

Release Notes for Financial Instruments 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 Financial Instruments Toolbox™

© COPYRIGHT 2012 by The MathWorks, Inc.

The software described in this document is furnished under a license agreement. The software may be used or copied only under the terms of the license agreement. No part of this manual may be photocopied or reproduced in any form without prior written consent from The MathWorks, Inc.

FEDERAL ACQUISITION: This provision applies to all acquisitions of the Program and Documentation by, for, or through the federal government of the United States. By accepting delivery of the Program or Documentation, the government hereby agrees that this software or documentation qualifies as commercial computer software or commercial computer software documentation as such terms are used or defined in FAR 12.212, DFARS Part 227.72, and DFARS 252.227-7014. Accordingly, the terms and conditions of this Agreement and only those rights specified in this Agreement, shall pertain to and govern the use, modification, reproduction, release, performance, display, and disclosure of the Program and Documentation by the federal government (or other entity acquiring for or through the federal government) and shall supersede any conflicting contractual terms or conditions. If this License fails to meet the government's needs or is inconsistent in any respect with federal procurement law, the government agrees to return the Program and Documentation, unused, to The MathWorks, Inc.

Trademarks

MATLAB and Simulink are registered trademarks of The MathWorks, Inc. See www.mathworks.com/trademarks for a list of additional trademarks. Other product or brand names may be trademarks or registered trademarks of their respective holders.

Patents

MathWorks products are protected by one or more U.S. patents. Please see www.mathworks.com/patents for more information.

R2012b

Merge of Fixed-Income Toolbox and Financial Derivatives Toolbox to Financial Instruments Toolbox	2
Cap and floor floating-rate note pricing using trees	3
Forward-swap pricing using trees or term structure	4
Functions for fitting and extracting calibrated parameters from <code>IRFunctionCurve</code> objects	5
LIBOR market model example	6
Counterparty credit risk example	7
Conversion of error and warning message identifiers	8

R2012b

Version: 1.0
New Features: Yes
Bug Fixes: No

Merge of Fixed-Income Toolbox and Financial Derivatives Toolbox to Financial Instruments Toolbox

Compatibility Considerations: Yes

Fixed-Income Toolbox™ and Financial Derivatives Toolbox™ are merged into the new product Financial Instruments Toolbox™.

Cap and floor floating-rate note pricing using trees

Support for pricing capped, collared, and floored floating-rate notes using the CapRate and FloorRate arguments.

Function	Purpose
floatbybdt	Price a capped floating-rate note using a Black-Derman-Toy interest-rate tree.
floatbyhjm	Price a capped floating-rate note using a Heath-Jarrow-Morton interest-rate tree.
floatbyhw	Price a capped floating-rate note using a Hull-White interest-rate tree.
floatbybk	Price a capped floating-rate note using a Black-Karasinski interest-rate tree.
instfloat	Create a capped floating-rate note instrument.
instadd	Add capped floating-rate note instruments to a portfolio.

Forward-swap pricing using trees or term structure

Support for interest-rate forward swaps using the new `StartDate` argument to define the future date for the swap instrument.

Function	Purpose
<code>swapbyzero</code>	Price a bond using a set of zero curves.
<code>swapbybdt</code>	Price a forward swap using a Black-Derman-Toy interest-rate tree.
<code>swapbyhjm</code>	Price a forward swap using a Heath-Jarrow-Morton interest-rate tree.
<code>swapbyhw</code>	Price a forward swap using a Hull-White interest-rate tree.
<code>swapbybk</code>	Price a forward swap using a Black-Karasinski interest-rate tree.
<code>instswap</code>	Create a forward swap instrument.
<code>instadd</code>	Add forward swap instruments to a portfolio.

Functions for fitting and extracting calibrated parameters from `IRFunctionCurve` objects

New enhancements for `IRFunctionCurve` object, including the ability to get calibrated parameters, the ability to specify linear inequality parameter constraints, and support for curve type in `fitSmoothingSpline` to be forward, zero, and discount.

LIBOR market model example

New example for mortgage prepayment that uses a LIBOR market model to generate interest-rate evolutions. For more information, see “Prepayment Modeling with a Two Factor Hull White Model and a LIBOR Market Model”.

Counterparty credit risk example

New example for computing the unilateral Credit Value (Valuation) Adjustment (CVA) for a bank holding a portfolio of vanilla interest-rate swaps with several counterparties. For more information, see “Counterparty Credit Risk and CVA”.

Conversion of error and warning message identifiers

Compatibility Considerations: Yes

For R2012b, error and warning message identifiers have changed in Financial Instruments 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, because Fixed-Income Toolbox and Financial Derivatives Toolbox merged to become Financial Instruments Toolbox, the `finfixed` and `finderiv` message identifiers have changed to `fininst`. If your code checks for `finfixed` or `finderiv` message identifiers, you must update it to check for `fininst` instead.

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

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

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`.