Real-time trading in MATLAB

Yair Altman

Undocumented Matlab.com

altmany@gmail.com
A common algo-trading challenge

• Trading platforms are relatively closed
  o Difficult to develop automated trading platforms
  o Vendor lock-in – algos are often un-portable
  o Internal algo customizability usually limited

• Common solutions:
  o Use Excel with trading-platform plugin
  o Use limited internal programmability (MT4, TS)
  o Develop custom C++/Java applications
Why use MATLAB?

• Numerous out-of-the-box analysis functionality
  o Much more functionality than Excel or C++/Java

• Dedicated toolboxes for specific uses
  o Financial, Data-feed, Statistics, Econometrics, Optimization, Trading, ...

• Tried-and-tested
  o Prevents risk of losses due to computational bugs
  o Most functions have editable source code – no secrets
  o Reduces total cost of ownership (develop/test/maintain)

• Easy learning curve – engineering power without needing to be a software developer

• Excellent at exactly the task taking most time/cost to develop: the algo strategy/model
  o All other components are usually far easier to develop

• mathworks.com/discovery/algorithmic-trading.html

© Yair Altman  UndocumentedMatlab.com
However...

- MATLAB could not until recently complete the trading loop –
  - Send automated trade orders to broker
  - Modify/cancel open orders
  - Track trade executions
  - Receive portfolio/account info
Solutions

• MATLAB 8.1 (R2013a): new Trading Toolbox
  - Windows only
  - Bloomberg EMSX
  - Trading Technologies X_TRADER
  - R2013b: Added CQG + IB interfaces
  - mathworks.com/products/trading

• MATLAB 7.1 (R14 SP3) onward: IB-MATLAB
  - Windows, Mac, Linux
  - Interactive Brokers only
  - UndocumentedMatlab.com/ib-matlab
General trading application design
Deployment in large institutions

Deployment server

Compiled code (MCC, Linux);
Java, or .NET MATLAB
Production Server client;
.C or HDL (FPGA) via
MATLAB Code Generator

Order Management System,
Crossing Network

Trading Strategy
Development & Automated
Back-Testing

Data feed server

FIX or API

Java glue layer
Today's demo
Aug 09
19:04:15

Demo trading application

<table>
<thead>
<tr>
<th>Symbol</th>
<th>Local</th>
<th>Exchange</th>
<th>Type</th>
<th>Pos</th>
<th>Cost</th>
<th>Latest</th>
<th>Value</th>
<th>Quotes</th>
</tr>
</thead>
<tbody>
<tr>
<td>AAPL</td>
<td>AAPL</td>
<td>NASDAQ</td>
<td>STK</td>
<td>-13</td>
<td>453.0946</td>
<td>456.1650</td>
<td>-5930</td>
<td>1171</td>
</tr>
<tr>
<td>AMZN</td>
<td>AMZN</td>
<td>NASDAQ</td>
<td>STK</td>
<td>-21</td>
<td>282.0983</td>
<td>297.4550</td>
<td>-6247</td>
<td>1012</td>
</tr>
<tr>
<td>CL</td>
<td>CLX5</td>
<td>NYMEX</td>
<td>FUT</td>
<td>1</td>
<td>103.3323</td>
<td>103.4500</td>
<td>103490</td>
<td>1185</td>
</tr>
<tr>
<td>EUR</td>
<td>EUR.USD</td>
<td>IDEALPRO</td>
<td>CASH</td>
<td>6</td>
<td>3.6338</td>
<td>1.3337</td>
<td>8</td>
<td>988</td>
</tr>
<tr>
<td>GOOG</td>
<td>GOOG</td>
<td>NASDAQ</td>
<td>STK</td>
<td>144</td>
<td>875.8424</td>
<td>891.9800</td>
<td>128445</td>
<td>986</td>
</tr>
<tr>
<td>USD</td>
<td>USD.JPY</td>
<td>IDEALPRO</td>
<td>CASH</td>
<td>13</td>
<td>98.0312</td>
<td>96.2575</td>
<td>1251</td>
<td>1014</td>
</tr>
</tbody>
</table>

18:54:55 Retrieving historical data for GOOG @ NASDAQ...
18:54:59 Retrieving historical data for USD.JPY @ IDEALPRO...
18:55:03 Starting update timer...
19:00:18 Buying 1 AMZN (STK) on NASDAQ ⇒ order ID=316735558
19:00:23 Buying 1 AMZN (STK) on NASDAQ ⇒ order ID=316735559
19:00:28 Buying 1 AMZN (STK) on NASDAQ ⇒ order ID=316735560
19:00:33 Selling 1 AMZN (STK) on NASDAQ ⇒ order ID=316735561
19:00:38 Decided not to trade in this cycle
19:00:43 Decided not to trade in this cycle
19:00:48 Decided not to trade in this cycle
19:00:53 Buying 1 EUR.USD (CASH) on IDEALPRO ⇒ order ID=316735562
19:00:53 Order Message: BUY 1 EUR.USD Forex Warning: Your order size is below the EUR 20000 IdealPro minimum and will be routed as an odd lot order.
19:00:58 Buying 1 EUR.USD (CASH) on IDEALPRO ⇒ order ID=316735563

Clear log
Close positions
Resume trading
Interactive Brokers (IB)

• Low-cost online broker

• Consistently ranked Best Online Broker by *Barron's*
  - commissions
  - execution prices
  - features
  - exchanges
  - reports

• Widely used worldwide

• Fully documented API
IB-MATLAB

• Connects MATLAB to IB
  o Receive account data (portfolio, cash, limits)
  o Receive market data feed (historic, snapshot, streaming quotes)
  o Send trade orders to market
  o Modify/cancel open orders
  o Track executions (MATLAB callback functions)
  o Synchronous + asynchronous modes
  o Fully supports IB’s API
  o 5-10 mS latency for IB events

• Works on all MATLAB platforms, Java-based API
• Hundreds of installations, trades $100M/day
IB-MATLAB: getting portfolio data

```matlab
>> data = IBMatlab('action','PORTFOLIO')
data =
    1x12 struct array with fields:
symbol
localSymbol
...

>> data(1)
ans =
    symbol: 'AMZN'
    localSymbol: 'AMZN'
    exchange: 'NASDAQ'
    secType: 'STK'
    currency: 'USD'
    right: '0'
    expiry: '
    strike: 0
    position: 9200
    marketValue: 1715800
    marketPrice: 186.5
    averageCost: 169.03183335
    contract: [1x1 struct]
```
IB-MATLAB: getting market data

```matlab
>> data = IBMatlab('action', 'QUERY', 'symbol', 'GOOG')
data =
    reqId: 22209874
    reqTime: '02-Dec-2010 00:47:23'
dataTime: '02-Dec-2010 00:47:23'
dataTimestamp: 734474.032914491
ticker: 'GOOG'
bidPrice: 563.68
askPrice: 564.47
    open: 562.82
    close: 555.71
    low: 562.4
    high: 571.57
lastPrice: -1
volume: 36891
tick: 0.01
bidSize: 3
askSize: 3
lastSize: 0
contractDetails: [1x1 struct]
```
>> data = IBMatlab('action','HISTORY', 'symbol','IBM', ... 'barSize','1 hour', 'useRTH',1)

data =
  dateNum: [1x7 double]
  dateTime: {1x7 cell}
    open: [161.08 160.95 161.66 161.17 161.57 161.75 162.07]
    high: [161.35 161.65 161.70 161.60 161.98 162.09 162.34]
    low: [160.86 160.89 161.00 161.13 161.53 161.61 161.89]
    close: [160.93 161.65 161.18 161.60 161.74 162.07 162.29]
    volume: [5384 6332 4580 2963 4728 4465 10173]
    count: [2776 4387 2990 1921 2949 2981 6187]
    WAP: [161.07 161.25 161.35 161.31 161.79 161.92 162.14]
  hasGaps: [0 0 0 0 0 0 0]

>> data.dateTime
ans =
  '20110225 16:30:00'  '20110225 17:00:00'  '20110225 18:00:00'
  '20110225 19:00:00'  '20110225 20:00:00'  '20110225 21:00:00'
  '20110225 22:00:00'
% Alternative #1: using a MATLAB struct
paramsStruct = [];
paramsStruct.action = 'BUY';
paramsStruct.symbol = 'GOOG';
paramsStruct.quantity = 100;
paramsStruct.limitPrice = 850;
orderId = IBMatlab(paramsStruct);

% Alternative #2: using name/value pairs
orderId = IBMatlab('action','BUY', 'symbol','GOOG', ...
                 'quantity',100, 'limitPrice',850);
% Set the callback function for IB trade execution events
orderId = IBMatlab('action','BUY', 'symbol','GOOG', ...
    'quantity',1, 'limitPrice',850,  ...
    'CallbackExecDetails',@myExecDetailsFcn);

% Sample event callback function
function myExecDetailsFcn(hObject, eventData)
    % Extract the basic event data components
    contractData  = eventData.contract;
    executionData = eventData.execution;

    % Now do something useful with this information...
end
Some design considerations (in no particular order)

• Build or buy
• Data feed provider (IB / IQFeed / eSignal / ...)
• Synchronous (periodic) or asynchronous (reactive)
• Latency/frequency
  → streaming quotes or periodic historical data requests
  → perhaps we need to use C / FPGA code (for µS latency)
• Level of robustness, failsafe mechanisms
• GUI or GUI-less
• Semi or fully automated
Example for a very simple application design

% Main application entry point
function tradingApplication()
    tradeSymbol('CLX3', 15*60, @timerCallbackFunction); % @ 15 mins
    tradeSymbol('GOOG', 10*60, @timerCallbackFunction); % @ 10 mins
    tradeSymbol('DAX', 5*60, @timerCallbackFunction); % @ 5 mins
    tradeSymbol('FTSE', 1*60, @timerCallbackFunction); % @ 1 min
end

% Start an endless timer at the specified frequency that will
% run the specified callbackFunc upon each invocation
function hTimer = tradeSymbol(symbolName, period, callbackFunc)
    % Create the timer object
    hTimer = timer('ExecutionMode','fixedRate', ..., 'Period',period, ..., 'TimerFcn',{callbackFunc,symbolName});

    % Start the timer
    start(hTimer);
end % tradeSymbol
Example for a very simple application design

```matlab
function timerCallbackFunction(hTimer, eventData, symbolName)
    try
        % Load previously-stored data for the specified contract
        persistentData = load(symbolName);

        % Get the latest data for this contract
        latestData = getLatestData(symbolName);

        % Process the data (secret sauce – algo strategy)
        [tradeParams, persistentData] = processData(latestData, ... persistentData);

        % Process trade signals (send orders to IB)
        IBMatlab(tradeParams{:});

        % Save decision-making data for next timer invocation
        save(symbolName, 'persistentData');
    catch
        processError(lasterror);
    end
end
```
Some additional bells & whistles

• **Main engine (non-GUI)**
  - Risk (open positions) management
  - Asynchronous trade execution tracking
  - FIX connection (rather than API)
  - Alerts via email/SMS (text-message)

• **Graphical User Interface (GUI)**
  - Open positions
  - Real-time market graph
    - TA indicators/bands, OHLC bars/candles, trade signals
  - Trades/executions log
  - P&L updates
  - Manual overrides ("panic button")
Sample advanced MATLAB GUI

![Gold Cockpit GUI](image)
### PC Ratios - GLD P-C RATIO analysis

#### Periods

<table>
<thead>
<tr>
<th>Periods</th>
<th>Any period returns</th>
<th>Avg return</th>
<th>Avg gain</th>
<th>Avg draw</th>
<th>Gain/loss</th>
<th>Max gain</th>
<th>Max draw</th>
<th>Random vol</th>
<th>Signal % pos</th>
<th>Payout % p-value</th>
<th>Lower vol cone</th>
<th>Upper vol cone</th>
<th>Realized vol after sig</th>
<th>Vol var after sig</th>
<th>% p-value</th>
</tr>
</thead>
<tbody>
<tr>
<td>3</td>
<td>0.3</td>
<td>0.1</td>
<td>-1.5</td>
<td>-1.7</td>
<td>-1.1</td>
<td>4.0</td>
<td>-4.3</td>
<td>58</td>
<td>56</td>
<td>0.1</td>
<td>19.0</td>
<td>24.5</td>
<td>16.2</td>
<td>121.9</td>
<td>8</td>
</tr>
<tr>
<td>5</td>
<td>0.5</td>
<td>0.6</td>
<td>2.2</td>
<td>-2.1</td>
<td>1.1</td>
<td>7.9</td>
<td>-10.1</td>
<td>58</td>
<td>64</td>
<td>0.6</td>
<td>12.5</td>
<td>20.2</td>
<td>24.7</td>
<td>17.5</td>
<td>90.3</td>
</tr>
<tr>
<td>10</td>
<td>1.0</td>
<td>2.0</td>
<td>3.9</td>
<td>-2.0</td>
<td>2.0</td>
<td>15.4</td>
<td>-5.8</td>
<td>62</td>
<td>67</td>
<td>2.0</td>
<td>13.7</td>
<td>20.8</td>
<td>24.2</td>
<td>17.8</td>
<td>100.9</td>
</tr>
<tr>
<td>15</td>
<td>1.5</td>
<td>2.8</td>
<td>4.3</td>
<td>-2.8</td>
<td>1.6</td>
<td>13.7</td>
<td>-6.9</td>
<td>65</td>
<td>78</td>
<td>2.8</td>
<td>3</td>
<td>14.5</td>
<td>20.9</td>
<td>24.5</td>
<td>16.4</td>
</tr>
<tr>
<td>20</td>
<td>1.9</td>
<td>3.7</td>
<td>4.9</td>
<td>-2.8</td>
<td>1.8</td>
<td>16.0</td>
<td>-6.4</td>
<td>67</td>
<td>64</td>
<td>3.7</td>
<td>1</td>
<td>14.5</td>
<td>20.9</td>
<td>25.1</td>
<td>16.2</td>
</tr>
<tr>
<td>30</td>
<td>2.7</td>
<td>4.1</td>
<td>5.2</td>
<td>-3.2</td>
<td>1.8</td>
<td>14.5</td>
<td>-5.9</td>
<td>70</td>
<td>67</td>
<td>4.1</td>
<td>3</td>
<td>14.7</td>
<td>21.0</td>
<td>24.8</td>
<td>15.5</td>
</tr>
<tr>
<td>40</td>
<td>3.4</td>
<td>5.6</td>
<td>6.9</td>
<td>-2.7</td>
<td>2.8</td>
<td>18.2</td>
<td>-8.6</td>
<td>71</td>
<td>68</td>
<td>5.5</td>
<td>1</td>
<td>14.7</td>
<td>21.0</td>
<td>25.6</td>
<td>18.4</td>
</tr>
<tr>
<td>60</td>
<td>5.1</td>
<td>8.9</td>
<td>8.7</td>
<td>-2.1</td>
<td>4.1</td>
<td>23.2</td>
<td>-2.9</td>
<td>76</td>
<td>65</td>
<td>8.3</td>
<td>0</td>
<td>14.7</td>
<td>21.1</td>
<td>24.3</td>
<td>18.3</td>
</tr>
</tbody>
</table>

#### Signal Dates

<table>
<thead>
<tr>
<th>Signal Dates</th>
<th>Periods</th>
<th>Max gain</th>
<th>Max draw</th>
</tr>
</thead>
<tbody>
<tr>
<td>02-Sep-2008</td>
<td>3</td>
<td>13.7</td>
<td>-6.6</td>
</tr>
<tr>
<td>1C-Sep-2008</td>
<td>5</td>
<td>18.0</td>
<td>-3.3</td>
</tr>
<tr>
<td>17-Oct-2008</td>
<td>10</td>
<td>5.5</td>
<td>-10.1</td>
</tr>
<tr>
<td>11-Nov-2008</td>
<td>15</td>
<td>14.7</td>
<td>-0.3</td>
</tr>
<tr>
<td>04-Dec-2008</td>
<td>20</td>
<td>19.4</td>
<td>0.0</td>
</tr>
<tr>
<td>19-Dec-2008</td>
<td>30</td>
<td>16.0</td>
<td>-21.0</td>
</tr>
<tr>
<td>13-Jan-2009</td>
<td>60</td>
<td>16.1</td>
<td>1.6</td>
</tr>
<tr>
<td>02-Apr-2009</td>
<td>3</td>
<td>13.7</td>
<td>-6.6</td>
</tr>
<tr>
<td>14-Apr-2009</td>
<td>5</td>
<td>18.0</td>
<td>-3.3</td>
</tr>
<tr>
<td>05-May-2009</td>
<td>10</td>
<td>5.5</td>
<td>-10.1</td>
</tr>
<tr>
<td>06-May-2009</td>
<td>15</td>
<td>14.7</td>
<td>-0.3</td>
</tr>
<tr>
<td>25-Jun-2009</td>
<td>20</td>
<td>19.4</td>
<td>0.0</td>
</tr>
<tr>
<td>08-Jul-2009</td>
<td>30</td>
<td>16.0</td>
<td>-21.0</td>
</tr>
<tr>
<td>10-Jul-2009</td>
<td>60</td>
<td>16.1</td>
<td>1.6</td>
</tr>
<tr>
<td>11-Aug-2009</td>
<td>3</td>
<td>13.7</td>
<td>-6.6</td>
</tr>
<tr>
<td>13-Aug-2009</td>
<td>5</td>
<td>18.0</td>
<td>-3.3</td>
</tr>
<tr>
<td>17-Aug-2009</td>
<td>10</td>
<td>5.5</td>
<td>-10.1</td>
</tr>
<tr>
<td>20-Aug-2009</td>
<td>15</td>
<td>14.7</td>
<td>-0.3</td>
</tr>
<tr>
<td>31-Aug-2009</td>
<td>20</td>
<td>19.4</td>
<td>0.0</td>
</tr>
<tr>
<td>13-Nov-2009</td>
<td>60</td>
<td>16.1</td>
<td>1.6</td>
</tr>
<tr>
<td>03-Feb-2010</td>
<td>3</td>
<td>13.7</td>
<td>-6.6</td>
</tr>
<tr>
<td>15-Mar-2010</td>
<td>5</td>
<td>18.0</td>
<td>-3.3</td>
</tr>
<tr>
<td>18-Mar-2010</td>
<td>10</td>
<td>5.5</td>
<td>-10.1</td>
</tr>
<tr>
<td>30-Mar-2010</td>
<td>15</td>
<td>14.7</td>
<td>-0.3</td>
</tr>
</tbody>
</table>

© Yair Altman

www.UndocumentedMatlab.com
Conclusion

• Technology is no longer a barrier to developing a relatively low-cost algorithmic trading engine

• We no longer need to handle connectivity plumbing

• We no longer need to prototype in MATLAB, deploy in C
  o MATLAB can handle entire investment management lifecycle
  o Simpler development, reduced cost & time-to-market

• The only major thing left to do is “just” to devise a winning strategy
  o With the analysis tools available in MATLAB this should be easier than ever
Resources

mathworks.com/products/trading
undocumentedmatlab.com/ib-matlab
interactivebrokers.com/en/software/ibapi.php
altmany@gmail.com