Release Notes for Optimization Toolbox™

How to Contact MathWorks

www.mathworks.com	Web
comp.soft-sys.matlab	Newsgroup
www.mathworks.com/contact_TS.html	Technical Support

suggest@mathworks.com	Product enhancement suggestions
bugs@mathworks.com	Bug reports
doc@mathworks.com	Documentation error reports
service@mathworks.com	Order status, license renewals, passcodes
info@mathworks.com	Sales, pricing, and general information

508-647-7000 (Phone)

508-647-7001 (Fax)

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

For contact information about worldwide offices, see the MathWorks Web site.

Release Notes for Optimization Toolbox™

© COPYRIGHT 2005–2012 by The MathWorks, Inc.

Trademarks

Patents

Contents

R2008b

R2008a

R2007b

R2007a

R2012b

Version: 6.2.1
New Features: No
Bug Fixes: Yes

Changing default algorithms for `fmincon` and `quadprog`
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

Enhanced Robustness in fminunc
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

New Quadratic Programming Algorithm
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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);
```

23

Output Structure Tweak
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

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 demo on modeling and solving optimization problems. View the first part from the MATLAB® command line by entering

```
playbackdemo('Optimization-Modeling-1','toolbox/optim/web/demos');
```

View the second part by entering

```
playbackdemo('Optimization-Modeling-2-Converting-To-Solver-Form',...
'toolbox/optim/web/demos');
```

31

R2010a

Version: 5.0
New Features: Yes
Bug Fixes: Yes

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
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

Enhanced Exit Messages in Selected Solvers
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

fsolve, lsqcurvefit, lsqnonlin Algorithm and Options Changes
Compatibility Considerations: Yes

- 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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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: Yes
Bug Fixes: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
Compatibility Considerations: Yes

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
New Features: No
Bug Fixes: Yes

R2007a

Version: 3.1.1
New Features: Yes
Bug Fixes: Yes

Changes to Outputs of Multiobjective Solvers
Compatibility Considerations: Yes

- `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: Yes
Bug Fixes: Yes

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
Compatibility Considerations: Yes

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
New Features: No
Bug Fixes: Yes

R14SP3

Version: 3.0.3
New Features: Yes
Bug Fixes: Yes

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.