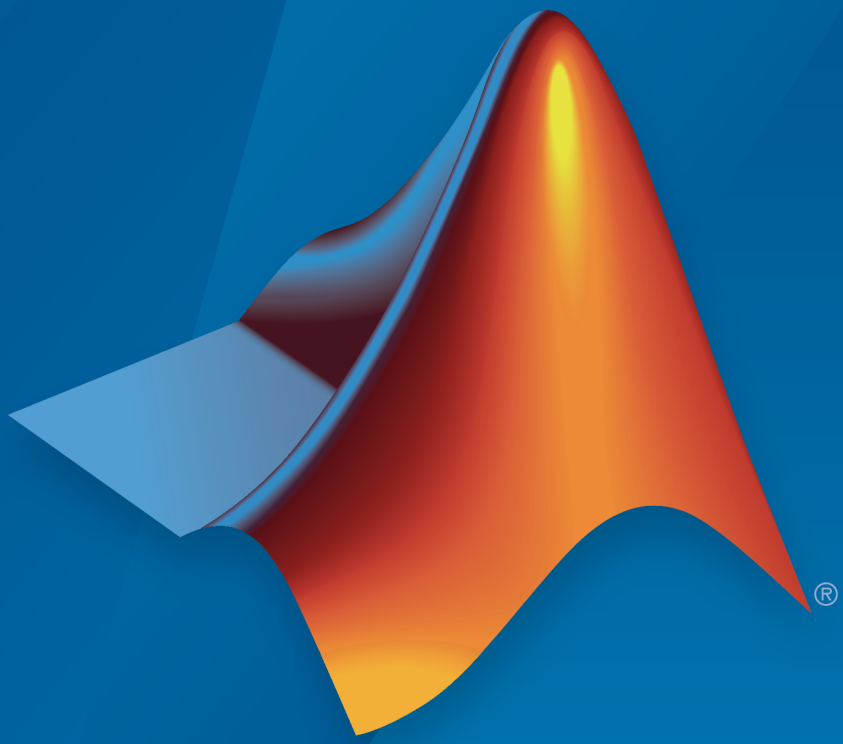
Optimization Toolbox™ Release Notes



MATLAB®

MathWorks®

How to Contact MathWorks

Latest news: www.mathworks.com

Sales and services: www.mathworks.com/sales_and_services

User community: www.mathworks.com/matlabcentral

Technical support: www.mathworks.com/support/contact_us

Phone: 508-647-7000

The MathWorks, Inc.
3 Apple Hill Drive
Natick, MA 01760-2098

Optimization Toolbox™ Release Notes

Contents

R2016a

Renamed Options: Use more expressive and consistent names for options 1-2

Parallel Computation: Accelerate `fminunc`, `fsolve`, `lsqcurvefit`, and `lsqnonlin` functions (using Parallel Computing Toolbox) 1-2

Option Changes: Distinguish between function tolerance and optimality tolerance, specify Hessians differently, more . 1-3

`resetoptions`: Restore default option values 1-3

Iterative display changes in `linprog` and `quadprog` 1-4

`fsolve 'trust-region-dogleg'` algorithm: Improved robustness 1-4

R2015b

Additional interior-point linear programming algorithm with improved performance and robustness 2-2

Mathematical Programming System (MPS) file reader for importing linear programming and mixed-integer linear programming problems 2-2

Updated output structure in several solvers 2-2

Optimization app will be removed in a future release **2-3**

Linear programming example **2-3**

R2015a

Improved performance and robustness of `intlinprog` `primal-simplex` algorithm **3-2**

`linprog dual-simplex` algorithm returns more information **3-2**

`quadprog active-set` algorithm will be removed **3-2**

`fmincon` allows problems without constraints **3-2**

Least-squares solvers allow equal upper and lower bounds . **3-3**

`quadprog` allows zero or empty quadratic term **3-3**

Changes to algorithm field of output structure **3-3**

Mixed-integer quadratic programming example **3-3**

R2014b

Dual-simplex algorithm in `linprog` linear programming solver **4-2**

Interior-point algorithm in `lsqlin` linear least-squares solver **4-2**

Plot functions and output functions for monitoring progress of `intlinprog` solver **4-2**

Levenberg-Marquardt `InitDamping` option 4-2

Changing default algorithm for fminunc 4-3

Mixed-integer linear programming example 4-3

Two `linprog` algorithms will be removed in the future 4-3

Two `fminunc` options will be removed in the future 4-3

`bintprog` removed 4-4

`ktrlink` removed 4-4

R2014a

Mixed-integer linear programming solver 5-2

New default algorithms in fmincon and quadprog 5-2

Nonlinear least-squares tweaks 5-3

Parallel option change 5-3

ktrlink default math library change 5-3

bintprog will be removed in the future 5-3

ktrlink will be removed in the future 5-3

R2013b

Solvers that check initial point more carefully 6-2

R2013a

optimoptions function for setting options with compact and comprehensive display 7-2

`Algorithm` option replaces `LargeScale` option 7-2

ktrlink supports KNITRO 8.1 7-3

R2012b

Changing default algorithms for fmincon and quadprog . . . 8-2

ktrlink supports KNITRO 8 8-2

R2012a

Enhanced Robustness in fminunc 9-2

FinDiffRelStep Option in Optimization Tool 9-2

Levenberg-Marquardt Algorithm Tweak 9-2

fmincon sqp Algorithm Tweak 9-2

R2011b

Derivative Estimate Changes 10-2

Gauss-Newton Algorithm Removed 10-2

DerivativeCheck Changes 10-3

fmincon ScaleProblem Default Changed 10-3

fsolve trust-region-dogleg Algorithm Change 10-4

Conversion of Error and Warning Message Identifiers 10-4

R2011a

New Quadratic Programming Algorithm 11-2

Enhanced Robustness in Nonlinear Solvers 11-3

New Defaults in DiffMinChange and DiffMaxChange Options 11-3

Output Structure Tweak 11-4

ktrlink Compatible with KNITRO 7 11-4

New quadprog Demo 11-4

R2010b

Enhanced fmincon Finite Difference Algorithms Add Robustness 12-2

ktrlink Available for Macintosh 64-Bit Systems 12-2

Output Structure Tweaks 12-2

New Video Demo on Modeling **12-2**

R2010a

New fmincon Algorithm **13-2**

lsqnonneg No Longer Uses x0 . . . . . **13-2**

R2009b

Enhanced Exit Messages in Selected Solvers **14-2**

fmincon Interior-Point Algorithm Robust to Certain Errors **14-2**

Changes in quadprog **14-2**

Changes in linprog **14-2**

Multiobjective optimValues Changes **14-3**

R2009a

Parallel Gradient Estimation Available in fmincon Interior-Point Algorithm **15-2**

Enhanced Exit Messages in Selected Solvers **15-2**
Links to More Information Window **15-2**
Link for More Detail in Command Window **15-2**
New Display Option Values Control Default Detail **15-2**
Messages in Output Structure **15-3**

Change in linprog Simplex Algorithm 15-3

Change in fminunc Exit Flag 15-3

New demos 15-4

R2008b

fsolve, lsqcurvefit, lsqnonlin Algorithm and Options Changes 16-2

Optimization Tool Enables Parallel Functionality 16-2

Central Finite Differences Available in Selected Solvers . . 16-2

lsqnonneg Refactored 16-3

Finite Difference Algorithm Tweaked 16-3

DerivativeCheck Tolerance Changed 16-4

R2008a

Parallel Computing Toolbox Support in fmincon, fminimax, and fgoalattain 17-2

Combined and Extended optimtool 17-2

New fmincon Solver, New Option Algorithm for fmincon, Option LargeScale Changed 17-2

External Interface to KNITRO Libraries 17-3

Default PrecondBandWidth = Inf in lsqcurvefit, lsqnonlin, and fsolve 17-3

New Option TolConSQP with Incompatible Default Value . **17-3**

Field constrviolation in Output Structure **17-3**

R2007b

Bug Fixes

R2007a

Changes to Outputs of Multiobjective Solvers **19-2**

R2006b

New Optimization Tool **20-2**

Plot Functions Option Added **20-2**

Output Function Option Enhanced to Accept Multiple Functions **20-2**

Changes to the Output Function **20-2**

R2006a

Bug Fixes

Notify Parameter Added to Display Option for Five Functions **22-2**

R2016a

Version: 7.4

New Features

Bug Fixes

Compatibility Considerations

Renamed Options: Use more expressive and consistent names for options

Many options have new names. The old names continue to work, but do not appear in documented examples. The renaming gives a more consistent set of option names that match those in Global Optimization Toolbox.

All legacy option names continue to work in both `optimoptions` and `optimset`.

For option name change details, see "Current and Legacy Option Name Tables".

Compatibility Considerations

`optimset` uses only the legacy option names, while `optimoptions` uses either legacy names or current names. However, `optimoptions` displays only current names, so if you set a legacy name, it displays the equivalent current name. See "Current and Legacy Option Name Tables".

`optimoptions` no longer displays some options. See "View Options".

Some option values differ when using current names. For example, any previous setting that was `'on'` or `'off'` is now `true` or `false`.

Parallel Computation: Accelerate `fminunc`, `fsolve`, `lsqcurvefit`, and `lsqnonlin` functions (using Parallel Computing Toolbox)

More nonlinear solvers can estimate gradients or Jacobians by parallel finite differences:

- `fminunc`
- `fsolve`
- `lsqcurvefit`
- `lsqnonlin`

To enable parallel finite differences, set the `UseParallel` option to `true`. For details, see "Parallel Computing".

Option Changes: Distinguish between function tolerance and optimality tolerance, specify Hessians differently, more

- Previously, the `TolFun` tolerance applied to both the change in function value and to the size of the first-order optimality measure. This role is now explicitly split into two new tolerances:
  - `FunctionTolerance` — Applies to changes in objective function value
  - `OptimalityTolerance` — Applies to first-order optimality measure
- The way to set Hessians for `fminunc` and `fmincon` has changed. For details, see "Including Hessians".
- Several solvers gained a `SubproblemAlgorithm` option for their `'trust-region'` or `'trust-region-reflective'` algorithm:
  - `fminunc`
  - `fsolve`
  - `lsqcurvefit`
  - `lsqnonlin`
- The former `Jacobian` option in `fsolve`, `lsqcurvefit`, and `lsqnonlin` is now the `SpecifyObjectiveGradient` option.

Compatibility Considerations

The Optimization app imports and exports only one option related to the former `TolFun` tolerance. It displays this option as **Optimality tolerance**, and uses it as the `OptimalityTolerance` option. You cannot import, export, or change the `FunctionTolerance` option in the Optimization app for Optimization Toolbox™ solvers. When you run problems in the Optimization app, `FunctionTolerance` has its default value.

However, Global Optimization Toolbox solvers do not have an `OptimalityTolerance` option. Those solvers can import, export, and set the `FunctionTolerance` option in the Optimization app.

`resetoptions`: Restore default option values

The `resetoptions` function resets selected `optimoptions` options to their default values. For details, see the function reference page.

Iterative display changes in `linprog` and `quadprog`

Iterative display has changed for the `linprog` `'dual-simplex'` and `'interior-point'` algorithms, and for the `quadprog` `'interior-point-convex'` algorithm. The changes give clearer and more consistent information. For iterative display details, see "linprog" and "quadprog".

`fsolve 'trust-region-dogleg'` algorithm: Improved robustness

The default `fsolve` `'trust-region-dogleg'` algorithm is now more robust to function evaluation failures during finite difference estimation of the Jacobian.

R2015b

Version: 7.3

New Features

Bug Fixes

Compatibility Considerations

Additional interior-point linear programming algorithm with improved performance and robustness

The linprog solver has a new algorithm named `'interior-point'`. The previous algorithm named `'interior-point'` is now named `'interior-point-legacy'`. Usually, the new `'interior-point'` algorithm is faster and more robust than the `'interior-point-legacy'` algorithm.

Compatibility Considerations

The default `linprog` algorithm is now named `'interior-point-legacy'`. If your code explicitly chooses the `'interior-point'` algorithm, `linprog` runs the new interior-point algorithm, and so can have different behavior than before.

Mathematical Programming System (MPS) file reader for importing linear programming and mixed-integer linear programming problems

The mpsread function reads files in the MPS format that specify linear programming problems or mixed-integer linear programming problems. For details, see the function reference page.

Updated output structure in several solvers

The `fmincon`, `fsolve`, `lsqcurvefit`, and `lsqnonlin` `'trust-region-reflective'` algorithms now return a `stepsize` field in their `output` structures. Similarly, the `fminunc` `'trust-region'` algorithm now returns a `stepsize` field in its `output` structure.

The `fmincon` `'sqp'` algorithm and `fminunc` `'quasi-newton'` algorithm now return a `stepsize` field that contains the size of the final step, and a `lssteplength` field that contains the relative step compared to a line-search prediction.

Compatibility Considerations

The meaning of the `stepsize` field has changed for the `output` structure of the `fmincon` `'sqp'` algorithm and `fminunc` `'quasi-newton'` algorithm. The former `stepsize` field, which measured the relative step compared to a line-search prediction, is now returned in the `lssteplength` field.

Optimization app will be removed in a future release

The Optimization app now warns that it will be removed in a future release.

Compatibility Considerations

Set and examine options using optimoptions, which streamlines viewing Optimization Toolbox options.

Linear programming example

There is a new featured example of linear programming, Maximize Long-Term Investments Using Linear Programming.

R2015a

Version: 7.2

New Features

Bug Fixes

Compatibility Considerations

Improved performance and robustness of `intlinprog` `primal-simplex` algorithm

The `'primal-simplex'` algorithm of the intlinprog solver can solve more problems than before, and has better performance on large problems. Select the algorithm in the `RootLPAlgorithm` option.

Compatibility Considerations

`intlinprog` iterations can differ from previous versions. This holds even when you select the `'dual-simplex'` algorithm, because `intlinprog` uses the `'primal-simplex'` algorithm for some calculations in any case.

`linprog` `dual-simplex` algorithm returns more information

The linprog `'dual-simplex'` algorithm now returns a Lagrange multiplier structure and an `output` structure containing the first-order optimality measure.

`quadprog` `active-set` algorithm will be removed

The quadprog `'active-set'` algorithm now warns that it will be removed in a future release.

Compatibility Considerations

To avoid this warning, use optimoptions to set the `Algorithm` option to `'interior-point-convex'` or `'trust-region-reflective'`. Or simply do not set the `Algorithm` option; `quadprog` defaults to the `'interior-point-convex'` algorithm.

`fmincon` allows problems without constraints

fmincon no longer throws an error if you run it on a problem without constraints. This change can make it easier for you to see the effect of constraints by running a problem both with and without constraints.

To run `fmincon` without constraints, you no longer need to set an artificial constraint, such as `lb = -Inf`, but doing so will not matter.

Compatibility Considerations

Any code that checks for this error will have different behavior than before.

Least-squares solvers allow equal upper and lower bounds

The lsqnonlin and lsqcurvefit solvers now allow you to fix variables by specifying equal upper and lower bounds. For example, lower bound `lb = [0,0]` and upper bound `ub = [Inf,0]` fix the solution `x` to have `x(2) = 0`.

`quadprog` allows zero or empty quadratic term

Previously, if you gave a zero or empty matrix as the `H` input, the quadprog solver would switch to `linprog`. Now `quadprog` solves the problem.

Changes to algorithm field of output structure

The `algorithm` field of the `output` structure of several solvers has changed. Now, all `output.algorithm` strings are the same as those you set in the `Algorithm` name-value pair. For example, the `output.algorithm` field of the `fmincon` `'active-set'` algorithm used to be `'medium-scale: SQP, Quasi-Newton, line-search'`. Now it is `'active-set'`.

The affected solvers are `fgoalattain`, `fminimax`, `fmincon`, `fseminf`, `fsolve`, `linprog`, `lsqcurvefit`, and `lsqnonlin`.

Compatibility Considerations

Any code that uses the string in `output.algorithm` can have different behavior than before.

Mixed-integer quadratic programming example

There is a new featured example of mixed-integer quadratic programming, Mixed-Integer Quadratic Programming Portfolio Optimization.

R2014b

Version: 7.1

New Features

Bug Fixes

Compatibility Considerations

Dual-simplex algorithm in `linprog` linear programming solver

linprog has a new algorithm option named `'dual-simplex'`. The algorithm saves memory when you specify sparse constraint matrices. For details, see the function reference page.

Currently, `'dual-simplex'` returns an empty Lagrange multiplier structure `lambda` and `firstorderopt` field in the `output` structure.

Interior-point algorithm in `lsqlin` linear least-squares solver

lsqlin has a new algorithm option named `'interior-point'`. The algorithm handles all types of constraints and saves memory when you specify sparse constraint matrices. For details, see the function reference page.

Plot functions and output functions for monitoring progress of `intlinprog` solver

intlinprog accepts `OutputFcn` and `PlotFcns` options. For details, see intlinprog Output Functions and Plot Functions.

There is a built-in output function that collects all integer feasible solutions that `intlinprog` encounters. Choose this output function by using optimoptions to set the `OutputFcn` option to `@savemilpsolutions`.

There is a built-in plot function that shows details of the solver iterations. Choose this plot function by using optimoptions to set the `PlotFcns` option to `@optimplotmilp`.

Levenberg-Marquardt `InitDamping` option

The fsolve, lsqcurvefit, and lsqnonlin solvers can use the `'levenberg-marquardt'` algorithm. To initialize the Levenberg-Marquardt parameter differently than its default, set the new `InitDamping` option using optimoptions.

Compatibility Considerations

Previously, to initialize the Levenberg-Marquardt parameter you would pass a cell array in the `Algorithm` option when using optimset, such as

```
options = optimset('Algorithm',{'levenberg-marquardt',0.1});
```

That method still works. But when using `optimoptions`, you cannot set the parameter in the `Algorithm` option, and instead must use the `InitDamping` option, such as

```
options = optimoptions(@fsolve,'Algorithm','levenberg-marquardt','InitDamping',0.1);
```

Changing default algorithm for `fminunc`

The fminunc default algorithm will change to `'quasi-newton'` in a future release.

Compatibility Considerations

`fminunc` now warns when you run it in cases where the default behavior will change. For details, see the fminunc function reference page or fminunc Algorithms.

Mixed-integer linear programming example

There is a new featured example of mixed-integer linear programming, Optimal Dispatch of Power Generators.

Two `linprog` algorithms will be removed in the future

The linprog `'active-set'` and `'simplex'` algorithms warn that they will be removed in a future release.

Compatibility Considerations

To avoid this warning, choose the `'interior-point'` or `'dual-simplex'` algorithms using optimoptions.

Two `fminunc` options will be removed in the future

fminunc now warns that the `InitialHessMatrix` and `InitialHessType` options will be removed in a future release.

Compatibility Considerations

To avoid these warnings, do not set values for these options. The Optimization app no longer has these options.

`bintprog` removed

The `bintprog` function has been removed.

Compatibility Considerations

To update code to use `intlinprog` instead of `bintprog`, see Tips.

`ktrlink` removed

The `ktrlink` interface to the KNITRO® third-party solver has been removed.

Compatibility Considerations

For a KNITRO interface, contact Ziena Optimization: `www.ziena.com`.

R2014a

Version: 7.0

New Features

Bug Fixes

Compatibility Considerations

Mixed-integer linear programming solver

The intlinprog function solves mixed-integer linear programming problems. For more information, see the reference page and the documentation.

There are several new featured examples of mixed-integer linear programming:

- Factory, Warehouse, Sales Allocation Model
- Travelling Salesman Problem
- Solve Sudoku Puzzles Via Integer Programming

Currently, you cannot run `intlinprog` in the Optimization app.

New default algorithms in `fmincon` and `quadprog`

When you do not specify an algorithm, fmincon and quadprog default to different algorithms than before. Usually, the new algorithms are faster and more robust than the algorithms they replace.

Solver	Previous Default Algorithm	New Default Algorithm
`fmincon`	`'trust-region-reflective'`	`'interior-point'`
`quadprog`	`'trust-region-reflective'`	`'interior-point-convex'`

Compatibility Considerations

Solvers can produce different results than before. To reproduce previous results, set the `Algorithm` option with optimoptions.

```
options = optimoptions('fmincon','Algorithm','trust-region-reflective');
% or
options = optimoptions('quadprog','Algorithm','trust-region-reflective');
```

Be sure to pass `options` in your function call.

```
x = fmincon(fun,x0,A,b,Aeq,beq,lb,ub,nlcon,options)
% or
x = quadprog(H,f,A,b,Aeq,beq,lb,ub,x0,options)
```

Nonlinear least-squares tweaks

The lsqcurvefit and lsqnonlin solvers have slightly different behavior than before when the internally calculated trust-region radius gets large in the `'trust-region-reflective'` algorithm, or the Levenberg-Marquardt parameter gets small in the `'levenberg-marquardt'` algorithm. There are no longer any arbitrary limits on these parameters. fsolve has the same change to its `'trust-region-reflective'` and `'levenberg-marquardt'` algorithms.

Parallel option change

The `UseParallel` option now accepts the values `true` and `false`. The option also accepts the former values `'always'` and `'never'`, and scalar values `1` and `0`.

The affected solvers are `fmincon`, `fgoalattain`, and `fminimax`.

`ktrlink` default math library change

The ktrlink function uses a different default math library within KNITRO than before. This change can enable `ktrlink` to run on more hardware versions.

Compatibility Considerations

With the new default math library, `ktrlink` runs slower than before. To use the previous math library, set the KNITRO format option `blasoption` to 1. See Setting Options.

`bintprog` will be removed in the future

intlinprog solves more problems than bintprog, and has better performance. So `bintprog` will be removed in a future release.

To update your existing `bintprog` code to use `intlinprog`, see Tips.

`ktrlink` will be removed in the future

ktrlink will be removed in a future release. For an updated KNITRO interface, contact Ziena Optimization: `www.ziena.com`.

R2013b

Version: 6.4

New Features

Bug Fixes

Compatibility Considerations

Solvers that check initial point more carefully

All nonlinear solvers now check whether derivatives of the objective and nonlinear constraint functions are well defined at the initial point, usually called `x0`. (This is in addition to the existing checks that the functions are well defined at `x0`.) Well defined means the value of each derivative is not `NaN`, `Inf`, or complex (nonlinear least-squares solvers allow complex values). Derivatives include both gradients and Jacobians. See Including Derivatives and Writing Vector and Matrix Objective Functions.

When the objective or nonlinear constraint functions do not include a derivative, solvers approximate derivatives by finite differences. This means that the functions must be well defined for points in a small neighborhood of `x0`.

If any derivative is not well defined at `x0`, the solver stops with an error, and does not attempt to find a solution.

In all tested cases, this behavior led to a clearer exit condition.

Compatibility Considerations

It is conceivable that a problem that previously ran to completion will now exit without completion. This can occur in the rare case when a derivative did not exist for a function at the initial point, but the solver was able to step to a point where the derivatives were well defined.

R2013a

Version: 6.3

New Features

Bug Fixes

Compatibility Considerations

`optimoptions` function for setting options with compact and comprehensive display

The new optimoptions function creates and modifies options for all solvers except the base MATLAB® solvers `fminbnd`, `fminsearch`, `fzero`, and `lsqnonneg`. Continue to use optimset for those functions.

`optimoptions` organizes options by solver, with a more focused and comprehensive display than `optimset`:

- Creates and modifies only the options that apply to a solver
- Shows your option choices and default values for a specific solver/algorithm
- Displays links for more information on solver options and other available solver algorithms

For details, see the optimoptions page, or Set Options.

Compatibility Considerations

If you export options or a problem from the Optimization Tool, the options are an object as created by `optimoptions`. Therefore, earlier software versions cannot import the options or problem. This consideration does not apply to the base MATLAB solvers, which continue to encapsulate options as a structure.

`Algorithm` option replaces `LargeScale` option

Use the `Algorithm` option to select the algorithm in all solvers that have multiple algorithms. In a future release, solvers will no longer use the `LargeScale` option. The `linprog` `Simplex` option will also change to the `Algorithm` option.

Compatibility Considerations

Update options as follows.

Solver	Previous Setting	Current Setting
fminunc	`'LargeScale' = 'on'`	`'Algorithm' = 'trust-region'`

Solver	Previous Setting	Current Setting
	`'LargeScale' = 'off'`	`'Algorithm' = 'quasi-newton'`
linprog	`'LargeScale' = 'on'`	`'Algorithm' = 'interior-point'`
	`'LargeScale' = 'off'`, `'Simplex' = 'off'` or unset	`'Algorithm' = 'active-set'`
	`'LargeScale' = 'off'`, `'Simplex' = 'on'`	`'Algorithm' = 'simplex'`
lsqlin	`'LargeScale' = 'on'`	`'Algorithm' = 'trust-region-reflective'`
	`'LargeScale' = 'off'`	`'Algorithm' = 'active-set'`

`ktrlink` supports KNITRO 8.1

The `ktrlink` function now supports KNITRO version 8.1. For details, see ktrlink: An Interface to KNITRO Libraries.

R2012b

Version: 6.2.1

Bug Fixes

Compatibility Considerations

Changing default algorithms for `fmincon` and `quadprog`

The fmincon and quadprog default algorithms will change in a future release.

- The default `fmincon` algorithm will become `'interior-point'`.
- The default `quadprog` algorithm will become `'interior-point-convex'`.

Compatibility Considerations

These solvers now warn when you run them in cases where the default behavior will change. For example, they warn when:

- You do not set the `Algorithm` option.
- You set incompatible `LargeScale` and `Algorithm` options.

To avoid these warnings:

- Do not set the `LargeScale` option.
- Set the `Algorithm` option appropriately.

For details, see the fmincon and quadprog function reference pages or Choosing the Algorithm.

`ktrlink` supports KNITRO 8

The ktrlink function now supports KNITRO version 8. For details, see ktrlink: An Interface to KNITRO Libraries.

R2012a

Version: 6.2

New Features

Bug Fixes

Compatibility Considerations

Enhanced Robustness in fminunc

The fminunc medium-scale algorithm now attempts to recover from failures when evaluating the objective function during iteration steps, or during gradient estimation. Failure means the objective function returns `NaN`, a complex value, or `Inf`. If there is such a failure, the algorithm attempts to take different steps.

As part of robustness, the `fminunc` medium-scale algorithm now uses the `ObjectiveLimit` tolerance.

Compatibility Considerations

When objective function values drop below `ObjectiveLimit` (default value: `-1e20`), iterations end with a `-3` exit flag. Use optimset to change the value of `ObjectiveLimit`. Set `ObjectiveLimit` to `-Inf` to disable this tolerance.

FinDiffRelStep Option in Optimization Tool

The `FinDiffRelStep` option for choosing relative finite difference step sizes is now available in the Optimization Tool, in the **Approximated derivatives** pane. This option lets you tune the gradient estimation step in most solvers.

Levenberg-Marquardt Algorithm Tweak

The fsolve, lsqcurvefit, and lsqnonlin solvers no longer use the magnitude of the Levenberg-Marquardt regularization parameter as a stopping criterion, so they no longer return an exit flag of `-3` when using the `levenberg-marquardt` algorithm. Instead, they use the `TolX` tolerance in all internal calculations.

Compatibility Considerations

The solvers now stop with exit flag `2` in most situations where previously they stopped with exit flag `-3`.

fmincon sqp Algorithm Tweak

The fmincon `sqp` algorithm calculates its Lagrange multiplier estimates somewhat differently than before.

Compatibility Considerations

The `fmincon` `sqp` algorithm can give slightly different results than before.

R2011b

Version: 6.1

New Features

Bug Fixes

Compatibility Considerations

Derivative Estimate Changes

- The fsolve, lsqcurvefit, and lsqnonlin solvers now accept the `FinDiffType` option. Set `FinDiffType` to `'central'` with optimset to enable derivative estimation by central finite differences. Central finite differences are more accurate, but take more time than the default `'forward'` finite differences.
- fsolve, lsqcurvefit, and lsqnonlin now use the `TypicalX` option when estimating dense Jacobians via finite differences. In previous releases, these solvers used `TypicalX` only when checking derivatives.
- For algorithms that obey bounds, finite difference steps for derivative estimation now stay within any bounds you set for the decision variables. See Iterations Can Violate Constraints.
- The new `FinDiffRelStep` option allows you to set a vector of finite difference step sizes to better handle problems whose components have different scales. Use `FinDiffRelStep` at the command line for any solver that uses finite differences. For details, see `FinDiffRelStep` in Options Structure.

Gauss-Newton Algorithm Removed

The fsolve, lsqcurvefit, and lsqnonlin functions no longer use the Gauss-Newton algorithm.

Compatibility Considerations

The previous way of selecting the Gauss-Newton algorithm was to set the `LargeScale` option to `'off'`, and in:

- `fsolve` — set the `NonlEqnAlgorithm` option to `'gn'`.
- `lsqcurvefit` or `lsqnonlin` — set the `LevenbergMarquardt` option to `'off'`.

To select an algorithm, use optimset to set the `Algorithm` option:

- `fsolve` — `trust-region-dogleg`, `trust-region-reflective`, or `levenberg-marquardt`
- `lsqcurvefit` or `lsqnonlin` — `trust-region-reflective` or `levenberg-marquardt`

Solvers no longer use the `LevenbergMarquardt`, `LineSearchType`, and `NonlEqnAlgorithm` options, since these options relate only to the Gauss-Newton algorithm.

DerivativeCheck Changes

The `DerivativeCheck` option checks whether a solver's finite-difference approximations match the gradient or Jacobian functions that you supply. When a solver finds a discrepancy between the computed derivatives and their finite-difference approximations, the solver now errors. Solvers used to pause in this situation instead of erroring.

Additionally, solvers now compare derivatives at a point near the initial point `x0`, but not exactly at `x0`. Previously, solvers performed the comparison at `x0`. This change usually gives more reliable `DerivativeCheck` decisions. For details, see Checking Validity of Gradients or Jacobians.

Solvers do not include the computations for `DerivativeCheck` in the function count. See Iterations and Function Counts.

Compatibility Considerations

Solvers compare the derivatives at a different point than before, so can change their decision on whether the derivatives match. Solvers now error instead of pause when they encounter a discrepancy.

fmincon ScaleProblem Default Changed

The fmincon `interior-point` and `sqp` algorithms can use the `ScaleProblem` option. The default value of `ScaleProblem` is now `'none'` instead of `'obj-and-constr'`.

Compatibility Considerations

Because of a bug in previous releases, when you did not provide gradients of the objective and nonlinear constraint functions, `fmincon` did not scale these functions. `fmincon` did scale linear constraints. So, if you do not provide gradients and have no linear constraints, the current `fmincon` behavior is the same as in previous releases. However, the current behavior can differ if you do provide gradients (`GradObj` or `GradConstr`

is 'on'). If you provide gradients, have no linear constraints, and want to obtain the previous behavior, set ScaleProblem to 'obj-and-constr' with optimset.

fsolve trust-region-dogleg Algorithm Change

The fsolve trust-region-dogleg algorithm no longer performs an internal calculation of conditioning. This change usually speeds fsolve.

Compatibility Considerations

fsolve iterations differ from previous versions. Additionally, the solution and all associated outputs can differ from previous versions. Usually, results are numerically equivalent to previous results.

Conversion of Error and Warning Message Identifiers

For R2011b, error and warning message identifiers have changed in Optimization Toolbox.

Compatibility Considerations

If you have scripts or functions that use message identifiers that changed, you must update the code to use the new identifiers. Typically, message identifiers are used to turn off specific warning messages, or in code that uses a try/catch statement and performs an action based on a specific error identifier.

For example, the 'optim:fmincon:ConstrainedProblemsOnly' identifier has changed to 'optimlib:fmincon:ConstrainedProblemsOnly'. If your code checks for 'optim:fmincon:ConstrainedProblemsOnly', you must update it to check for 'optimlib:fmincon:ConstrainedProblemsOnly' instead.

To determine the identifier for a warning, run the following command just after you see the warning:

```
[MSG,MSGID] = lastwarn;
```

This command saves the message identifier to the variable MSGID.

To determine the identifier for an error, run the following command just after you see the error:

```
exception = MException.last;
MSGID = exception.identifier;
```

Tip Warning messages indicate a potential issue with your code. While you can turn off a warning, a suggested alternative is to change your code so it runs warning free.

R2011a

Version: 6.0

New Features

Bug Fixes

Compatibility Considerations

New Quadratic Programming Algorithm

quadprog has a new algorithm named `'interior-point-convex'`. It has these features:

- The algorithm has fast internal linear algebra.
- The algorithm handles sparse problems.
- There is a new presolve module that can improve speed, numerical stability, and detection of infeasibility.
- The algorithm handles large convex problems, and accepts and uses sparse inputs. See Large-Scale vs. Medium-Scale Algorithms.
- The algorithm optionally gives iterative display.
- The algorithm has enhanced exit messages.

For details on the algorithm, see interior-point-convex quadprog Algorithm. For help choosing the algorithm to use, see Quadratic Programming Algorithms.

Compatibility Considerations

You now choose the `quadprog` algorithm by using optimset to set the `Algorithm` option instead of the `LargeScale` option. If you don't set `Algorithm` or `LargeScale`, `quadprog` behaves as before.

`Algorithm` option choices are:

- `trust-region-reflective` (formerly `LargeScale = 'on'`), the default
- `active-set` (formerly `LargeScale = 'off'`)
- `interior-point-convex`

The previous way of choosing the `quadprog` algorithm at the command line was to set the `LargeScale` option to `'on'` or `'off'`. `quadprog` now ignores the `LargeScale` option, except when you set the inconsistent values `LargeScale` = `'off'` and `Algorithm` = `'trust-region-reflective'`. In this case, to avoid backward incompatibility, `quadprog` honors the `LargeScale` option, and uses the `'active-set'` algorithm.

`quadprog` now checks whether any inputs are complex, and, if so, it errors. The only exception is the `Hinfo` argument for the `HessMult` option is allowed to be complex.

Enhanced Robustness in Nonlinear Solvers

More solvers now attempt to recover from errors in the evaluation of objective functions and nonlinear constraint functions during iteration steps, or, for some algorithms, during gradient estimation. The errors include results that are `NaN` or `Inf` for all solvers, or complex for `fmincon` and `fminunc`. If there is such an error, the algorithms attempt to take different steps. The following solvers are enhanced:

- fmincon `trust-region-reflective` algorithm (the `interior-point` and `sqp` algorithms already had this robustness)
- fminunc `LargeScale` algorithm
- fsolve `trust-region-reflective`, `trust-region-dogleg`, and `levenberg-marquardt` algorithms
- lsqcurvefit `trust-region-reflective` and `levenberg-marquardt` algorithms
- lsqnonlin `trust-region-reflective` and `levenberg-marquardt` algorithms

New Defaults in DiffMinChange and DiffMaxChange Options

The `DiffMinChange` and `DiffMaxChange` options set the minimum and maximum possible step sizes for finite differences in gradient estimation. The defaults are now:

- `DiffMinChange = 0` (formerly `1e-8`)
- `DiffMaxChange = Inf` (formerly `0.1`)

Solvers have mechanisms that ensure nonzero and non-infinite step sizes, so the new defaults simply mean that the step size adjustment algorithms have fewer constraints.

The new defaults remove the previous arbitrary choices. The previous values can be inappropriate when components are too large or small in magnitude. Tests show these new defaults are good for most situations.

Compatibility Considerations

Some solver iterations can differ from previous ones. To obtain the previous behavior:

```
options = optimset('DiffMinChange',1e-8,'DiffMaxChange',0.1);
```

Output Structure Tweak

For the `trust-region-reflective` algorithm, the `algorithm` field of the `output` structure is now `'trust-region-reflective'`. This value differs slightly from the previous values returned by `fmincon`, `fsolve`, `lsqcurvefit`, `lsqnonlin`, and `quadprog`.

Compatibility Considerations

To avoid errors or unexpected results, update any code that depends on the exact value of the `output.algorithm` string.

ktrlink Compatible with KNITRO 7

ktrlink is compatible with KNITRO 7. For details, see ktrlink: An Interface to KNITRO Libraries, or the Ziena Optimization web site `http://www.ziena.com/`.

New quadprog Demo

A new demo shows how to solve portfolio optimization problems using `quadprog`. View the demo at the command line by entering

```
showdemo portfoptimdemo
```

R2010b

Version: 5.1

New Features

Bug Fixes

Enhanced fmincon Finite Difference Algorithms Add Robustness

The fmincon `interior-point` and `sqp` algorithms now attempt to recover from errors in the evaluation of objective functions and nonlinear constraint functions during gradient estimation. The errors include results that are `NaN`, `Inf`, or complex. If there is such an error, the finite differencing routines attempt to take different steps.

ktrlink Available for Macintosh 64-Bit Systems

The ktrlink function now works with Macintosh 64-bit systems. Therefore, `ktrlink` works on the same systems as all other Optimization Toolbox functions.

Output Structure Tweaks

All linprog and quadprog algorithms now create a `firstorderopt` field in the output structure. This field contains the value of the first-order optimality measure at the final point.

All fmincon and quadprog algorithms now create a `constrviolation` field in the output structure. This field contains the largest value of the constraint functions at the final point: bounds, linear constraints, and nonlinear constraints. (Some algorithms return the larger of the constraint functions and 0.) See Writing Constraints.

New Video Demo on Modeling

There is a new two-part video on modeling and solving optimization problems:

- Mathematical Modeling with Optimization, Part 1
- Optimization Modeling 2: Converting to Solver Form

R2010a

Version: 5.0

New Features

Bug Fixes

Compatibility Considerations

New fmincon Algorithm

fmincon has a new algorithm called `SQP` for Sequential Quadratic Programming. The algorithm has the following features:

- Honors bounds at all iterations
- Attempts a different step if one leads to an objective or constraint function returning a `NaN`, `Inf`, or complex result
- Fast internal linear algebra for solving quadratic programs

Choose the algorithm at the command line by setting the `Algorithm` option to `'sqp'` with optimset. For more information about the algorithm, see fmincon SQP Algorithm in the Optimization Toolbox documentation.

lsqnonneg No Longer Uses x0

The lsqnonneg solver no longer accepts a start point `x0` as an optional input.

Compatibility Considerations

The Optimization Tool no longer has an input region for accepting a start point. If you import or run a problem that contains a start point `x0`, MATLAB issues a warning. Also, the Optimization Tool and `lsqnonneg` ignore `x0`, and instead use a start point of a vector of zeroes. If you export a problem structure from the Optimization Tool, there is no `x0` field.

R2009b

Version: 4.3

New Features

Bug Fixes

Compatibility Considerations

Enhanced Exit Messages in Selected Solvers

Enhanced, clearer exit messages in `fsolve`, `lsqnonlin`, and `lsqcurvefit`, with links for more information. For more information about the enhancements, see Exit Flags and Exit Messages.

Compatibility Considerations

For solvers with enhanced exit messages, the content of `output.message` contains many more characters than before. User code that relies on this field might need to be modified in order to display the larger exit message satisfactorily.

fmincon Interior-Point Algorithm Robust to Certain Errors

The `fmincon` interior-point algorithm attempts to continue when a user-supplied objective or constraint function returns `Inf`, `NaN`, or a complex result. For more information, see fmincon Interior Point Algorithm.

Changes in quadprog

The large-scale `quadprog` algorithm now uses the `TolFun` and `MaxIter` tolerances for deciding when to end iterations when there are only linear equality constraints, instead of the `TolPCG` and `MaxPCGIter` tolerances.

The `quadprog` `output` structure now contains the `constrviolation` field, which reports the maximum constraint function at the final point.

Compatibility Considerations

For large-scale linear equality constrained problems, the default values of the tolerances are much tighter than before, so `quadprog` can take more iterations, but the resulting solution should be more accurate.

Changes in linprog

The large-scale interior-point algorithm of `linprog` now has a backtracking mechanism for the case of stalling, and performs LDL factorization when there is rank deficiency. For more information, see Large Scale Linear Programming.

The `linprog` `output` structure now contains the `constrviolation` field, which reports the maximum constraint function at the final point.

Compatibility Considerations

The interior-point algorithm of `linprog` might arrive at different solutions than before, and can solve more problems than before.

Multiobjective optimValues Changes

The `optimValues` structure, used by output functions, has two new fields to better reflect the state of multiobjective solvers:

- For fgoalattain, the `optimValues.attainfactor` field contains the value of γ, the attainment factor.
- For fminimax, the `optimValues.maxfval` field contains the value $\max_i F_i$, where F is the vector of objectives.

Furthermore, the value stored in `optimValues.fval` has changed. Now `optimValues.fval` contains the vector F of objective function values. For a complete description of the current `optimValues` structure, see Fields in optimValues.

Compatibility Considerations

User code that uses the `optimValues.fval` field within an output function in `fgoalattain` and `fminimax` might need to be updated to avoid errors

R2009a

Version: 4.2

New Features

Bug Fixes

Compatibility Considerations

Parallel Gradient Estimation Available in fmincon Interior-Point Algorithm

The `fmincon` solver's `interior-point` algorithm can now compute finite differences in parallel in order to speed the estimation of gradients. For details on how to use this parallel gradient estimation, see the Parallel Computing for Optimization chapter in the User's Guide.

Enhanced Exit Messages in Selected Solvers

Solvers print exit messages by default at the end of their runs. The exit messages are different in R2009a for several solvers, and the messages have been enhanced with new functionality. The following sections describe the new features and changes. There is more information in the Exit Flags and Exit Messages section of the User's Guide.

The following solvers have enhanced exit messages:

- `fgoalattain`
- `fmincon`
- `fminimax`
- `fminunc`
- `fseminf`

Links to More Information Window

The enhanced exit messages include hyperlinks within their exit messages. These hyperlinks bring up a window containing further information about the terms used in the exit messages.

Link for More Detail in Command Window

A `<stopping criteria details>` hyperlink may appear at the end of an exit message, depending on the solver and setting of the `Display` option. This link causes the solver to print more detail about the exit conditions to the MATLAB Command Window.

New Display Option Values Control Default Detail

There are new values of the `Display` option to control whether detailed exit messages appear instead of the default (simpler) messages. The new values are:

- `'final-detailed'`

- `'iter-detailed'`
- `'notify-detailed'`

These settings have the same effect as the corresponding settings without `'-detailed'`, but give detailed exit messages instead of the default exit messages. For solvers without the new exit messages, the `'-detailed'` options give the same behavior as without `'-detailed'`.

Messages in Output Structure

For solvers with enhanced exit messages, the `message` field of the output structure contains both the default (simpler) and the detailed exit messages, separated by a line of text stating `Stopping criteria details:`. The message field does not contain hyperlinks; it contains only text.

Compatibility Considerations

For solvers with enhanced exit messages, the content of `output.message` contains many more characters than before. User code that relies on this field may need to be modified in order to display the larger exit message satisfactorily.

Change in linprog Simplex Algorithm

The simplex algorithm of `linprog` now detects when there is no progress in the solution process. It attempts to continue by performing bound perturbation.

Compatibility Considerations

The simplex algorithm of `linprog` might arrive at different solutions than before, and can solve more problems than before.

Change in fminunc Exit Flag

One exit flag in the `fminunc` medium-scale solver was changed from –2 to 5. This flag appears when the solver predicts a change in function value at the next step in its iterations will be less than the `TolFun` tolerance. This condition can occur at a relative minimum, which should be reported by a positive flag.

Compatibility Considerations

This change might cause users (or code) that examine exit flags to evaluate a result more favorably than previously, since positive exit flags represent normal termination of solvers.

New demos

There are two new demos:

- A demo showing how to use Symbolic Math Toolbox™ functions to help calculate gradients and Hessians. Run the demo at the MATLAB command line by entering `echodemo symbolic_optim_demo`.
- A demo showing how to use `fseminf` for investigating the effect of parameter uncertainty. Run the demo at the MATLAB command line by entering `echodemo airpollution`.

Furthermore, the optimization tutorial demo now shows how to include extra parameters. Run the demo at the MATLAB command line by entering `echodemo tutdemo`.

R2008b

Version: 4.1

New Features

Bug Fixes

Compatibility Considerations

fsolve, lsqcurvefit, lsqnonlin Algorithm and Options Changes

- The Levenberg-Marquardt algorithm was refactored in the solvers fsolve, lsqcurvefit and lsqnonlin. It is now a more standard implementation, that accepts and preserves sparse Jacobians.
- Choose between the algorithms used in fsolve, lsqcurvefit and lsqnonlin using the new `Algorithm` option.
- There is a new `ScaleProblem` option that can sometimes help the Levenberg-Marquardt algorithm converge.
- The default fsolve algorithm, `'trust-region-dogleg'`, has been validated to work with sparse Jacobians.

Compatibility Considerations

- The refactored Levenberg-Marquardt algorithm can cause fsolve, lsqcurvefit and lsqnonlin to yield different answers than before.
- The previous way of choosing the algorithm at the command line was to set the `LargeScale` option to `'on'` or `'off'`, and, for all solvers but `fsolve`, to set the `LevenbergMarquardt` option to `'on'` or `'off'`. For `fsolve`, in addition to the `LargeScale` option, you needed to set the `NonlEqnAlgorithm` option appropriately. `LargeScale`, `NonlEqnAlgorithm`, and `LevenbergMarquardt` are now ignored, except when choosing to use the Gauss-Newton algorithm.
- The Gauss-Newton algorithm warns that soon it may no longer be available.
- The default value of the `MaxFunEvals` option in the refactored Levenberg-Marquardt algorithm is now 200*`numberOfVariables`; the previous value was 100*`numberOfVariables`.

Optimization Tool Enables Parallel Functionality

You can now access built-in parallel functionality in Optimization Tool for relevant Optimization Toolbox solvers and, if licensed, Global Optimization Toolbox solvers. The option is available when you have a license for Parallel Computing Toolbox™ functions.

Central Finite Differences Available in Selected Solvers

The following solvers can now use central finite differences for gradient estimation:

- fgoalattain

- fmincon
- fminimax
- fminunc
- fseminf

The `fmincon` active-set algorithm and `fminunc` medium-scale algorithm gained central finite differences this release. The `fmincon` interior-point algorithm already had them, and the trust-region-reflective algorithm for both solvers requires a user-supplied gradient, so does not use finite differences.

To use central finite differences, use `optimset` to set the `FinDiffType` option to `'central'` instead of the default `'forward'`. This causes the solver to estimate gradients by formulae such as

$$\nabla f(x) \approx \left[\frac{f(x+\Delta_1 e_1) - f(x-\Delta_1 e_1)}{2\Delta_1}, \ldots, \frac{f(x+\Delta_n e_n) - f(x-\Delta_n e_n)}{2\Delta_n} \right],$$

instead of

$$\nabla f(x) \approx \left[\frac{f(x+\Delta_1 e_1) - f(x)}{\Delta_1}, \frac{f(x+\Delta_2 e_2) - f(x)}{\Delta_2}, \ldots, \frac{f(x+\Delta_n e_n) - f(x)}{\Delta_n} \right].$$

Central finite differences take twice as many function evaluations as forward finite differences, but are usually much more accurate.

Central finite differences can work in parallel for gradient estimation in fgoalattain, fmincon active-set algorithm, and fminimax. For details on how to use this parallel gradient estimation, see the Parallel Computing for Optimization chapter in the User's Guide.

lsqnonneg Refactored

lsqnonneg was refactored. It can now use sparse matrices, and it preserves sparsity during its execution.

Finite Difference Algorithm Tweaked

A subroutine for gradient estimation by forward finite differences in nonlinear solvers had a bug that affected it when the current point `x` had a component with the value

0. Forward finite differences are typically calculated with a step size proportional to `sqrt(eps)`, which is about $1.5*10^{-8}$. When a component of `x` was 0, the step size would instead be proportional to `DiffMinChange`, which has a default value of 10^{-8}. There is now no difference in step size when `x` is 0.

Compatibility Considerations

Nonlinear solvers can run slightly differently whenever an iteration causes a component of `x` to be zero, and gradients are estimated by forward finite differences.

DerivativeCheck Tolerance Changed

The `DerivativeCheck` option enables you to ascertain whether the derivative (gradient) functions that you supply for objective or constraint functions give *approximately* the same values as those estimated by a solver using finite differences. The meaning of "approximately" has changed. Now it means the relative error of each component of the gradient is less than 10^{-6}, unless the size of an analytically given component is smaller than 1, in which case it means the absolute difference is less than 10^{-6}. Previously, the gradients were considered approximately equal if the maximum absolute error in any component of the gradient was less than (10^{-6} * norm of analytic gradient) + 10^{-5}.

Compatibility Considerations

Some problems will now report violations of the `DerivativeCheck` condition, when previously they would not.

R2008a

Version: 4.0

New Features

Bug Fixes

Compatibility Considerations

Parallel Computing Toolbox Support in fmincon, fminimax, and fgoalattain

fmincon, fminimax, and fgoalattain can take finite differences in parallel in order to speed the estimation of gradients. For details on how to use this parallel gradient estimation, see the Parallel Computing for Optimization chapter in the User's Guide.

Combined and Extended optimtool

The Global Optimization Toolbox GUIs `gatool` and `psearchtool` have been combined into the Optimization Tool GUI. To access these GUIs, type `optimtool` at the command line, and choose the appropriate solver.

Furthermore, three new Global Optimization Toolbox solvers were added to Optimization Tool: gamultiobj, simulannealbnd, and `threshacceptbnd`.

Optimization Tool shows Global Optimization Toolbox solvers only if these solvers are licensed.

New fmincon Solver, New Option Algorithm for fmincon, Option LargeScale Changed

The new interior-point algorithm is a large-scale algorithm that can handle all types of constraints. It has several new options, explained in the fmincon function reference pages.

fmincon now has three algorithms. Choose between them by setting the new option `Algorithm` to:

- `'trust-region-reflective'` (formerly known as `'large scale'`)
- `'active-set'` (formerly known as `'medium scale'`)
- `'interior-point'`

By default, `Algorithm` = `'trust-region-reflective'`.

Compatibility Considerations

The previous way of choosing the algorithm at the command line was to set option `LargeScale` to `'on'` or `'off'`. `LargeScale` is now ignored, except when `LargeScale`

= 'off' and Algorithm = 'trust-region-reflective'. In this case, the 'active-set' algorithm is used, to minimize backward incompatibility.

External Interface to KNITRO Libraries

Use the new ktrlink function to call KNITRO optimization libraries from Ziena Optimization, Inc. KNITRO libraries must be purchased separately. The External Interface chapter of the User's Guide describes the ktrlink function.

Default PrecondBandWidth = Inf in lsqcurvefit, lsqnonlin, and fsolve

The default value of the PrecondBandWidth option changed from 0 to Inf for the lsqcurvefit, lsqnonlin, and fsolve solvers. This change was beneficial in the vast majority of tested problems.

In Optimization Tool, the default in **Algorithm settings > Subproblem algorithm** is now **Cholesky factorization**, instead of **Preconditioned CG** = 0.

Compatibility Considerations

The new default can lead to slower performance for problems with high-dimensional nonlinearities. If this happens, change the default to another value such as 0 (the previous default).

New Option TolConSQP with Incompatible Default Value

The new TolConSQP option exposes a parameter that was fixed at eps before. The parameter is used in the fmincon, fminimax, fgoalattain, and fseminf solvers.

Compatibility Considerations

The new default value is TolConSQP = 1e–6. This did not affect a vast majority of tested cases, and was beneficial in some. If you want exactly the same behavior as before, set TolConSQP = eps using optimset.

Field constrviolation in Output Structure

The constrviolation field now exists in the output structure for the fgoalattain, fmincon, fminimax, and fseminf functions; it measures the nonlinear constraint violation.

R2007b

Version: 3.1.2

Bug Fixes

R2007a

Version: 3.1.1

New Features

Bug Fixes

Compatibility Considerations

Changes to Outputs of Multiobjective Solvers

- fminimax now returns the value of `max(fval)` in the output `maxfval`.
- The iterative display of fminimax and fgoalattain have changed.

Compatibility Considerations

- The third output argument of the solver `fminimax`, `maxfval`, is described in the documentation as the maximum of the objective functions in the input `fun` evaluated at the solution `x`, that is, `max(fval)`. Before this release, `fminimax` actually returned the maximum of the objective functions in the reformulated minimax problem internally constructed by the algorithm. This value was typically very close to, but not necessarily equal to, `max(fval)`. `fminimax` now returns the exact value of `max(fval)` in the output `maxfval`.
- The iterative display for `fminimax` includes a new column with header `Objective value` that reports the objective function value of the nonlinear programming reformulation of the minimax problem. The column header `Max{F,constraints}` has been changed to `Max constraint`, and the column now contains the maximum violation among all constraints, both internally constructed and user-provided.

  The iterative display for `fgoalattain` now shows the value of the attainment factor in the `Attainment factor` column. A new column, `Max constraint`, contains the maximum violation among all constraints, both internally constructed and user-provided.

R2006b

Version: 3.1

New Features

Bug Fixes

Compatibility Considerations

New Optimization Tool

The Optimization Tool is a graphical user interface (GUI) for performing common optimization tasks with the Optimization Toolbox. Using the optimtool, you can do the following:

- Select a solver and define your optimization problem.
- Set and inspect optimization options and their default values.
- Run problems and visualize results.
- Import and export problem definitions, algorithm options, and results between the MATLAB workspace and the Optimization Tool.
- Automatically generate M-code to capture, automate, and recreate your problem.
- Access built-in help.

Plot Functions Option Added

You can now specify the `PlotFcns` option in the `optimset` function or using the Optimization Tool for use with an Optimization Toolbox solver. With this option, you can plot various measures of progress while the algorithm executes. You can select from several predefined plots, or you can write your own.

Output Function Option Enhanced to Accept Multiple Functions

You can now specify more than one output function in the `OutputFcn` option.

Changes to the Output Function

The output function input `x` and fields in the `optimValues` structure have the following changes that address bugs in previous releases:

- `residual` now returns the residual vector for `lsqnonlin` and `lsqcurvefit`.
- `resnorm` contains the sum of squares and has been added for `lsqnonlin` and `lsqcurvefit`. The previous field `fval` has been removed for these functions.
- `procedure` has been removed for `lsqnonlin`, `lsqcurvefit`, and `fsolve`.
- `x` now returns the expected shape and size for `fgoalattain` and `fminimax`.

Compatibility Considerations

The above changes to the input `x` and `optimValues` structure have the following compatibility considerations in the output function:

- If you have references to the `residual` in a previous version, note that the value of this field has changed for `lsqnonlin` and `lsqcurvefit`. This fixes the problem addressed by the bug report S-289285.
- Any references to `fval` for `lsqnonlin` and `lsqcurvefit` need to be updated to `resnorm`. This fixes the problem addressed by the bug report S-289285.
- Any references to `procedure` for `lsqnonlin` and `lsqcurvefit` need to be removed. This fixes the problem addressed by the bug report S-291974.
- Previously, for `fgoalattain` and `fminimax`, `x` returned a column vector with an additional last element. If you have references to the values for `x` in a previous version, the extra element must be removed and the output vector may need to be reshaped. This fixes the problem addressed by the bug report S-315658.

R2006a

Version: 3.0.4

Bug Fixes

R14SP3

Version: 3.0.3

New Features

Bug Fixes

Notify Parameter Added to Display Option for Five Functions

You can now set the optimization option `Display` to `'notify'` for the functions fmincon, fminunc, fminimax, fgoalattain, and fseminf. When `Display` is set to `'notify'`, the output is displayed only if the function does not converge.