

ADAPTIVE INDICATOR EXAMPLE: LINEAR TIME REGRESSION SLOPE

Most indicators we use work with use a single period value to do their job, e.g., we set a period value for RSI at 5, or a Stochastic%K at 10, and in NST this value is often well chosen by the optimizer. This generally works well, but it would be interesting to have an indicator with the ability to move more briskly when needed, then slower and more deliberate when that behavior is appropriate, thus hoping better to follow the market's intrinsic speed changes.

An adaptive indicator does exactly this; it will use fewer periods (quicker response) if told to do so, and use more periods (slower, more careful response) if required at other times. It does this via an additional input, called an "adaptor". This input - in addition to the usual close (and maybe others) - varies according to some rule (often volatility) and is another time series input to the indicator math which then causes variable periods to be used in the indicator's calculations instead of the usual fixed single value.

All that is really required to do this is to rewrite the indicator's software routine to accept an additional input which will govern the number of periods for any given bar. And we need two more simple values: the maximum and minimum periods we think the indicator should ever use - the limits of its "adaptation".

The software included in this tip is written for PowerBasic, and is an adaptive version of the Linear Time Regression Slope - an effective and commonly used indicator. The same technique is applicable to C, of course. The idea is to generate a DLL (Dynamic Link Library) that NST can access to create the indicator. DLL structure is beyond this tip, but one is easily created in PowerBasic without further knowledge.

SOFTWARE NOTES

Creation and installation of custom indicators is covered in other NST tips, so will not be repeated here, but the passing of parameters between NST and the DLL is of fundamental interest.

Note in the header portion of the code (where the name is declared and following), each parameter to be passed in is declared along with its passing technique and variable type. Time series values are not passed by their entire value stream (which would be slow and inefficient), but instead are simply pointed to by (oddly enough) a pointer value, and so are passed by referring to that pointer. PowerBasic is informed in this line that the value being received is a pointer, and can be used as such. We will pass the value of the pointer itself (ByVal).

Integer values such as the maximum value of the periods and the number of bars are not time series, but are a fixed value and so are passed as their actual value (ByVal), and their type (long or double) is declared as well.

Note also that those variables used as pointers to time series values begin with lower case “p” as an indication and reminder that this is not a variable value, but a *pointer* to a variable. (This is only a convention and convenience to programmers - NOT a PowerBasic or C requirement!) When the code wishes to refer to an actual value within the time series, this pointer name is preceded by a “@”, causing the code to act on the value pointed to.

Further, the position in the time series stream is indicated by the value in square brackets following the pointer name, e.g., @pInputStream[4] would act on the value that is the fourth member of the InputStream series.

Another interesting portion of the code is the line which determines the period value for each pass through the loop (the current bar). Observe that it overlays the ratio of max/min periods and the incoming adaptor value onto values from 0 to 100. Note especially that it only uses the integer part of this calculation (the FIX operator); a period must have only integer values.

Experienced programmers will note that there are other, somewhat more sophisticated ways to structure this code. The intent in this example is readability for the newcomer, and less to demonstrate higher-level techniques.

While the math for both the indicator and adaptor here are easily seen in the code, both are in the public domain, and a quick internet search will give several sources. One good source for the LTRS is in the “help” area for this indicator in NST.

ADAPTORS

The user is encouraged to experiment with many time series as adaptors. The indicator in this tip expects an adaptor to move in a bounded way between 0 and 100, with lower values causing smaller period values (faster indicator response) and larger values causing slower response (greater period values). Note in the indicator definition that these max and min values can be specified by the user. Note too that if an adaptor only moves between, say, 20 and 70, then the indicator will only ever achieve a limited range of periods corresponding to that limited adaptor range. So an indicator with a max/min period range of 18/3 may only actually use periods from 14 to 5 or similar.

The choice of adaptors is a very interesting and important area, and three are suggested to begin.

r(squared): This is found in the Regression category in NST, and a period of 10 is a good start. Note that since it moves between 0 and 1, the actual adaptor function should be $100 \cdot (r(\text{sqr}))$ to satisfy this indicator's input expectations.

Absolute value of Polarized Fractal Efficiency (ABS(PFE)): The PFE is found in NST's Indicator Set #1, and the math is available in the user's guide for it.

Vertical Horizontal Filter: The VHF is intended to be a measure of market consolidation or trending, and is thought to improve on Wilde's ADX. See these web sites for the math and discussion:

<http://www.stator-afm.com/vertical-horizontal-filter.html>

http://www.traders.com/Documentation/FEEDbk_docs/Archive/072000/Abstracts_new/Gopalakrishnan/Gopalakrishnan.html

and many others.

Since the first two are available in NST resources, the latter is included in this code for convenience.

NOTE: The overarching point is that the user will want to choose an adaptor that is felt to characterize some market behavior that is suitable for governing indicator agility.

INSTALLATION

As with any indicator or group, the DLL is simply dropped into NST's "Template" folder, then NST must be rebooted to discover this new DLL. When the chart containing the indicators is loaded, the indicators become available.

CAUTION: When inserting this (or any other) adaptive indicator into a chart, note that its default "adaptor" input is the close - hardly a useful adaptor function. You must choose an appropriate adaptor time series when inserting onto a chart!

FURTHER THOUGHTS

Many users prefer indicators that are smoothed, which means passing the output through an averaging filter (SMA, EMA, etc). More sophisticated techniques, such as Gaussian filters can be used as well. Do note that a smooth indicator may look pleasing, but NST may not care a whit. Smoothing will always add a delay in response, which may be less desirable than an unattractive output curve!

That being said, the output of the adaptor is frequently smoothed to avoid rough transitions of the indicator, but is not required. The user may insert any smoothing he desires at the end of the software just prior to the line where the output is returned.

Many other additional features can be added as well, such as scaling or normalization of adaptors and inputs.

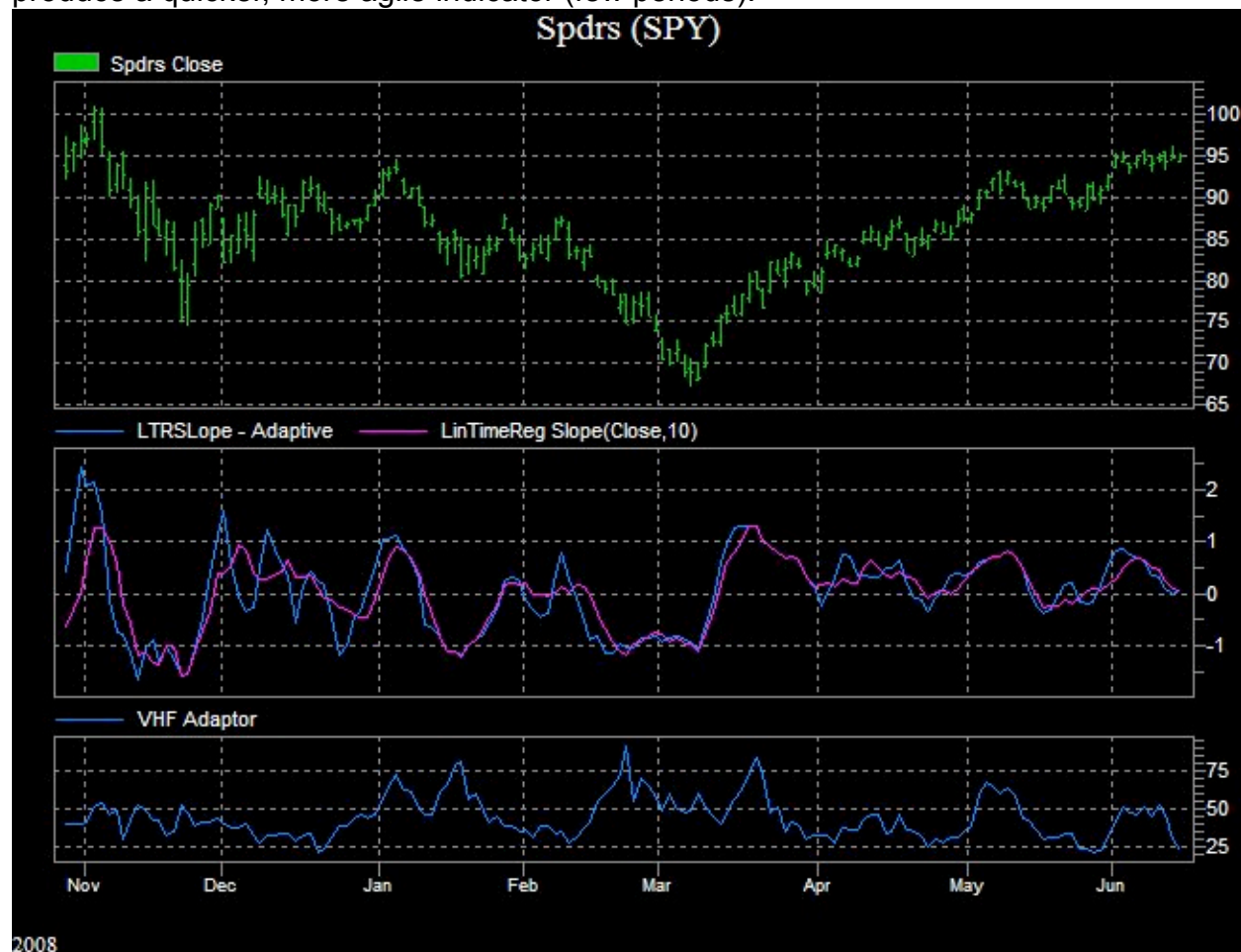
EXAMPLE

In the attached chart below, the ETF SPY is shown with the adaptive Linear Time Regression Slope indicator and the fixed-period version overlaid. Note that there are times where they differ quite a lot, and weeks where they virtually overlay each other. In this example, note especially where they diverge:

- In December, the adaptive version more faithfully catches the two upswings, where the fixed version all but ignores them.
- The rise in early January is seen well by the adaptive version, and is late in the fixed.
- Second week in February is interesting.
- The upswing at the first of June is late in the fixed version.
- Activity in April and May are interesting as well.

Now some of these apparently poor showings by the fixed version are indeed eliminated by using a faster version, i.e., a period of 5 instead of 10. But doing so will simply insert errors elsewhere of a different nature, usually a too-fast response or an overshoot in amplitude.

The VHF adaptor function is shown at the bottom of the chart for interest only. Again, larger values produce slower indicator response (longer periods), smaller values produce a quicker, more agile indicator (few periods).



SUMMARY

Not all indicators benefit equally from being made adaptive. Some remain virtually indistinguishable from their fixed parents, others simply don't perform as hoped. But

others - such as the one in this example - do indeed perform well. The user is encouraged to experiment with his favorites.

Also remember that no indicator will make or break our trading! The best we hope for - and work very hard to achieve - is some small advantage.