure and the BFGS formula for updating the approximation of the Hessian matrix.

Nelder-Mead is a direct-search method that uses only function values (does not require derivatives) and handles non-smooth objective functions.

Trust-region is used for large-scale problems where sparsity or structure can be exploited.

Constrained Nonlinear Optimization

Constrained nonlinear optimization problems are composed of nonlinear objective functions and may be subject to linear and nonlinear constraints. Optimization Toolbox uses three methods to solve these problems: trust-region, active-set sequential quadratic programming, and interior-point.
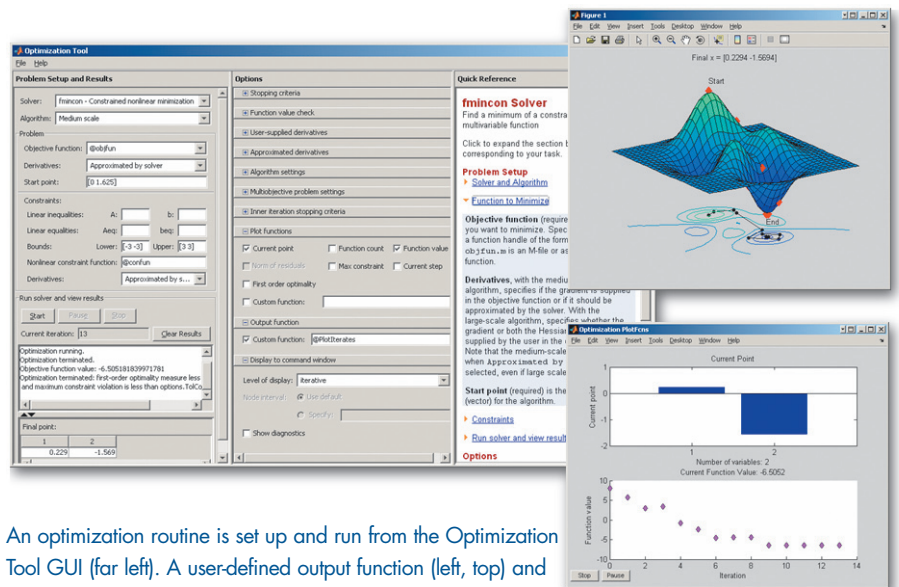
Trust-region is used for bound constrained problems or linear equalities only.

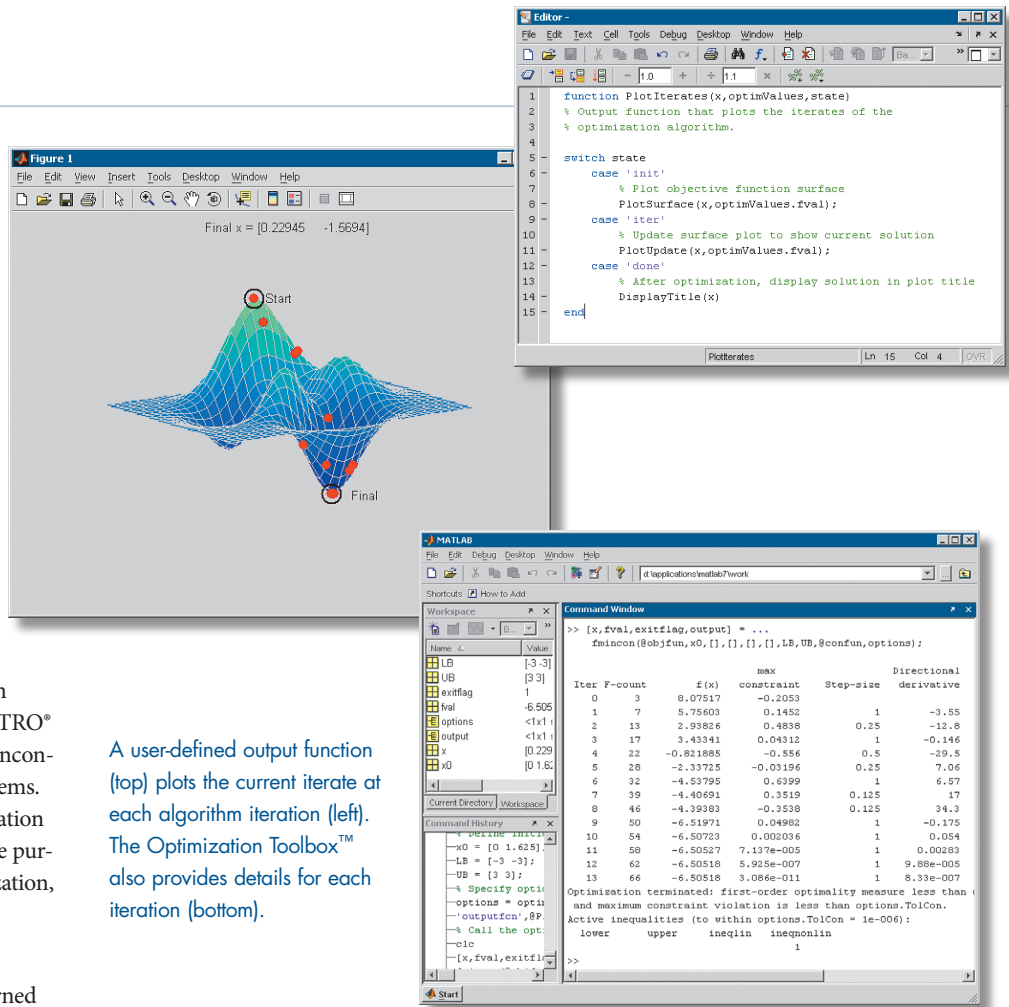**Active-set** sequential quadratic programming is used for general nonlinear optimization.

Interior-point is used for general nonlinear optimization. It is especially valuable for large-scale problems. It is based on a barrier function, and optionally keeps all iterates strictly feasible with respect to bounds during the optimization run.

The trust-region method, which is based on an interior-reflective Newton method, allows you to estimate Hessians with known sparsity structure using sparse finite differences of gradients. Additionally, the trust-region and interior-point methods enable you to calculate Hessian-times-vector products in a function without having to form the Hessian matrix explicitly. You can also adjust the bandwidth of the preconditioner used in the Newton iteration.

```
function PlotIterates(x,optimValues,state)
% Output function that plots the iterates of the
% optimization algorithm.

switch state
    case 'init'
        % Plot objective function surface
        PlotSurface(x,optimValues.fval);
    case 'iter'
        % Update surface plot to show current solution
        PlotUpdate(x,optimValues.fval);
    case 'done'
        % After optimization, display solution in plot title
        DisplayTitle(x)
end
```

A user-defined output function (top) plots the current iterate at each algorithm iteration (left). The Optimization Toolbox™ also provides details for each iteration (bottom).

Optimization Toolbox also includes an interface to Ziena Optimization's KNITRO® libraries for solving constrained and unconstrained nonlinear optimization problems. Use of the interface requires an installation of the KNITRO libraries, which can be purchased separately from Ziena Optimization, Inc (http://www.ziena.com/).

Multi-objective Optimization

Multi-objective optimization is concerned with the minimization of multiple objective functions that are subject to a set of constraints. Optimization Toolbox provides functions for solving two formulations of multi-objective optimization problems: goal attainment and minimax.

The **goal attainment** problem involves reducing the value of a linear or nonlinear vector function to attain the goal values given in a goal vector. The relative importance of the goals is indicated using a weight vector. The goal attainment problem may also be subject to linear and nonlinear constraints.

The **minimax** problem involves minimizing the worst-case value of a set of multivariate functions, possibly subject to linear and non linear constraints.

Optimization Toolbox transforms both types of multi-objective problems into standard constrained optimization problems and then solves them using a sequential quadratic programming approach.

Nonlinear Least Squares, Data Fitting, and Nonlinear Equations

Optimization Toolbox can solve nonlinear least squares problems, data fitting problems, and systems of nonlinear equations.

The toolbox uses three methods for solving nonlinear least squares problems: trust-region, Levenberg-Marquardt, and Gauss-Newton.

Levenberg-Marquardt is a method whose search direction is a cross between the Gauss-Newton and steepest descent directions.

Trust-region implements the Levenberg-Marquardt algorithm using trust regions. It is used for unconstrained and bound constrained problems.

Gauss-Newton is a line search method that chooses a search direction based on the solution to a linear least- squares problem.

The toolbox also includes a specialized inter face for data-fitting problems to find the member of a family of nonlinear functions that best fits a set of data points. The toolbox uses the same methods for data-fitting problems as it uses for nonlinear least-squares problems.

Optimization Toolbox implements a dogleg trust region method for solving a system of nonlinear equations where there are as many equations as unknowns. The toolbox can also solve this problem using either the trust-region, the Levenberg-Marquandt, or the Gauss-Newton method.

Quadratic and Linear Programming
Quadratic Programming

Quadratic programming problems involve minimizing a multivariate quadratic function subject to linear equality and inequality constraints. The toolbox implements three methods for solving these problems: trust-region, preconditioned conjugate gradient, and active set.

Trust-region is used for bound constrained problems.

Preconditioned conjugate gradient is used for problems subject to equality constraints.

Active set minimizes the objective at each iteration over the active set (a subset of the constraints that are locally active) until it reaches a solution.

Linear Programming

Linear programming problems consist of a linear expression for the objective function and linear equality and inequality constraints. Two methods are used to solve this type of problem: simplex and interior point.

The interior point method is based on a primal-dual predictor-corrector algorithm, and is used for large-scale linear problems.

The simplex method is a systematic procedure for generating and testing candidate vertex solutions to a linear program.

Binary Integer Programming

Binary integer programming problems involve minimizing a linear objective function subject to linear equality and inequality constraints. Each variable in the optimal solution must be either a 0 or a 1.

Optimization Toolbox solves these problems using a branch-and-bound algorithm that:

- Searches for a feasible binary integer solution
- Updates the best binary point found as the search tree grows
- Verifies that no better solution is possible by solving a series of linear programming relaxation problems

Solving Optimization Problems Using Parallel Computing

Optimization Toolbox can be used in conjunction with Parallel Computing Toolbox™ for solving problems that benefit from parallel computation. You can use parallel computing to decrease time to solution by enabling built-in parallel computing support or by defining a custom parallel computing implementation of your optimization problem.

Built-in support for parallel computing allows you to accelerate the gradient estimation step in select solvers for constrained nonlinear optimization problems and multi-objective goal attainment and minimax problems.

Customizable support for parallel computing involves explicitly defining the optimization problem to use parallel computing functionality. You can define either your objective function or constraint function to use parallel computing, allowing you to decrease the time required to evaluate the objective/constraint.

Required Products
MATLAB®

Related Products

Curve Fitting Toolbox™. Create curve fits and regression models of data

Genetic Algorithm and Direct Search Toolbox™. Solve optimization problems using genetic algorithms, simulated annealing, and direct search

Neural Network Toolbox™. Design and simulate neural networks

Parallel Computing Toolbox™. Perform parallel computations

Spline Toolbox™. Create and manipulate spline approximation models of data

Statistics Toolbox™. Perform statistical analysis, modeling, and algorithm development

For more information on related products, visit *www.mathworks.com/products/optimization*

Platform and System Requirements

For information on platform and system requirements, visit *www.mathworks.com/products/optimization* ▪