Automated Trading with MATLAB

Webinar: http://www.mathworks.com/wbnr68672

Automated trading is a complex and multi-dimensional problem; there are a large number of different challenges that need to be addressed and solved.

At its heart you need to be able to develop, build and test a robust trading algorithm, but this process requires you to solve a range of surrounding issues including data gathering, preparation and visualization, model development, backtesting, calibration, integration with existing systems and ultimately deployment.

The files and demo are for illustration/instructional purposes only. They are intended to show the steps involved in building an automated trading system in MATLAB, but does not imply the approach is sound, or suggest a viable investment strategy.

Requirements

This seminar is designed for use in MATLAB R2012a and later. Earlier versions of MATLAB will not have all of the functionality that this seminar requires. Furthermore, the Statistics, Optimization, Global Optimization, Financial, and Econometrics Toolboxes are required. The Parallel Computing Toolbox is highly recommended to speed up calculations; if it is not installed, then the "matlabpool" commands in demos 3 and 5 may need to be removed. You will also need to have X_Trader® and Quickfix/J installed to run demos 6 and 7. See steps 4 and 5 for directions.

Database Toolbox was used in the live seminar, but it is not required to run these files; see step 3 in the installation directions below.

Datafeed Toolbox (as well as some external software applications) is required to reproduce the "live" trading engine environment; see steps 4 and 5 in the installation directions below.

Directions for installing / setting up

1. Unzip all files from this archive, preserving the relative file structure.
2. All folders that are created (Charts, Data, gaFiles, Models, and TradingSystem) must be added to the MATLAB file path. You may right-click on the folders in the MATLAB Current Folder window and select "Add to path" or use commands like PATH or PATHTOOL to do this.
3. During the seminar, we show how to download data from a database. This requires several setup steps in order to create, define, and connect to the database. Rather than force this upon you, the code has instead been adapted to simply read from a MATLAB data file (oilData.mat).

   Users are welcome to add this data to the database of their choosing, at which point they can

use either the Visual Query Builder in Database Toolbox or the file Data\getMinuteDataFromDB.m as a starting point for reading the data into MATLAB. The section "Working with Data Sources" in the Database Toolbox documentation may be useful for setting up such a database.

4. Most of the files from this seminar require no external software to run, but if you wish to reproduce the trading engine environment (in the folder TradingSystem), then you must have both the Reuters playback engine and a Microsoft Messaging Queue (MSMQ) installed and initialized.

   The connection with Reuters is through the Reuters Market Data System (https://customers.reuters.com/Home/RMDS.aspx; contact Reuters for a license or a possible trial). Once RMDS is installed, consult the MATLAB Datafeed Toolbox documentation for directions on associating it with MATLAB.

   MSMQ is typically a standard component of Windows, although it may have to be installed before its first use. You can find resources on MSMQ on Microsoft's website (for instance, at http://msdn.microsoft.com/en-us/library/ms711472%28v=VS.85%29.aspx).

   An example of installation steps (in this case, on Windows 7) involves going to Control Panel -> Add/Remove Programs -> Add/Remove Windows Components, selecting "Message Queuing," and following the on-screen prompts.

   You may also need to manually start the service on subsequent Windows sessions, particularly if you receive errors that "Message Queue service is not available." Again, consult Microsoft's support for details on how to do this; typical steps are going to Control Panel -> Administrative Tools -> Services -> Message Queuing and choosing to "Start" the service.

   X_Trader®, from Trading Technologies was shown as a desktop order execution system and for in Demo 6. You can learn more about it and request trials from https://www.tradingtechnologies.com/xtrader/.

   Quickfix/J, an open-source FIX library, was shown as an alternative order execution system to X_Trader®. You will need to download the jar (Java) files from http://www.quickfixj.org/. The demo references the location c:\sandbox\java\quickfixj as the installation directory. It you install it in a different location, you will need to change runTradingAlog.m and Demo7 to reference the new location. This demo used version 1.5.2.

5. Once these other programs have been installed, then the following additional steps must be taken to make the playback engine aware of our data:
   a. Take note of the folder in which the Reuters playback engine ("playback.exe") is installed. It will be referred to as REUTERSROOT below.

b. Create a new folder in the Reuters playback directory: REUTERSROOT\data_files_brent. Copy the file BRENT from the TradingSystem folder into this new data files folder.
c. Copy the config_file_brent.txt file from the TradingSystem folder into REUTERSROOT. Note that there may already be config_files there corresponding to other playback files: this is fine as long as no file names match exactly.
d. Edit line 10 of TradingSystem\runTradingAlgo.m to reflect the REUTERSROOT directory location.

Directions for running

1. Files in the Charts, Data, gaFiles, and Models folders correspond to utility functions, helper functions, and data files. You may explore them at your leisure, but you will rarely need to call them directly. Instead, you may step through the main scripts (they start with the word "Demo"), which will call the other files as needed.
2. The TradingSystem folder contains all of the commands needed for the trading engine environment. It calls many of the same functions as before. The function "helloWorldNET.m" is a quick introduction to MATLAB's integration with the .NET environment, which is how we send messages to the MSMQ. Most files in this folder are again helper functions, this time called from the main function "runTradingAlgo.m". To run this file, take the following steps:
    a. Change the MATLAB current folder to the TradingSystem folder while ensuring that the other folders are still on the MATLAB file path.
    b. Ensure that TradingSystem is a "trusted location" for Office files or that macros are always enabled, or else the Excel spreadsheet will be unable to pick up trading signals from the MSMQ.
    c. Execute the command "conn = runTradingAlgo;"
    d. Several things will now happen: first, Excel will open "orderHistory.xlsm" and listen for trading signals on a messaging queue. Then, a separate command window will be opened by the Reuters playback engine. Within a few seconds, data will start to stream into MATLAB from Reuters every 100ms. Any messages in red on the MATLAB screen may be ignored if they are INFO messages from the Reuters engine.
    e. Within about 1 minute, the "burn-in" period for the trading strategies should end, and MATLAB will start sending BUY and SELL messages to the queue, where they will be picked up and displayed by the Excel spreadsheet.
    f. When you wish to stop the engine, select the Reuters window and press "Ctrl-c" to halt the batch job. Once this is done, close the connection to Reuters within MATLAB by running the command "close(conn);". At this point, it is safe to close all windows.
    g. If you wish to restart the trading algorithm, you should "clear all" within MATLAB first. This resets the burn-in period properly.