Optimization Toolbox Release Notes

The Optimization Toolbox Release Notes describe the changes introduced in the latest version of the Optimization Toolbox. For a description of the changes introduced in Version 3.0.3, see Chapter 1, "Optimization Toolbox 3.0.3 Release Notes".

The Optimization Toolbox Release Notes also provide information about recent versions of the product, in case you are upgrading from a version that was released prior to Release 14 with Service Pack 2.

- Chapter 2, "Optimization Toolbox 3.0.2 Release Notes"
- Chapter 3, "Optimization Toolbox 3.0.1 Release Notes"
- Chapter 4, "Optimization Toolbox 3.0 Release Notes"
- Chapter 5, "Optimization Toolbox 2.3 Release Notes"
- Chapter 6, "Optimization Toolbox 2.2 Release Notes"
- Chapter 7, "Optimization Toolbox 2.1.1 Release Notes"
- Chapter 8, "Optimization Toolbox 2.1 Release Notes"

Contents

Optimization Toolbox 2.3 Release Notes

5

Optimization Toolbox 2.2 Release Notes

6

Optimization Toolbox 2.1.1 Release Notes

7

Optimization Toolbox 2.1 Release Notes

8

Optimization Toolbox 3.0.3 Release Notes

New Features

The Optimization Toolbox Version 3.0.3 has the following enhancement.

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.

Major Bug Fixes

To view major bug fixes made in R14 with Service Pack 3 for Optimization Toolbox 3.0.3, use the Bug Reports interface on the MathWorks Web site.

Note If you are not already logged into Access Login when you link to the Bug Reports interface, you will be prompted to log in or create an Access Login account.

After you are logged in, use this Bug Reports link. You will see the bug report list for the Optimization Toolbox. The report is sorted with fixed bugs listed first, and then open bugs.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Optimization Toolbox 3.0.2 Release Notes

Major Bug Fixes

To view major bug fixes made in R14 with Service Pack 2 for Optimization Toolbox 3.0.2, use the Bug Reports interface on the MathWorks Web site.

Note If you are not already logged into Access Login when you link to the Bug Reports interface, you will be prompted to log in or create an Access Login account.

After you are logged in, use this Bug Reports link. You will see the bug report list for the Optimization Toolbox. The report is sorted with fixed bugs listed first, and then open bugs.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Optimization Toolbox 3.0.1 Release Notes

Major Bug Fixes

The Optimization Toolbox 3.0.1 includes bug fixes made since Version 3.0. This section describes the particularly important Version 3.0.1 bug fixes.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Optimization Toolbox 3.0 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the Optimization Toolbox 3.0.

If you are upgrading from a release earlier than Release 13 with Service Pack 1, then you should also see "New Features" on page 5-2 in the Optimization Toolbox 2.3 Release Notes.

The Optimization Toolbox 3.0 contains the following new features:

- "Binary Integer Programming" on page 4-2
- "New Medium Scale Algorithm for fminunc" on page 4-3
- "Output hessian Now Returns Finite-Difference Approximation" on page 4-3
- "New Values for exitflag" on page 4-3
- "New Field for Output Argument" on page 4-3
- "New Option FunValCheck" on page 4-4
- "New Way to Provide Parameters to Functions Called by Optimization Functions" on page 4-4

Binary Integer Programming

Version 3.0 provides a new function, `bintprog`, that solves binary integer progamming problems of the form

$$\min_{x} f^{T}x \qquad \text{such that} \qquad A \cdot x \le b$$
$$Aeq \cdot x = beq$$

where x is required to be binary vector, that is, a vector of 0s and 1s. For example, the following commands

```
f = [-9; -5; -6; -4];
A = [6 3 5 2; 0 0 1 1; -1 0 1 0; 0 -1 0 1];
b = [9 1 0 0];
```

```
x = bintprog(f,A,b)
```

return the solution

```
x =

     1
     1
     0
     0
```

Type help bintprog for information on how to use the function.

New Medium Scale Algorithm for fminunc

Version 3.0 provides an improved algorithm for the unconstrained optimization function fminunc when you solve medium-scale problems.

Output hessian Now Returns Finite-Difference Approximation

In Version 3.0, the medium-scale algorithm for fminunc returns a finite-difference approximation for the output argument hessian. This is more accurate than the quasi-Newton approximation used in previous releases.

New Values for exitflag

The output argument exitflag now returns more values that provide more information about why an optimization function terminated. See the command-line help for an individual optimization functions for information on the values of exitflag for that function, or see "Output Arguments" in the online documentation for the Optimization Toolbox.

New Field for Output Argument

The argument output now has a new field, output.message, which is a string describing the reason the optimization terminated.

New Option FunValCheck

The new option FunValCheck returns an error message and terminates the optimization if the objective function or other functions called by the solver return Inf, Nan, or complex values.

To use this option, set

```
options = optimset('FunValCheck', 'on');
```

Note FunValCheck does not return a warning for Inf when used with fminbnd, fminsearch, or fzero, which handle Inf appropriately.

New Way to Provide Parameters to Functions Called by Optimization Functions

In Version 3.0, you can provide additional parameters to any function called by an optimization function, such as the objective function, using anonymous or nested functions. This is a more flexible approach that avoids passing these parameters as input arguments to the optimization function. See Avoiding Global Variables Via Anonymous and Nested Functions in the online documentation for the Optimization Toolbox for examples.

Major Bug Fixes

The Optimization Toolbox 3.0 includes several bug fixes made since Version 2.3. This section describes the particularly important Version 3.0 bug fixes.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

If you are upgrading from a release earlier than Release 13 with Service Pack 1, then you should also see "Major Bug Fixes" on page 5-3 in the Optimization Toolbox 2.3 Release Notes.

Optimization Toolbox 2.3 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the Optimization Toolbox 2.3.

If you are upgrading from a release earlier than Release 13, then you should also see "New Features" on page 6-2 of the Optimization Toolbox 2.2 Release Notes.

Output Function

In Version 2.3, you can now create an output function that an optimization function calls at each iteration of its algorithm. You can use the output function to obtain information about the data at each iteration or to stop the algorithm based on the current values of the data. See "Calling an Output Function Iteratively" for an example of how to use the output function.

Revised Simplex Algorithm

You now have the option of using a revised simplex algorithm, which is new for Version 2.3, when calling the linear programming function linprog. See "Revised Simplex Algorithm" for a description of the algorithm.

DerivativeCheck Option

In Version 2.3, the DerivativeCheck option is available for all methods (medium- and large-scale) in fmincon, fminunc, lsqnonlin, lsqcurvefit, fsolve, fseminf, fminimax, and fgoalattain. In previous versions, DerivativeCheck was not available for the large-scale methods of fminunc, fmincon, lsqnonlin, lsqcurvefit, and fsolve.

Major Bug Fixes

The Optimization Toolbox 2.3 includes several bug fixes made since Version 2.2. This section describes the particularly important Version 2.3 bug fixes.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Optimization Toolbox 2.2 Release Notes

New Features

This section summarizes the new features and enhancements introduced in the Optimization Toolbox 2.2.

If you are upgrading from a release earlier than Release 12.1, then you should also see "New Features" on page 7-2 of the Optimization Toolbox 2.1.1 Release Notes.

New fsolve Default Algorithm

The `fsolve` function, which is used to solve systems of nonlinear equations, has a new default algorithm for medium-scale systems where the number of equations is equal to the number of variables. The new algorithm uses a trust-region dogleg method that has improved convergence properties over the previous default method. It is based on the algorithm described in [1].

Medium-Scale Is Now the Default Method

In keeping with the new default trust-region dogleg algorithm, `fsolve` now defaults to the medium-scale method. The default for the `'LargeScale'` parameter is changed to `'off'`.

To use the large-scale `fsolve` method, which was the default in Versions 2.0 and 2.1, specify the large-scale method using code similar to

```
options = optimset('LargeScale','on');
xsol = fsolve(myfun,x0,options);
```

New 'NonlEqnAlgorithm' Parameter

A new 'NonlEqnAlgorithm' `fsolve` parameter enables you to choose the Levenberg-Marquardt or Gauss-Newton algorithm over the trust-region dogleg algorithm. 'NonlEqnAlgorithm' has three options: `'dogleg'` (default), `'lm'` (Levenburg-Marquart), and `'gn'` (Gauss-Newton). Specify the use of an algorithm other than the default with code similar to

```
options = optimset('NonlEqnAlgorithm','gn');
xsol = fsolve(myfun,x0,options);
```

See "Upgrading from an Earlier Release" on page 6-5 for more information.

References

[1] Moré, J.J., B.S. Garbow, and K.E. Hillstrom, *User Guide for MINPACK - 1*, Argonne National Laboratory, Rept. ANL-80-74, 1980

Function Summary

Version 2.2 of the Optimization Toolbox provides the following functions with new or changed capabilities.

Optimization Functions with New or Changed Capabilities

Function	Enhancement or Change
fsolve	The trust-region dogleg algorithm is now the default for medium-scale systems of equations where the number of equations is equal to the number of variables. See "New fsolve Default Algorithm" on page 6-2 for more information.
optimset	A new 'NonlEqnAlgorithm' parameter, which applies only to fsolve, enables you to choose the Levenberg-Marquardt or Gauss-Newton algorithm over the new default trust-region dogleg algorithm.

Major Bug Fixes

The Optimization Toolbox 2.2 includes several bug fixes made since Version 2.1.1. This section describes the particularly important Version 2.2 bug fixes.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Upgrading from an Earlier Release

This section describes the upgrade issues involved in moving from the Optimization Toolbox 2.1.1 to Version 2.2.

If you are upgrading from a version earlier than 2.1.1, then you should also see "Upgrading from an Earlier Release" on page 8-5 in the Optimization Toolbox 2.1. Release Notes.

fsolve Now Defaults to New Medium-Scale Method

In keeping with the new default trust-region dogleg algorithm, `fsolve` now defaults to the medium-scale method. The default for the `'LargeScale'` parameter is changed to `'off'`.

Using the Large-Scale Method

If you want to continue using the large-scale `fsolve` method, which was the default in Versions 2.0 and 2.1, you must explicitly specify the large-scale method using code similar to

```
options = optimset('LargeScale','on');
xsol = fsolve(myfun,x0,options);
```

Using the Gauss-Newton Method

In prior releases to get the Gauss-Newton method you did:

```
options = optimset('LargeScale','off',...
                   'LevenbergMarquardt','off');
```

For this release you do:

```
options = optimset('LargeScale','off',...
                   'NonlEqnAlgorithm','gn');
```

Note It is unnecessary to set `'LargeScale'` to `'off'`, unless you have previously set it `'on'`.

Using the Levenberg-Marquardt Method

In prior releases to get the Levenberg-Marquardt method you used

```
options = optimset('LargeScale','off',...
                   'LevenbergMarquardt','on');
```

For this release you use

```
options = optimset('largescale','off',...
                   'NonlEqnAlgorithm','lm');
```

Use playshow Command to Run Slideshow Style Demos

Starting in Release 13, to run slideshow style demos from the command line, you must use the playshow command. For example,

```
playshow circustent
```

You can continue to run other styles of demos from the command line by typing just the demo name. Three Optimization Toolbox demos are affected: circustent, molecule, and optdeblur.

Optimization Toolbox 2.1.1 Release Notes

New Features

This section introduces the new features and enhancements added in the Optimization Toolbox 2.1.1 since the Optimization Toolbox 2.1 (Release 12.0).

For information about Optimization Toolbox features that are incorporated from Version 2.1, see "New Features" on page 8-2 in the Optimization Toolbox 2..

Support for Large Problems that Are Not Well-Scaled

In Version 2.1.1, large-scale finite differencing is improved numerically to handle cases when an optimization problem is not well-scaled. These changes potentially improve the speed and accuracy of results when using the large-scale versions of lsqnonlin, lsqcurvefit, fsolve, fmincon and fminunc, particularly if the objective function is not well-scaled.

Major Bug Fixes

The Optimization Toolbox includes several bug fixes, including the following descriptions of particularly important bug fixes for both Version 2.1 (Release 12.0) and Version 2.1.1 (Release 12.1).

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the links provided.

Upgrading from an Earlier Release

There are no upgrade issues moving from the Optimization Toolbox 2.1 to the Optimization Toolbox 2.1.1.

See "Upgrading from an Earlier Release" on page 8-5 for upgrade issues involved in moving from the Optimization Toolbox 2.0 (Release 11.0) to the Optimization Toolbox 2.1 (Release 12.0).

Optimization Toolbox 2.1 Release Notes

New Features

This section introduces the new features and enhancements added in the Optimization Toolbox 2.1 since the Optimization Toolbox 2.0 (Release 11.0):

- Overall toolbox speed is improved.
- Functions that take a function as an argument now support the use of function handles.
- Large structured problems are supported.
- Several other existing functions have been enhanced.

Toolbox Speed

By improving the speed of `optimset` and `optimget`, the overall speed of the toolbox is improved. Applications that call these functions repeatedly should exhibit improved execution time.

Function Handles

For any optimization function that expects a function as an argument, you can now specify that argument as a function handle. These optimization functions also accept additional parameters, which they pass to the passed-in function.

For information about function handles, see the `function_handle` (@), `func2str`, and `str2func` reference pages, and the "Function Handles" section of "Programming and Data Types" in the MATLAB documentation.

Large Structured Problems

The functions `fmincon`, `fminunc`, `fsolve`, `lsqcurvefit`, `lsqlin`, `lsqnonlin`, and `quadprog` now support solving large structured problems, i.e., problems that have large dense Hessian or Jacobian matrices that you do not want to form explicitly, but for which Hessian-matrix (or Jacobian-matrix) products are efficient to compute.

Two new `options` parameters, `HessMult` and `JacobMult`, provide access to this new feature.

See "Large-Scale Examples" in the Optimization Toolbox User's Guide, and the respective function reference pages for more information.

Functions with New or Changed Capabilities

Function	New or Changed Capability
fminbnd, fminsearch, fzero, lsqnonneg	A new Display options parameter value, 'notify', displays output only if the function does not converge. For these functions, 'notify' is the new default.
fmincon, fminunc, quadprog	A new options parameter, HessMult, enables you to provide a function that computes the Hessian-matrix product for large structured problems.
fsolve, lsqcurvefit, lsqlin, lsqnonlin	A new options parameter, JacobMult, enables you to provide a function that computes the Jacobian-matrix product for large structured problems.

Major Bug Fixes

The Optimization Toolbox includes several bug fixes, including the following descriptions of particularly important bug fixes.

If you are viewing these Release Notes in PDF form, refer to the HTML form of the Release Notes, using either the Help browser or the MathWorks Web site, and use the link provided.

Upgrading from an Earlier Release

The Optimization Toolbox uses an options structure to access various algorithm options. The Version 2.0 (Release 11) version of the options structure may be incompatible with the 2.1 (Release 12.0) version in some cases. In particular, if you have saved an options structure in a MAT-file from Release 11 and load it into Release 12.0 you may get an error similar to

```
??? Error using ==> subsref
Reference to non-existent field 'MaxSQPIter'.
```

To avoid this error, wrap a call to `optimset` around the old options structure to update it. For example,

```
optionsnew = optimset(optionsold);
```

and then use the `optionsnew` options structure.

If you were using `optimset` to create your options structure, as opposed to loading it from an older MAT-file, you should not need to make any changes.