This user guide attempts to document the various settings and options in the MTF version of HGIMatrix. It does *not* discuss how to trade the various HGI signals.

Before continuing, I first want to thank Bob (nanningbob) for creating/designing the HGI-indicator and Denis (Elixe), Tommaso (Milanese) and Baluda (Paul) for coding the various parts of the indicator and library. The HGI indicator and discussion on how to interpret the signals can be found in the HGI thread.

The other point I want to make, is that HGIMatrix is meant to be a DashBoard, or overview Tool, to quickly see the signals for all selected symbols and timeframes. However, it is highly recommended, before taking a trade, to load the chart of a selected symbol/timeframe, to get the real picture of what has been going on. HGIMatrix makes this easy, as you only have to click on the signal (arrow) for the symbol and time frame you want to load the chart for. So, don't trade of the HGIMatrix alone.

There are two versions of the HGIMatrixMTF-indicator:

- The 'normal' version, that uses the HGI-indicator (by default: HGI_v17.00) to get its signals
- The 'library' version, that uses the HGI-library (hgi_lib.ex4 and hgi_lib.mqh) to calculate the signals internally. The 'library' version has the suffix 'L' behind the version number in the filename.

The difference between these versions is subtle, but can have a profound impact. The HGI-indicator was designed by Elixe and Tommaso to NOT repaint. That means that all signals and waves that were calculated for candles 1 and above (i.e. all *historic* candles), won't ever change, unless the indicator is reloaded. The 'normal' version of the HGIMatrix will therefore also not repaint signals from before candle 1, again, as long as it is not reloaded. However, the 'library' version calls the signal-generating code directly from the HGI-library and it is therefore possible that, due to market action, a signal or wave for a historic candle is replaced or removed (i.e. is repainted).

Additionally, the library version will load and refresh must faster and has a smaller memory footprint, as it is much more efficient to directly call the HGI-library functions, than it is to load a gazillion instances of the HGI-indicator.

Installation

Normal, Non-Repainting version:

> Download HGIMatrixMTF<version>.mq4 from Post 1 of the HGIMatrix thread, and save the file inside your \MQL4\Indicators folder. Download the HGI_v17.00 indicator from the HGI thread and save the file inside your \MQL4\Indicators folder. Open the MetaEditor (from Tools->MetaQuotes Language Editor or F4) and navigate to MQL4\Indicators. You should see the HGIMatrixMTF<version>.mq4 file listed. Right-click on the HGIMatrixMTF<version>.mq4 filename and select 'Compile'. This should generate the HGIMatrixMTF<version>.ex4 file, and the indicator is now ready to be used inside MT4.

Library, Repainting version:

Download HGIMatrixMTF<version>**L**.mq4 from Post 1 of the HGIMatrix thread, and save the file inside the \MQL4\Indicators folder. Also download hgi_lib.ex4 and save it inside the \MQL4\Libraries folder and download hgi_lib.mqh and save it inside the \MQL4\Include folder. Open the MetaEditor (from Tools->MetaQuotes Language Editor or F4) and navigate to MQL4\Indicators. You should see the HGIMatrixMTF<version>L.mq4 file listed. Right-click on the HGIMatrixMTF<version>L.mq4 filename and select 'Compile'. This should generate the HGIMatrixMTF<version>**L**.ex4 file, and the indicator is now ready to be used inside MT4.

Parameters

The HGIMatrix has gone through various iterations, during which many new parameters were added.

IncludeSymbols: Leave it blank if you want to include all symbols in the market panel, or specify just the symbols you want. The string 'NULL' is translated to the symbol of the current chart.

ExcludeSymbols: Leave it blank if you don't want to exclude any symbols, or specify the symbols you want to exclude. Again 'NULL' is translated to the symbol of the current chart.

ExcludeExtensions: This is to specify, which symbol extensions you want to exclude. A symbol extension is defined as the string of characters that appears after the 6th character in the symbol name. Eg. in 'AUDUSD.', the extension is the '.' (dot) character.

The symbols HGIMatrix selects for inclusion, are determined as follows:
If *IncludeSymbols* is empty: You will get all symbols in the market panel, minus those in *ExcludeSymbols* and minus those which have an extension in the *ExcludeExtentions* list.

If *IncludeSymbols* is non-empty: You will only get the symbols in *IncludeSymbols*, as long as they are listed in the market panel and as long as their extension is not in the *ExcludeExtentions* list.

TimeFrames = "240,15";

Set the *TimeFrames* parameter to the list of time frames you want HGIMatrix to monitor. Currently, due to CPU and memory constraints, HGIMatrix can only handle a maximum of 5 time frames. HGIMatrix ignores time frames it can not recognise as a valid time frame.

TemplateName = "default";

One can open a chart for a symbol and timeframe, by (single) clicking the signal-arrow on the cell for the corresponding symbol and timeframe. The *TemplateName* parameter is used to specify with what template that chart should be opened.

CellWidth = 90;
CellHeight = 60;

The *CellWidth* and *CellHeight* parameters can be used to configure the width and height (in pixels) of the matrix cells. There is code inside the HGIMatrix-indicator to attempt to scale everything else (symbols, waves, symbols names, timeframes) with the given *CellWidth* and *CellHeight.*

XOffSet = 10;
YOffSet = 10;

The *XOffSet* and *YOffSet* are used to physically place the indicator on the screen with the given offset in pixels. This is useful if you want to have two or more HGIMatrix indicators on the same screen.

UseClosedCandle = false;

If *UseClosedCandle* is true, HGIMatrix will not look at the current candle (candle=0). All signals/waves have a candle number of at least 1.

ShowMostRecentSignal = true;

If *ShowMostRecentSignal* is false, HGIMatrix won't bother looking back for the most recent signal. It will either display a signal for the closed candle (candle=1), if *UseClosedCandle* is true, or a signal for the currently open candle (candle=0), if *UseClosedCandle* is false.

If *ShowMostRecentSignal* is true, HGIMatrix will go back in time (up to *MaxBarLookback* candles), and retrieve the first (i.e. the most recent) signal, for each symbol and timeframe. If *UseClosedCandle* is true, it will start the search at candle=1, otherwise, if *UseClosedCandle* is false it will start the search at candle=0 (the current candle).

ShowBarNo = true;

If *ShowBarNo* is true, HGIMatrix will display (next to the signal) the candle number where it discovered that particular signal. If *ShowBarNo* is false, that candle number won't be displayed.

ShowHurst = true;

Hurst is an indicator very similar to the TMA. It was observed that Hurst, with period 5 (denoted as Hurst(5), snakes around Hurst(11). Therefore, if Hurst(5) < Hurst(11), look for BUYs and if Hurst(5) > Hurst(11), look for SELLs.

If *ShowHurst* is true, the cell background is colored DarkGreen, if the symbol arrows generated by HGI are in the same direction as what the Hurst-condition would suggest. So, if

> Hurst(5) < Hurst(11), ***and*** the HGI arrow is pointing UP, the cell background is DarkGreen, or Hurst(5) > Hurst(11), and the HGI arrow is pointing DOWN, the cell background is DarkGreen.

ShowOnlyHurst = true;

If *ShowOnlyHurst* is true, HGIMatrix will not display any signals or waves, for cells for which the Hurst-condition is not valid. It simply leaves those cells blank. In addition, if for a certain symbol, there is not a valid signal for *any* of the selected time frames, HGIMatrix will skip that symbol altogether (rather than displaying the symbol with a row of blank cells). So, if you think that

HGIMatrix is 'missing' symbols, check the *ShowHurst* and *ShowOnlyHurst* parameters.

ShowTrend = true;

If *ShowTrend* is true, HGIMatrix colors the *symbol name* in the following manner:

> if Price Action > LWMA240 ***and*** Price Action > LWMA60 ==> GREEN (Up Trend)
> if Price Action < LWMA240 ***and*** Price Action < LWMA60 ==> RED (Down Trend)
> otherwise ==> YELLOW (Ranging)
>
> The LWMA is calculated for the H4 time frame, with the OPEN price field. This conforms to the 10.4 Trend definition

ShowFlash = true;

If *ShowFlash* is true, HGIMatrix will 'flash' signals and waves for the current candle (candle=0)

ShowATR = true;

If *ShowATR* is true, HGIMatrix will display the Average True Range value (in PIPs) for the given symbol, right underneath the symbol name. The ATR value is calculated for the given *ATRTimeFrame* (default: Daily) and *ATRPeriod* (default: 20)

ATRThreshold = 100.0;

The ATRThreshold is used, so that HGIMatrix can filter out (i.e. remove) symbols that exceed that threshold. The process of filtering against the ATRThreshold can be switch on/off using the *ShowOnlyATR* parameter.

ShowOnlyATR = false;

If *ShowOnlyATR* is true, HGIMatrix will only include symbols that have an ATR value smaller than the *ATRThreshold.*

ShowTrendSignal = true;

If *ShowTrendSignal* is set to false, HGIMatrix will not display Trend signals. If, on its search for signals, it discovers a TREND arrrow (either UP or DOWN), and *ShowTrendSignal* was set to false, HGIMatrix will continue looking further backwards in search of other (non TREND) signals. It will look back as far as *MaxBarLookback* candles.

ShowRangeSignal = true;

If *ShowRangeSignal* is set to false, HGIMatrix will not display Range signals. If, on its search for signals, it discovers a RANGE arrrow (either UP or DOWN), and *ShowRangeSignal* was set to false, HGIMatrix will continue looking further backwards in search of other (non RANGE) signals. It will look back as far as *MaxBarLookback* candles.

ShowRadSignal = true;

If *ShowRadSignal* is set to false, HGIMatrix will not display Rad signals. If, on its search for signals, it discovers a RAD arrrow (either UP or DOWN), and *ShowRadSignal* was set to false, HGIMatrix will continue looking further backwards in search of other (non RAD) signals. It will look back as far as *MaxBarLookback* candles.

ScreenHeight = 720;

This is the screen height in pixels. HGIMatrix needs to know how much vertical space is available to draw cells. If the vertical limit is about to be exceeded, HGIMatrix starts a new column of symbols/timeframes.

MaxBarLookback = 200;

MaxBarLookback is the number of candles, HGIMatrix will look back in search for signals and waves. If you set *MaxBarLookback* to a small number (eg. 10), and there haven't been any signals in the last 10 candles, HGIMatrix won't find any and therefore won't display a signal or wave.

SymbolFont = "WingDings";

The arrows on the HGIMatrix come from the *SymbolFont* character-set. In fact they are taken from the extended ASCII group of the specified *SymbolFont.* Normally, you would have no reason to change this parameter. However, on Chinese (and other double-byte character set (dbcs) operating systems), the extended ASCII characters of the WingDings-font are not displayed correctly. Therefore, the *SymbolFont* parameter was introduced, to allow people on DBCS O/S to create their own font and set the *SymbolFont* parameter accordingly.

ClrRangeWave = clrOrange;

ClrRangeWave: to change the color of the Range Wave

ClrTrendWave = clrDodgerBlue;

ClrTrendWave: to change the color of the Trend Wave

ClrAlert = clrDarkRed;

ClrAlert: When a signal or wave changes for a certain symbol and time frame, the cell background color of the corresponding cell is changed for a minute to the *ClrAlert* color.

ClrSelect = clrNavy;

ClrSelect: When you click the symbol name , the corresponding cell background color is set to the *ClrSelect* color. This is a visual marker for yourself, to keep an eye on that symbol.

ClrHurst = clrDarkGreen;

ClrHurst: When the signal direction for a certain symbol and time frame is in agreement with the Hurst-condition, ***and*** if *ShowHurst* was set to true, the cell background color of the corresponding cell is changed to the *ClrHurst* color.

Alerts

HGIMatrix refreshes all its signals and waves every minute. It 'remembers' the old signal or wave from a minute ago, and compares the new signal/wave with the old one. If there is a difference, an Alert could be sent.

There are three parameters to specify where Alerts should be sent to:

AlertsOn = false;

AlertsOn: if true Alerts will be popped-up on the screen with the MT4 Alert-function

NotificationsOn = false;

If *NotificationsOn* is true, a notification is sent to your mobile device as and when an alert is generated. You must have set the MetaQuotes ID and checked 'Enable Push Notifications' in the Notifications-tab of the Options menu. You can get your MetaQuotes ID from the MetaTrader 4 App (Android or iOS) under the Settings menu.

AlertsToFile = true;

AlertsToFile: if true Alerts will be appended to a file in the \MQL4\Files area. The file name will be like: <indicatorname>.txt, where <indicatorname> represents the name of the HGIMatrix indicator file.

You can have all of these parameters true, so that you will get Pop-Up alerts, Notifications on your Phone and Alerts in the file. Or you can have all of these parameters false and you won't get any Alerts at all.

There are 5 parameters to specify what type of Alert and on what timeframe you want to receive.

AlertOnTrendChange: a comma-separated list of timeframes, to indicate for which timeframe you want to receive a TREND (big arrow) change alert.

AlertOnRangeChange: a comma-separated list of timeframes, to indicate for which timeframe you want to receive a RANGE (small arrow) change alert.

AlertOnRadChange: a comma-separated list of timeframes, to indicate for which timeframe you want to receive a RAD (small arrow pointing diagonally up or down) change alert.

AlertOnRangeWaveChange: a comma-separated list of timeframes, to indicate for which timeframe you want to receive an alert when a RANGE WAVE (Golden Rod wave) appears.

AlertOnTrendWaveChange: a comma-separated list of timeframes, to indicate for which timeframe you want to receive an alert when a TREND WAVE (Dodger Blue wave) appears.

And there is a *AlertOnlyHurst* parameter. The Hurst-condition is only 'calculated' if *ShowHurst* is true. Otherwise, HGIMatrix wouldn't have a clue, whether a signal direction agrees with Hurst or not. If you'd set *ShowHurst* = false, the *AlertOnlyHurst* parameter is ignored (ie. you would still get Alerts, even if for a given signal and timeframe, the Hurst condition does not agree). If you have

ShowHurst = true and *AlertOnlyHurst* = true you will only get changes to signals/waves if the new value of the signal/wave agrees with the Hurst-condition.