VOLUME 13

STAD Club Strategies Revisited: Looking Back at Volumes 1-12

Information in this document is subject to change without notice.

THE TRADING STRATEGIES IN THIS BOOK ARE EXAMPLES ONLY, AND HAVE BEEN INCLUDED SOLELY FOR EDUCATIONAL PURPOSES. OMEGA RESEARCH DOES NOT RECOMMEND THAT YOU USE ANY SUCH TRADING STRATEGY, AS THE USE OF ANY SUCH TRADING STRATEGY DOES NOT GUARANTEE THAT YOU WILL MAKE PROFITS, INCREASE PROFITS, OR MINIMIZE LOSSES. THE SOLE INTENDED USES OF THE TRADING STRATEGIES INCLUDED IN THIS BOOK ARE TO DEMONSTRATE THE WAYS IN WHICH EASYLANGUAGE CAN BE USED TO DESIGN PERSONAL TRADING STRATEGIES AND TO SHOW SOME EXAMPLES OF HOW CERTAIN POPULAR, WELL-KNOWN TRADING STRATEGIES MAY BE INCORPORATED INTO PERSONAL TRADING STRATEGIES. OMEGA RESEARCH, INC. IS NOT ENGAGED IN RENDERING ANY INVESTMENT OR OTHER PROFESSIONAL ADVICE. IF INVESTMENT OR OTHER PROFESSIONAL ADVICE IS REQUIRED, THE SERVICES OF A COMPETENT PROFESSIONAL SHOULD BE SOUGHT.

Contents

INTRODUCTION

Welcome to Volume 13

Welcome to Volume 13 of the Omega Research Strategy Trading & Development Club. In this very special edition of STAD Club, we will revisit ten strategies and three articles selected from STAD Volumes 1-12.

We chose the strategies that we believe are the best educational examples of the art and science of designing, testing, and optimizing trading strategies. Based on all we've learned by producing more than 100 STAD Club trading strategies since January, 1998, we've revised, added and deleted rules, brought testing and optimizing up to date (April, 2000), simplified the EasyLanguage whenever possible, and updated any code that was written before the release of TradeStation 2000i from its 16-bit architecture to 32-bit architecture. Much more than mere "reprints," these overhauled strategies and articles are designed to take you to the next level in your ability to create your own winning strategies.

The thirteen chapters in this volume incorporate many of the essentials you need to know to maximize your potential as a trader of stocks or commodities. The articles introduce setups, entries, exits, stops, and orders, and show you how to design, test, optimize, and evaluate your own trading strategies. The strategies show you both conventional and unusual ways to employ popular indicators such as Accumulation-Distribution, the Directional Movement Index, MACD, Momentum, four types of Moving Averages, RSI, Stochastics, and Volume. The strategies also demonstrate how you can use concepts such as Fibonacci Retracements, intermarket analysis, and pyramiding to improve your trading results.

As you work your way through this special edition of the Omega Research STAD Club, concentrate not only on what we've written and tested for you, but also on how you can combine your own ideas with ours. The best strategies you'll ever trade will be the ones you've designed, tested, and optimized for yourself.

We had an interesting and enjoyable time putting this "retrospective" issue together for you, and we learned a lot by reworking and improving these ten strategies. Our work on STAD 13 is done; yours is just beginning. Thanks for letting us be a springboard for you as you grow in your ability to build and trade your own winning strategies.

Contents at a Glance

Additional Educational Services

Omega Research is committed to enhancing individual trading potential through quality education. To learn more about strategy trading, an Omega Research product, or EasyLanguage, visit our web site at www.omegaresearch.com or call **(800) 439-7995** (outside US 305-485-7000) and ask about the following educational services:

EasyLanguage Resource Center

One of the best ways to learn is by example, and the EasyLanguage Resource Center on our web site is an excellent source of examples. In this Resource Center, we list all the analysis techniques — indicators and trading strategies — published in the *Technical Analysis of Stocks and Commodities* magazine, as well as popular analysis techniques worth taking a look at. Access to this Resource Center is free of charge. Feel free to download and review any of the analysis techniques and their descriptions. Our web site address is **www.omegaresearch.com**.

Getting Started

To begin reviewing your strategies, transfer the analysis techniques into your TradeStation® library and then apply the strategy you want to review to a chart. Use the Strategy Performance Report to view the strategy's results and take a look at the EasyLanguage instructions by opening the strategy in the PowerEditor™.

To transfer the analysis techniques into TradeStation:

1. Place the Strategy Testing and Development Club CD in the CD-ROM drive.
2. Start the PowerEditor. In **Windows**, click **Start,** choose **Programs,** choose **Omega Research (OMGA)** and choose **EasyLanguage PowerEditor.**
3. In the PowerEditor, use the **File - Import and Export** menu sequence**.**
4. Select the **Import EasyLanguage Archive File (ELA and ELS)** option and click **NEXT.**
5. Click **Scan.**
6. In the **Enter drive letter to scan** edit box, enter the drive letter for your CD-ROM drive (normally D), and click **OK.**
7. Choose **STAD13.ELS** from the list and click **NEXT.**
8. Below the **Analysis Types** box choose the **Select All** button and click **NEXT.**
9. Below the **Available Analysis Techniques** box choose the **Select All** button and click **FINISH.**
10. Once the files are transferred and verified, a dialog box appears informing you that the transfer was performed successfully. Click **OK.**

For your convenience, the names of the strategies in this volume all begin with STAD13 (although the signals will not have this prefix). You can now open the strategies in the PowerEditor and view the EasyLanguage instructions and/or apply them to a chart in TradeStation. You can remove your CD from the CD-ROM drive and store it in a safe place. As you apply the strategies and work with them, refer to this book for detailed explanations of the strategies and the EasyLanguage used to create them. For instructions on applying strategies and viewing the Strategy Performance Report, please refer to your user's manual.

Note to SuperCharts® Users: *To transfer the strategies into SuperCharts, use the Tools - QuickEditor menu sequence and select Transfer. Keep in mind, however, that although you can apply the strategies in SuperCharts, you will not be able to view the EasyLanguage instructions in the QuickEditor. This is because the strategies were designed in the EasyLanguage PowerEditor. Also, if you are using SuperCharts End of Day, some of the strategies will not apply as they are designed for intraday trading. Since the purpose of the STAD Club is to provide you with a learning tool, and viewing the EasyLanguage instructions is an essential part of this learning process, the use of this club for SuperCharts users is limited.*

Note to TradeStation or SuperCharts 3.x Users: *The strategies for the Club were designed using TradeStation 2000i. As such, some of the features used, such as automatic drawing of trendlines and/or text, are not available in previous versions of TradeStation (or SuperCharts). An effort is made to provide a variety of strategies that incorporate both long standing and new features; however, keep in mind that as new features are developed, we will naturally want to showcase and educate users on these features; therefore, users of the most recent version of our software will be able to make the most use of the Club.*

Obtaining Technical Support

Depending on your question, there are two resources at your disposal: the EasyLanguage Support Department and the STAD Club E-Mail Address.

EasyLanguage Support Department

The EasyLanguage Support Department provides EasyLanguage support via e-mail or fax and is designed to help you troubleshoot an analysis technique or trading strategy you are currently working on. For example, if you are incorporating a trading strategy from the Club into your own and have a question about the implementation, the EasyLanguage Support Department can answer it.

Please keep in mind that while this department can answer any EasyLanguage question, it cannot answer questions about the STAD Club specifically, such as the theory behind a strategy in the STAD Club, why a strategy was developed a certain way, or why the strategy is not performing as you expect it to, etc.

Fax Number: **(305) 485-7598**

E-Mail Address: **easylang@omegaresearch.com**

Be sure to include the following information in your fax or e-mail:

- Name
- Security Block or Customer ID Number
- Telephone Number
- Fax Number
- Product you own
- EasyLanguage instructions you are working on
- Detailed description of your problem

Please allow 48 hours for a response.

STAD Club E-Mail Address

Another resource at your disposal is the STAD Club e-mail address.

Please realize that when you send a message to this e-mail address, you will not receive a response directly; your message will be reviewed and the answer incorporated into the next volume of the STAD Club, when applicable. Therefore, if you need technical support on EasyLanguage, please use the above fax number or e-mail address.

stadclub@omegaresearch.com

Please send any comment, suggestion, or question regarding the strategies in the Club to the STAD Club e-mail address, and in each subsequent volume we will publish the most common suggestions and questions.

Benefits of Strategic Trading

There are at least five major benefits of trading in a strategic manner as opposed to trading in a discretionary manner:

1. You'll have a strategy that is compatible with your own personality and trading style — a strategy that you are comfortable with and that you can follow.

2. You will eliminate overly emotional trading and reduce the stress of constantly making subjective, spur-of-the-moment trading decisions.

3. You will have objective entry and exit criteria that have been validated by historical testing of quantifiable data.

4. You will know the maximum peak-to-valley drawdown that your strategy has experienced in the past, and you can make sure that you are adequately capitalized (both financially and psychologically) to withstand another worst-case drawdown.

5. You will gain confidence in both your strategy and yourself, thus strengthening your ability to follow your strategy and to trade in a highly disciplined manner.

As you continue to become more proficient as a strategy trader, you will almost certainly discover even more benefits of a strategic approach.

SECTION 1

Articles

CHAPTER 1

Understanding the Basics of Strategy Trading

When you are developing your strategies, there are certain rules of thumb that you should follow. These are guidelines that our users have discovered through years of trial and error, and we are providing them for you here. For example, you should know that there are three basic types of strategies — trending, countertrending and event — and no one strategy can work well in all three. The key is to develop a strategy that works well in one type of market activity and limits your losses in the other market types. This is a basic but very important concept.

Another concept is that you don't need to look at the strategy as a whole at the beginning. For example, you can have a great idea for an entry signal and really no idea as to how you will want to exit. That doesn't mean you can't start writing a great strategy. It means you can start by developing your entries until they work really well, then, once that's done, start working on the exits. Maybe you even have several favorite exits that you try with your different entries.

Select Your Strategy Type

The very first step in developing a strategy is to decide on the type of market activity you want to trade. This is an important decision because it determines the type of strategy you will be developing. This section will help you to understand some of the conditions that can occur in the different types of markets and the types of strategies that complement those markets. Once you are familiar with the basic strategy types, you will be able to select the one that you want to use.

Let's take a look at the strategies that are appropriate to each type of market.

Trending Market

Trending markets are characterized by a large sustained increase or decrease in price. Figure 1 shows an example of a trending market [Figure 1, Example of a Trending Market]. This market has been in an uptrend since before 1994. The price has almost tripled over this three-year period. This trending market was characterized by sustained up moves with small, short-lived corrections.

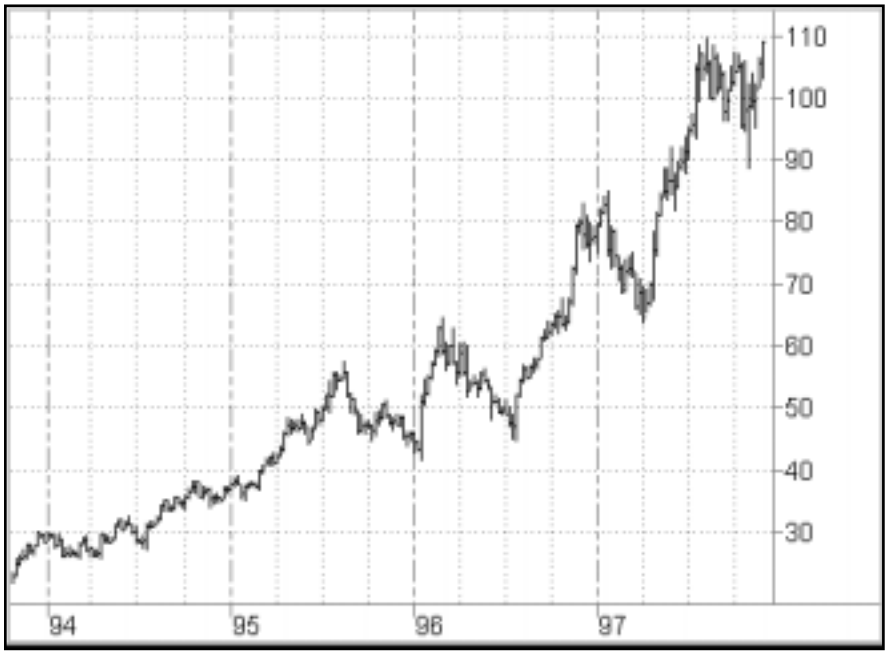


Figure 1. Example of a Trending Market

Trending Strategies

Like the name implies, trending strategies are designed for trending markets and to take advantage of all the big trending moves that may occur. In creating a trending strategy, the number one priority should be that the strategy never misses a big move. The easiest way to accomplish this is to always be in the market, that is, to always be either short or long. If you always have a position, you will always be there when the big move takes place.

The other method is to always have a "stop" order in the market, resting either above or below the current price (this is the same order as a stop loss, but it is used to enter the market rather than exit). Using a stop to enter the market will protect you because if the market moves quickly in either direction you will be stopped in before the big move begins.

Keep in mind that trending strategies tend to lose money in choppy, or directionless phases of the market. They have a small percentage of winning trades; that is, they make their money in a few big trades. This means that if you miss a big move, you may not have enough capital to hold out through the drawdown as you wait for the next big move.

Another design characteristic should be to limit your losses during the market's sideways mode. Remember, no strategy will make money in every market condition. If the strategy is designed to make money in a trending market, it will lose money in the choppy phase. Your priority should be to minimize the losses in the directionless market.

Many trending strategies make their money in one or two months of the year and break even or lose money for the rest. The most common indicators used in trend following strategies are moving averages, most often two — a short moving average and a longer moving average.

Trending strategies have the following characteristics:

- They make 80% of their profits on 20% of their trades. This is the difficult part of trend trading — a low percentage of winning trades. You need a lot of positive self-esteem and confidence in your abilities to trade a strategy that loses money on 60 or 65% of its trades. You should also be able to sit through significant drawdowns as the market drifts through a directionless period.
- Many researchers have estimated that any market is in the trend mode 15% of the time and is directionless 85% of the time. A trend following strategy then, by definition, has a low percentage of profitable trades. A trend following strategy is psychologically difficult to trade, but if you think you can successfully trade without constant positive feedback, it can prove to be very profitable.
- They attempt to limit losses during the market's sideways mode; no strategy will make money in every market condition, but a good strategy will limit losses in market conditions for which it was not designed.

Trending strategies are the most popular type of strategy. With a high percentage of losing trades, you might be wondering why it is so popular. Very simply, trend-following strategies can be very profitable overall, and this is why people attempt to trade them. Another reason is that people like to follow (and make money on) the big trends. It is human nature to want to cash in on the big moves in the market. It is innately satisfying to get in early on a trend and watch your profits soar.

Directionless Market

A directionless market is characterized by smaller, insignificant up and down movements in price, with the general movement sideways. The key is that the up and down movements are insignificant. Figure 2 shows a chart exhibiting sideways movement [Figure 2, Example of a Sideways Market].

Countertrend Strategies

The main focus of a countertrend strategy is to profit from the price swings that occur in directionless markets. Countertrend strategies start with the premise that markets are directionless 85% of the time. The strategy attempts to take advantage of this price movement and catch the small swings that take place in sideways or choppy markets. This type of strategy has a higher number of winning trades, with small profits on each winning trade. It misses the full trend because it exits early in the trend as the market becomes quickly overbought or oversold.

A countertrend strategy is built on the premise of buying low and selling high. As you are buying when prices are low and selling when prices go up, you are actually trading against the trend. Essentially, you are picking tops and bottoms.

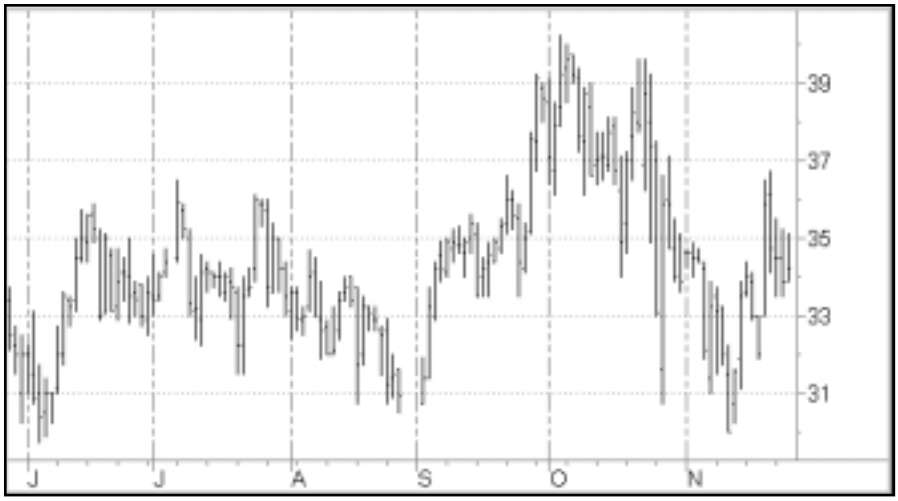


Figure 2. Example of a Sideways Market

Although a countertrend strategy is easier to trade emotionally, many traders don't trade this type of strategy because they miss the big move. The drawback of countertrend strategies is that they usually have small profits and larger losses, losing money as the market trends.

Volatile Market

A volatile market is characterized by sharp jumps in price. Figure 3 contains a volatile market. Notice the opening gaps [Figure 3, Example of a Volatile Market].

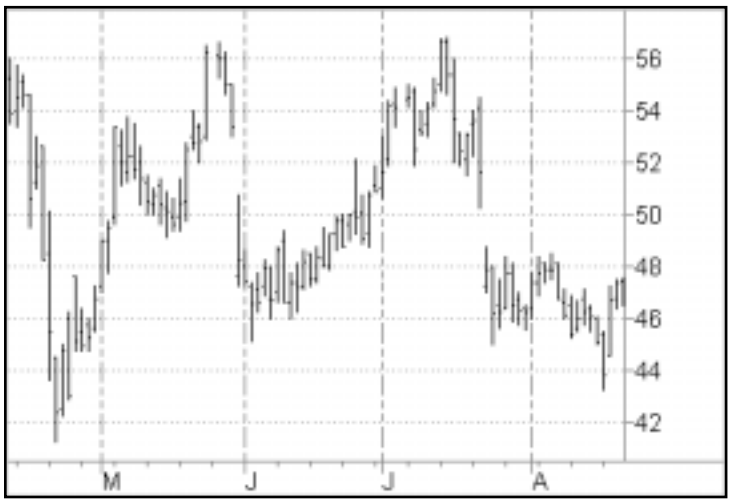


Figure 3. Example of a Volatile Market

Event Strategies

Event strategies are designed to take advantage of the sharp jumps in price characteristic of volatile markets. Event strategies profit from market action like the movement depicted in the chart in Figure 3. Basically the strategy measures recent volatility and attempts to trade an immediate increase by buying an upside breakout with increased volatility or selling a downside breakout as the volatility increases. Another measure of increasing volatility might be the difference or spread between two moving averages, gap openings, or an increase in the daily range.

The trades generated by event strategies are usually short term, and when trading this strategy you should be out of the market a significant amount of time. Event strategies generate a high percentage of winning trades, with small profits per trade. When designing this type of strategy, the key is to effectively anticipate and take advantage of a significant change in volatility and then exit the position before a loss of profit occurs.

Select your Time Frame

As you look at a chart and are evaluating market action type, it is important to consider the time frame of your chart. In fact, choosing which time frame is appropriate for you is almost as important as the type of market action and strategy you want to trade. You can take the same chart and time period, and when you change the time frame, say from daily to weekly, the market action type may be completely different. We'll discuss three basic types of charts: daily, weekly and intraday.

Daily Charts

The most common chart used by traders is the daily chart. Daily charts are the most common for several reasons. First, as most traders also have day jobs, they want to keep abreast of the market as much as possible without it intruding into their workday. The daily chart is perfect for this type of trader. You are able to review the markets each night and make your decisions for the next day.

Weekly Charts

Weekly charts are much more difficult to trade because it takes more discipline. To trade weekly charts you make your decisions on the weekends and don't make any changes until the next weekend. For most traders, this is very difficult to do. It's hard not to look at the market during the week and be tempted to move a stop loss or a money management stop, or not to want to keep your profits and exit the market early.

However, most people don't think of trading weekly charts, and a lot of traders agree that to make money in the markets you have to tread where the average traders fear to tread. Weekly charts are one of those places.

Also, some strategies work better on a weekly chart than on a daily. Very few people have the patience and the discipline to trade weekly charts. By their very nature, weekly charts smooth the price fluctuations of the daily chart. If there is a long trending market, we should be in the trend longer. We might get into and out of the trend a little later than on the daily chart, but we will probably not get whipsawed as much in the directionless markets.

IntraDay Charts

Intraday charts are the 5, 10, 30, and 60-minute charts that are compiled from intraday tick data. Intraday data, if used correctly, can give you a distinct advantage over using daily charts. If you have the time and energy, you can take advantage of the microscopic look at the markets.

However, to trade intraday charts, you must give almost your full attention to the markets during the day. It is virtually impossible to have a full-time job and trade intraday charts well.

Once you select and the strategy type you want to use and the time frame you will be trading, you're ready to start defining your trading rules.

Tips for Defining Your Trading Rules

A large part of strategy development involves writing your rules clearly, without any ambiguity. In fact, you should be able to state your rules so clearly that anyone, using only the Data Window and a calculator, should be able to generate buy and sell signals. It sounds simple until you try it. Unless you have stated your rules in a way that is completely unambiguous, you won't be able to do it. And certainly, TradeStation won't be able to interpret your rules either.

When people first start developing strategies, they tend to focus on the visual aspect. They try to describe their entry criteria, for example, based on the pattern they see in the chart. However, EasyLanguage is mathematical, so eliminating the visual aspect and focusing on the mathematical aspect goes a long way to making sure your strategy does exactly what you want it to do.

Another stumbling block for beginning strategy developers is that sometimes they try to develop the entire strategy all at once. If you find an idea that you think is significant, don't rush to build a strategy around it. First, see if it's a valid idea. For example, you may want to write a ShowMe study to identify the criteria on the chart. Then, you can see if it's worth pursuing.

Once you decide an idea is worth pursuing and start writing a strategy, you still don't have to build the entire strategy. For instance, you can define your entry points only and use just a standard exit like a trailing stop. Or maybe work on the short side only, leaving out the long side. Many strategy developers have a handful of favorite exits which they use with the entries they develop. In other words, they spend their time developing the entries.

TradeStation Basics

Keep the following in mind when you are developing your strategies

You can develop what are called reversal strategies, which will reverse your position but never exit you from the market.

To do this, you use Buy and Sell orders without ExitLong or ExitShort. This is not to say that you wouldn't get stopped out of the market, but your buy and sell orders do not exit you from the market. For example, if you are long 1 share, and the sell criteria are met, the strategy would sell 2 shares. If you are short 1 share, and the buy criteria are met, the strategy would buy 2 shares.

When testing a strategy, make sure you take into account commissions, margin (if applicable) and slippage. Many times, you'll be evaluating a strategy, and it will look really good until you factor in the above costs.

Most importantly, don't throw away an idea because it didn't work right away. Play with it — some ideas just need refining before you can see their true worth.

Developing Your Entry and Exit Rules

Once you have clearly defined the direction you're heading with your strategy, that is, once you've decided on the strategy type and market type you are going to trade, and have a feel for the types of patterns you want to capitalize on, it's time for a brainstorming session. You should sit down in front of your computer with TradeStation and start to develop the set of rules that actually make up your trading strategy.

Many traders at one time or another have become frustrated with strategy development. Not because they don't like it, but because they have run out of new ideas to test or haven't found anything that works for them. For example, most traders have tested the Dual Moving Average Crossover strategy sometime in their trading career. Usually, the trader will look at this strategy and believe that the only thing to test is the length of the two averages; they will experiment with many different lengths for the averages. When they don't find any that work to their satisfaction, they discard the Dual Moving Average concept entirely and move on to something else. They keep looking for that Holy Grail indicator that they can instantly make into a strategy. We have all been there and have all discarded a lot of great ideas. However, more often than not, the discarding of an idea is a mistake.

For the most part, any indicator can be made into a profitable strategy. Yes, any indicator. When we discard the moving averages, it is usually a mistake because the moving averages by themselves only represent one half of the strategy development puzzle. This half is what we refer to as the "Setup" of a strategy. The second half of a strategy, the half that most traders ignore completely, is what we call the "Entry."

The Magic of Setup and Entry

The secret to successful strategy development is to look at a method or indicator in an unconventional manner. The trick is to use it in a different and unique way. With Setup and Entry, you will look at strategy development in a completely different way. As you'll soon see, it can provide you with a whole new world of exciting possibilities and ideas to test. It will lift you out of the rut of simply optimizing standard indicators and give you a method of organizing your creativity.

The Setup

The Setup is the condition or set of conditions that are necessary prior to considering taking a position in the market. It is the indicator or group of indicators that tell you to get ready to buy or sell. Setups don't get you in the market, they simply make you aware that a trade is in the making.

Examples of setups for a trendfollowing strategy:

- A fast moving average crossing a slow moving average
- The ADX indicator in an uptrend
- Prices moving outside of a price channel

Examples of setups for a countertrend strategy:

- The RSI moving into oversold territory (e.g. below 20) or into overbought territory (e.g. above 80)
- SlowK crossing SlowD when using the Stochastic indicator
- Prices reaching the upper or lower line of a moving average envelope

Examples of setups for an event strategy:

- An opening price gap over the high of the previous bar
- The current bar's range is greater than the average range of the last three bars
- The difference between two moving averages on the current bar is greater than the average difference of the last 10 bars

There are countless other indicators and conditions that could be used as setups. In the final analysis, you are limited only by your creativity. There is only one constraint that you should impose upon yourself. It is essential to recognize the type of strategy you are trying to develop and use the different indicators accordingly. You do not want to use a moving average crossover for a countertrend strategy unless you are using it in a unique way. You would not choose to use the Stochastic indicator for a trendfollowing strategy unless you had completely re-configured how it is used. Most strategy traders do not recognize that these indicators only set up the trade. They are unaware that there are a multitude of ways to actually get into the market once the setup has occurred. They are not aware that setups are only part of the equation and are not particularly profitable in and of themselves.

Beginning strategy developers get discouraged when they try to develop profitable strategies from setups only. They quickly run out of ideas to test, because they use up all their ideas as setups without trying to combine them with various complementary entries. By trading only setups, you lose the added precision, accuracy and increased profitability of a strategy that uses both a setup and an entry. If trading setups by themselves worked, trading would be easy, and all traders would be rich.

The Entry

An entry is the signal by which the strategy purchases the contract in the market. It is the technique that you use to take a market position once the rules for the setup have been met. Entry selection is dependent on the type of setup you've designed. You may choose to trade a trendfollowing strategy, a countertrend strategy, or an event strategy. The entries are designed differently depending on the type of strategy you choose to trade. Many beginning traders devise strategies that only trade entries. These are not as effective and are usually less profitable than strategies that utilize both a setup and an entry. Strategies based only on entries tend to have too many trades and a low percentage of profitable trades. There are two rules to which all entries should adhere:

Entry Rules

1. Prices should confirm the direction indicated by the setup before a taking a position.
2. The entry should guarantee that a strategy will capture every price move for which it is designed.

Entry Rule 1 requires prices to move in the expected direction before entering the market. If our setup indicates a long position, we would require the price action to move up in some specified manner before we would be comfortable taking a position. We want the price action to confirm the setup and force us into taking a position. For instance, let's assume that on today's close our setup has given us a long signal. We might require a breakout above the high of today's bar to confirm that the market is in a bullish mode. With this breakout as a condition of entry, we have now required specific market action in the direction of the setup before we risk taking a market position.

Some examples of buy entries are:

- A buy stop one tick above the current bar's high
- A buy stop over the highest high of the last three bars
- Buy at market after several consecutive up closes
- Buy at market after a close over the previous bar's high
- Buy on a close that is greater than the open
- Buy on a stop, one tick above the last swing high
- Buy at the market on the close of a key reversal bar

When deciding which type of signal to use as your entry, it is important to keep in mind the type of strategy you are trying to create. There are certain types of entries you don't want to use with setups because they have basic flaws that may allow the strategy to miss the big move.

Entry Rule 2 is to make sure that your entry guarantees that you will be in on every move that the strategy was designed to catch. The strength of this guarantee is the criterion by which you should judge the viability of all entries. A trendfollowing entry, for example, is flawed if there is even a slight chance that there could be a big move that the entry would miss. This is a very important strategy development principle that you should think about.

For example, you do not want to use a key reversal signal as the only entry for a trendfollowing strategy. There is absolutely no assurance that once the trend setup has occurred that a key reversal will follow. It is possible that after the moving averages have crossed, giving us a buy setup, the market may very well embark on a long uptrend without ever having a key reversal bar. Without the key reversal bar, we would not enter the market even though the trend setup has given us a signal. Without the key reversal bar, we will miss the big move. And missing the big move is the worst outcome for the trend trader.

Another example of a faulty entry is an entry that consists of three consecutive up or down closes. There is no guarantee that given a setup, this pattern will occur. The market may embark on a long trend without having three up or down closes in a row. It is possible that a trendfollowing strategy with this entry could miss the big move, and this possibility is a flaw in the strategy design that should be avoided. That is not to say however that key reversals or consecutive closes should not be used. You could compensate for their shortcomings as entries by including an additional entry or entries in the strategy that would serve as a backup. The entry or combination of entries must guarantee that the strategy will be in the market should any large trend develop.

Understanding Different Order Types

The only limit to creating viable entries is your creativity. There are potentially many techniques that make interesting entries. However, entries are also dependent on the type of order used. There are four basic orders that are commonly used for entries: Market orders, Stop Close Only orders, Stop orders, and Limit orders. Not all of these orders are available on every exchange. You should check the exchange you will be trading on for a list of the available order types.

Market Orders

A market order is used to enter the market without any restrictions on what the price should be. This order is commonly placed on the open of the day (market on open) or close of the day (market on close). However, market orders may also be placed anytime during the day by calling your broker and either buying or selling “at market.”

Although market orders fulfill the criteria for Entry Rule 2, they are deficient because they violate Entry Rule 1. Market orders (market on close, market on open) are not entries at all. They are simply the obvious and easiest way to put on a trade. Market orders may be turned into viable entries by adding another condition to them that will signal an implied direction.

For example, an effective use of a market order would be to “buy tomorrow at market if the open tomorrow is greater than the high of today.” This forces the market to indicate a direction, presumably in the direction of the setup (in this case up) before we enter the market. A market order may be used to enter the market, but should always be used with at least one more condition in order to fulfill Entry Rule 1.

Stop Close Only Orders

Stop Close Only (SCO) orders are market orders with an important twist. The twist is that to enter long, the market must close above a price that we have selected. For a sell, the market must close below our selected price. An example is to buy a contract on the close at 856.30 SCO. This means that if the price closes at or above 856.30, your order will be filled at the market. The idea is that with an SCO order, you have placed an important restriction on the market order, making it a viable entry. This forces the strategy developer to find a price that the market must close above (or below) before the strategy takes a position. By placing this restriction on a market order, we have turned it into a valid type of entry.

Stop Orders

The easiest way to create a valid entry is to use a stop order. By its nature, a stop requires the market to pass through a certain price before a contract is bought or sold. Using a stop order is the best way to create innovative entries and confirm the entry rules. The reason stop orders are generally superior to SCO orders is that they guarantee that your strategy will enter the market regardless of when during the day the price is hit. You will not have to wait for the close, and you may catch a big intraday move that would be lost if you waited for an SCO.

An example of using a stop order as an entry is the bar breakout entry. If today our setup turns bearish, we would place a sell order one tick below today's low. Unless prices move below this price, forcing a confirmation of the set-up, the strategy would not take a short position in the market. The same mechanics would hold true for a long signal. Stop orders are also the best guarantee that the strategy will be in for the big move.

Placing a sell stop (good until cancelled) below the current price provides the best assurance that you will be in on any move beyond that price. The floor brokers must fill your order as soon as they can once that price is hit. This guarantees you will be in on the move, although there is no guarantee as to the exact price (this difference between the stop and the fill price is called slippage).

Limit Orders

Limit orders are the opposite of stop orders. By their nature, limit orders require prices to be traveling in a direction opposite the setup. The primary intent of the limit order is to place a resting buy order somewhere below the present market price. This is an attempt to pick off a lower and better price than where the market is currently. You may also place a resting sell order above the current price to sell at better than current prices.

Limit orders are primarily used in countertrend strategies and are not always effective. Assume that the market is now trading at 258.00. The mechanics of a limit order are to place an order to buy a contract or share at 256.50 limit (or better). This means that the floor brokers who are filling your order will only attempt to buy your contract at a price equal to or less than 256.50. If the broker cannot buy the contract at or below the specified price, you will not be in the market. The same strategy is used with the limit sell order.

The limit order does not conform to Entry Rule 1 because it does not force prices to move in the direction of the setup before entry. There could be a case in which a lower limit price was not reached before the market took off in a big up move. Even if the market by chance should hit this price, there still is no guarantee that the broker will be able to fill the order at that price. Unlike a stop, which becomes a market order at the prescribed price, a limit order must be filled at or better than the prescribed price. The market may trade at that price for only one or two trades, and then move away quickly. You may not get filled even though the market traded at your prescribed price.

Limit orders violate both Entry Rule 1 and Entry Rule 2; therefore, we do not recommend them for use as an entry.

Evaluating Each Component

The basic premise is that most new strategy developers do not organize their strategies in this manner. They use either setups or entries, but not both. Using a setup or entry on its own generally does not work. The power comes when you combine the two.

A very effective method of evaluation is to compare the performance of each component of a strategy by itself as well as the final strategy-take a look at how the setups and entries perform on their own. Compare them with the performance of the final strategy, which includes both the setups and entries. With this type of comparison, you are able to gauge what the different characteristics of each component are and what they add to the mix. This will let you determine whether or not the whole is greater than the sum of its parts.

What you should find is that the combined setup and entry strategy is the most profitable. You should find that setup and entry worked their magic and gave you a better strategy than either of the components could deliver by itself.

By a better strategy, we mean one that you could trade with confidence. Ultimately, the question you have to ask yourself is could you trade this strategy? Could you stick with the strategy you have designed? Just because the strategy is profitable and meets our strategy development criteria does not mean it is one we could or would want to trade. Just because it is profitable does not mean that you are emotionally able to trade it. Many traders create or purchase very profitable strategies, but because their personality doesn't match the strategy, they still lose money, all the while lamenting the fact that they can't stick to the strategy.

Summary

Trading the setup and entry concept and making sure that you follow the rules gives far superior results when compared to trading either setups or entries by themselves. Using both a setup and an entry together enhances the performance of a strategy. Here's how you can summarize how you should think about setups and entries:

AIM WITH THE SETUP

PULL THE TRIGGER WITH THE ENTRY

Always use the concept of setup and entry to develop strategies. There are two distinct parts to strategy writing, and keeping these two components in mind will help you to organize your thoughts and design a sound strategy. Above all, this blueprint for strategy development opens up a whole new range of possibilities for you to test.

CHAPTER 2

Optimizing Your Trading Strategies

What Is Optimization?

Optimization is the process of testing different inputs and stops to fine-tune a trading strategy. You use historical data to test the effects of slight changes in your strategy's criteria.

When you optimize a strategy, TradeStation runs a series of tests based on different values for strategy inputs and/or stops and then automatically picks the parameters that yielded the best results according to the criteria you specify.

When you optimize a strategy, you are not limited to optimizing based on the best net profit. You can optimize for return on account, overall drawdown, or on any of the fields in the Strategy Performance Report, as discussed in the section titled, "Choosing the Criteria for Best Result."

Optimizing Simple Numeric Inputs

For examples of simple numeric inputs, take a look at the strategy STAD13: 3 MA Cross (included on the CD). This strategy uses the inputs Fast, Medium and Slow. The input Fast refers to the number of bars used to calculate the fastest moving average, Medium the second fastest moving average, and Slow the third fastest moving average.

The default values for this strategy are 4, 9, and 18, respectively, which when we tested it on a daily IBM chart generated a net profit of $2,688.10. Figure 1 shows the resulting Strategy Performance Report:

System Report: Performance Summary

MovAvg (3) Crossover International Busines-Daily 07/17/90 - 07/17/98

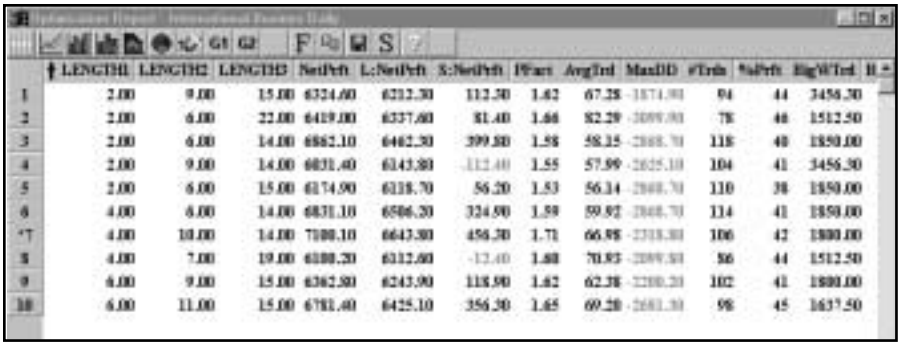
Performance Summary: All Trades

Total net profit	$ 2688.10	Open position P/L	$ 500.00
Gross profit	$ 14731.60	Gross loss	$ -12043.50
Total # of trades	84	Percent profitable	42%
Number winning trades	35	Number losing trades	49
Largest winning trade	$ 1675.00	Largest losing trade	$ -775.00
Average winning trade	$ 420.90	Average losing trade	$ -245.79
Ratio avg win/avg loss	1.71	Avg trade(win & loss)	$ 32.00
Max consec. winners	4	Max consec. losers	5
Avg # bars in winners	36	Avg # bars in losers	14
Max intraday drawdown	$ -3668.50		
Profit factor	1.22	Max # contracts held	100
Account size required	$ 3668.50	Return on account	73%

Figure 1. Strategy Performance Report

However, you may want to test various combinations of input values for the 3 MA Cross strategy. For example, for the Fast input you could test the values from 2 to 6, for Medium the values from 6 to 12, and for Slow the values from 14 to 22, all in increments of one.

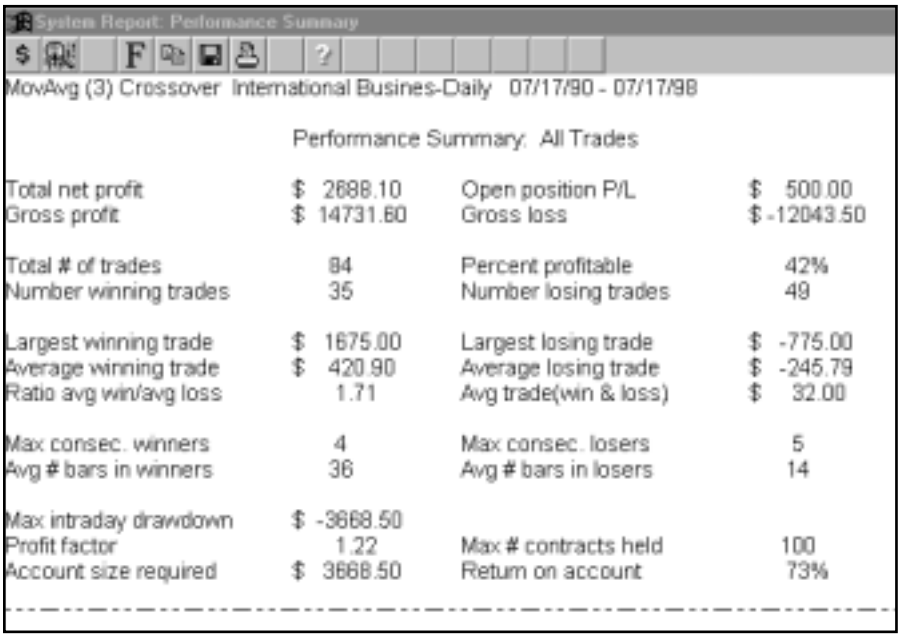
Figure 2 shows a sample Optimization Report for such a test. For this test, we optimized for net profit. In other words, we wanted to find which lengths result in the highest net profit. In this case, the lengths 4, 10 and 14 were the most profitable, as is indicated by the asterisk next to the test number. Notice the net profit rose to $7,100.10.



	LENGTH1	LENGTH2	LENGTH3	NetPrft	L:NetPrft	S:NetPrft	PFact	AvgTrd	MaxDD	#Trds	%Prft	BigWTrd
1	2.00	9.00	15.00	6324.60	6212.30	112.30	1.62	67.28	-1874.90	94	44	3456.30
2	2.00	6.00	22.00	6419.00	6337.60	81.40	1.66	82.29	-2099.80	78	46	1512.50
3	2.00	6.00	14.00	6662.10	6462.30	399.80	1.58	58.35	-2868.70	118	40	1850.00
4	2.00	9.00	14.00	6031.40	6143.80	-112.40	1.55	57.99	-2625.10	104	41	3456.30
5	2.00	6.00	15.00	6174.90	6118.70	56.20	1.53	56.14	-2868.70	110	38	1850.00
6	4.00	6.00	14.00	6831.10	6506.20	324.90	1.59	59.92	-2868.70	114	41	1850.00
*7	4.00	10.00	14.00	7100.10	6643.80	456.30	1.71	66.98	-2318.80	106	42	1800.00
8	4.00	7.00	19.00	6180.20	6312.60	-132.40	1.68	70.93	-2099.80	86	44	1512.50
9	6.00	9.00	15.00	6362.80	6243.90	118.90	1.62	62.38	-2280.20	102	41	1800.00
10	6.00	11.00	15.00	6781.40	6425.10	356.30	1.65	69.20	-2681.30	98	45	1637.50

Figure 2. Optimization Report

Figure 3 shows the improved Strategy Performance Report generated by the optimized strategy.



System Report: Performance Summary

MovAvg (3) Crossover International Busines-Daily 07/17/90 - 07/17/98

Performance Summary: All Trades

Total net profit	$ 2688.10	Open position P/L	$ 500.00
Gross profit	$ 14731.60	Gross loss	$ -12043.50
Total # of trades	84	Percent profitable	42%
Number winning trades	35	Number losing trades	49
Largest winning trade	$ 1675.00	Largest losing trade	$ -775.00
Average winning trade	$ 420.90	Average losing trade	$ -245.79
Ratio avg win/avg loss	1.71	Avg trade(win & loss)	$ 32.00
Max consec. winners	4	Max consec. losers	5
Avg # bars in winners	36	Avg # bars in losers	14
Max intraday drawdown	$ -3668.50		
Profit factor	1.22	Max # contracts held	100
Account size required	$ 3668.50	Return on account	73%

Figure 3. Strategy Performance Report using Optimized Inputs

Determining the Number of Tests

The range of values you specify for inputs and stops when you optimize a trading strategy determines the number of tests TradeStation will perform. The more values for inputs or stops that are optimized, the more tests must be run.

For example, if you want to optimize only one input for the strategy you are testing, and six different values are to be used for that input, TradeStation will perform six tests (1 x 6 = 6). If you optimize two inputs, and you decide to test six different values for each input, the number of tests increases exponentially to 36 (6 x 6 = 36). Each possible combination is tested to determine which performs best. If, in addition to the two inputs, you also included a money management stop with six different values, the number of tests would climb to 216 (6 x 6 x 6).

Depending on the number of tests you run, an optimization can take seconds, minutes, or even hours. You can cut down the time required for testing by either reducing the number of values being tested for each input or stop, or reducing the number of inputs or stops being optimized. For example, if you are optimizing a strategy with three unrelated entry signals, you should optimize the inputs for each entry signal separately. If the inputs are related (so that it's not logical to optimize them separately), you can increase the size of the increment to reduce the number of tests. For example, you might decide to test the input values 10, 20, 30, and 40 rather than all the values between 10 and 40. Then, when you determine the range of input values that performs best, you can re-optimize using this reduced range and a smaller increment.

Choosing the Criterion for Best Result

TradeStation enables you to choose the criterion for the best result when you optimize a trading strategy. The default criterion for the best result of a strategy optimization is the value or combination of values that produces the highest total net profit. However, the value that produces the highest net profit is not always the best result. Would the value that yielded the highest net profit but also produced the largest drawdown and the smallest winning percentage necessarily be the best choice?

TradeStation offers more than 60 criteria for you to choose from for determining the best result of an optimization. For example, instead of Total net profit, you could select Average trade (Wins + Losses), Percent profitable, or Profit factor (how many dollars won for each dollar lost). The criteria for best result are calculated for three categories: All, Long, and Short. You can, for example, optimize for Total net profit in long positions only, Ratio of average win to average loss in short positions only, or Profit factor in both long and short positions. (The criteria within each category are identical.)

Avoiding the Over-Optimization Trap

If you over-optimize your trading strategy's inputs or stops, you run the risk of curve-fitting the parameters to the data you used to test your strategy. Since it is highly unlikely that a lengthy series of price fluctuations will repeat itself exactly in the future, you must avoid the trap of over-optimization. Following are some suggestions for constructive optimization as opposed to curve-fitting optimization:

Do not rely on optimization to create a trading strategy. Optimization should be one of the last steps you take when you are developing a strategy. It is not time to optimize a trading strategy until the strategy is already profitable as it stands.

Develop a trading strategy that is based on a logical trading idea. Avoid trading strategies that yield good optimized results but that are not based on a sound market theory.

Keep your trading strategies as simple as possible. The markets don't pay you any extra for designing and trading a more complex strategy.

Test your trading idea on several stocks and/or commodities. If your test results are poor on other markets, the strategy is probably curve-fitted to one set of data.

The best value discovered by optimization should have a range of profitable values around it. For example, if the optimum value of an input is 20, values on either side of it (e.g. 18, 19, 21, and 22) should also produce good results.

Include both backward and forward testing in your strategy evaluation. The optimal parameters you discovered when testing one set of data should also be profitable on both earlier and later data.

If you optimize your trading strategies with these suggestions in mind, you'll be well on your way to effective, meaningful optimization, and you'll avoid ineffective, curve-fitted optimization

Universalization - A Modified Optimization Approach

In the April, 1993 issue of Technical Analysis of Stocks & Commodities magazine, Adam White wrote an excellent article called "Filtering Breakouts." One topic White introduced in the article was Universalization, which he described as "a modified optimization approach."

At Omega Research, we believe strongly in the value of optimization. We believe that optimization, when applied properly, can be very useful in improving trading strategies. One point we emphasize every chance we get is that optimization should be used to fine-tune trading strategies that are already performing well, not to create trading strategies. Optimization should be one of the last steps in the development of a trading strategy, not one of the first steps. We think that White's Universalization idea may prove helpful to many traders who want to use the power of TradeStation to optimize their strategies, but who are also wary of the dangers of over-optimization.

Following is an example of the Universalization process applied to the STAD13: C-Breakout strategy (included on the CD).

First, select the markets you want to trade with your strategy. For this example, we chose U.S. Bonds, British Pound, Cotton, Crude Oil, Deutsche Mark, Eurodollars, Value Line Index, and Wheat.

Second, run standard optimizations for the value you want to improve. We chose to optimize the channel length of our Channel Break Intrabar system.

Third, rank the optimized values from largest to smallest. Here's our example:

Market Optimized Channel Length

- Crude Oil 47
- Wheat 45
- Cotton 32
- U.S. Bonds 29
- British Pound 27
- Value Line Index 25
- D-Mark 21
- Eurodollars 13

Fourth, drop the largest and the smallest values. In our example, drop 47 and 13.

Fifth, average the remaining values. In our example, we add 45, 32, 29, 27, 25, and 21 for a total of 179 and an average of 29.8, which we round up to 30.

The Universalized parameter is 30. Thirty wasn't the top-performing value for any of the eight markets in our example, but the Universalization process strongly suggests that it is a reasonable, ballpark number for trading this strategy. Applied to our portfolio of eight markets, the 30-bar channel breakout may not turn out to be the best possible choice (with the benefit of hindsight), but it is very unlikely to rank among the worst choices.

The following table shows the results of three tests: (1) a test of the Channel Break Intrabar default value of 10; (2) a test to optimize for the most profitable value; and (3) a test of the Universalized value.

In this example, Universalization produced $77,433 more net profit than the default value, gaining $221,843 compared to the default value's $144,410. Note that the Universalized value produced more profits than the default value in six of the eight markets we tested. Of course, the optimized values made a lot more money than either the default values or the Universalized value; however, we would want to carefully backward and forward test the optimized values to see how well they hold up on out-of-sample data before we would trade with them.

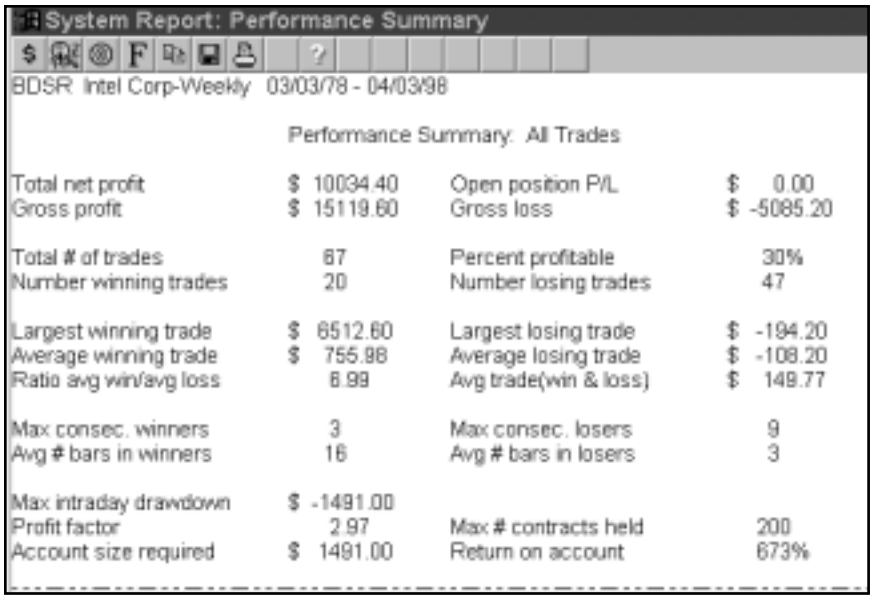
The technique of applying Universalization to a portfolio of optimized values seems to have some merit. Universalization is clearly not a replacement for optimization. It is, however, an interesting step that can be added to our optimization routine as we try to make our strategies more stable and robust.

Market 1/87-12/96	Ten-Bar Default Net Profit	Optimized Value/Net Profit	Universalized Value (30) Net Profit
Crude Oil	$10,970	47/$41,190	$26,620
Wheat	7,031	45/17,331	2,481
Cotton	16,570	32/53,705	18,375
Bonds	16,688	29/59,656	56,344
British Pound	52,063	27/87,350	78,425
Value Line	(9,825)	25/69,725	2,525
D-Mark	37,988	21/45,225	24,088
Eurodollars	12,925	13/25,825	12,975

CHAPTER 3

Making The Most of Your Strategy Performance Reports

TradeStation's Strategy Performance Report provides you with in-depth information about how well or how poorly your trading strategies performed on historical data [Figure 1, Sample Performance Summary]**. The evaluation of a strategy should encompass many important factors in addition to the obvious Net Profit or Net Loss. Here's a brief rundown on several of the most important items featured in the Strategy Performance Report.**


System Report: Performance Summary

BDSR Intel Corp-Weekly 03/03/78 - 04/03/98

Performance Summary: All Trades

Total net profit	$ 10034.40	Open position P/L	$ 0.00
Gross profit	$ 15119.60	Gross loss	$ -5085.20
Total # of trades	67	Percent profitable	30%
Number winning trades	20	Number losing trades	47
Largest winning trade	$ 6512.60	Largest losing trade	$ -194.20
Average winning trade	$ 755.98	Average losing trade	$ -108.20
Ratio avg win/avg loss	6.99	Avg trade(win & loss)	$ 149.77
Max consec. winners	3	Max consec. losers	9
Avg # bars in winners	16	Avg # bars in losers	3
Max intraday drawdown	$ -1491.00		
Profit factor	2.97	Max # contracts held	200
Account size required	$ 1491.00	Return on account	673%

Figure 1. Sample Performance Summary

Total Net Profit

Don't rely solely on this number; it's important, but it doesn't tell the whole story. How a strategy earned the profit is crucial. For example, if a strategy with an acceptable Total Net Profit had 15% winners and a 70% drawdown before hitting a grand slam homerun or two, the strategy probably wouldn't have been tradable in the real world.

Open Position P/L

Include the Last Bar Exit signal (which you'll find in StrategyBuilder) so performance numbers aren't skewed by a large open profit or loss. This signal will automatically exit positions on the last bar of your test data.

Total # of Trades

Every trade incurs a commission, and most trades (except for limit orders) suffer slippage (the difference between the price you want and the price you get when your order is filled). Many strategies are not powerful enough to overcome the costs of making a large number of trades. Emphasize the quality of trades, not the quantity of trades.

Percent Profitable

This number is closely linked to the Ratio Avg Win/Avg Loss. All traders would like to have 90% winners with a 10-to-1 reward-to-risk ratio, but that is not attainable (yet). A trader can achieve a high winning percentage with a low reward-to-risk ratio, a low winning percentage with a high reward-to-risk ratio, or an average winning percentage with an average reward-to-risk ratio. A good, middle-of-the-road strategy would have 40 - 50% winners with a ratio of average win to average loss of 3 or 3.5 to 1.

Largest Winning Trade

This result is more important in a long-term trendfollowing strategy than in a countertrend or event-based strategy (trendfollowing strategies buy high and try to sell higher, countertrend strategies buy low and try to sell high, and event-based strategies buy or sell a condition, pattern, etc. without regard to its location within a trend). Trendfollowing depends on a few “outlier” winning trades (trades more than three standard deviations greater than the average trade) to compensate for a large number of small losses, while the other strategies strive for consistency (lots of small profits vs. fewer losses). Even in a trendfollowing strategy, the largest winning trade shouldn't account for too large a percentage of a strategy's Total Net Profit (more than 40 - 45%): what if that one incredible trade hadn't occurred, or if (for whatever reason) the trader missed that trade? There's also no guarantee that a similar opportunity for a windfall profit will occur again in the reasonably near future for the same strategy in the same market. A good strategy maintains acceptable numbers even when the largest winning trade is deleted from the Performance Summary.

Largest Losing Trade

For professionals, the Largest Losing Trade lost so much because of a gap, a limit move, excessive slippage, or another event beyond the trader's immediate control. Skilled traders do not accept large losing trades as normal occurrences in their strategies. In many profitable strategies, the small wins and small losses approximately cancel each other out, there are no (or very few) big losses, and the Total Net Profit approximates the dollars gained on the big wins.

Ratio Avg Win/Avg Loss

The desired ratio of the average win to the average loss depends on the type of strategy followed. A profitable trendfollowing strategy generally achieves a ratio of at least 3 or 4 to 1, while countertrend and event-based strategies can often make money with ratios of 1 to 1 or even less (with a very high winning percentage).

Avg Trade (win & loss)

The dollar amount won or lost on the average trade may be the most important of the performance measurements. What's the expectation in dollars when a trader makes a trade? It must, of course, be a positive number after deductions are made for slippage and commissions. In addition, traders should consider the other costs of trading (hardware, software, data, books, seminars, etc). The average trade must also earn enough money to compensate the trader adequately for the time he or she devotes to trading, the stress that the trader endures, and the financial risks the trader assumes

Max Consecutive Winners/Max Consecutive Losers

These numbers are not meaningful in isolation from other performance measurements. For example, a strategy that won eight times in a row over the course of 100 trades may not be as profitable as a strategy that only won three times in a row but that had a higher winning percent and/or a higher reward-to-risk ratio. A trendfollowing strategy generally has a greater number of consecutive losers than consecutive winners, while countertrend and event-based strategies usually have a greater number of consecutive winners than consecutive losers. It's not uncommon for a profitable trendfollowing system to suffer ten consecutive small losses, and it's not too unusual for a countertrend or event-based strategy to post ten consecutive small wins.

Avg # Bars in Winners/Avg # Bars in Losers

“Let profits run and cut losses short” is one of the doctrines practiced by successful traders. Most profitable trendfollowing strategies hold winning trades at least three times as long as losing trades; most profitable countertrend and event-based strategies keep winners at least as long as they keep losers.

Max Intraday Drawdown

This statistic represents the largest peak-to-valley decline in equity experienced over the course of testing and/or trading a strategy. Drawdown is one of the most important factors in evaluating a strategy. Most professional traders would prefer a strategy that earned a 35% annual return with a 15% drawdown to a strategy that earned 45% with a 25% drawdown. Although the amount of drawdown that can be tolerated varies among individuals, a strategy that suffers a drawdown greater than 30% in back-testing or actual trading should be re-evaluated and modified to reduce the drawdown. Keep in mind that a 50% drawdown requires a 100% return on the remaining equity just to get back to breakeven!

Profit Factor

Profit Factor, which is calculated by dividing Gross Profit by Gross Loss, represents the number of dollars won for each dollar lost. Obviously, the minimum requirement for a strategy's Profit Factor is that it's a positive number. A guideline to consider is that a robust strategy should generate a Profit Factor of at least 2.0 ($2 won for each $1 lost).

Account Size Required

Account Size Required applies only to strategies that trade futures. In the Performance Report, Account Size Required is calculated by multiplying the margin required to trade one futures contract by the maximum number of contracts held, and adding the result to the Max Intraday Drawdown. Account Size Required is not a recommendation that a strategy be traded with only the required amount of capital behind it. Ideally, an account should be funded with at least two or three times the amount of money specified in Account Size Required to protect against worse-than-anticipated drawdowns.

Return on Account

This number is calculated by dividing Total Net Profit by Account Size Required. The percent return must be interpreted correctly, or the trader will have an inflated idea of the return he or she should expect. Remember that Account Size Required is the minimum amount necessary for trading a strategy, not an ideal amount. Also, the Return on Account represents the percent return over the entire test period — not the percent return on an annual basis.

In Conclusion

TradeStation's most valuable feature is that it allows you to back-test your trading ideas to discover exactly how your idea performed historically before you risk any money in the markets. We hope these brief notes and guidelines will help you to make the best possible use of your Strategy Performance Reports.

SECTION 2

Trending Strategies

CHAPTER 4

Open-Close Histogram

The *Open-Close Histogram* trading strategy is one of the few strategies we've presented that includes the opening price in its calculations. The strategy idea is based on the observation that in an uptrend, a market generally closes above its open, and in a downtrend, a market generally closes below its open. This relationship is certainly not true on every bar, because the market tends to go through correction periods during trending periods, but it's true on average.

Therefore, we will use the relationship between the open and close prices to determine whether the market is in an uptrend on in a downtrend. We will calculate an exponential moving average of the last 10 opens and the exponential moving average of the last 10 closes. We will then subtract the exponential average of the opens from the exponential average of the closes and draw the difference as a histogram.

When the histogram crosses from below zero to above zero, it means that the average close is greater than the average open and the market is in an uptrend; conversely, when the histogram crosses from above zero to below zero, it means that the average close is less than the average open and the market is in a downtrend.

To enter a long position, we will wait for the histogram to cross from below zero to above zero. Then, we'll place a buy stop at the high of that bar plus half the 10-bar average true range (Average true range is the largest of the following: today's high minus today's low, yesterday's close minus today's low, or today's high minus yesterday's close). The buy stop will remain in effect at that price for 10 bars or until the histogram crosses from above zero to below zero, whichever occurs first.

Once we enter a long position, we'll set a protective stop at the low of the bar where the histogram crossed over zero minus half of the 10-bar average true range. Also, we will exit from the long position whenever the histogram crosses under zero.

To enter a short position, we will wait for the histogram to cross from above zero to below zero. Then, we'll place a sell stop at the low of that price bar minus half of the 10-bar average true range. The sell stop will remain in effect at that price for 10 bars or until the histogram crosses from below zero to above zero, whichever occurs first.

Once we enter into a short position, we'll set our protective stop at the high of the bar on which the histogram crossed under zero plus half of the average true range. We will exit from the short position when the histogram crosses above zero.

Figure 1 shows the *Open-Close Histogram* strategy and indicator applied to a daily chart of IBM [Figure 1, IBM Chart].

Figure 1. IBM Chart

Defining Our Trading Rules

In this strategy, we defined long and short entries and exits. The long and short entries reverse your position, whereas the exits close out your existing position. We also performed some setup work, which involved calculating the exponential averages and calculating their difference. The setup, entry and exits are described next.

Setup

a) Calculate a 30-bar exponential average of the open prices.

b) Calculate a 30-bar exponential average of the close.

c) Subtract the average of the open prices from the average of the close.

Long and Short Entries

a) When the histogram crosses from below zero to above zero, we will place a buy stop at the high of that bar plus half the 10-bar average true range. The buy stop will remain in effect at that price for 10 bars or until the histogram crosses from above zero to below zero, whichever occurs first.

b) When the histogram crosses from above zero to below zero, we'll place a sell stop at the low of that price bar minus half of the 10-bar average true range. The sell stop will remain in effect at that price for 10 bars or until the histogram crosses from below zero to above zero, whichever occurs first.

Long and Short Exits

a) We will exit from the long position whenever the histogram crosses under zero

b) Also, on the first bar of the long position, we'll set a protective stop at the low of the bar where the histogram crossed over zero minus half of the 10-bar average true range.

c) We will exit from the short position when the histogram crosses above zero.

d) Also, once we enter into a short position, we'll set a protective stop at the high of the bar on which the histogram crossed under zero plus half of the average true range.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Open-Close Histogram (STAD13: OC Histogram)

Strategy Inputs (STAD13: OC Histogram)

INPUT	DEFAULT	DESCRIPTION
Length	30	The length used to determine the calculation of the exponential moving average for both Open and Close values, and the average true range calculation
EntryFactor	.5	The multiple of average true ranges used to determine the values for entry into a Long or Short position
ExitFactor	.5	The multiple of average true ranges used to determine the values for exit from a Long or Short position

Strategy Components:

1. *Open-Close Histogram*

2. Last Bar Exit

EasyLanguage Signal: Open-Close Histogram:

```
Inputs: Length(10), EntryFactor(.5), ExitFactor(.5);
Variables: Histogram(0), ATR(0), BuyPrice(0), SellPrice(0), LongExitPrice(0), ShortExitPrice(0);
```

```
{ Setup calculation - calculating the exponential averages spread }
Histogram = XAverage(Close, Length) - XAverage(Open, Length);
ATR = AvgTrueRange(Length);

If Histogram Crosses Over 0 Then Begin
        BuyPrice = High + (ATR * EntryFactor);
        LongExitPrice = Low - (ATR * ExitFactor);
End;

If Histogram Crosses Under 0 Then Begin
        SellPrice = Low - (ATR * EntryFactor);
        ShortExitPrice = High + (ATR * ExitFactor);
End;

{ Long entry orders }
If MRO(Histogram Crosses Over 0, 10, 1) > -1 AND Histogram > 0 Then
        Buy ("LE") next bar at BuyPrice Stop;

{ Short entry orders }
If MRO(Histogram Crosses Under 0, 10, 1) > -1 AND Histogram < 0 Then
        Sell ("SE") next bar at SellPrice Stop;

{ Exits }
If Histogram crosses under 0 Then
        ExitLong this bar at Close;
ExitLong from entry ("LE") next bar at LongExitPrice Stop;

If Histogram Crosses Over 0 Then
        ExitShort this bar at Close;
ExitShort from entry ("SE") next bar at ShortExitPrice Stop;
```

Signal Inputs (Open-Close Histogram)

INPUT	DEFAULT	DESCRIPTION
Length	10	The length used to determine the calculation of the exponential moving average for both Open and Close values, and the average true range calculation
EntryFactor	.5	The multiple of average true ranges used to determine the values for entry into a Long or Short position
ExitFactor	.5	The multiple of average true ranges used to determine the values for exit from a Long or Short position

Signal Variables (Open-Close Histogram)

INPUT	DEFAULT	DESCRIPTION
Histogram	0	[Numeric] Stores the difference between the exponential average of Close prices and the exponential average of Open prices
ATR	0	[Numeric] Stores the calculation of the average true range
BuyPrice	0	[Numeric] Stores the value used to enter a Long position
SellPrice	0	[Numeric] Stores the value used to enter a Short position
LongExitPrice	0	[Numeric] Stores the value used to exit a Long position
ShortExitPrice	0	[Numeric] Stores the value used to exit a Short position

Setup

The *Open-Close Histogram* is calculated by taking the exponential average of the Open prices and subtracting that from the exponential average of the Close prices. That result is stored in the variable Histogram, and the average true range is calculated and stored in the variable ATR.

```
Histogram = XAverage(Close, Length) - XAverage(Open, Length);
ATR = AvgTrueRange(Length);
```

When the Histogram crosses above zero, that bar is used to determine any future entry and exit values for a Long position. BuyPrice, used for the entry, is calculated as the High of the bar plus EntryFactor (Input) times the current ATR. Conversely, LongExitPrice, used for the exit, is calculated as the Low minus ExitFactor (Input) times the ATR.

```
If Histogram Crosses Over 0 Then Begin
        BuyPrice = High + (ATR * EntryFactor);
        LongExitPrice = Low - (ATR * ExitFactor);
End;
```

When the Histogram reverses and crosses under zero, a similar initialization of values for a Short position is taken. On the bar where the Histogram crosses below zero, SellPrice is initialized to the Low minus EntryFactor times the ATR and ShortExitPrice is calculated as the High plus ExitFactor times the ATR.

```
If Histogram Crosses Under 0 Then Begin
        SellPrice = Low - (ATR * EntryFactor);
        ShortExitPrice = High + (ATR * ExitFactor);
End;
```

Long and Short Entries

The Long entry should be close enough to the crossover that the setup still holds value. The use of the MRO() function (most recent occurrence) allows for determining how long ago a condition occurred, if at all. Using a length of 10 and looking for the first condition, the MRO() function will return the number of bars ago a condition occurred within the length specified, or -1 if the condition did not occur. Testing that the MRO() function does not return -1 means that the condition occurred. If the Histogram recently crossed above zero (within the last ten bars) and is still above zero, a Buy order is placed on a Stop at the previously calculated BuyPrice.

```
If MRO(Histogram Crosses Over 0, 10, 1) > -1 AND Histogram > 0 Then
        Buy ("LE") next bar at BuyPrice Stop;
```

If the Histogram recently crossed below zero (within the last ten bars) and is still below zero, a Sell order is placed on a Stop at the previously calculated SellPrice.

```
If MRO(Histogram Crosses Under 0, 10, 1) > -1 AND Histogram < 0 Then
        Sell ("SE") next bar at SellPrice Stop;
```

Long and Short Exits

There are two exit scenarios for a Long or Short position. Because the setup requirements for a Long position is the Histogram greater than zero, if the Histogram crosses below zero the setup requirement is not met and the position will be exited. Additionally, on each bar a Long exit order will be generated at the value of LongExitPrice, generated at the time the Histogram crossed above zero.

```
If Histogram crosses under 0 Then
        ExitLong this bar at Close;
ExitLong from entry ("LE") next bar at LongExitPrice Stop;
```

On the short side, the exits are generated when the Histogram crosses above zero, negating the setup requirement for a short position, and also at a price of ShortExitPrice, calculated when the Histogram most recently crossed below zero.

```
If Histogram Crosses Over 0 Then
        ExitShort this bar at Close;
ExitShort from entry ("SE") next bar at ShortExitPrice Stop;
```

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing & Improving

We tested the *Open-Close Histogram* strategy on daily data for IBM (long side only) and Crude Oil (CL). The test period for IBM was 6/92 - 4/00; for CL it was 1/95 - 3/00. We set the Max number of bars strategy will reference to 50 and did not make any deductions for slippage or commission.

The strategy's default values and testing parameters are as follows:

Length = 30, testing 10 - 50 in increments of 10

EntryFactor = .50, testing .10 - 1.0 in increments of .10

ExitFactor = .50, testing .10 - 1.0 in increments of .10

Let's take a look at the strategy's test results in IBM. Here are the optimized values:

Length = 50

EntryFactor = .90

ExitFactor = .10

Buying 100 shares per trade, the strategy earned $5,169 on 21 trades, with 67% of the trades profitable [Figure 2, IBM Performance Summary]. The average winning trade was 2.27 times the dollar amount of the average losing trade, and the strategy earned $4.54 for each $1.00 it lost (Profit Factor). The Annual Trading Summary table shows that the strategy was profitable in every year of the test period [Figure 3, IBM Annual Trading Summary]. The Equity Curve displays two periods of relatively flat performance and two periods of very strong performance [Figure 4, IBM Equity Curve].


TradeStation Strategy Performance Report

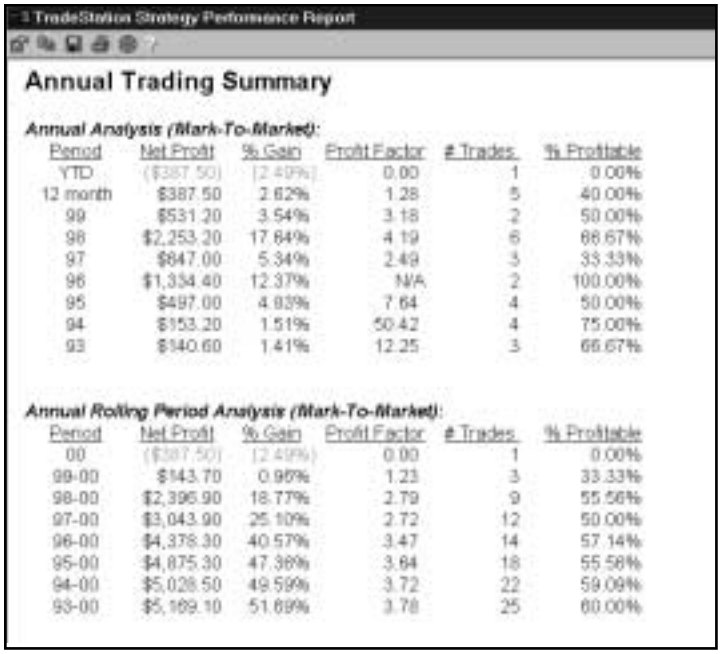
TradeStation Strategy Performance Report - STAD13: OC Histogram IBM-Daily

Performance Summary: All Trades

Total Net Profit	$5,169.10	Open position P/L	$0.00
Gross Profit	$6,628.40	Gross Loss	($1,459.30)
Total # of trades	21	Percent profitable	66.67%
Number winning trades	14	Number losing trades	7
Largest winning trade	$2,362.50	Largest losing trade	($650.00)
Average winning trade	$473.46	Average losing trade	($208.47)
Ratio avg win/avg loss	2.27	Avg trade (win & loss)	$246.15
Max consec. Winners	5	Max consec. losers	2
Avg # bars in winners	49	Avg # bars in losers	11
Max intraday drawdown	($960.70)		
Profit Factor	4.54	Max # contracts held	100
Account size required	$338,490.69	Return on account	1.53%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win/Loss | Time | Graphs | Settings

Figure 2. IBM Performance Summary



TradeStation Strategy Performance Report

Annual Trading Summary

Annual Analysis (Mark-To-Marked):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
YTD	($387.50)	[2.49%]	0.00	1	0.00%
12 month	$387.50	2.62%	1.28	5	40.00%
99	$531.20	3.54%	3.18	2	50.00%
98	$2,253.20	17.64%	4.19	6	66.67%
97	$847.00	5.34%	2.49	3	33.33%
96	$1,334.40	12.37%	N/A	2	100.00%
95	$497.00	4.83%	7.64	4	50.00%
94	$153.20	1.51%	50.42	4	75.00%
93	$140.60	1.41%	12.25	3	66.67%

Annual Rolling Period Analysis (Mark-To-Marked):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
00	($387.50)	[2.49%]	0.00	1	0.00%
99-00	$143.70	0.96%	1.23	3	33.33%
98-00	$2,396.90	18.77%	2.79	9	55.56%
97-00	$3,043.90	25.10%	2.72	12	50.00%
96-00	$4,378.30	40.57%	3.47	14	57.14%
95-00	$4,875.30	47.36%	3.64	18	55.56%
94-00	$5,028.50	49.59%	3.72	22	59.09%
93-00	$5,169.10	51.69%	3.78	25	60.00%

Figure 3. IBM Annual Trading Summary

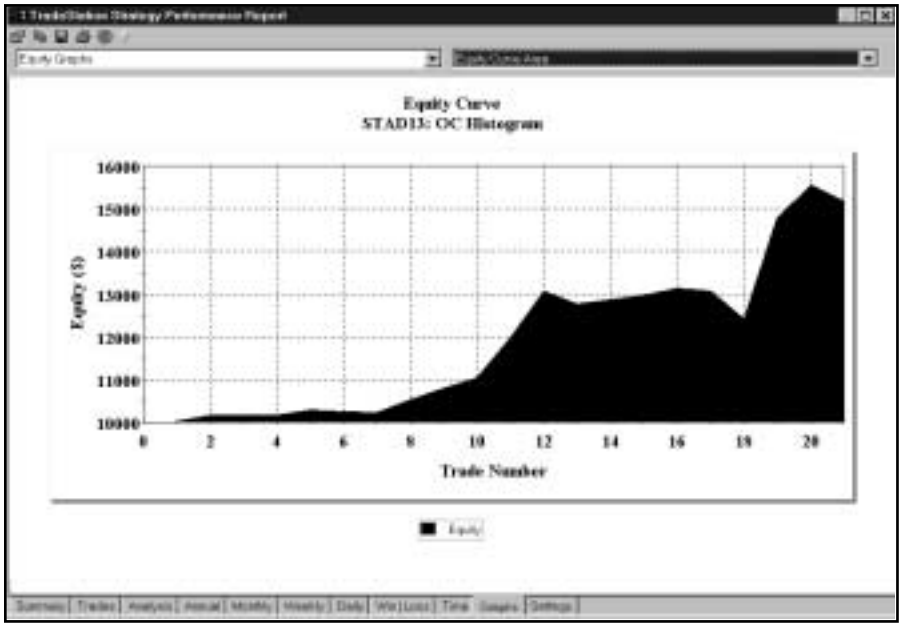


Figure 4. IBM Equity Curve

The Underwater Equity Curve is shown in Figure 5 [Figure 5, IBM Underwater Equity Curve]. Strategies always look bad on this graph because it is designed to paint the most pessimistic portrait possible of a strategy's performance. Think of the Underwater Equity Curve as a kind of reality check for your strategy. The small bars rising above the zero line represent new equity peaks, often referred to as high-water marks. Because the bars are not drawn to scale-and they're all the same height-they don't provide a true sense of how much money we're making when the strategy is performing well. The equity drawdowns, which extend down from the zero line, are drawn to scale, depicting the duration and magnitude of the strategy's losses when it's performing poorly. Note that the worst drawdown for our strategy in IBM was only about eight percent, and the second worst drawdown was less than six percent-not as bad as it looks on the Underwater Equity Curve graph.

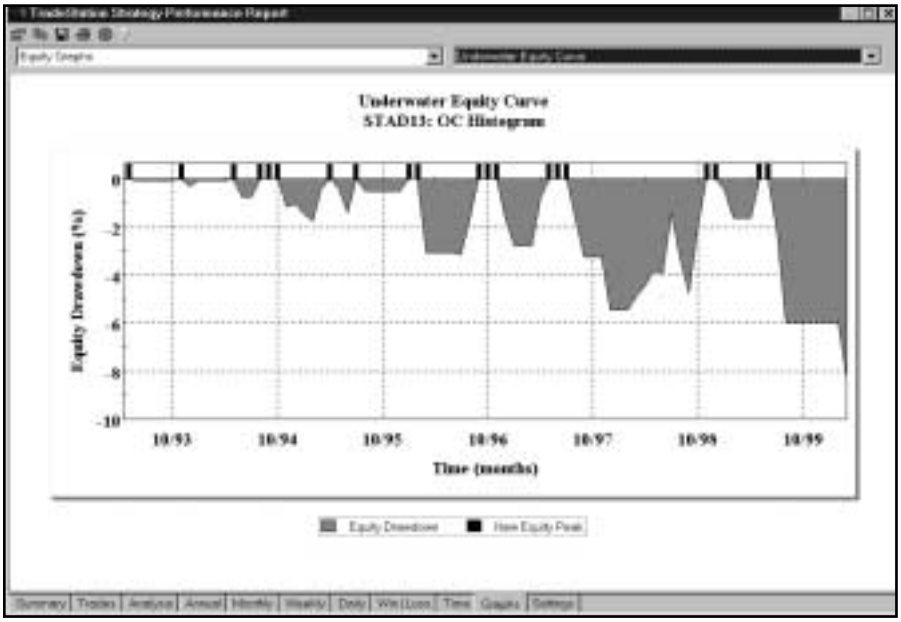


Figure 5. IBM Underwater Equity Curve

The Average Profit by Month graph shows our strategy's monthly performance when each month's returns are averaged over the length of the test period [Figure 6, IBM Average Profit by Month]. Nine months were profitable and only three months were unprofitable when performance was averaged for each month over the seven years of test data.

Next, let's see how well the Open-Close Histogram strategy handled the Crude Oil futures market (CL).

The optimized values are as follows:

Length = 40

EntryFactor = .10

ExitFactor = .20

Figure 7 is a chart of CL with the *Open-Close Histogram* strategy and indicator applied [Figure 7, CL Chart]. The strategy bought CL in February of 1999 after the histogram crossed from below zero to above zero, and prices climbed to the high of the setup bar plus 10 percent of the 10-bar average true range. In October, the strategy exited when the histogram crossed below the zero line.

Figure 6. IBM Average Profit by Month

Figure 7. CL Chart

The Performance Summary is encouraging: the net profit was $38,029 on 36 trades, of which 42% were profitable [Figure 8, CL Performance Report]. The largest winner ($13,090) far eclipsed the largest loser ($1,522), and the average winner ($3,250) was 6.37 times as large as the average loser ($511). The average trade earned a hefty $1,056, and the strategy earned $4.55 for each $1.00 it lost (Profit Factor). Keeping true to the trading maxim “Let profits run, and cut losses short,” the strategy rode winners for an average of 71 bars, while it abandoned losers in an average of only five bars.


TradeStation Strategy Performance Report

TradeStation Strategy Performance Report - STAD13: OC Histogram CL.LNG-Daily

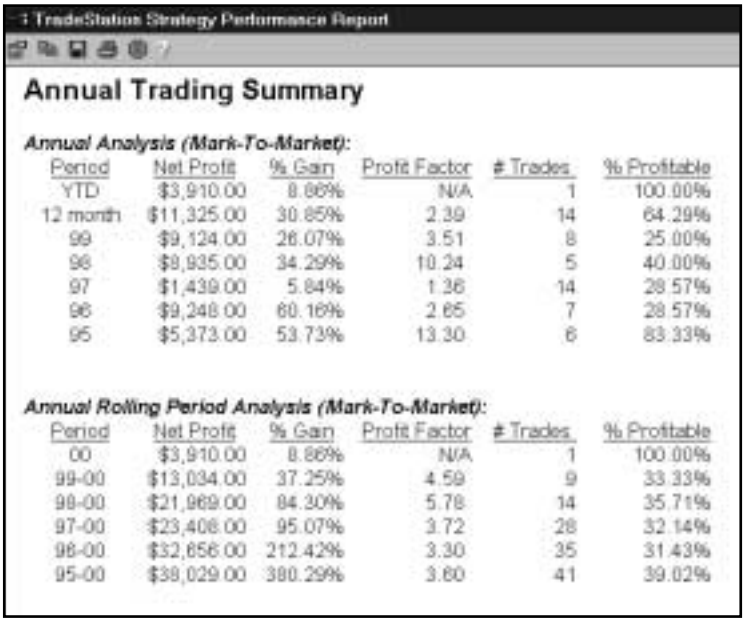
Performance Summary: All Trades

Total Net Profit	$38,029.00	Open position P/L	$0.00
Gross Profit	$48,752.00	Gross Loss	($10,723.00)
Total # of trades	36	Percent profitable	41.67%
Number winning trades	15	Number losing trades	21
Largest winning trade	$13,090.00	Largest losing trade	($1,522.00)
Average winning trade	$3,250.13	Average losing trade	($510.82)
Ratio avg win/avg loss	6.37	Avg trade (win & loss)	$1,056.36
Max consec. Winners	4	Max consec. losers	4
Avg # bars in winners	71	Avg # bars in losers	5
Max intraday drawdown	($4,084.00)		
Profit Factor	4.55	Max # contracts held	1
Account size required	$8,039.00	Return on account	473.06%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win | Loss | Time | Graphs | Settings

Figure 8. CL Performance Report

The *Open-Close Histogram* strategy also demonstrated consistent performance year-by-year in the test period, posting a net profit every year [Figure 9, CL Annual Trading Summary]. The Equity Curve shows three periods of relatively flat performance and three periods of strong equity growth [Figure 10, CL Equity Curve]. This is important because it demonstrates that our strategy earned substantial profits when the Crude Oil market was trending but didn't give back too much of the profits when the market was choppy.


TradeStation Strategy Performance Report

Annual Trading Summary

Annual Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
YTD	$3,910.00	8.86%	N/A	1	100.00%
12 month	$11,325.00	30.85%	2.39	14	64.29%
99	$9,124.00	26.07%	3.51	8	25.00%
98	$8,935.00	34.29%	10.24	5	40.00%
97	$1,439.00	5.84%	1.36	14	28.57%
96	$9,248.00	60.16%	2.65	7	28.57%
95	$5,373.00	53.73%	13.30	6	83.33%

Annual Rolling Period Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
00	$3,910.00	8.86%	N/A	1	100.00%
99-00	$13,034.00	37.25%	4.59	9	33.33%
98-00	$21,969.00	84.30%	5.78	14	35.71%
97-00	$23,408.00	95.07%	3.72	28	32.14%
96-00	$32,656.00	212.42%	3.30	35	31.43%
95-00	$38,029.00	380.29%	3.60	41	39.02%

Figure 9. CL Annual Trading Summary

The drawdown in 1996 and 1997 looks forbidding as depicted in the Underwater Equity Curve, but at its worst the drawdown reached only about 17%, not too bad for a trendfollowing strategy [Figure 11, CL Underwater Equity Curve]. The graph of Average Profit by Month shows that our strategy demonstrated fairly consistent performance when monthly returns were averaged over the length of the test period: nine months returned profits overall, compared to only three that returned losses [Figure 12, CL Average Profit by Month]. The Total Trades graph places each trade in sequence on the horizontal axis and each trade's dollar gain or loss on the vertical axis [Figure 13, CL Total Trades]. The bold horizontal line represents the average trade, and the large ball just before trade 10 represents a positive outlier — a trade more than three standard deviations greater than the average trade. Trendfollowing strategies often make a substantial portion of their total net profits on a few positive outliers, while posting few, if any, negative outliers.

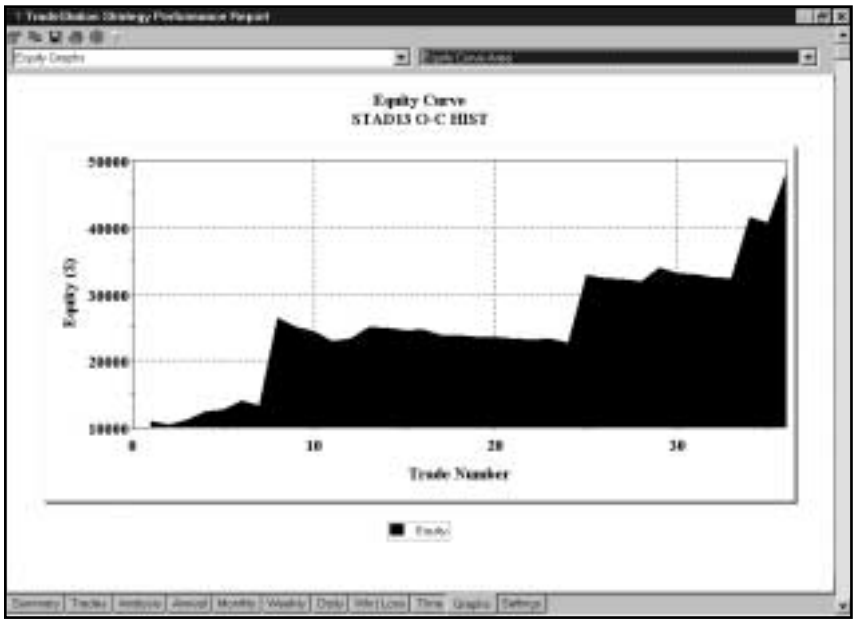


Figure 10. CL Equity Curve

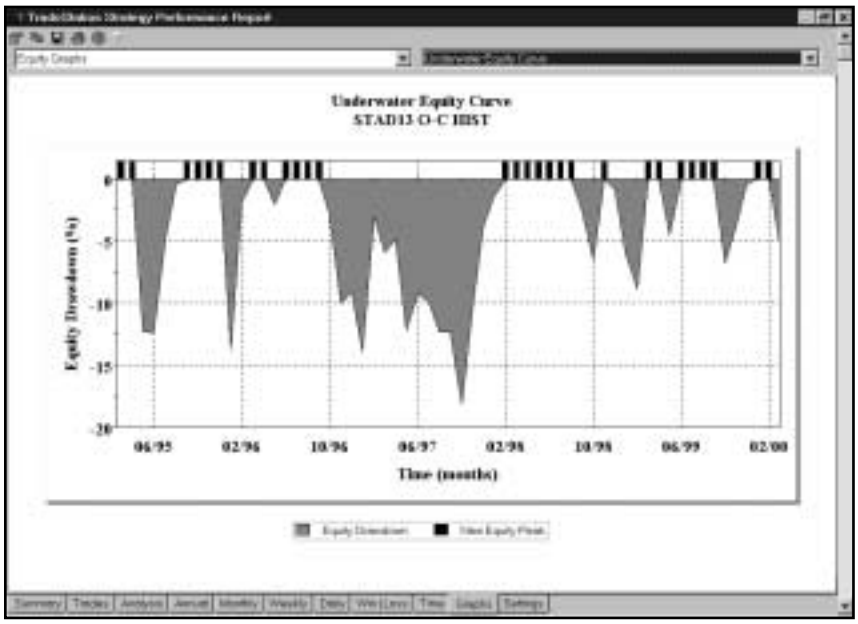


Figure 11. CL Underwater Equity Curve

Figure 12. CL Average Profit by Month

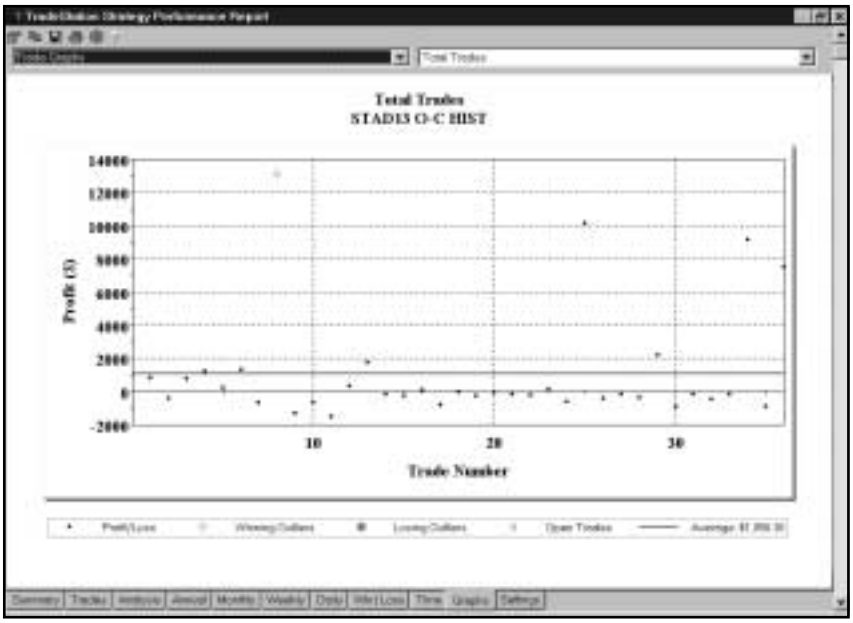


Figure 13. CL Total Trades

Summary

The *Open-Close Histogram* is a good example of a relatively simple strategy generating favorable results. Especially if you're new to strategy trading, don't make your strategies overly complex. Base your strategy on an idea you understand and test it thoroughly. That's the only way you'll know if your idea has the potential to become a winning trading strategy.

CHAPTER 5

DMA and Range Leaders

Our *DMA and Range Leaders* strategy (*DMARL*, for short) constructs a displaced moving-average channel of highs and lows and then looks for a simple chart pattern known as a Range Leader.

A displaced moving average (DMA) differs from other types of moving averages in that it is shifted forward a specified number of bars rather than plotted on the bar for which it was calculated. This allows us to know the numeric value the moving average will have on a bar in the future. Many traders believe that displaced moving averages can reduce the "whipsaws" (the many small losses due to false signals) that occur when a standard (not displaced) moving average is applied to a price series. For this strategy, we'll construct a displaced moving-average channel by calculating moving averages of highs and lows and shifting them forward a number of bars.

Next, we'll look for the Range Leader pattern. A bullish Range Leader is a bar with a midpoint above the previous bar's high and a range greater than the previous bar's range; a bearish Range Leader has a midpoint below the previous bar's low and a range greater than the previous bar's range.

The setup to buy is a bullish Range Leader with a close above the DMA of highs, and the long entry is on the next bar at the high of the Range Leader. The setup to sell short is a bearish Range Leader with a close below the DMA of lows, and the short entry is on the next bar at the low of the Range Leader. We'll also add a money-management stop, a breakeven stop, and a dollar risk trailing stop in StrategyBuilder.

Figure 1 is a daily bar chart of Intel with the DMA channel and a very profitable long trade [Figure 1, INTC chart].

Figure 1. INTC Chart

Defining Our Trading Rules

For this strategy, we defined long and short setups, entries, and exits. These items are described next.

Long and Short Setup

a) Our setup to buy is a bullish Range Leader with a close above the DMA of highs.

b) Our setup to sell short is a bearish Range Leader with a close below the DMA of lows.

Long and Short Entries

a) Our long entry is on the next bar at the Range Leader's high plus one point.

b) Our short entry is on the next bar at the Range Leader's low minus one point.

Long and Short Exits

a) Exit a long position on a decline to the Stop Loss, the Breakeven Stop, or the Dollar Risk Trailing Stop.

b) Exit a short position on a rally to the Stop Loss, the Breakeven Stop, or the Dollar Risk Trailing Stop.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: DMA and Range Leaders (STAD13: RangeLeaders)

Strategy Inputs (STAD13: RangeLeaders)

INPUT	DEFAULT	DESCRIPTION
PositionBasis	True	Determines if the Breakeven, Dollar Risk and Stop Loss exit criteria will be based on a position basis (True) or on a per contract/share basis (False)
DllrRiskTrailing	500	The amount of loss from the highest profit, in dollars, at which the position will be closed
StopLoss	500	The strict amount of loss, in dollars, at which point the position will be closed
BreakevenFloor	500	The amount of profit that must be met before the Breakeven stop takes effect
DMAvgLength	20	Used as the length parameter in calculating the average of the High and Low prices
Displacement	5	Used as the displacement for the calculation of the upper and lower bands and the exit value

Strategy Components:

1. *DMA & Range Leaders*
2. Breakeven Stop - floor
3. Dllr Risk Trailing
4. Stop Loss
5. Last Bar Exit

EasyLanguage Signal: DMA & Range Leaders:

```
Inputs:
        Length(20), Displacement(5);
Variables:
        UpperBand(0), LowerBand(0), BuyRangeLeader(False), SellRangeLeader(False);

UpperBand = Average(High, Length)[Displacement];
LowerBand = Average(Low, Length)[Displacement];
```

```
BuyRangeLeader = MedianPrice > High[1] AND Range > Range[1];
SellRangeLeader = MedianPrice < Low[1] AND Range > Range[1];

If BuyRangeLeader AND Close > UpperBand Then
        Buy Next Bar at High + 1 point Stop;

If SellRangeLeader AND Close < LowerBand Then
        Sell Next Bar at Low - 1 point Stop;
```

Signal Inputs (DMA & Range Leaders)

INPUT	DEFAULT	DESCRIPTION
Length	20	Used as the length parameter in calculating the average of the High and Low prices
Displacement	5	Used as the displacement for the calculation of the upper and lower bands and the exit value

Signal Variables (DMA & Range Leaders)

INPUT	DEFAULT	DESCRIPTION
UpperBand	0	[Numeric] Stores the calculation of the average of High prices
LowerBand	0	[Numeric] Stores the calculation of the average of Low prices
BuyRangeLeader	False	[True/False] Used to determine a Buy setup occurring on a Range Leader bar and trigger a Long entry order
SellRangeLeader	False	[True/False] Used to determine a Sell setup occurring on a Range Leader bar and trigger a Short entry order

Setup

In order to calculate the values of the upper and lower bands, the displaced average of the High and Low prices are taken. The averages are calculated using the function Average() and they are calculated at Displacement (Input) bars ago.

```
UpperBand = Average(High, Length)[Displacement];
LowerBand = Average(Low, Length)[Displacement];
```

If the current bar is a range leader that should generate Long entry orders, the midpoint of the bar will be greater than the High of the previous bar and the range will also be greater than previous. These conditions are evaluated together using AND. The result is stored in the variable BuyRangeLeader.

```
BuyRangeLeader = MedianPrice > High[1] AND Range > Range[1];
```

For a bar that should generate Short entry orders, the midpoint of the bar will be below that of the previous bar's Low and the range will be greater than the previous. These conditions are evaluated together using AND. The result is stored in the variable SellRangeLeader.

```
SellRangeLeader = MedianPrice < Low[1] AND Range > Range[1];
```

Long and Short Entries

If the current bar is a range leader that will generate a Long entry order, the final condition is dependent on the position of the Close of the bar. If the Close of the bar is also greater than the upper average band, then an entry order is generated for the next bar at the High of the current bar plus one point.

```
If BuyRangeLeader AND Close > UpperBand Then
        Buy next bar at High + 1 point Stop;
```

Using the same structure as the order generation for a Long entry position, SellRangeLeader is evaluated. If the Close of the bar is also less than the lower average band, then an entry order is generated for the next bar at the Low of the current bar minus one point.

```
If SellRangeLeader AND Close < LowerBand Then
        Sell Next Bar at Low - 1 point Stop;
```

EasyLanguage Signal: Breakeven Stop - floor:

```
Inputs: PositionBasis(True), FloorAmnt(0);

If PositionBasis Then
        SetStopPosition
Else
        SetStopContract;

SetBreakeven(FloorAmnt);
```

Signal Inputs (Breakeven Stop - floor)

INPUT	DEFAULT	DESCRIPTION
PositionBasis	True	Determines if the exit criteria will be based on a position basis (True) or on a per contract/share basis (False)
FloorAmnt	0	The amount of profit that must be met before the stop takes effect

This Signal does not contain any Variables.

Setup

If PositionBasis (Input) is True, the breakeven point will be calculated on a position basis. If it is False, the breakeven point will be calculated on a per contract/share basis.

```
If PositionBasis Then
        SetStopPosition
Else
        SetStopContract;
```

Long and Short Exits

The SetBreakeven statement calculates the Breakeven Stop value for either a Long or Short position, based on the FloorAmnt (Input) specified.

```
SetBreakeven(FloorAmnt);
```

EasyLanguage Signal: DIlr Risk Trailing:

```
Inputs: PositionBasis(True), Amount(0);

If PositionBasis Then
        SetStopPosition
Else
        SetStopContract;

SetDollarTrailing(Amount);
```

Signal Inputs (DIlr Risk Trailing)

INPUT	DEFAULT	DESCRIPTION
PositionBasis	True	Determines if the exit criteria will be based on a position basis (True) or on a per contract/share basis (False)
Amount	0	The amount of loss from the highest profit, in dollars at which the position will be closed

This Signal does not contain any Variables.

Setup

If PositionBasis (Input) is True, the dollar risk is calculated on a position basis. If it is False, the dollar risk is calculated on a per contract/share basis.

```
If PositionBasis Then
        SetStopPosition
Else
        SetStopContract;
```

Long and Short Exits

The SetDollarTrailing statement calculates the Dollar Risk Trailing Stop for either a Long or Short position based on the Amount (Input) specified.

```
SetDollarTrailing(Amount);
```

EasyLanguage Signal: Stop Loss

** See Common Stops Appendix

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing & Improving

We tested the *DMARL* strategy on daily bars of Intel (INTC, long side only) and Coffee futures (KC). The optimized values for INTC (100 shares) are as follows:

StopLoss = $700

BreakevenFloor = $700

DllrRiskTrailing = $900

DMAvgLength = 30

Displacement = 10

Applied to INTC, our *DMARL* strategy earned $9,314 (per 100 shares) on 16 trades, for an average trade of $582 [Figure 2, INTC Performance Summary]. The strategy was 63% correct, and the average winning trade ($1,001) was 8.65 times as large as the average losing trade ($582). The $2,125 largest winner far exceeded the $391 largest loser.

The Annual Trading Summary also shows strong results, with the strategy trading profitably for every year in the test period [Figure 3, INTC Annual Trading Summary]. Although never dipping into negative territory, the Equity Curve [Figure 4, INTC Equity Curve] depicts slow gains over the first ten trades but impressive gains thereafter.


TradeStation Strategy Performance Report

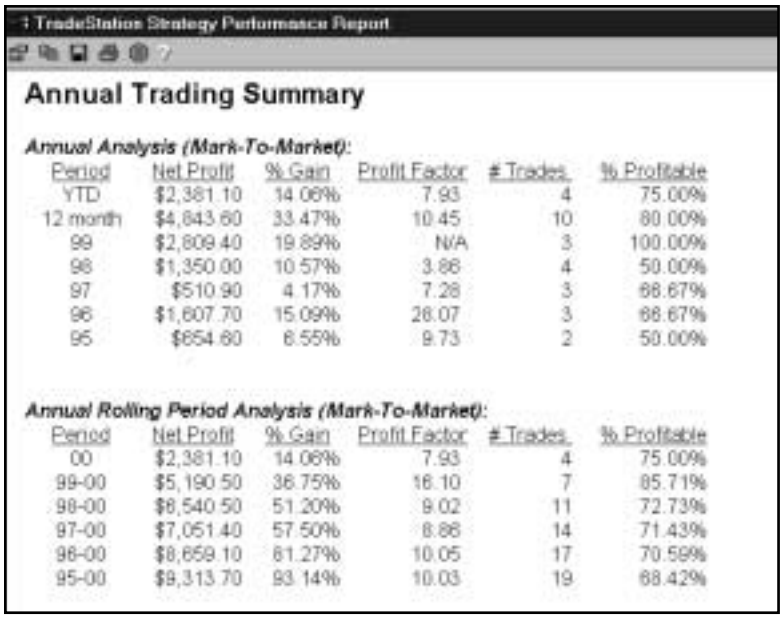
TradeStation Strategy Performance Report - STAD13: RangeLeaders INTC-Daily

Performance Summary: All Trades

Total Net Profit	$9,313.70	Open position P/L	$0.00
Gross Profit	$10,007.60	Gross Loss	($693.90)
Total # of trades	16	Percent profitable	62.50%
Number winning trades	10	Number losing trades	6
Largest winning trade	$2,125.00	Largest losing trade	($390.60)
Average winning trade	$1,000.76	Average losing trade	($115.65)
Ratio avg win/avg loss	8.65	Avg trade (win & loss)	$582.11
Max consec. Winners	3	Max consec. losers	2
Avg # bars in winners	60	Avg # bars in losers	19
Max intraday drawdown	($593.80)		
Profit Factor	14.42	Max # contracts held	100
Account size required	$338,093.81	Return on account	2.75%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win/Loss | Time | Graphs | Settings

Figure 2. INTC Performance Summary


TradeStation Strategy Performance Report

Annual Trading Summary

Annual Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
YTD	$2,381.10	14.06%	7.93	4	75.00%
12 month	$4,843.60	33.47%	10.45	10	80.00%
99	$2,809.40	19.89%	N/A	3	100.00%
98	$1,350.00	10.57%	3.86	4	50.00%
97	$510.90	4.17%	7.28	3	66.67%
96	$1,607.70	15.09%	26.07	3	66.67%
95	$654.60	6.55%	9.73	2	50.00%

Annual Rolling Period Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
00	$2,381.10	14.06%	7.93	4	75.00%
99-00	$5,190.50	36.75%	16.10	7	85.71%
98-00	$8,540.50	51.20%	9.02	11	72.73%
97-00	$7,051.40	57.50%	8.86	14	71.43%
96-00	$8,659.10	61.27%	10.05	17	70.59%
95-00	$9,313.70	93.14%	10.03	19	68.42%

Figure 3. INTC Annual Trading Summary

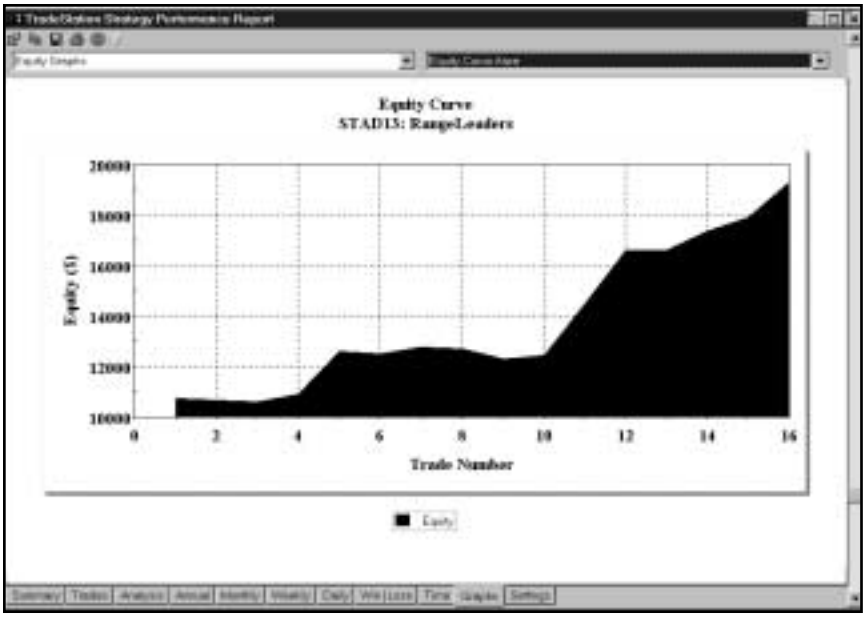


Figure 4. INTC Equity Curve

As usual, the Underwater Equity Curve looks intimidating, but the worst drawdown was only about eight percent [Figure 5, INTC Underwater Equity Curve]. The graph of Average Profit by Month illustrates the strategy's consistent performance as it posted profits in ten of twelve months when monthly returns were averaged over the length of the test period [Figure 6, INTC Average Profit by Month].

Figure 5. INTC Underwater Equity Curve

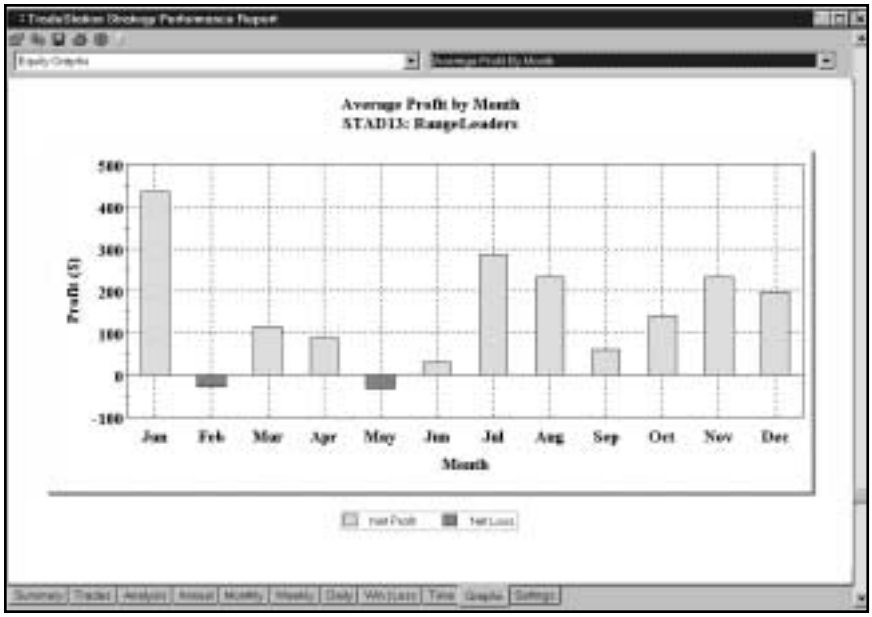


Figure 6. INTC Average Profit by Month

Next, let's turn our attention to the *DMARL*'s performance in Coffee futures. The optimized values are as follows:

StopLoss = $3,000

BreakevenFloor = $3,000

DllrRiskTrailing = $5,000

DMAvgLength = 50

Displacement = 10

In the Coffee market, our *DMARL* strategy produced net profits of $91,152 on 81 trades, with 52% of the trades profitable [Figure 7, KC Performance Summary]. The largest winning trade made $22,243, compared to the largest losing trade of $3,319. The average trade (counting both wins and losses) earned a healthy $1,125, and the strategy earned $2.11 for each $1.00 it lost.

The Equity Curve tells us that *DMARL* only broke even from trade 50 to trade 80 [Figure 8, KC Equity Curve]. At least the strategy didn't lose money while waiting for the next major trend to develop in this market. Figure 9 shows the impact a few sustained trends can have on a trendfollowing strategy [Figure 9, KC Total Trades]. The two filled-in circles represent positive outliers — trades more than three standard deviations greater than an average trade. Note that the two positive outliers were approximately 30 trades apart, and that it's been about 30 trades since the last outlier. It will be interesting to see how soon the next good trend develops.

Figure 7. KC Performance Summary

Figure 8. KC Equity Curve

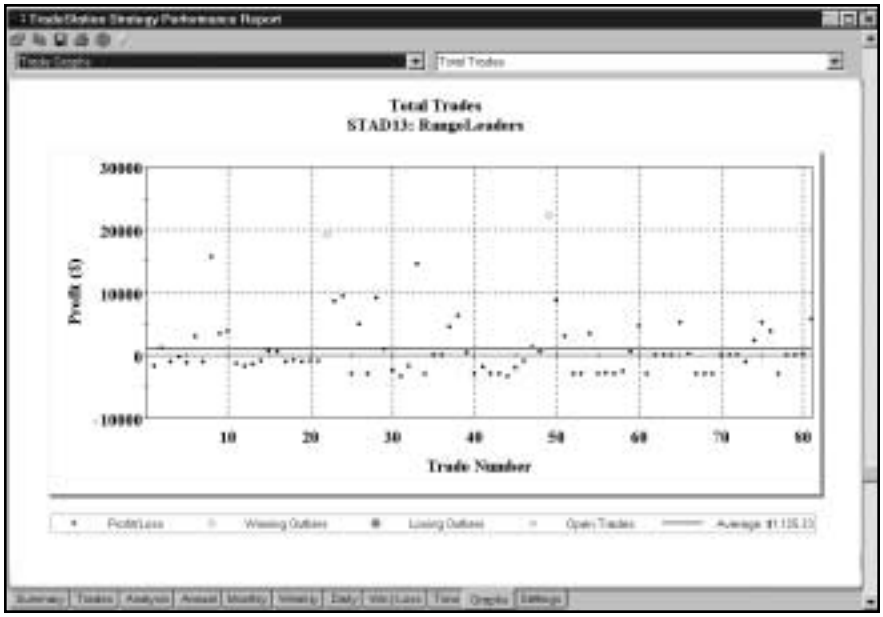


Figure 9. KC Total Trades

Summary

We like to use indicators in unusual ways--if our testing yields encouraging results — because we don't want to be doing the same thing the majority of other traders are doing in the markets. In the *DMARL* strategy, we calculated moving averages of highs and lows (instead of just closing prices), displaced the averages a number of bars into the future, and required the Range Leader pattern to complete the setup. This combination gave us a simple yet unusual strategy that generated significant profits in our historical testing.

CHAPTER 6

Linear Regression and Momentum

The Linear Regression indicator plots a line through the prices of a stock or commodity in an attempt to minimize the distance between the line and each individual point. The method used to accomplish this is called the "least squares" method. The indicator is based on the theory that prices are pulled back to the regression line after they stray above it or below it.

The Linear Regression indicator can also be used to monitor the current trend. If it's rising the trend is up, and if it's falling the trend is down. For our Linear Regression and Momentum strategy, we'll smooth the linear regression line with a moving average so that the indicator does not switch back and forth between uptrends and downtrends too often.

Momentum, the second indicator in this strategy, compares the current bar's closing price with the closing price a specified number of bars in the past. To calculate a 5-bar momentum line, for example, subtract the close of 5 bars ago from the current bar's close.

When the 5-bar Momentum indicator is above its zero line and rising, the 5-bar price changes are positive and increasing - that is, the trend is bullish and accelerating. If the momentum line turns flat, it implies that the 5-bar price changes are about equal during the Momentum indicator's period of sideways movement. When the Momentum indicator begins to decline from above zero, the market's gains during the past 5 bars are less than the corresponding gains in the preceding bars - that is, the uptrend is decelerating.

When the 5-bar Momentum indicator falls below its zero line, the current close is below the close 5 bars ago. As the downtrend gains bearish velocity, momentum accelerates downward from the zero line. An upturn of the indicator in negative territory means that the magnitude of 5-bar price declines is decreasing - that is, the downtrend is decelerating. Momentum is a leading indicator - it levels off while prices are still rising in an uptrend or falling in a downtrend, and it reverses its direction when the trend begins to slow.

In this strategy, we'll use momentum to identify countertrend declines in an uptrend and countertrend rallies in a downtrend. As mentioned previously, the trend will be determined by the direction of a smoothed linear regression line (i.e., a moving average of linear regression). Figure 1 shows a daily chart of Cisco Systems (CSCO) with a linear regression line and momentum indicator applied [Figure 1, CSCO Chart].

Figure 1. CSCO Chart

For a buy setup, we'll require the smoothed linear regression line to be rising and for momentum to be below zero but rising. For a sell setup, we'll require the smoothed linear regression line to be falling and for momentum to be above zero but falling.

After a buy setup, we'll add 50% of the ten-bar average true range to the high of the setup bar to determine our long entry price. After a sell setup, we'll subtract 50% of the ten-bar average true range from the low of the setup bar to determine our short entry price.

After we've entered a position, we'll manage our trade with a series of ATR (Average True Range) stops from StrategyBuilder: the ATR Protective Stop, ATR Breakeven Stop, ATR Trailing Stop, ATR Volatility Stop, and ATR Big Profit Stop. This is an extremely effective group of stops, which we use over and over with different setups and entries.

Defining Our Trading Rules

In this strategy, we defined long and short setups, entries, and stops. We also did some setup work to calculate the linear regression line, its smoothed average, and the momentum indicator. The setup, entries, and stops are described next:

Long and Short Setups

a) For a buy setup, we'll require the smoothed linear regression line to be rising and for the close to be above the linear regression line. Also, momentum must be below zero but rising (greater on this bar than on the previous bar).

b) For a sell setup, we'll require the smoothed linear regression line to be falling and for the close to be below the linear regression line. Also, momentum must be greater than zero but falling (less on this bar than on the previous bar).

Long and Short Entries

a) After a buy setup, we'll add 50% of the ten-bar Average True Range to the high of the setup bar to determine our long entry price. The order will remain active for four bars.

b) After a sell setup, we'll subtract 50% of the ten-bar Average True Range from the low of the setup bar to determine our short entry price. The order will remain active for four bars.

Long and Short Exits

Once we've entered a position, we'll implement our series of ATR Stops. Our exit for each bar will be at the closest of the ATR protective stop, breakeven stop, trailing stop, volatility stop, or big profit stop.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Linear Regression and Momentum (STAD13: LinReg - Mom)

Strategy Inputs (STAD13: LinReg - Mom)

INPUT	DEFAULT	DESCRIPTION
BreakevenATRs	3	The number of average true ranges above/below the EntryPrice at which the breakeven stop becomes active
ATRLength	10	The length parameter used to calculate the average true range for the ATR stops
Price	Close	The price value used to base calculations for the average linear regression line and for momentum
Regression	60	The length parameter used to calculate the linear regression line
XMALength	30	The length parameter used to calculate the average of the linear regression line
MomentumLength	20	The length parameter used to calculate momentum
EntryPercent	.5	The factor applied to the average true range to determine protective stop values
EntryCounter	3	The maximum number of bars after an entry setup that will generate entry orders
VolatilityATRs	3	The number of average true ranges that are used to determine the required volatility to place an exit order
TrailingATRs	5	The number of average true ranges that are risked from the highest/lowest price of the position
ProtectiveATRs	3	The number of average true ranges used to determine protective stop values
BigProfitATRs	8	Number of average true ranges used to determine the “Big Profit” level
ExitBarLength	4	The length parameter used to determine the trailing stop after the “Big Profit” level has been achieved

Strategy Components:

1. Linear Reg Momentum
2. ATR Big Profit Stop
3. ATR Trailing Stop
4. ATR Volatility Stop
5. ATR Breakeven Stop
6. ATR Protective Stop
7. Last Bar Exit

EasyLanguage Signal: Linear Reg Momentum:

```
Inputs:
        Price(Close), Regression(20), XMALength(15), MomentumLength(10), EntryPercent(.25),
        EntryCounter(4);
Variables:
        XLRAverage(0), Moment(0), BuySetup(False), SellSetup(False), BuyPrice(0), SellPrice(0),
        LongCounter(0), ShortCounter(0);

XLRAverage = XAverage(LinearRegValue(Price, Regression, 0), XMALength);
Moment = Momentum(Price, MomentumLength);
ATR = AvgTrueRange(10);

LongCounter = LongCounter + 1;
ShortCounter = ShortCounter + 1;

BuySetup = XLRAverage > XLRAverage[1] AND Close > XLRAverage AND Moment < 0 AND
        Moment > Moment[1];
SellSetup = XLRAverage < XLRAverage[1] AND Close < XLRAverage AND Moment > 0 AND
        Moment < Moment[1];

If BuySetup Then Begin
        BuyPrice = High + ( EntryPercent * ATR );
        LongCounter = 1;
End;

If SellSetup Then Begin
        SellPrice = Low - ( EntryPercent * ATR );
        ShortCounter = 1;
End;


If LongCounter <= EntryCounter Then
        Buy next bar at BuyPrice Stop;
If ShortCounter <= EntryCounter Then
        Sell next bar at SellPrice Stop;
```

Signal Inputs (Linear Reg Momentum)

INPUT	DEFAULT	DESCRIPTION
Price	Close	The price value used to base calculations for the average linear regression line and for momentum
Regression	60	The length parameter used to calculate the linear regression line
XMALength	30	The length parameter used to calculate the average of the linear regression line
MomentumLength	20	The length parameter used to calculate momentum
EntryPercent	.5	The factor applied to the average true range to determine protective stop values
EntryCounter	3	The maximum number of bars after an entry setup that will generate entry orders

Signal Variables (Linear Reg Momentum)

INPUT	DEFAULT	DESCRIPTION
XLRAverage	0	[Numeric] Used to store the exponential average of the linear regression line
Moment	0	[Numeric] Used to store the momentum of the price data
BuySetup	False	[True/False] Used to evaluate the condition of a buy setup
SellSetup	False	[True/False] Used to evaluate the condition of a sell setup
BuyPrice	0	[Numeric] Used to calculate the price of entry for a Long position
SellPrice	0	[Numeric] Used to calculate the price of entry for a Short position
LongCounter	0	[Numeric] Used to count the number of bars after a Buy setup to limit the time allowed for entry
ShortCounter	0	[Numeric] Used to count the number of bars after a Sell setup to limit the time allowed for entry

Setup

To determine a setup, the exponential average of the linear regression line must be determined. Here, the calculation of the linear regression line is nested in the function XAverage(). Doing this allows the storage of the calculated value on one line, storing the value in the variable XLRAverage.

```
XLRAverage = XAverage(LinearRegValue(Price, Regression, 0), XMALength);
```

The momentum of the price data can be determined using the Momentum() function. This is stored in the variable Moment.

```
Moment = Momentum(Price, MomentumLength);
```

The average true range of the last ten bars is calculated and stored in the variable ATR. This allows a future calculation of the entry price.

```
ATR = AvgTrueRange(10);
```

For the purposes of limiting the number of bars after a setup that entry orders will be generated, the counter variables LongCounter and ShortCounter need to be incremented on every bar. This is done with the following two statements.

```
LongCounter = LongCounter + 1;
ShortCounter = ShortCounter + 1;
```

The setup for a Long position is based on a rising smoothed linear regression line, a rising momentum and the close of the current bar being above the smoothed linear regression line. To determine that a value is rising, it can be compared to its value one bar ago. The entire evaluation is combined with AND, forcing all three conditions to be True in order to make the BuySetup variable True.

```
BuySetup = XLRAverage > XLRAverage[1] AND Close > XLRAverage AND Moment < 0 AND
        Moment > Moment[1];
```

In contrast, setup for a Short position is based on a falling smoothed linear regression line, a falling momentum and the close of the current bar being below the smoothed linear regression line. Again, values are compared to their value one bar ago. The entire evaluation is combined with AND, forcing all three conditions to be True in order to make the SellSetup variable True.

```
SellSetup = XLRAverage < XLRAverage[1] AND Close < XLRAverage AND Moment > 0 AND
        Moment < Moment[1];
```

The final portion of the setup is to determine the actual prices to be used for entry and to reset the counters used to limit the number of bars that a setup will generate an order. Because more than action is taken, the block statement If-Then-Begin is used to make all of the statements between the Begin and End dependent on the same condition.

```
If BuySetup Then Begin
```

The price of the Long entry is the High of the setup bar plus EntryPercent (Input) percent of the average true range. That value is calculated and stored in the variable BuyPrice. The LongCounter variable is reset to 1, allowing a fresh count for the generation of orders based on the new price.

```
        BuyPrice = High + ( EntryPercent * ATR );
        LongCounter = 1;
```

The word End is needed as a separate statement to determine the end of the block that is dependent on the condition “If BuySetup Then..”.

```
End;
```

For the Short side, another block statement is used to make all of the statements between the Begin and End dependent on the same condition.

```
If SellSetup Then Begin
```

The price of the Short entry is the Low of the setup bar minus EntryPercent percent of the average true range. That value is stored in the variable SellPrice. The ShortCounter variable is reset to 1, allowing a fresh count for the generation of orders based on the new price.

```
        SellPrice = Low - ( EntryPercent * ATR );
        ShortCounter = 1;
```

The word End is needed as a separate statement to determine the end of the block that is dependent on the condition “If SellSetup Then..”.

```
End;
```

Long and Short Entries

The Long entry is only dependent on the variable LongCounter being equal to or less than EntryCounter (Input). If this is the case, a Buy order is generated for the next bar on a stop at the price stored in the variable BuyPrice.

```
If LongCounter <= EntryCounter Then
        Buy next bar at BuyPrice Stop;
```

The Short entry is only dependent on the variable ShortCounter being equal to or less than EntryCounter. If this is the case, a Sell order is generated for the next bar on a stop at the price stored in the variable SellPrice.

```
If ShortCounter <= EntryCounter Then
        Sell next bar at SellPrice Stop;
```

EasyLanguage Signal: ATR Big Profit Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Breakeven Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Protective Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Trailing Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Volatility Stop:

** See Common Stops Appendix

EasyLanguage Signal: Last Bar Exit:

** See Common Stops Appendix

Testing and Improving

We tested the *Linear Regression and Momentum* strategy on daily data for Cisco Systems (CSCO, long side only) from 6/92 to 4/00 and Japanese Yen futures (JY) from 1/95 to 4/00, with the Max number of bars strategy will reference set to 100. The default values and test protocol were as follows:

Regression = 60, testing from 40 -80 in increments of 10

XMALength = 30, testing from 20-40 in increments of 10

MomentumLength = 20, testing from 10 to 30 in increments of 10

ProtectiveATRs = 3, testing from 2-4 in increments of 1

BreakevenATRs = 3, testing from 2-4 in increments of 1

TrailingATRs = 5, testing from 4-6 in increments of 1

VolatilityATRs = 3, testing from 2-4 in increments of 1

BigProfitATRs = 8, testing from 6-10 in increments of 1

ExitBarLength = 4, testing from 3-5 in increments of 1

EntryPercent = .50, testing from .25 to 1.0 in increments of .25

EntryCounter = 3, testing from 1-5 in increments of 1

Note that we optimized the 11 inputs in closely related groups of two or three rather than in one great frenzy of number crunching. The first group comprises the indicators, the second group the basic ATR stops, the third group the more esoteric ATR stops, and the fourth group the two minor factors. This approach to optimization yields good results in much less time than it would require to optimize all 11 inputs at once.

Let's see how our *Linear Regression and Momentum* (*LRM*) strategy performed in CSCO. The optimized values are as follows:

Regression = 80

XMALength = 30

MomentumLength = 10

ProtectiveATRs = 3

BreakevenATRs = 3

TrailingATRs = 5

VolatilityATRs = 4

BigProfitATRs = 9

ExitBarLength = 3

EntryPercent = .50

EntryCounter = 5

Applied to CSCO, the *LRM* strategy produced profits of $2,458 (per 100 shares) on 13 trades [Figure 2, CSCO Performance Summary]. Seventy-seven percent of the trades were profitable, while the average winner ($287) was 2.09 times the amount of the average loser ($138). The Profit Factor was outstanding — *LRM* won $6.96 for each $1.00 it lost.

The Equity Curve paints a familiar picture: a relatively flat line until the stock began to trend well and then a generous accumulation of profits during the trending period [Figure 3, CSCO Equity Curve]. Ironically, the biggest drawdowns also occurred during the trending period because of the stock's greatly increased volatility recently [Figure 4, CSCO Underwater Equity Curve]. Still, the worst drawdown — about three percent — isn't bad at all.

Next, let's examine *LRM*'s performance on Japanese Yen futures. Figure 5 displays two substantial winning trades in the Yen [Figure 5, JY Chart]. *LRM* garnered $67,778 in net profits on 26 trades [Figure 6, JY Performance Summary]. Sixty-five percent of the trades were profitable, and the average winner ($5,070) was 2.48 times as much as the average loser ($2,047). The largest winning trade ($25,320) far surpassed the largest losing trade ($3,851), and the average trade (wins and losses) earned $2,607. *LRM*'s longest winning streak reached six, compared to its longest losing streak of only two. The Profit Factor of 4.68 means that *LRM* earned $4.68 for each $1.00 it lost.

Figure 2. CSCO Performance Summary

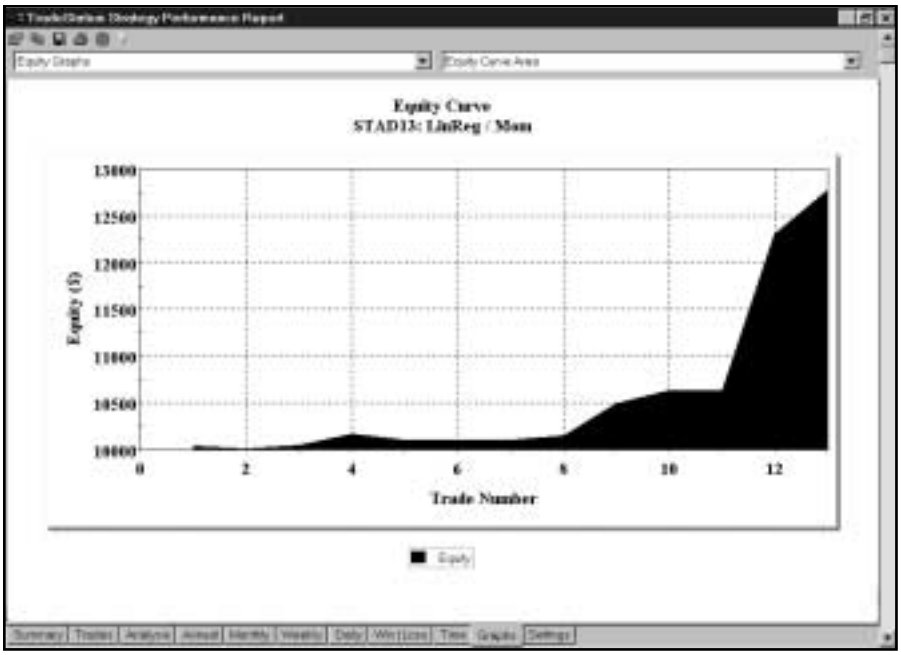


Figure 3. CSCO Equity Curve

Unlike the typical Equity Curve we've seen in stocks-a long period of relatively flat performance until a recent string of big gains — *LRM*'s Equity Curve in the Japanese Yen shows relatively slow but sure progress over the test period [Figure 7, JY Equity Curve]. Also diverging from the typical Underwater Equity Curve we've seen in stocks, the Underwater Equity Curve for *LRM* applied to the Yen depicts the worst drawdowns (22% and 16%) near the beginning of the test period and very modest drawdowns (5 to 7%) more recently [Figure 8, JY Underwater Equity Curve].

Figure 4. CSCO Underwater Equity Curve

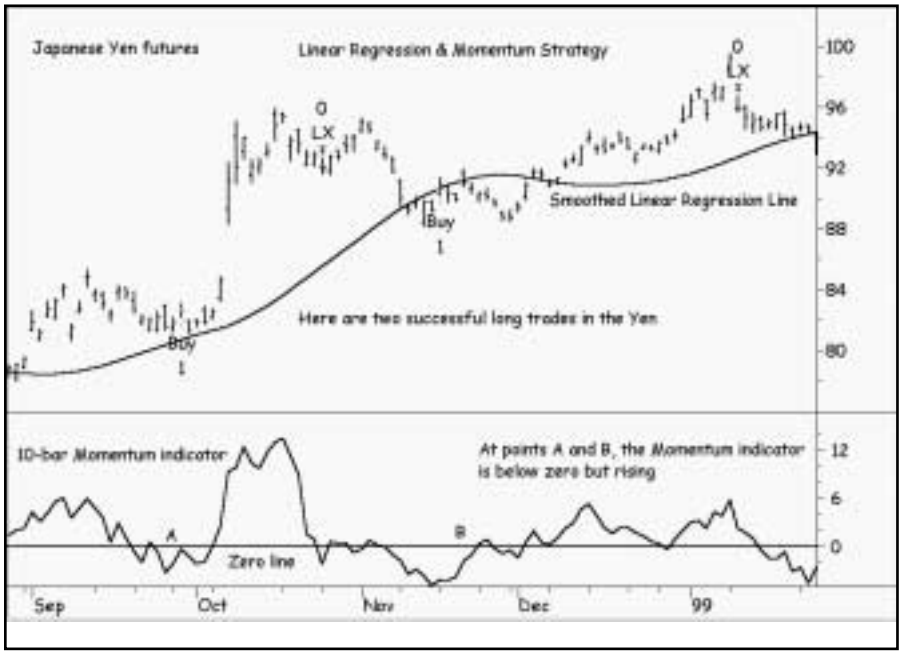


Figure 5. JY Chart

Figure 6. JY Performance Summary

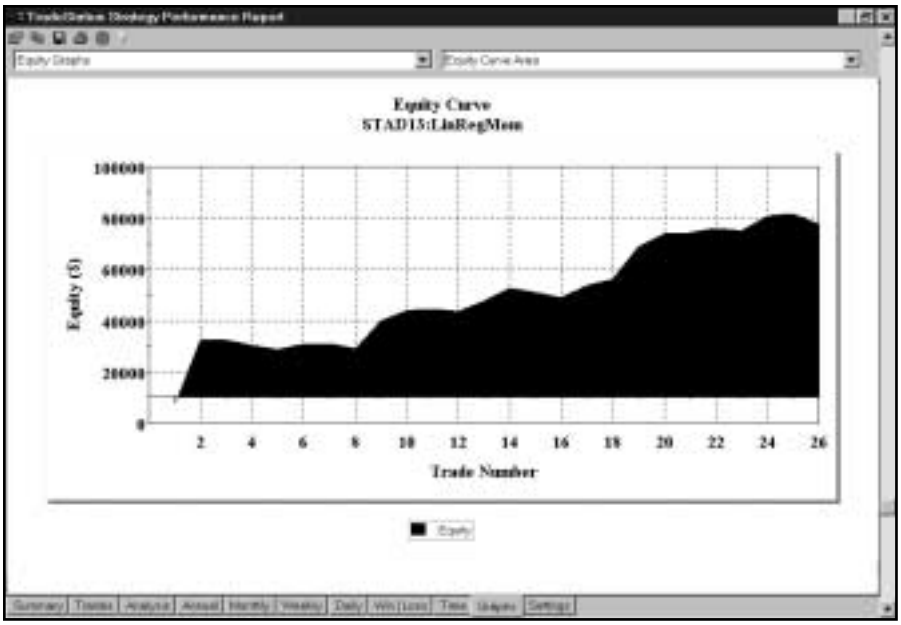


Figure 7. JY Equity Curve

The Annual Trading Summary indicates consistent gains, with *LRM* trading profitably in all five years of the test period [Figure 9, JY Annual Trading Summary]. Our strategy was also a steady performer on the graph of Average Profit by Month: only one month (April) suffered a loss when monthly returns were averaged over the five-year test period [Figure 10, JY Annual Profit by Month].

Figure 8. JY Underwater Equity Curve

TradeStation Strategy Performance Report

Annual Trading Summary

Annual Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
YTD	($5,691.25)	(6.82%)	0.00	2	0.00%
12 month	$3,296.25	4.43%	1.35	11	45.45%
99	$8,407.50	11.20%	4.83	6	66.67%
98	$21,968.75	41.38%	6.42	6	66.67%
97	$20,246.25	61.64%	23.40	7	85.71%
96	$2,528.75	8.34%	1.66	5	60.00%
95	$20,317.50	203.18%	5.06	4	50.00%

Annual Rolling Period Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
00	($5,691.25)	(6.82%)	0.00	2	0.00%
99-00	$2,716.25	3.62%	1.34	8	50.00%
98-00	$24,685.00	46.49%	3.07	14	57.14%
97-00	$44,931.25	136.79%	4.50	21	66.67%
96-00	$47,460.00	156.54%	3.85	26	65.38%
95-00	$67,777.50	677.78%	4.13	30	63.33%

Figure 9. JY Annual Trading Summary

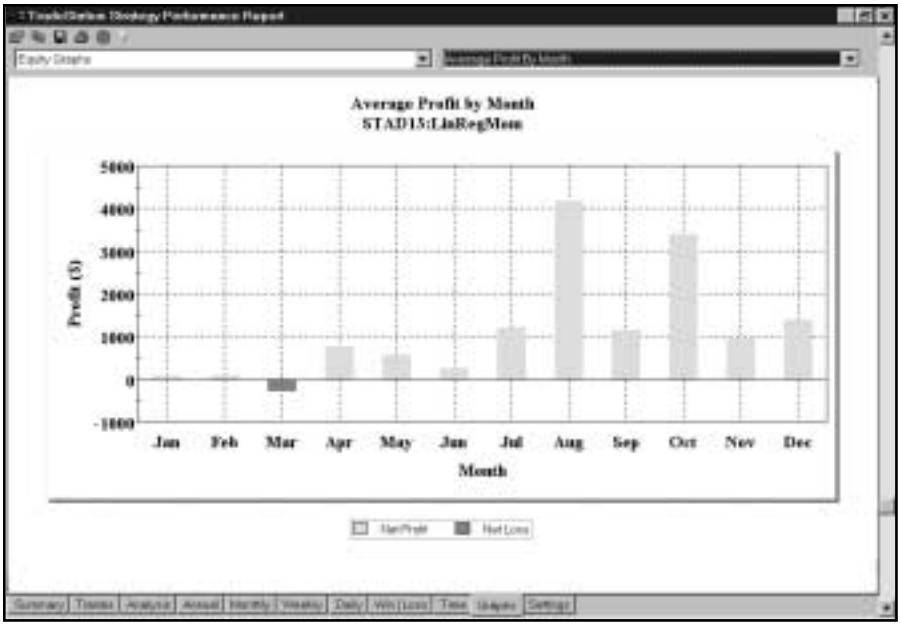


Figure 10. JY Annual Profit by Month

Summary

We like the *Linear Regression and Momentum* strategy for three main reasons:

1. Buying declines in an uptrend and selling rallies in a downtrend is one of our favorite ways to enter trades
2. Applying our series of five ATR stops is one of our most effective ways to manage trades
3. The strategy performed well in both stocks and futures, indicating its robustness and versatility.

CHAPTER 7

LUXOR

Our *Luxor* strategy employs the technique of pyramiding to increase profits in a strongly trending market. Since a typical market (stock or commodity) is in a trending mode only about 15 to 25 percent of the time, traders need to maximize returns during trends.

Pyramiding (adding shares or contracts to a position after the initial position is established) is the most common method traders use to take full advantage of trending moves.

Luxor identifies setups for new trades by the crossing of two moving averages — a fast one and a slow one. Of course, there are many types of moving averages; *Luxor,* which was introduced in *STAD Club 9,* is the first strategy in STAD Club to use Triangular Moving Averages.

The purpose of the Triangular Moving Average (TMA) is to increase the smoothing of the price data without also increasing the lag time between prices and the indicator. TMAs begin with the calculation of a simple arithmetic average of prices (the close is the price field most commonly averaged). Then, the TMA indicator calculates a simple arithmetic average of the first average.

The length of each average is equal to one more than half the value specified as the input length.

A 20-bar TMA, for example, first calculates an 11-bar simple arithmetic average; then, it calculates an 11-bar average of the first average. The resulting average of the average is usually plotted as a line in the same subgraph as the price data.

After the two TMAs are calculated, we wait for the fast average (a 5-bar average, for example) to cross above the slow average (a 20-bar average, for example) for a buy setup or for the fast average to cross below the slow average for a sell setup. The setup is in effect until the fast average crosses the slow average in the opposite direction. In the case of a setup to buy, we enter a long position at the high of the setup bar plus one point; after a setup to sell, we enter a short position at the low of the setup bar minus one point. Our initial and trailing stops are set at the slow TMA minus one point for a long position and at the slow TMA plus one point for a short position.

The pyramiding feature of our *Luxor* trading strategy uses the fast TMA and the ADX indicator (Average Directional Index) to identify pyramiding opportunities. The fast TMA tells us when a market has retraced a little so that we can add contracts or shares at a more favorable price; the ADX tells us when a market is in a strong trend that can be exploited by adding contracts or shares.

Here are *Luxor*'s rules for pyramiding: When in a long position, if the high of the current bar is less than the fast TMA, and ADX is rising, then buy at the fast TMA plus one point. To qualify as rising, ADX must be greater than it was on the bar the same number of bars ago as the length of the fast TMA. For example, if we're using a length of five bars for the fast TMA, the ADX must be greater than it was five bars ago.

When in a short position, if the low of the current bar is greater than the fast TMA, and ADX is rising, then sell at the fast TMA minus one point. If we're using a five-bar fast TMA, ADX must be greater than it was five bars ago to qualify as rising. The default value for the maximum number of pyramid entries is three per trade.

Stops for the positions that were added with our pyramiding strategy are identical to the stops for our initial positions. In an uptrend, the stops are set one point below the slow TMA; in a downtrend, the stops are set one point above the slow TMA.

Figure 1 is a daily bar chart of NXTL with the Fast and Slow Triangular Moving Averages, the ADX, an initial long trade and a pyramiding long trade [Figure 1, NXTL Chart].

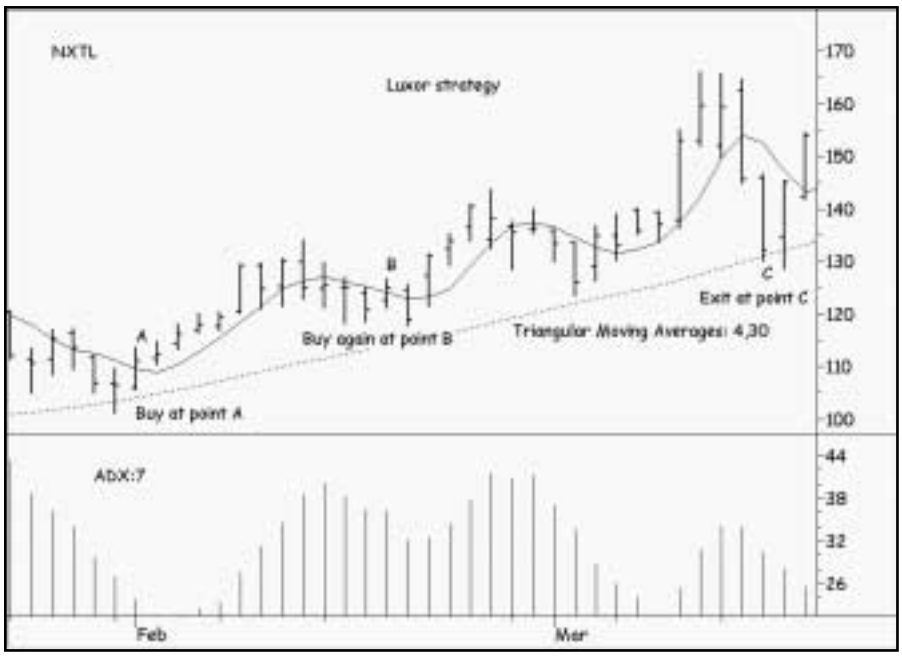


Figure 1. NXTL Chart

Defining Our Trading Rules

In this strategy, we defined long and short setups, entries, pyramiding conditions, and exits. We also calculated the fast and slow Triangular Moving Averages and the ADX. The setups, entries, pyramiding conditions and exits are described next.

Long and Short Setups

a) The setup for a long position is the fast TMA crossing above the slow TMA. The long setup remains in effect while the fast TMA is above the slow TMA.

b) The setup for a short position is the fast TMA crossing below the slow TMA. The short setup remains in effect while the fast TMA is below the slow TMA.

c) The long setup for a pyramid position is that the high of the current bar is below the fast TMA, and ADX is greater than it was n-bars ago.

d) The short setup for a pyramid position is that the low of the current bar is above the fast TMA, and ADX is greater than it was n-bars ago.

Long and Short Entries

a) The long entry is one point above the high of the setup bar; the long pyramid entry is one point above the fast TMA. The default value for the maximum number of long pyramid entries is 3.

b) The short entry is one point below the low of the setup bar; the short pyramid entry is one point below the fast TMA. The default value for the maximum number of short pyramid entries is 3.

Long and Short Exits

a) The initial and trailing stops for long positions are one point below the slow TMA.

b) The initial and trailing stops for short positions are one point above the slow TMA.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: LUXOR (STAD13: LUXOR)

Strategy Inputs (STAD13: LUXOR)

INPUT	DEFAULT	DESCRIPTION
FastLength	5	Used as the length parameter to calculate the fast moving triple average
SlowLength	30	Used as the length parameter to calculate the slow moving triple average
ADXLength	15	Used as the length parameter of the ADX

Strategy Components:

1. *LUXOR*

2. Last Bar Exit

EasyLanguage Signal: LUXOR:

```
Inputs:
        FastLength(5), SlowLength(30), ADXLength(12);
Variables:
        MP(0), Fast(0), Slow(0), GoLong(False), GoShort(False), BuyStop(0), SellStop(0),
        ADXValue(0), Trending(False);

MP = MarketPosition;

Fast = TriAverage(Close, FastLength);
Slow = TriAverage(Close, SlowLength);

ADXValue = ADX(ADXLength);
Trending = ADXValue > ADXValue[FastLength];

GoLong = Fast > Slow;
GoShort = Fast < Slow;

If Fast crosses above Slow Then
        BuyStop = High + 1 point;
If Fast crosses below Slow Then
        SellStop = Low - 1 point;

If GoLong AND MP = 0 Then
```

```
        Buy ("Long") next bar at BuyStop Stop;
If GoShort AND MP = 0 Then
        Sell ("Short") next bar at SellStop Stop;

If MP = 1 Then Begin
        ExitLong next bar at Slow - 1 point Stop;
        If High < Fast AND Trending Then
                Buy ("Longer") next bar at Fast + 1 point Stop;
End;

If MP = -1 Then Begin
        ExitShort next bar at Slow + 1 point Stop;
        If Low > Fast AND Trending Then
                Sell ("Shorter") next bar at Fast - 1 point Stop;
End;
```

Signal Inputs (LUXOR)

INPUT	DEFAULT	DESCRIPTION
FastLength	5	Used as the length parameter to calculate the fast moving triple average
SlowLength	20	Used as the length parameter to calculate the slow moving triple average
ADXLength	12	Used as the length parameter of the ADX

Signal Variables (LUXOR)

INPUT	DEFAULT	DESCRIPTION
MP	0	[Numeric] Used to store market position on a bar by bar basis
Fast	0	[Numeric] Used to store the calculation of the fast moving triple average
Slow	0	[Numeric] Used to store the calculation of the slow moving triple average
GoLong	False	[True/False] Used to determine that the fast moving average is greater than the slow
GoShort	False	[True/False] Used to determine that the fast moving average is below the slow
BuyStop	0	[Numeric] Used to store the entry point for a Long position
SellStop	0	[Numeric] Used to store the entry point for a Short position
ADXValue	0	[Numeric] Used to store the value of the ADX calculation
Trending	False	[True/False] Used to determine that the strength of the trend is increasing

Setup

MarketPosition is stored into the variable MP in order to allow references on a bar-by-bar basis. MarketPosition will return 1 for a Long position, -1 for a Short position and 0 for no position. When used alone the reserved word returns the current position and when using a parameter (ex. MarketPosition(n)) will return the position n number of bars ago.

```
MP = MarketPosition;
```

Using FastLength and SlowLength (Inputs), the fast and slow moving Triple Averages are calculated and stored in the variables Fast and Slow, respectively.

```
Fast = TriAverage(Close, FastLength);
Slow = TriAverage(Close, SlowLength);
```

The ADXValue is calculated using the ADX() function. The length parameter used is ADXLength (Input). The determination of trending is when the ADX is greater than it was FastLength bars ago. This determination is made using the bars ago notation within square brackets ([ ]). The result is stored in the True/False variable Trending.

```
ADXValue = ADX(ADXLength);
Trending = ADXValue > ADXValue[FastLength];
```

The strategy rules that determine that a Long position is favorable are when the Fast average is greater than the Slow average, vice-versa for a Short position. These evaluations are made and stored in the True/False variables GoLong and GoShort, respectively.

```
GoLong = Fast > Slow;
GoShort = Fast < Slow;
```

The final part of the setup occurs by determining the initial entry price, determined at the point of crossover of the two averages. For a Long position, the price is determined as the High of the bar of cross over plus one point. For a Short position, the Low of the bar of cross over minus one point. The Long and Short entry evaluations are stored in BuyStop and SellStop, respectively.

```
If Fast crosses above Slow Then
        BuyStop = High + 1 point;
If Fast crosses below Slow Then
        SellStop = Low - 1 point;
```

Long Entries and Exit

As long as the Fast average is above the Slow average (determined by GoLong) and there is no other open position, a Long entry order is placed for the next bar at the price of BuyStop on a Stop.

```
If GoLong AND MP = 0 Then
        Buy ("Long") next bar at BuyStop Stop;
```

Once a Long position has been established, an exit order is generated at the current value of the Slow moving average. This level will trail the position as the price rises and will exit the position before the Fast average can cross below the Slow. Other positions will be entered into (after the initial entry) if the price movement shows signs of strengthening. If the High of the current bar is greater than the Fast average and the Trending determination evaluates to True, a Long entry order is generated at one point above the Fast moving average.

```
If MP = 1 Then Begin
        ExitLong next bar at Slow - 1 point Stop;
        If High < Fast AND Trending Then
                Buy ("Longer") next bar at Fast + 1 point Stop;
End;
```

Short Entries and Exit

As long as the Fast average is below the Slow average (determined by GoShort) and there is no other open position, a Short entry order is placed for the next bar at the price of SellStop on a Stop.

```
If GoShort AND MP = 0 Then
        Sell ("Short") next bar at SellStop Stop;
```

Once a Short position is taken, an exit order is generated at the current value of the Slow moving average plus one point. Other entries will be made (after the initial entry) if the price movement shows signs of strengthening. If the Low of the current bar continues below the Fast average and the Trending determination evaluates to True, a Short entry order is generated at one point below the Fast moving average.

```
If MP = -1 Then Begin
        ExitShort next bar at Slow + 1 point Stop;
        If Low > Fast AND Trending Then
                Sell ("Shorter") next bar at Fast - 1 point Stop;
End;
```

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing and Improving

We tested *Luxor* on daily data for Nextel (NXTL) from 4/96 to 4/00 and US Treasury Bonds (US) from 1/95 to 4/00. We set the Max number of bars strategy will reference to 50 and did not make a deduction for slippage and commission. The default values and testing protocol were as follows:

FastLength = 5, testing from 3 to 7 in increments of 1

SlowLength = 30, testing from 20 to 40 in increments of 10

ADXLength = 15, testing from 7 to 21 in increments of 2

Let's find out how *Luxor* handled NXTL. The optimized values are as follows:

FastLength = 4

SlowLength = 30

ADXLength = 7

Our strategy reaped profits of $7,850 (per 100 shares) on 47 trades [Figure 2, NXTL Performance Summary]. Fifty-one percent of the trades were profitable, while the average winner ($502) was 2.75 as large as the average loser ($788). The strategy enjoyed a winning streak of five consecutive profitable trades and suffered a losing streak of only three consecutive unprofitable trades. *Luxor* let profits run for an average of 21 bars but cut losses extremely short in an average of only four bars. The Profit Factor was 2.87: the strategy earned $2.87 for each $1.00 it lost.


TradeStation Strategy Performance Report

TradeStation Strategy Performance Report - STAD13: LUXOR NXTL-Daily

Performance Summary: All Trades

Total Net Profit	$7,850.00	Open position P/L	$0.00
Gross Profit	$12,037.40	Gross Loss	($4,187.40)
Total # of trades	47	Percent profitable	51.06%
Number winning trades	24	Number losing trades	23
Largest winning trade	$2,400.00	Largest losing trade	($787.50)
Average winning trade	$501.56	Average losing trade	($182.06)
Ratio avg win/avg loss	2.75	Avg trade (win & loss)	$167.02
Max consec. Winners	5	Max consec. losers	3
Avg # bars in winners	21	Avg # bars in losers	4
Max intraday drawdown	($2,056.30)		
Profit Factor	2.87	Max # contracts held	300
Account size required	$1,009,800.00	Return on account	0.78%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win | Loss | Time | Graphs | Settings

Figure 2. NXTL Performance Summary

The Equity Curve shows only a modest profit of about $2,000 over the first 32 trades but a decent profit of more than $8,000 after trade 40 [Figure 3, NXTL Equity Curve].

Let's turn to our test results of *Luxor* applied to US Treasury Bonds. The optimized values are as follows:

FastLength = 6

SlowLength = 30

ADXLength = 21

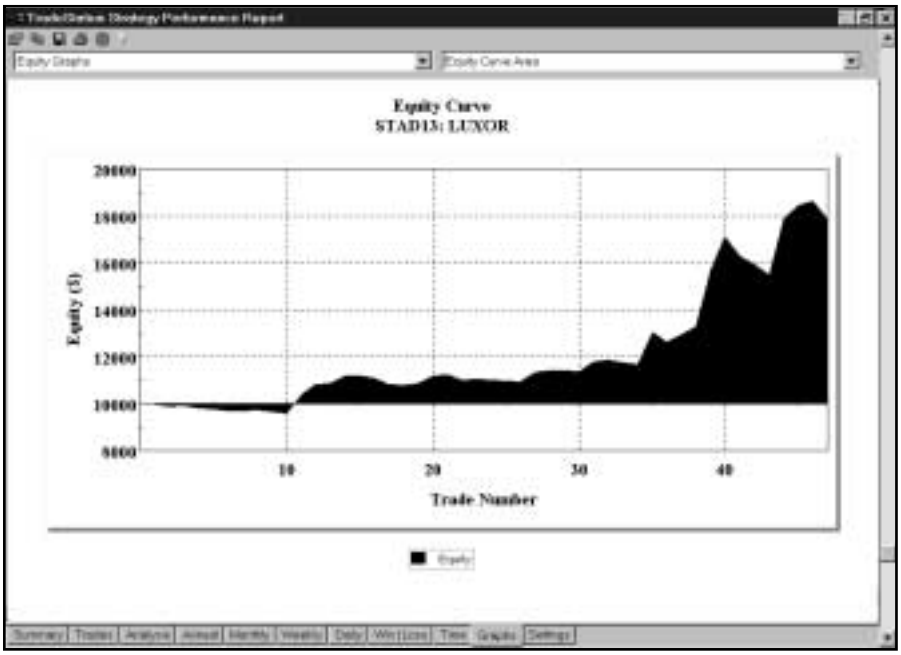


Figure 3. NXTL Equity Curve

Figure 4 is a daily bar chart of US with *Luxor*'s fast and slow Triangular Moving Averages, ADX, an initial short entry, two pyramiding entries, and a respectable exit [Figure 4, US Chart].

Figure 4. US Chart

Figure 5. US Performance

Trading US with the *Luxor* strategy earned a total net profit of $84,043 [Figure 5, US Performance Summary]. Of the 229 trades, 46% were profitable, and the average winner ($1,711) was 2.22 times the amount of the average loser ($771). The largest winning trade ($10,002) far eclipsed the largest losing trade ($2,252).

The Annual Trading Summary is impressive: *Luxor* traded the Bonds profitably in all ten years of the test period [Figure 6, US Annual Trading Summary]. The Equity Curve shows a few dips into negative territory during the first 35 trades but fairly steady growth over the rest of the series [Figure 7, US Equity Curve].

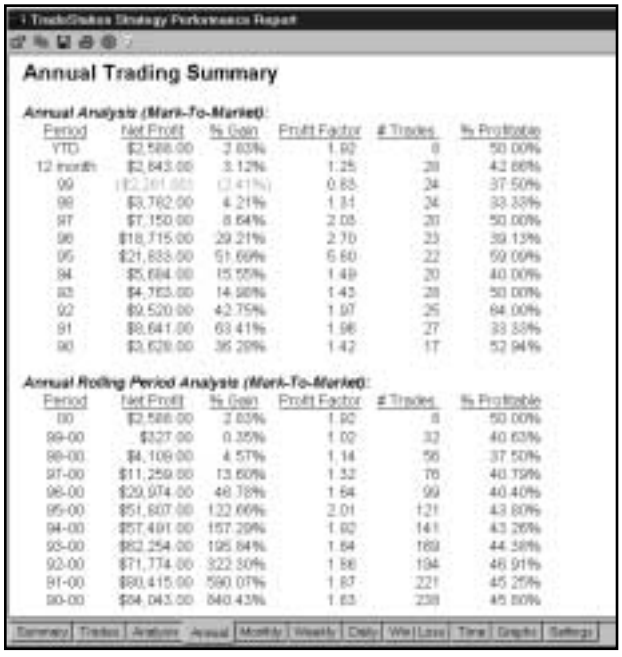


Figure 6. US Annual Trading Summary

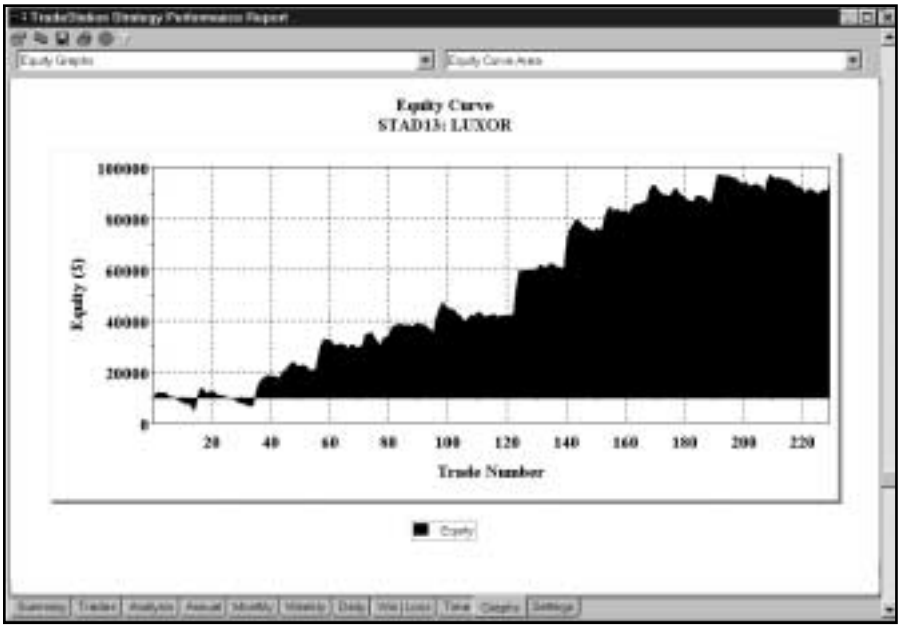


Figure 7. US Equity Curve

A trendfollowing strategy needs to hit at least a few homeruns to post acceptable profits, and *Luxor* knocked four balls out of the park [Figure 8, US Total Trades]. The four balls between trades 95 and 140 represent winning trades that were more than three standard deviations greater than the average winning trade.

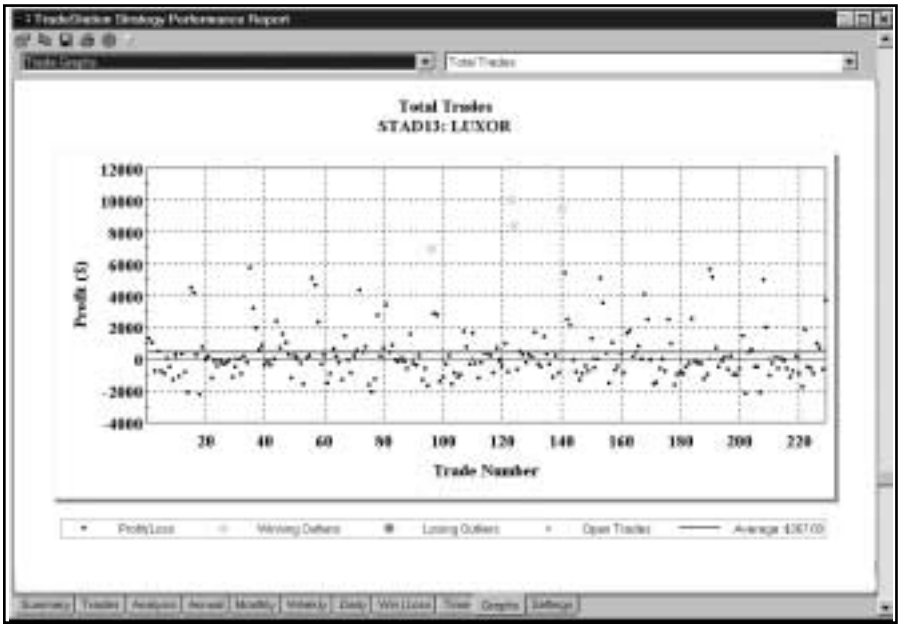


Figure 8. US Total Trades

Summary

The *Luxor* strategy tries to exploit trends by adding positions on small countertrend moves in strongly trending markets. Pyramiding can greatly enhance your trading profits, or it can put you out of the trading business if it's implemented recklessly. The most important caution is to only pyramid if all your current open entries are profitable; in other words, don't add to losers.

CHAPTER 8

Currency/Bonds/Dollar Index Strategy

The *Currency/Bonds/Dollar Index* Strategy (*CBI*) is based on the positive correlation between foreign currencies and US Treasury Bonds and the negative correlation (obviously) between foreign currencies and the US Dollar Index. When the Bond market is bullish, foreign currencies tend to do well against the US Dollar; when Bonds are bearish, foreign currencies usually are weak in comparison to the Dollar. When the US Dollar Index is strong, there's a broad-based weakness in most of the foreign currencies, and when the US Dollar Index is weak, there's strength across the board in most of the foreign currencies.

We use three exponential moving averages to determine the trend of the currency. Next, we use an exponential moving average of Bonds to confirm (or not to confirm) the trend of the currency. Finally, we use an exponential moving average of the Dollar Index to determine if the currencies in general are strong or weak relative to the US Dollar.

If the moving averages indicate an uptrend, we have a setup to buy. Our buy entry will be at the high of the setup bar plus 50% of the average true range. When the moving averages indicate a downtrend, we have a setup to sell short. Our sell entry will be at the low of the setup bar minus 50% of the average true range. The setup will remain in effect for five bars. Our signal to exit the position will be a crossover of the two moving averages calculated on the currency.

Figure 1 shows the strategy applied to a daily Swiss Franc chart accompanied by Bonds and the Dollar Index [Figure 1, Swiss Franc Chart].

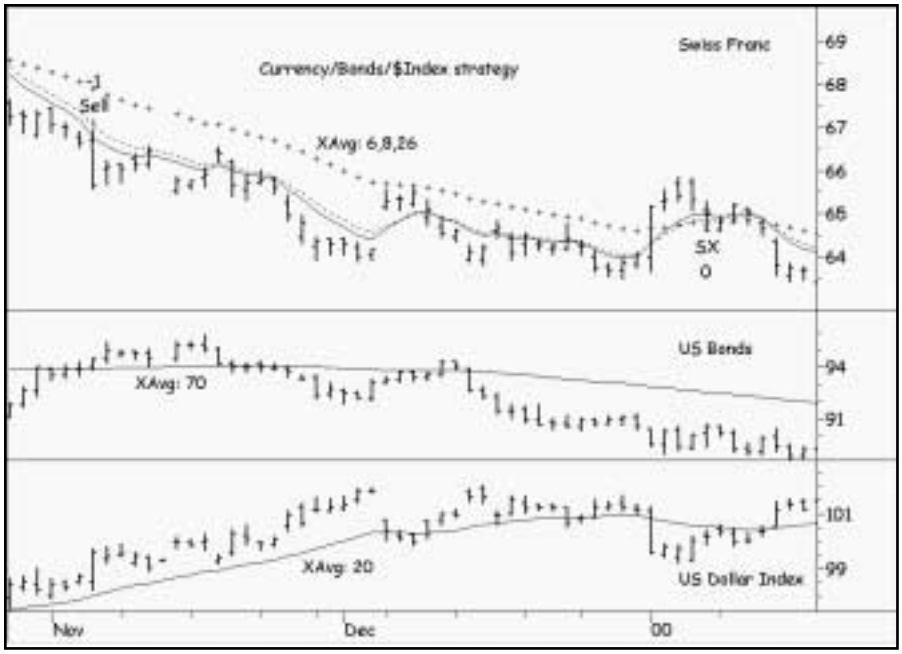


Figure 1. Swiss Franc Chart

Defining Our Trading Rules

In this strategy, we defined long and short setups, entries and exits. These components are described next.

Long and Short Setups

a) For a long position, check for all of the following:
- the four-bar exponential average to be greater than it was one bar ago
- the 12-bar exponential average to be above the 23-bar exponential average
- the 50-bar exponential average of Bonds to be greater than it was one bar ago
- the 50-bar exponential average of the Dollar Index to be less than it was one bar ago

b) For a short position, check for all of the following:
- the four-bar exponential average to be less than it was one bar ago
- the 12-bar exponential average to be below the 23-bar exponential average
- the 50-bar exponential average of Bonds to be less than it was one bar ago
- the 50-bar exponential average of the Dollar Index to be greater than it was one bar ago

Long and Short Entries

a) Once the trends of the currency and Bonds are up, and the trend of the Dollar Index is down, add 50% of the 10-bar ATR to the high of the setup bar. That's our buy point: we'll go long when the market reaches that price. The buy point will remain active for five bars.

b) Once the trends of the currency and Bonds are down, and the trend of the Dollar Index is up, subtract 50% of the 10-bar ATR from the low of the setup bar. That's our sell point: we'll go short when the market reaches that price. The sell point will remain active for five bars.

Long and Short Exits

a) We'll exit a long position on the next open when the 12-bar exponential average crosses below the 23-bar exponential average.

b) We'll exit a short position on the next open when the 12-bar exponential average crosses above the 23-bar exponential average.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Currency / Bonds / Dollar Index (STAD13: BondCurrency)

Strategy Inputs (STAD13: BondCurrency)

INPUT	DEFAULT	DESCRIPTION
FastLength	4	Used as the length parameter for the fast moving exponential average
MediumLength	12	Used as the length parameter for the medium moving exponential average
SlowLength	23	Used as the length parameter for the slow moving exponential average
BondsLength	50	Used as the length parameter for the exponential average of the Bonds (Data2)
IndexLength	50	Used as the length parameter for the exponential average of the Index (Data3)
ATRLength	10	Used as the length parameter for the average true range
BarsActive	5	Used to limit the number of bars that each occurrence of an entry setup will generate an order

Strategy Components:

1. Bond Currency

2. Last Bar Exit

EasyLanguage Signal: Bond Currency:

```
Input:
        FastLength(4), MediumLength(12), SlowLength(23), BondsLength(50), IndexLength(50),
        ATRLength(10), BarsActive(5);
Variables:
        Fast(0), Medium(0), Slow(0), Bonds(0), Index(0), ATRs(0), GetInLong(BarsActive),
        GetInShort(BarsActive), BuyPoint(0), SellPoint(0), Bullishness(False), Bearishness(False);

{Assignment of Exponential Average calculations}
Fast = XAverage(Close, FastLength);
Medium = XAverage(Close, MediumLength);
Slow = XAverage(Close, SlowLength);
Bonds = XAverage(Close, BondsLength) of Data2;
Index = XAverage(Close, IndexLength) of Data3;

ATRs = AvgTrueRange(ATRLength) / 2;
GetInLong = GetInLong + 1;
GetInShort = GetInShort + 1;

Bullishness = Bonds > Bonds[1] AND Index < Index[1];
Bearishness = Bonds < Bonds[1] AND Index > Index[1];

{Buy Setup}
If Fast > Fast[1] AND Medium > Slow AND Bullishness Then Begin
        BuyPoint = High + ATRs;
        GetInLong = 1;
End;

{Sell Setup}
If Fast < Fast[1] AND Medium < Slow AND Bearishness Then Begin
        SellPoint = Low - ATRs;
        GetInShort = 1;
End;

If GetInLong < BarsActive Then
        Buy next bar at BuyPoint Stop;
If GetInShort < BarsActive Then
        Sell next bar at SellPoint Stop;

{Exits}
If Medium crosses below Slow Then
        ExitLong next bar at market;
If Medium crosses above Slow Then
        ExitShort next bar at market;
```

Signal Inputs (Bond Currency)

INPUT	DEFAULT	DESCRIPTION
FastLength	4	Used as the length parameter for the fast moving exponential average
MediumLength	12	Used as the length parameter for the medium moving exponential average
SlowLength	23	Used as the length parameter for the slow moving exponential average
BondsLength	50	Used as the length parameter for the exponential average of the Bonds (Data2)
IndexLength	50	Used as the length parameter for the exponential average of the Index (Data3)
ATRLength	10	Used as the length parameter for the average true range
BarsActive	5	Used to limit the number of bars that each occurrence of an entry setup will generate an order

Signal Variables (Bond Currency)

INPUT	DEFAULT	DESCRIPTION
Fast	0	[Numeric] Used to store the fast moving exponential average
Medium	0	[Numeric] Used to store the medium moving exponential average
Slow	0	[Numeric] Used to store the slow moving exponential average
Bonds	0	[Numeric] Used to store the exponential average of the Bonds (Data2)
Index	0	[Numeric] Used to store the exponential average of the Dollar Index (Data3)
ATRs	0	[Numeric] Used to store one half of the average true range calculation
GetInLong	BarsActive	[Numeric] Used as a counter to determine what bars a long entry setup is good for
GetInShort	BarsActive	[Numeric] Used as a counter to determine what bars a short entry setup is good for
BuyPoint	0	[Numeric] Used to store the price for a long entry stop
SellPoint	0	[Numeric] Used to store the price for a short entry stop
Bullishness	False	[True/False] Used to store that the condition of the supporting data streams reflect a bullish situation for the primary data set
Bearishness	False	[True/False] Used to store that the condition of the supporting data streams reflect a bearish situation for the primary data set

Setup

The determination of trend is in part based on three exponential averages. Based on the Inputs FastLength, MediumLength and SlowLength, the Fast, Medium and Slow exponential moving averages are calculated using the function XAverage().

```
Fast = XAverage(Close, FastLength);
Medium = XAverage(Close, MediumLength);
Slow = XAverage(Close, SlowLength);
```

Additionally, other determinations of trend will be based on the behavior of two longer term exponential averages based on the U.S. Bonds (Data2) and the Dollar Index (Data3). These averages are also calculated with the function XAverage(). The calculations are tied to a specific data stream by adding "of Datax" immediately after the calculation, where x is the number referring the data stream.

```
Bonds = XAverage(Close, BondsLength) of Data2;
Index = XAverage(Close, IndexLength) of Data3;
```

The average true range is calculated by the AvgTrueRange() function and divided in half. The result is stored in the variable ATRs.

```
ATRs = AvgTrueRange(ATRLength) / 2;
```

GetInLong and GetInShort are used as counters. On every bar they are incremented by one.

```
GetInLong = GetInLong + 1;
GetInShort = GetInShort + 1;
```

A determination of a bullish condition is when the exponential average of the Bonds is rising and the exponential average of the Dollar Index is falling. The opposite combination is an indication of a bearish condition. The comparisons of rising and falling can be made by comparing the current value to its value one bar ago. The bracket notation of [1] represents the phrase "one bar ago". The results are stored in the True/False variables Bullishness and Bearishness.

```
Bullishness = Bonds > Bonds[1] AND Index < Index[1];
Bearishness = Bonds < Bonds[1] AND Index > Index[1];
```

The determination of a rising trend is based on the fast exponential average rising and the intermediate average greater than the slow average. Combined with a bullish outlook on Bonds and Index, the combination completes a Long setup and triggers the calculation of a price for Long entry and resets the counter used for generating a Long entry order. In order to integrate more than one statement to be based on the same condition, the Begin-End block statement is used.

```
If Fast > Fast[1] AND Medium > Slow AND Bullishness Then Begin
```

The two statements that are dependent is the calculation and storage of the Long entry price, calculated as the High of the current bar plus ATRs. This value is stored in the variable BuyPoint. The counter GetInLong is reset to one.

```
        BuyPoint = High + ATRs;
        GetInLong = 1;
```

The word End is included to cut off the statements that are dependent on the same trigger condition.

```
End;
```

On the Short side, the determination of a falling trend is based on a falling fast exponential average and the intermediate average lower than the slow average. If there is also an evaluation of Bearishness based on the U.S. Bonds and Dollar Index, then the setup for a Short entry is complete. Again, the statements dependent on a Short entry are blocked off using the Begin-End combination. SellPoint is used to store the value used as a Short entry stop, the Low of the setup bar minus ATRs and GetInShort is reset to one.

```
If Fast < Fast[1] AND Medium < Slow AND Bearishness Then Begin
        SellPoint = Low - ATRs;
        GetInShort = 1;
End;
```

Long Entry

As long as the variable GetInLong has not incremented to be greater than BarsActive (Input), a Buy order will be generated for the next bar at a value of BuyPoint on a Stop.

```
If GetInLong < BarsActive Then
        Buy next bar at BuyPoint Stop;
```

Short Entry

On the Short side, if the variable GetInShort has not incremented to be greater than BarsActive (Input), a Sell order will be generated for the next bar at a value of SellPoint on a Stop.

```
If GetInShort < BarsActive Then
        Sell next bar at SellPoint Stop;
```

Long and Short Exits

The exit criteria of the strategy is to exit a Long position when the intermediate moving average crosses below the slow average…

```
If Medium crosses below Slow Then
        ExitLong next bar at market;
```

and to exit a Short position when the intermediate average crosses above the slow average.

```
If Medium crosses above Slow Then
        ExitShort next bar at market;
```

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing and Improving

We tested the *CBI* strategy on daily bars of the Swiss Franc and the Japanese Yen from 1/2/95 to 4/4/00. We set the Max number of bars strategy will reference to 100 and didn't deduct for slippage and commission. The default values and testing protocol were as follows:

FastLength = 4, testing 2-6 in increments of 1

MediumLength = 12, testing 8-16 in increments of 2

SlowLength = 23, testing 17-29 in increments of 3

BondsLength = 50, testing 20-80 in increments of 10

IndexLength = 50, testing 20-80 in increments of 10

BarsActive = 5, testing 1-10 in increments of 1

Note that we optimized this strategy's inputs in related groups, instead of optimizing all the inputs at once. The first group is the length of the currency's averages, the second group is the length of the averages applied to Bonds and the Dollar Index, and the last is the number of bars to keep the setup active. This approach to optimizing our strategies yields good results in much less time than optimizing everything all at once.

Let's evaluate *CBI*'s performance on the Swiss Franc (SF). Here are the optimized values:

FastLength = 6

MediumLength = 8

SlowLength = 26

BondsLength = 70

IndexLength = 20

BarsActive = 8

Trading the Swiss Franc, the *CBI* strategy generated a net profit of $28,656 on 18 trades [Figure 2, SF Performance Summary]. Forty-four percent of the trades were profitable, while the ratio of the average winner ($5,601) to the average loser ($1,183) was a very respectable 4.28 to 1. The average trade, including both wins and losses, made $1,592 per contract. *CBI* won $3.42 for each $1.00 it lost in SF.


TradeStation Strategy Performance Report

TradeStation Strategy Performance Report - STAD13: Cur/Bnd/Indx SF,LNG-Daily

Performance Summary: All Trades

Total Net Profit	$28,656.25	Open position P/L	$0.00
Gross Profit	$40,490.00	Gross Loss	($11,833.75)
Total # of trades	18	Percent profitable	44.44%
Number winning trades	8	Number losing trades	10
Largest winning trade	$16,950.00	Largest losing trade	($2,237.50)
Average winning trade	$5,061.25	Average losing trade	($1,183.38)
Ratio avg win/avg loss	4.28	Avg trade (win & loss)	$1,592.01
Max consec. Winners	2	Max consec. losers	3
Avg # bars in winners	71	Avg # bars in losers	21
Max intraday drawdown	($8,146.25)		
Profit Factor	3.42	Max # contracts held	1
Account size required	$11,521.25	Return on account	248.73%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win | Loss | Time | Graphs | Settings

Figure 2. SF Performance Summary

Equity growth stalled between trades five and 14 but forged ahead nicely from trades 15 to 18 [Figure 3, SF Equity Curve]. *CBI* traded profitably in nine of the 12 months when monthly returns were averaged over the length of the test period [Figure 4, SF Average Profit by Month].

Next, let's examine *CBI*'s results in the Japanese Yen (JY). The optimized values are as follows:

FastLength = 3

MediumLength = 16

SlowLength = 29

BondsLength = 20

IndexLength = 30

BarsActive = 10

Figure 5 is a daily bar chart of JY, US, and DX with the *CBI* indicators and strategy [Figure 5, JY Chart]. Note the strategy's excellent entry in July, 1999, and its timely exit in January, 2000.

Trading the Yen, our *CBI* strategy netted a profit of $77,428 on 13 trades [Figure 6, JY Performance Summary].

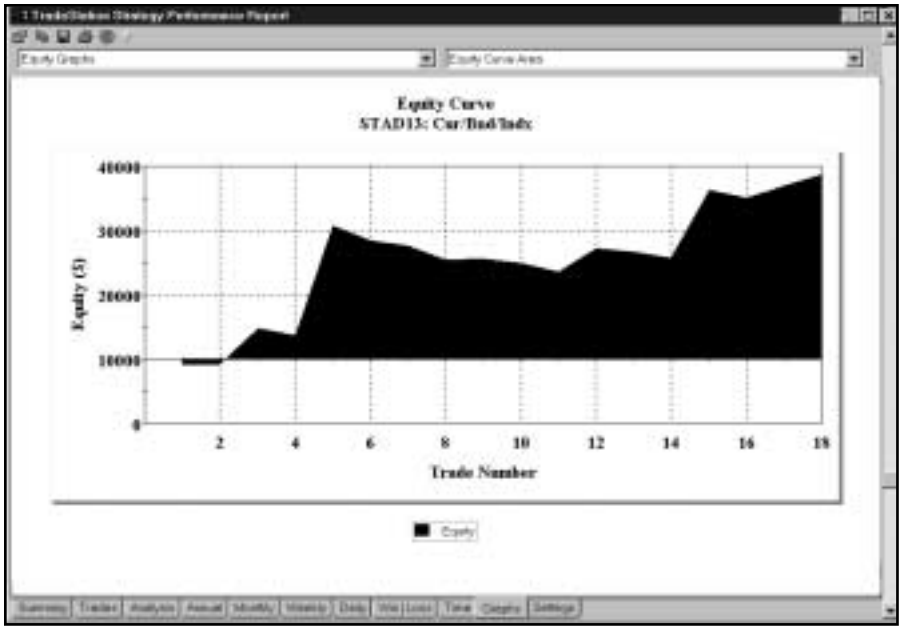


Figure 3. SF Equity Curve

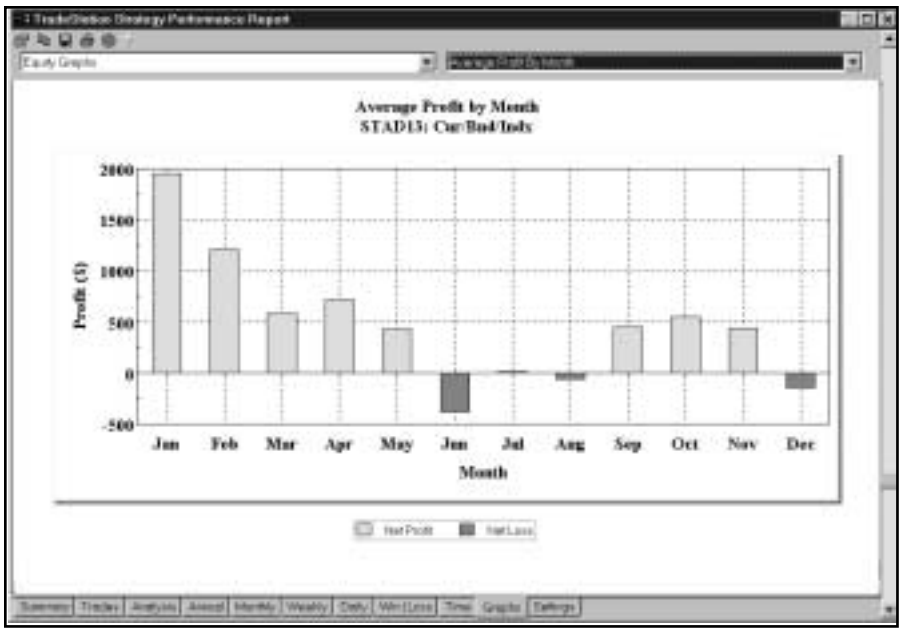


Figure 4. SF Average Profit by Month

An impressive 77% of the trades were winners-a very high percentage for a strategy producing an average winning trade that was 4.73 times the size of the average losing trade. The $27,088 largest winning trade dwarfed the $3,135 largest losing trade. *CBI* lets profits run for an average of 95 bars in winners, while cutting losses short in an average of 33 bars. Our strategy's average trade — counting both the wins and the losses — earned an impressive $5,956 per contract.

Figure 5. JY Chart

Figure 6. JY Performance Summary

The Equity Curve shows steady growth through the first seven trades and relatively flat performance from trades eight to 13 [Figure 7, JY Equity Curve]. The Underwater Equity Curve shows that the three largest drawdowns weren't bad at all: 9% in 1996, 10% in 1997, and 8% in 1999 [Figure 8, JY Underwater Equity Curve].

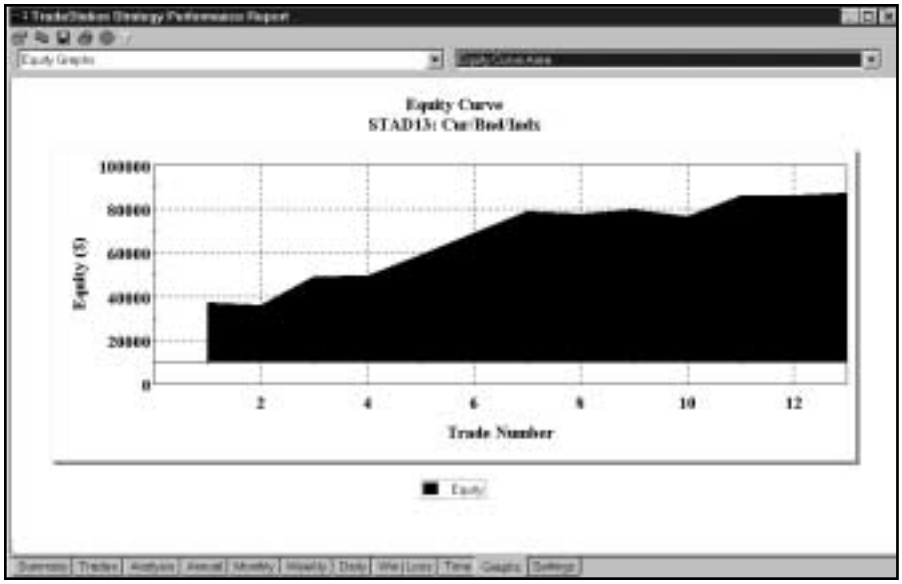


Figure 7. JY Equity Curve

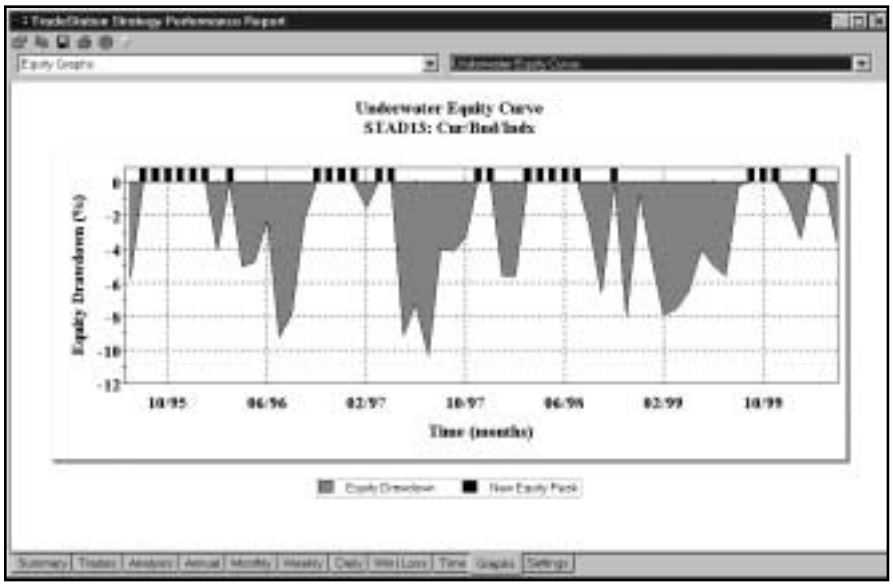


Figure 8. JY Underwater Equity Curve

Summary

Intermarket analysis can significantly improve trading results. Once you've designed a strategy that's working fairly well, you might want to try adding another market or two to the mix as we did in the *Currency/Bonds/Dollar Index* strategy.

CHAPTER 9

Momentum Retracement

The *Momentum-Retracement Trading Strategy* (*MRTS*) demonstrates a way of trading within an established trend. The strategy consists of seven components: 1) trend, 2) momentum, 3) directional movement, 4) retracements, 5) entries, 6) stops, and 7) exits.

The first step in applying *MRTS* is to determine the trend. We use a moving-average channel and the MACD indicator to accomplish this task.

The moving-average channel includes exponential moving averages of 15 highs, 15 lows, and five closes. The trend is up when the five-bar average of the closes is above the 15-bar average of the highs, or when the five-bar average of the closes was above the 15-bar average of the highs more recently than it was below the 15-bar average of the lows.

Conversely, trend is down when the five-bar average of the closes is below the 15-bar average of the lows, or when the five-bar average of the closes was below the 15-bar average of the lows more recently than it was above the 15-bar average of the highs.

The MACD indicator must confirm the trend identified by the moving-average channel. We use a 3-10-15 MACD. This means that the MACD line represents the difference between a three-bar and a 10-bar exponential moving average, and that the signal line of the indicator is a 15-bar exponential moving average of the MACD line. When the signal line is above zero, the trend is up, and when the signal line is below zero, the trend is down.

The second step is to evaluate the market's momentum. In this strategy, we use the Relative Strength Index (RSI). RSI compares the relative strength of price gains on bars that close above the previous bar's close to price losses on bars that close below the previous bar's close. A five-bar RSI rising to 70 or higher signifies strong bullish momentum and the RSI falling to 30 or lower indicates strong bearish momentum.

The third step is to measure the market's directional movement. In this strategy, we use the Directional Movement Index (DMI). DMI consists of two lines - the DMIPlus line and the DMIMinus line. These two lines represent the amount of consistent bullish "trendiness" and consistent bearish "trendiness" respectively. We construct an indicator called the DMI Spread by subtracting the DMIMinus line from the DMIPlus line. A DMI spread of +15 or higher indicates a persistent uptrend, and a DMI spread of -15 or lower indicates a persistent downtrend.

The fourth step is to identify a retracement. A retracement refers to a countertrend decline in an uptrend or a countertrend rally in a downtrend. In this strategy, we specify three conditions that must be met for a qualifying retracement. In an uptrend, the three conditions are: 1) Prices decline into the moving-average channel (in other words, the low of a price bar crosses below the exponential moving average of 15 highs), 2) MACD crosses below the signal line, and 3) RSI declines from above 70 to below 50 or declines by at least 30 RSI points.

In a downtrend, the three conditions are: 1) Prices rally into the moving-average channel (i.e., the high of a price bar crosses above the exponential moving average of 15 lows), 2) MACD crosses above the signal line, and 3) RSI rises from below 30 to above 50 or rises by at least 30 RSI points.

These three conditions for retracements do not have to occur on the same bar. The requirement is that all three conditions occur within 10 bars of the highest high (in an uptrend) or within 10 bars of the lowest low (in a downtrend).

The fifth step is to determine the entry price. After a qualified retracement in an uptrend, we enter a long position when prices rally above the high of the previous bar. After a qualified retracement in a downtrend, we enter a short position when prices fall below the previous bar's low.

The setup for a buy entry is cancelled if the moving average of five closes crosses below the moving average of 15 lows or if the signal line of the MACD crosses below zero. The setup to sell short is cancelled if the moving average of five closes crosses above the moving average of 15 highs or if the signal line of the MACD crosses above zero.

The sixth step is to determine our stops. In StrategyBuilder we add the following ATR (Average True Range) stops: the Protective stop, Breakeven stop, Trailing stop, Volatility stop, and Big Profit stop.

The seventh step is the exit. We exit from a long position on the next open when both the signal line of the MACD and the DMI spread fall below zero; we exit from a short position on the next open when both the MACD signal line and the DMI spread rally above zero.

Our *Momentum-Retracement Trading Strategy* is a fairly long and complex one. We believe, however, that the time you devote to studying this strategy will be time well spent. Figure 1 shows *MRTS* applied to a daily chart of Crude Oil [Figure 1, Crude Oil Chart].

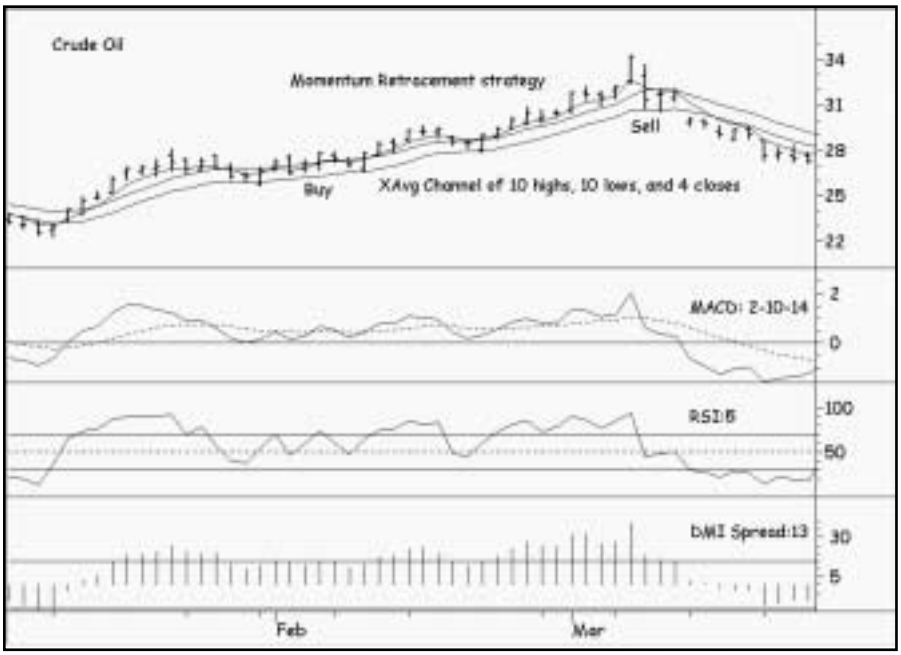


Figure 1. Crude Oil Chart

Defining Our Trading Rules

In this strategy, we defined long and short setups, entries, stops, and exits. These components are described next:

Long Setup

a) Determine the trend by comparing the moving averages. The trend is up when: the five-bar moving average of closes is above 15-bar moving average of highs, or when the five-bar moving average of closes has been above the 15-bar moving average of highs more recently than it was below the 15-bar moving average of lows.

b) Confirm the trend using the MACD indicator: when the signal line is above zero, the trend is up.

c) Evaluate the momentum of the market using the RSI indicator: RSI rising to 70 or higher signifies strong bullish momentum.

d) Determine the persistence of the trend using the DMI Spread. We construct the spread by subtracting the DMIMinus line from the DMIPlus line. A DMI Spread of +15 or higher indicates a persistent uptrend.

e) Check for a retracement. In an uptrend, the three conditions are:

- Prices decline into the moving-average channel (in other words, the low of a price bar crosses below the moving average of 15 highs)
- MACD crosses below the signal line
- RSI declines from above 70 to below 50 or declines by at least 30 RSI points.

The requirement is that all three conditions occur within 10 bars of the highest high of the current uptrend.

Short Setup

a) Determine the trend by comparing the moving averages. The trend is down when the five-bar moving average of closes is below the 15-bar moving average of lows or when the five-bar moving average of closes has been below the 15-bar moving average of lows more recently than it was above 15-bar moving average of highs.

b) Confirm the trend using the MACD indicator. When the signal line is below zero, the trend is down.

c) Evaluate the momentum of the market using the RSI indicator. RSI falling to 30 or lower indicates strong bearish momentum.

d) Determine the persistence of the trend using the DMI Spread. We construct the Spread by subtracting DMIMinus from DMIPlus. A DMI spread of -15 or lower indicates a persistent downtrend.

e) Check for a retracement. In a downtrend, the three conditions are:

- Prices rally into the moving-average channel (i.e., the high of a price bar crosses above the moving average of 15 lows)
- MACD crosses above the signal line
- RSI rises from below 30 to above 50 or rises by at least 30 RSI points.

The requirement is that all three conditions occur within 10 bars of the lowest low of the current downtrend.

Long and Short Entries

a) Enter a long position at one point above the previous bar's high.

b) Enter a short position at one point below the previous bar's low.

Long and Short Stops

In StrategyBuilder, we added the following ATR stops: the Protective stop, Breakeven stop, Trailing stop, Volatility stop, and Big Profit stop.

Long and Short Exits

a) Exit from a long position on the next open when both the signal line of the MACD and the DMI spread fall below zero.

b) Exit from a short position on the next open when both the signal line of the MACD and the DMI spread rally above zero.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Momentum Retracement (STAD13: Mom-Retrace)

Strategy Inputs (STAD13: Mom-Retrace)

INPUT	DEFAULT	DESCRIPTION
BreakevenATRs	4	The number of average true ranges used by the ATR Breakeven stop
ATRLength	10	Used as a length parameter for the stops based on the average true range
ChannelLength	15	Used as a length parameter for the exponential average of the High and Low prices
CloseLength	5	Used as a length parameter for the exponential average of the Close price
Fast	3	Used as the fast length parameter for the MACD calculation
Slow	10	Used as the slow length parameter for the MACD calculation
Signal	15	Used as a length parameter for the exponential average of the MACD line
RSILength	5	Used as a length parameter for the RSI calculation
OBZone	70	The overbought range for the RSI
OSZone	30	The oversold range for the RSI
DMILength	15	Used as a length parameter for the DMI calculation
TimetoRetrace	10	Used as the maximum number of bars to determine a counter trend retracement
RSIChange	30	A minimum change in the RSI value used to determine a counter trend retracement (optional)
BestEntry	2	Used as the length parameter to determine the best entry price for a Long or Short position
VolatilityATRs	3	The number of average true ranges that are used to determine the required volatility to place an exit order
TrailingATRs	6	The number of average true ranges that are risked from the highest/lowest price of the position
ProtectiveATRs	3	The number of average true ranges used to determine protective stop values
BigProfitATRs	9	Number of average true ranges used to determine the “Big Profit” level
BigProfitExitLength	4	The length parameter used to determine the trailing stop after the “Big Profit” level has been achieved

Strategy Components:

1. Momentum Retracement

2. ATR Big Profit Stop

3. ATR Breakeven Stop

4. ATR Protective Stop

5. ATR Trailing Stop

6. ATR Volatility Stop

7. Last Bar Exit

EasyLanguage Signal: Momentum Retracement:

```
Inputs:
        ChannelLength(15), CloseLength(5), Fast(3), Slow(10), Signal(15), RSILength(5),
        OBZone(70), OSZone(30), DMILength(15), TimetoRetrace(10), RSIChange(30),
        BestEntry(1);
Variables:
        HighBand(0), LowBand(0), CloseLine(0), MACDLine(0), SignalLine(0), RSIValue(0),
        ZeroLine(0), ChannelUptrend(False), ChannelDowntrend(False), MACDUptrend(False),
        MACDDowntrend(False), BullishMomentum(False), BearishMomentum(False), DMISpread(0),
        Uptrend(False), Downtrend(False), UptrendHigh(0), DowntrendLow(0),
        CheckRetracement(False), CounterTrendDecline(False), CounterTrendRally(False),
        HighCounter(0), LowCounter(0), HighRSIValue(0), LowRSIValue(0), LowofDecline(0),
        HighofRally(0), BuyStop(0), SellStop(0), MP(0), ATR(0);

HighBand = XAverage(High, ChannelLength);
LowBand = XAverage(Low, ChannelLength);
CloseLine = XAverage(Close, CloseLength);
MACDLine = MACD(Close, Fast, Slow);
SignalLine = XAverage(MACDLine, Signal);
RSIValue = RSI(Close, RSILength);
DMISpread = DMIPlus(DMILength) - DMIMinus(DMILength);
ATR = AvgTrueRange(10);

MP = MarketPosition;

If MP <> 0 Then Begin
        CheckRetracement = False;
        If MP = 1 Then
                CounterTrendDecline = False
        Else
                CounterTrendRally = False;
End;

If CloseLine crosses above HighBand Then Begin
```

```
        ChannelUptrend = True;
        ChannelDowntrend = False;
End;

If CloseLine crosses below LowBand Then Begin
        ChannelDowntrend = True;
        ChannelUptrend = False;
End;

MACDUptrend = SignalLine > ZeroLine;
MACDDowntrend = SignalLine < ZeroLine;

Uptrend = ChannelUptrend AND MACDUptrend;
Downtrend = ChannelDowntrend AND MACDDowntrend;
BullishMomentum = RSIValue > OBZone AND DMISpread > 15;
BearishMomentum = RSIValue < OSZone AND DMISpread < -15;

If Uptrend Then Begin
        If Uptrend[1] = False Then Begin
                UptrendHigh = High;
                HighCounter = 0;
                HighRSIValue = RSIValue;
                CheckRetracement = False;
                CounterTrendDecline = False;
        End
        Else Begin
                If High > UptrendHigh[1] Then Begin
                        UptrendHigh = High;
                        HighCounter = 0;
                End
                Else If HighCounter < TimetoRetrace Then
                        HighCounter = HighCounter + 1;
                If RSIValue > HighRSIValue Then
                        HighRSIValue = RSIValue;
        End;
        If BullishMomentum Then
                CheckRetracement = True;
        If CheckRetracement Then Begin
                If HighCounter <> 0 AND HighCounter < TimetoRetrace Then Begin
                        Condition1 = MRO(Low < HighBand, HighCounter, 1) <> -1;
                        Condition2 = MRO(MACDLine crosses below SignalLine, HighCounter, 1) <> -1;
                        Condition3 = MRO(RSIValue crosses below 50 OR
                                RSIValue crosses below HighRSIValue - RSIChange, HighCounter, 1) <> -1;
                        If Condition1 AND Condition2 AND Condition3 Then Begin
                                CounterTrendDecline = True;
                                BuyStop = Highest(High, BestEntry);
                                Condition1 = False;
                                Condition2 = False;
                                Condition3 = False;
                                CheckRetracement = False;
```

```
                End;
            End;
        End;
        If CounterTrendDecline Then
            Buy next bar at BuyStop Stop;
End;

If Downtrend Then Begin
        If Downtrend[1] = False Then Begin
            DowntrendLow = Low;
            LowCounter = 0;
            LowRSIValue = RSIValue;
            CheckRetracement = False;
            CounterTrendRally = False;
        End
        Else Begin
            If Low < DowntrendLow[1] Then Begin
                DowntrendLow = Low;
                LowCounter = 0;
            End
            Else If LowCounter < TimetoRetrace Then
                LowCounter = LowCounter + 1;
            If RSIValue < LowRSIValue Then
                LowRSIValue = RSIValue;
        End;
        If BearishMomentum Then
            CheckRetracement = True;
        If CheckRetracement Then Begin
            If LowCounter <> 0 AND LowCounter < TimetoRetrace Then Begin
                Condition1 = MRO(High > LowBand, LowCounter, 1) <> -1;
                Condition2 = MRO(MACDLine crosses above SignalLine, LowCounter, 1) <> -1;
                Condition3 = MRO(RSIValue crosses above 50 OR
                        RSIValue crosses above LowRSIValue + RSIChange, LowCounter, 1) <> -1;
                If Condition1 AND Condition2 AND Condition3 Then Begin
                    CounterTrendRally = True;
                    SellStop = Lowest(Low, BestEntry);
                    Condition1 = False;
                    Condition2 = False;
                    Condition3 = False;
                    CheckRetracement = False;
                End;
            End;
        End;
        If CounterTrendRally Then
            Sell next bar at SellStop Stop;
End;

If MACDDowntrend AND DMISpread < 0 Then
        ExitLong ("Below 0") next bar at market;

If MACDUptrend AND DMISpread > 0 Then
        ExitShort ("Above 0") next bar at market;
```

Signal Inputs (Momentum Retracement)

INPUT	DEFAULT	DESCRIPTION
ChannelLength	15	Used as a length parameter for the exponential average of the High and Low prices
CloseLength	5	Used as a length parameter for the exponential average of the Close price
Fast	3	Used as the fast length parameter for the MACD calculation
Slow	10	Used as the slow length parameter for the MACD calculation
Signal	15	Used as a length parameter for the exponential average of the MACD line
RSILength	5	Used as a length parameter for the RSI calculation
OBZone	70	The overbought range for the RSI
OSZone	30	The oversold range for the RSI
DMILength	15	Used as a length parameter for the DMI calculation
TimetoRetrace	10	Used as the maximum number of bars to determine a counter trend retracement
RSIChange	30	A minimum change in the RSI value used to determine a counter trend retracement (optional)
BestEntry	1	Used as the length parameter to determine the best entry price for a Long or Short position
TrailingATRs	1	The number of ATRs used to determine a possible trailing stop

Signal Variables (Momentum Retracement)

INPUT	DEFAULT	DESCRIPTION
HighBand	0	[Numeric] Used to store the exponential average of High prices
LowBand	0	[Numeric] Used to store the exponential average of Low prices
CloseLine	0	[Numeric] Used to store the exponential average of Close prices
MACDLine	0	[Numeric] Used to store the calculation of the MACD line
SignalLine	0	[Numeric] Used to store the exponential average of the MACD line, or the MACD Signal line
RSIValue	0	[Numeric] Used to store the calculation of the Relative Strength Index
ZeroLine	0	[Numeric] Used to store the zero line
ChannelUptrend	False	[True/False] Used to evaluate the condition of an uptrend based on the exponential average channel
ChannelDowntrend	False	[True/False] Used to evaluate the condition of a downtrend based on the exponential average channel
MACDUptrend	False	[True/False] Used to evaluate the condition of an uptrend based on the MACD Signal line
MACDDowntrend	False	[True/False] Used to evaluate the condition of a downtrend based on the MACD Signal line
BullishMomentum	False	[True/False] Used to determine that the price behavior exhibits a bullish momentum
BearishMomentum	False	[True/False] Used to determine that the price behavior exhibits a bearish momentum
DMISpread	0	[Numeric] Used to store the calculation of the DMI spread
Uptrend	False	[True/False] Used to determine an overall uptrend
Downtrend	False	[True/False] Used to determine an overall downtrend
UptrendHigh	0	[Numeric] Stores the highest High of the current Uptrend
DowntrendLow	0	[Numeric] Stores the lowest Low of the current Downtrend
CheckRetracment	False	[True/False] Used to determine that the conditions have been met to check for a counter-trend retracement
CounterTrendDecline	False	[True/False] Used to determine that a counter-trend decline has occurred
CounterTrendRally	False	[True/False] Used to determine that a counter-trend rally has occurred
HighCounter	0	[Numeric] Used to keep track of the number of bars since the highest High of the current Uptrend
LowCounter	0	[Numeric] Used to keep track of the number of bars since the lowest Low of the current Downtrend
HighRSIValue	0	[Numeric] Stores the highest RSIValue of the current Uptrend
LowRSIValue	0	[Numeric] Stores the lowest RSIValue of the current Downtrend
LowofDecline	0	[Numeric] Stores the lowest value of the counter-trend decline
HighofRally	0	[Numeric] Stores the highest value of the counter-trend rally
BuyStop	0	[Numeric] Used to store the value used for a Long entry stop
SellStop	0	[Numeric] Used to store the value used for a Short entry stop
MP	0	[Numeric] Used to store MarketPosition
ATR	0	[Numeric] Used to store the average true range

Setup

The Momentum Retracement Signal uses a channel based on the exponential averages of High and Low prices. A faster moving exponential average of Close prices is used to determine when the price behavior exhibits an up or downtrend. The values of the exponential averages are calculated and stored in the HighBand and LowBand, using ChannelLength (Input) as the length parameter. The exponential average of Close prices is stored in CloseLine and uses CloseLength (Input) as the length parameter.

```
HighBand = XAverage(High, ChannelLength);
LowBand = XAverage(Low, ChannelLength);
CloseLine = XAverage(Close, CloseLength);
```

In order to duplicate the calculations of the MACD indicator, two variables are calculated. First, MACDLine is generated using Fast and Slow (Inputs) and then SignalLine is calculated as an exponential average of the MACDLine.

```
MACDLine = MACD(Close, Fast, Slow);
SignalLine = XAverage(MACDLine, Signal);
```

To determine the RSIValue, the RSI() function is used and RSILength (Input) is used as the length parameter.

```
RSIValue = RSI(Close, RSILength);
```

DMISpread is calculated as the DMI+ value minus the DMI- value. DMILength (Input) is used as the length parameter for both functions.

```
DMISpread = DMIPlus(DMILength) - DMIMinus(DMILength);
```

The average true range is calculated and stored in the variable ATR. This will be used in generating a possible exit price for the “Trailer” exit.

```
ATR = AvgTrueRange(10);
```

MarketPosition is stored into the variable MP in order to allow references on a bar-by-bar basis. MarketPosition will return 1 for a Long position, -1 for a Short position and 0 for no position. When used alone the reserved word returns the current position and when using a parameter (ex. MarketPosition(n)) will return the position n number of bars ago.

```
MP = MarketPosition;
```

If the strategy enters into a position, MP will reflect a value other than zero. If this is the case there are a number of True/False variables that require a reset to False.

```
If MP <> 0 Then Begin
```

First, CheckRetracement needs to be reset.

```
CheckRetracement = False;
```

Then, depending on whether the position is Long or Short, CounterTrendDecline or CounterTrendRally is reset. Notice that because this If-Then-Else statement is nested under the condition "MP <> 0", it can be known that if MP is currently not equal to 1, reflecting a Long position, it must be set to -1, reflecting a Short position.

```
If MP = 1 Then
        CounterTrendDecline = False
Else
        CounterTrendRally = False;
```

The word End is used to block off the statements that reset the True-False variables.

```
End;
```

The determination of an uptrend based on the activity of the exponential average channel is if the last time CloseLine crossed out from within the channel it went above HighBand. If this is the case, ChannelUptrend is set to True and ChannelDowntrend is set to False. Because both statements are dependent on the same condition, they are "blocked" together by Begin-End.

```
If CloseLine crosses above HighBand Then Begin
        ChannelUptrend = True;
        ChannelDowntrend = False;
End;
```

Similarly, the determination of a downtrend is based on if the last time CloseLine crossed out from within the channel it went below LowBand. If this is the case, ChannelDowntrend is set to True and ChannelUptrend is set to False. Because both statements are dependent on the same condition, they are "blocked" together by Begin-End.

```
If CloseLine crosses below LowBand Then Begin
        ChannelDowntrend = True;
        ChannelUptrend = False;
End;
```

There is also a separate determination of up or downtrend based on the MACD Signal line. If SignalLine is greater than zero, then the MACD calculations indicate an uptrend. If SignalLine is less than zero, MACD calculations indicate a downtrend. These evaluations are made and stored in the True/False variables MACDUptrend and MACDDowntrend.

```
MACDUptrend = SignalLine > ZeroLine;
MACDDowntrend = SignalLine < ZeroLine;
```

The overall determination of Uptrend and Downtrend is made when both the channel and MACD determinations are the same. This can be done by using the relational operator AND, with the results stored in the variables Uptrend and Downtrend, respectively.

```
Uptrend = ChannelUptrend AND MACDUptrend;
Downtrend = ChannelDowntrend AND MACDDowntrend;
```

Bullish and Bearish momentum are determined by the RSI Oscillator and the strength of the DMI spread. If RSIValue is greater than the determined OBZone (Input) and the DMISpread is greater than 15, BullishMomentum exists. If RSIValue is less than the determined OSZone (Input) and the DMISpread is less than -15, BearishMomentum exists.

```
BullishMomentum = RSIValue > OBZone AND DMISpread > 15;
BearishMomentum = RSIValue < OSZone AND DMISpread < -15;
```

Long Entry

The Long entry for this strategy is dependent on the existence of an overall Uptrend. All of the statements involved in the Long entry are contained within the Begin-End block, which is dependent on the status of the True/False variable Uptrend.

```
If Uptrend Then Begin
```

If this is the first bar of a new Uptrend, the value of Uptrend on the previous bar will be False. If this is the case, the strategy calls for keeping track of the High of the Uptrend and the highest RSIValue in the Uptrend. In addition, two True/False variables need to be reset to False to prevent erroneous order generation. Nested in the Begin-End block of statements is the storing of the High of the first bar of the Uptrend into UptrendHigh. Subsequent entry conditions require that they occur within ‘X’ number of bars from the High of the Uptrend, so a variable called HighCounter is reset to zero to keep track of the number of bars that elapse from the new UptrendHigh. The current RSIValue is stored into HighRSIValue and CheckRetracement and CounterTrendDecline are reset to False. Because this block is actually the first portion of an If-Then-Else statement, there is no semi-colon (;) after the reserved word End at the end of the block.

```
    If Uptrend[1] = False Then Begin
        UptrendHigh = High;
        HighCounter = 0;
        HighRSIValue = RSIValue;
        CheckRetracement = False;
        CounterTrendDecline = False;
    End
```

The Else portion of the If-Then-Else statement is executed on any continuing bar of an Uptrend. Again, there is more than one action to take, causing a use of the Begin-End block.

```
    Else Begin
```

In order to keep track of UptrendHigh, if the current High is greater than UptrendHigh of the previous bar, then UptrendHigh is reset to the value of the current High and the HighCounter is reset to zero. If this is not the case, there is an Else portion of this If-Then statement that will increment the HighCounter variable by one if it is less than the limit of TimetoRetrace (Input).

```
If High > UptrendHigh[1] Then Begin
        UptrendHigh = High;
        HighCounter = 0;
End
Else If HighCounter < TimetoRetrace Then
        HighCounter = HighCounter + 1;
```

The second statement that is executed on all consecutive bars of an Uptrend is a test of the RSIValue. If the current RSIValue is greater than HighRSIValue, then HighRSIValue is reset to the current RSIValue. The End after this statement contains a semi-colon, bringing a close to the larger If-Then-Else statement that begins with "Uptrend[1] = False".

```
        If RSIValue > HighRSIValue Then
                HighRSIValue = RSIValue;
End;
```

If BullishMomentum occurs during an Uptrend, it sets the trigger of CheckRetracement to True. This is done via an If-Then statement.

```
If BullishMomentum Then
        CheckRetracement = True;
```

A number of the following statements are dependent on the CheckRetracement variable being set to True.

```
If CheckRetracement Then Begin
```

If the HighCounter is not currently zero and the HighCounter is less than TimetoRetrace, the current bar is within TimetoRetrace bars of the bar that contains the High of the Uptrend. This allows the proper testing and evaluation of the conditions that should occur during a retracement.

```
        If HighCounter <> 0 AND HighCounter < TimetoRetrace Then Begin
```

The three conditions made are stored in the predefined True/False variables Condition1, 2 and 3. Using the MRO() function, a condition is evaluated for the occurrence of the Low breaking below HighBand, the MACDLine crossing below SignalLine and either the RSIValue crossing below 50 or dropping 30 points from its high point of the Uptrend. The MRO() function will return the number of bars ago a condition occurred if it occurred within the specified length, or a -1 if it did not occur. By using HighCounter as the length parameter, if any of these conditions did not occur since the High of the Uptrend, the function will return -1. If all three items occur, the functions will not return -1 and all three conditions will return True.

```
        Condition1 = MRO(Low < HighBand, HighCounter, 1) <> -1;
        Condition2 = MRO(MACDLine crosses below SignalLine, HighCounter, 1) <> -1;
        Condition3 = MRO(RSIValue crosses below 50 OR
                RSIValue crosses below HighRSIValue - RSIChange, HighCounter, 1) <> -1;
```

If all three conditions return True, then the CounterTrendDecline has occurred. By testing the three conditions with AND, a number of actions can be taken. First, the variable CounterTrendDecline is set to True and then the price of entry is calculated and stored in BuyStop. Conditions 1-3 and CheckRetracement are reset to False. The BuyStop is calculated as the highest High of the last BestEntry (Input) bars.

```
If Condition1 AND Condition2 AND Condition3 Then Begin
        CounterTrendDecline = True;
        BuyStop = Highest(High, BestEntry);
        Condition1 = False;
        Condition2 = False;
        Condition3 = False;
        CheckRetracement = False;
End;
```

The next two End; statements block off all of the statements dependent on “HighCounter <> 0 AND HighCounter < TimetoRetrace” and “CheckRetracement”, respectively.

```
        End;
End;
```

If CounterTrendDecline is evaluated to True at any point, a Long entry order is generated for the next bar at the calculated value of BuyStop on a Stop. The End; statement closes the block of statements dependent on “Uptrend”.

```
        If CounterTrendDecline Then
                Buy next bar at BuyStop Stop;
End;
```

Short Entry

In a similar fashion to the Long entry, the Short entry for this strategy is dependent on the existence of an overall Downtrend. All of the statements involved in the Short entry are contained within the Begin-End block, which is dependent on the status of the True/False variable Downtrend.

```
If Downtrend Then Begin
```

If this is the first bar of a new Downtrend, the value of Downtrend on the previous bar will be False. If this is the case, the strategy calls for keeping track of the Low of the Downtrend and the lowest RSIValue in the Downtrend. In addition, two True/False variables need to be reset to False to prevent erroneous order generation. Nested in the Begin-End block of statements is the storing of the Low of the first bar of the Downtrend into DowntrendLow. Subsequent entry conditions require that they occur within ‘X’ number of bars from the Low of the Downtrend, so a variable called LowCounter is reset to zero to keep track of the number of bars that elapse from the new DowntrendLow. The current RSIValue is stored into LowRSIValue and CheckRetracement and CounterTrendRally are reset to False. Because this block is actually the first portion of an If-Then-Else statement, there is no semi-colon (;) after the reserved word End at the end of the block.

```
If Downtrend[1] = False Then Begin
        DowntrendLow = Low;
        LowCounter = 0;
        LowRSIValue = RSIValue;
        CheckRetracement = False;
        CounterTrendRally = False;
End
```

The Else portion of the If-Then-Else statement is executed on any continuing bar of an Downtrend. Again, there is more than one action to take, causing a use of the Begin-End block.

```
Else Begin
```

In order to keep track of DowntrendLow, if the current Low is less than DowntrendLow of the previous bar, then DowntrendLow is reset to the value of the current Low and the LowCounter is reset to zero. If this is not the case, there is an Else portion of this If-Then statement that will increment the LowCounter variable by one if it is less than the limit of TimetoRetrace (Input).

```
        If Low < DowntrendLow[1] Then Begin
                DowntrendLow = Low;
                LowCounter = 0;
        End
        Else If LowCounter < TimetoRetrace Then
                LowCounter = LowCounter + 1;
```

The second statement that is executed on all consecutive bars of a Downtrend is a test of the RSIValue. If the current RSIValue is greater than LowRSIValue, then LowRSIValue is reset to the current RSIValue. The End after this statement contains a semi-colon, bringing a close to the larger If-Then-Else statement that begins with "Downtrend[1] = False".

```
        If RSIValue < LowRSIValue Then
                LowRSIValue = RSIValue;
End;
```

If BearishMomentum occurs during an Downtrend, it sets the trigger of CheckRetracement to True. This is done via an If-Then statement.

```
If BearishMomentum Then
        CheckRetracement = True;
```

A number of the following statements are dependent on the CheckRetracement variable being set to True.

```
If CheckRetracement Then Begin
```

If the LowCounter is not currently zero and the LowCounter is less than TimetoRetrace, the current bar is within TimetoRetrace bars of the bar that contains the Low of the Downtrend. This allows the proper testing and evaluation of the conditions that should occur during a retracement.

```
If LowCounter <> 0 AND LowCounter < TimetoRetrace Then Begin
```

The three conditions made are stored in the predefined True/False variables Condition1, 2 and 3. Using the MRO() function, a condition is evaluated for the occurrence of the High breaking above LowBand, the MACDLine crossing above SignalLine and either the RSIValue crossing above 50 or rising 30 points from its low point of the Downtrend. The MRO() function will return the number of bars ago a condition occurred if it occurred within the specified length, or a -1 if it did not occur. By using LowCounter as the length parameter, if any of these conditions did not occur since the Low of the Downtrend, the function will return -1. If all three items occur, the functions will not return -1 and all three conditions will return True.

```
Condition1 = MRO(High > LowBand, LowCounter, 1) <> -1;
Condition2 = MRO(MACDLine crosses above SignalLine, LowCounter, 1) <> -1;
Condition3 = MRO(RSIValue crosses above 50 OR
        RSIValue crosses above LowRSIValue + RSIChange, LowCounter, 1) <> -1;
```

If all three conditions return True, then the CounterTrendRally has occurred. By testing the three conditions with AND, a number of actions can be taken. First, the variable CounterTrendRally is set to True and then the price of entry is calculated and stored in SellStop. Conditions 1-3 and CheckRetracement are reset to False. The SellStop is calculated as the lowest Low of the last BestEntry (Input) bars.

```
If Condition1 AND Condition2 AND Condition3 Then Begin
        CounterTrendRally = True;
        SellStop = Lowest(Low, BestEntry);
        Condition1 = False;
        Condition2 = False;
        Condition3 = False;
        CheckRetracement = False;
End;
```

The next two End; statements block off all of the statements dependent on “LowCounter <> 0 AND LowCounter < TimetoRetrace” and “CheckRetracement”, respectively.

```
        End;
End;
```

If CounterTrendRally is evaluated to True at any point, a Short entry order is generated for the next bar at the calculated value of SellStop on a Stop. The End; statement closes the block of statements dependent on “Downtrend”.

```
    If CounterTrendRally Then
        Sell next bar at SellStop Stop;
End;
```

Long Exits

An exit that is based on market conditions is generated if a downtrend is indicated by the MACD Signal line and if DMISpread falls below zero. By using MACDDowntrend and DMISpread < 0, an exit order will be generated for the next bar at market.

```
If MACDDowntrend AND DMISpread < 0 Then
        ExitLong ("Below 0") next bar at market;
```

Short Exits

An exit that is based on market conditions is generated if an uptrend is indicated by the MACD Signal line and if DMISpread rises above zero. By using MACDUptrend and DMISpread > 0, an exit order will be generated for the next bar at market. The End; statement terminates the block of statements dependent on a Short position.

```
If MACDUptrend AND DMISpread > 0 Then
        ExitShort ("Above 0") next bar at market;
```

EasyLanguage Signal: ATR Big Profit Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Breakeven Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Protective Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Trailing Stop:

** See Common Stops Appendix

EasyLanguage Signal: ATR Volatility Stop:

** See Common Stops Appendix

EasyLanguage Signal: Last Bar Exit:

** See Common Stops Appendix

Testing and Improving

We tested the *Momentum-Retracement Trading Strategy* (*MRTS*) on daily data for Qualcomm (QCOM, long side only) from 6/92 to 4/00 and for Coffee futures (KC) from 1/95 to 4/00. We set the Max number of bars strategy will reference to 50 and didn't make any deductions for slippage or commission. The default values and the test protocol were as follows:

ChannelLength = 15, testing 10-20 in increments of 5

CloseLength = 5, testing 3-7 in increments of 1

RSILength = 5, testing 3-7 in increments of 1

OBZone = 70, testing 60-80 in increments of 10

OSZone = 30, testing 20-40 in increments of 10

RSIChange = 30, testing 20-40 in increments of 10

Fast = 3, testing 2-4 in increments of 1

Slow = 10, testing 8-12 in increments of 2

Signal = 15, testing 12-18 in increments of 2

DMILength = 15, testing 12-18 in increments of 1

TimetoRetrace = 10, testing 5-15 in increments of 2

BestEntry = 2, testing 1-3 in increments of 1

ProtectiveATRs = 3, testing 2-4 in increments of 1

BreakevenATRs = 3, testing 2-4 in increments of 1

TrailingATRs = 6, testing 4-8 in increments of 1

VolatilityATRs = 3, testing 2-4 in increments of 1

BigProfitATRs = 9, testing 7-11 in increments of 2

BigProfitExitLength = 4, testing 2-6 in increments of 1

Note that we conducted our optimizations on small, closely related groups of inputs, rather than on all 18 inputs at once. The group-by-group method saves a lot of time and still gives good results.

Let's see how well *MRTS* handled QCOM. The optimized values are as follows:

ChannelLength = 20

CloseLength = 3

RSILength = 3

OBZone = 80

OSZone = 20

RSIChange = 30

Fast = 2

Slow = 12

Signal = 16

DMILength = 17

TimetoRetrace = 9

BestEntry = 1

ProtectiveATRs = 3

BreakevenATRs = 3

TrailingATRs = 9

VolatilityATRs = 4

BigProfitATRs = 7

BigProfitExitLength = 4

Figure 2 is a daily bar chart of QCOM with the *MRTS* and its indicators applied [Figure 2, QCOM Chart].

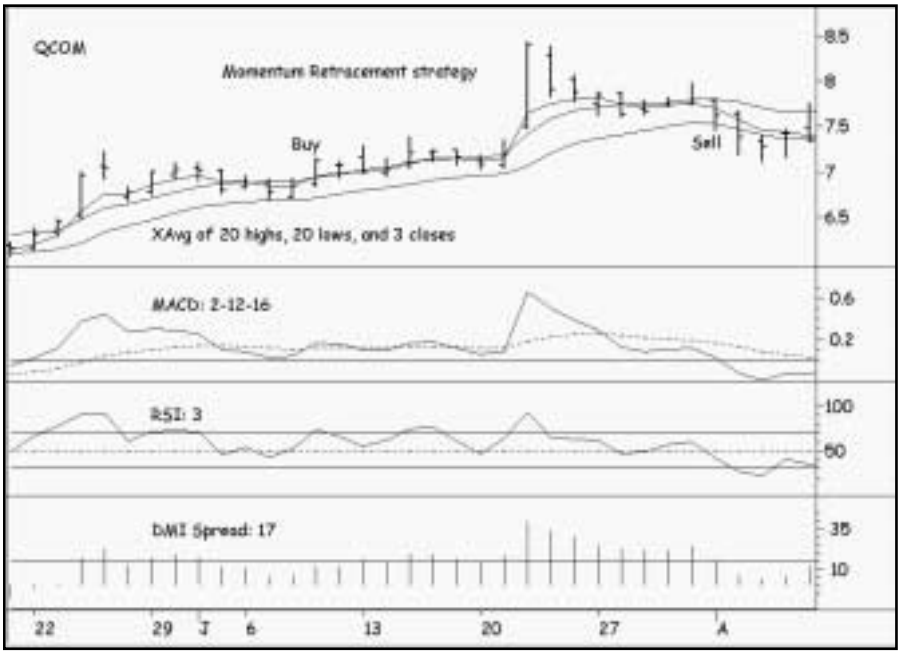


Figure 2. QCOM Chart

Figure 3. QCOM Performance Summary

Applied to QCOM, *MRTS* earned $12,155 on 12 trades [Figure 3, QCOM Performance Summary]. Seventy-five percent of the trades were profitable, with the average winner ($1,360 per 100 shares) 50.35 times the amount of the average loser ($27.00). The average trade made $1,013. The Profit Factor was far beyond any rule of thumb or guideline: *MRTS* won $151 for each $1.00 it lost!

Following are *MRTS*'s optimized values for Coffee futures:

ChannelLength = 20
CloseLength = 7

RSILength = 5
OBZone = 60
OSZone = 30
RSIChange = 40

Fast = 4
Slow = 12
Signal = 12

DMILength = 16
TimetoRetrace = 7
BestEntry = 1

ProtectiveATRs = 2
BreakevenATRs = 3

TrailingATRs = 4
VolatilityATRs = 4
BigProfitATRs = 9
BigProfitExitLength = 2

Figure 4 shows consecutive big winners in Coffee [Figure 4, Coffee Chart]. Note the nearly perfect exit in March, the prompt re-entry in April, and the extraordinary exit one bar off the high in May.

MRTS produced net profits of $76,217 on just 13 trades [Figure 5, Coffee Performance Summary]. Sixty-nine percent of the trades were profitable, while the average winning trade ($9,608) was 3.75 times the size of the average losing trade ($2,563). The Profit Factor was also very strong at 8.43-our strategy earned $8.43 in Coffee for each $1.00 it lost. *MRTS* performed fairly consistently as well, trading profitably in nine of the twelve months when monthly returns were averaged over the length of the test period [Figure 6, Coffee Average Profit by Month].

Summary

MRTS is a very promising strategy. It performs several tasks well: identifying the trend, evaluating its quality, finding a retracement, picking an appropriate entry point, limiting the initial risk, moving the stop to breakeven quickly, trailing a reasonable stop, accounting for changes in recent volatility, and locking in profits when trades become big winners. It's not really a difficult or complex strategy — it's just a long one that offers lots of valuable lessons for the conscientious student.

Figure 4. Coffee Chart

TradeStation Strategy Performance Report

TradeStation Strategy Performance Report - STAD13: Mom-Retrace KC.LNG-Daily

Performance Summary: All Trades

Total Net Profit	$76,216.88	Open position P/L	$0.00
Gross Profit	$86,469.38	Gross Loss	($10,252.50)
Total # of trades	13	Percent profitable	69.23%
Number winning trades	9	Number losing trades	4
Largest winning trade	$40,837.50	Largest losing trade	($3,581.25)
Average winning trade	$9,607.71	Average losing trade	($2,563.13)
Ratio avg win/avg loss	3.75	Avg trade (win & loss)	$5,862.84
Max consec. Winners	3	Max consec. losers	2
Avg # bars in winners	26	Avg # bars in losers	14
Max intraday drawdown	($6,793.13)		
Profit Factor	8.43	Max # contracts held	1
Account size required	$10,168.13	Return on account	749.57%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win/Loss | Time | Graphs | Settings

Figure 5. Coffee Performance Summary

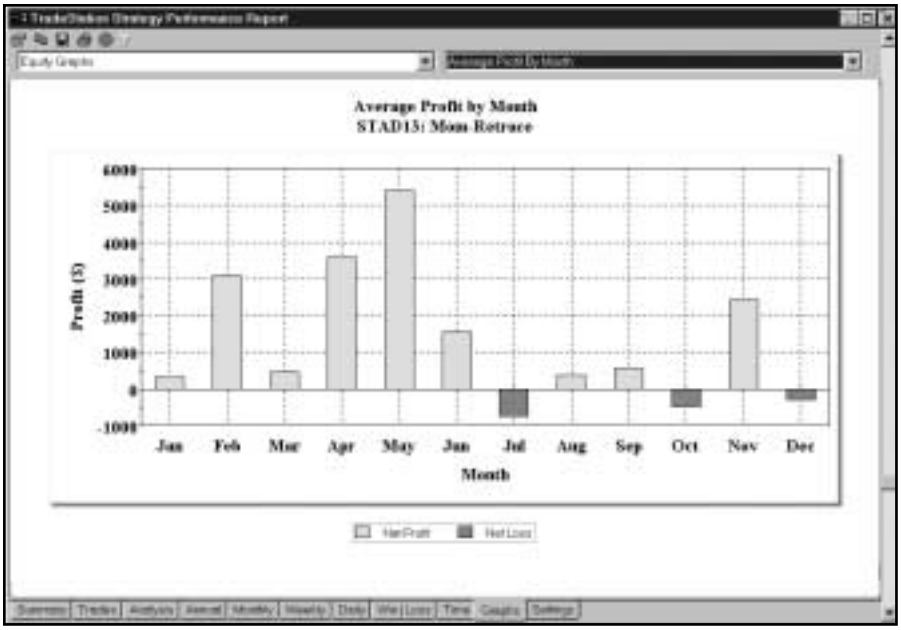


Figure 6. Coffee Average Profit by Month

SECTION 3

Countertrend Strategies

CHAPTER 10

Advance-Decline Divergence

Our *Advance-Decline Divergence* strategy (*ADD*) is based on the premise that an increase in price movement will be accompanied by an increase in volume. Therefore, in order for a rally to evolve into a trend it will most likely be accompanied by an increase in volume. The strategy looks for a divergence between price movement and volume in order to identify tops and bottoms in price.

The Accumulation Distribution indicator is an analysis technique that incorporates both price movement and volume. When the price movement for the day is positive, the daily value of the indicator is added to a running total, and when the price movement is negative, the daily value of the indicator is subtracted to this running total. The formula used by the Accumulation Distribution indicator is as follows:

AccumDist of 1 bar ago + ((Close - Open) / (High - Low) * Volume)

As a result of the formula, when the close of the current bar is greater than the open, the higher the close and the lower the open, the more the indicator is going to advance. Likewise, when the close is lower than the open, the lower the close and the higher the open, the more the indicator will decline.

One of the most commonly used techniques for picking tops and bottoms is to find divergence between price and volume. However, volume will increase when prices move abruptly in either direction, so we need to find a way to express "negative volume," or volume when the price declined. We could have simply used the following formula:

(Close - Open) * Volume

However, the problem with this formula is that the net change in price will have a marked influence on the result; in other words, the result will change significantly based on the width of the range between the close and the open. Since our intention is to compare price with volume in order to find divergences, instead we used the following formula:

(Close - Open) / (High - Low) * Volume

With this new formula, the price factor will always be a number ranging from -1 to 1 regardless of the net move of the price. Because this produces a very choppy line that often crosses over and under zero, we will smooth this plot by applying a nine-bar exponential moving average to it. This formula is identical to that used by the Accumulation Distribution indicator, except that we removed the running total.

Our plan is to enter the market long when price and volume show a divergence. We will buy 100 shares when the price has made a new 30-day Low and the volume move has not made a 30-day low in the last five days. We will buy 200 shares when the price has made a new 30-day low and the volume move has made a new 30-day high in the last five days. Our thinking is that while the first scenario indicates a divergence, it is not as strong a divergence as the second scenario. Therefore, we'll get into the market with the first scenario, but we won't take on as much risk as we will when the second scenario occurs.

Rules for the short side are analogous. We will sell 100 shares when price has made a new 30-day high and volume move has not made a 30-day high in the last five days, we will sell 200 shares when price has made new 30-day high and volume has made a 30-day low in the last five days.

To exit from a long position, we will subtract a four-bar average of the range from the low of the bar and use that as our exit point for the first bar. We will use inputs so that we can tighten this stop. Then, we will add a third of the distance between the low of the bar and the previous exit point to the current exit point to determine the next bar's exit point. We will use this for all bars except our bar of entry.

We will do the opposite for short positions. We will add a four-bar average of the range to the high of the bar of entry and use this as a stop for the first bar. Thereafter, we will subtract a third of the distance between the previous stop and the high of the bar from the previous stop point to determine next bar's exit price. We will use this for all bars except our bar of entry.

Figure 1 shows the ADD strategy and indicator applied to a daily bar chart of GE [Figure 1, GE Chart].

Figure 1. GE Chart

Defining Our Trading Rules

For this strategy, we defined both long and short entries and exits. These components are described next:

Long Entries

a) If price has made a new 30-day low and the volume has not made a 30-day low in the last five days, buy 100 shares.

b) If price has made a new 30-day low and the volume made a 30-day high in the last five days, buy 200 shares.

Short Entries

a) If price has made a new 30-day high and volume has not made a 30-day high in the last five days, sell 100 shares.

b) If price has made a new 30-day high and volume made a 30-day low in the last five days, sell 200 shares.

Long Exits

a) When in a long position, once the bar of entry is closed, we calculate the four-bar average of the range, divide the average by four, and then subtract this value from the low of the current bar. This is our long exit price for bar two.

b) When in a long position, for all bars after bar two, we obtain the previously calculated exit price and add to it a third of the difference between the low and the previous stop price. We repeat this operation at the end of every bar to give us the exit price for all subsequent bars.

Short Exits

a) When in a short position, once the bar of entry is closed, we calculate the four-bar average of the range and divide it by four. Then, we add this value to the high of the current bar. This is our short exit price for bar two.

b) When in a short position, for all bars after bar two, we obtain the previously calculated exit price and subtract from it a third of the difference between the previous stop price and the high of the current bar. Repeating this operation at the end of every bar gives us the exit price for bar three and beyond.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Advance - Decline Divergence (STAD13: Adv-Dec Div)

Strategy Inputs (STAD13: Adv-Dec Div)

INPUT	DEFAULT	DESCRIPTION
LongTerm	30	The number of bars used to determine that a new high or low price has been reached
ShortTerm	5	The number of recent bars used to check that Volume has not made a long term high or low
ExAvgLength	9	The Length parameter used to calculate the exponential average
TradeAmount	2	The number of contracts that will be used to enter into a position
PositionBasis	True	Used to determine if the Stop Loss exit criteria should be based on a position basis or a per contract basis
StopLoss	500	The strict amount of loss, in dollars, at which point the position will be closed

Strategy Components:

1. Adv-Dec Divergence

2. Stop Loss

3. Last Bar Exit

EasyLanguage Signal: Adv-Dec Divergence:

```
Inputs: LongTerm(30), ShortTerm(5), ExAvgLength(9), TradeAmount(2);
Variables: MP(0), StopPrice(0), ADLine(0), ADVolume(0), LowADVolume(0), HighADVolume(0);

MP = MarketPosition;

If High - Low <> 0 Then Begin
        ADLine = (Close - Open) / (High - Low) * Volume;
        ADVolume = XAverage(ADLine, ExAvgLength);
```

```
End;

LowADVolume = Lowest(ADVolume, LongTerm);
HighADVolume = Highest(ADVolume, LongTerm);

If BarNumber > 1 Then Begin
        If Low < Lowest(Low, LongTerm)[1] Then Begin
                If ADVolume > HighADVolume[1] Then
                        Buy ("Strong Buy") TradeAmount contracts next bar at market
                Else If MRO(ADVolume < LowADVolume[1], ShortTerm, 1) = -1 Then
                        Buy ("Weak Buy") TradeAmount / 2 contracts next bar at market;
        End;
        If High > Highest(High, LongTerm)[1] Then Begin
                If ADVolume < LowADVolume[1] Then
                        Sell ("Strong Sell") TradeAmount contracts next bar at market
                Else If MRO(ADVolume > HighADVolume[1], ShortTerm, 1) = -1 Then
                        Sell ("Weak Sell") TradeAmount / 2 contracts next bar at market;
        End;
End;

If MP = 1 Then Begin
        If MP[1] <> 1 Then
                StopPrice = Low - Average(Range, 4);
        ExitLong ("AD LX") next bar at StopPrice Stop;
        StopPrice = StopPrice + (Low - StopPrice) / 3;
End;

If MP = -1 Then Begin
        If MP[1] <> -1 Then
                StopPrice = High + Average(Range, 4);
        ExitShort ("AD SX") next bar at StopPrice stop;
        StopPrice = StopPrice - (StopPrice - High) / 3;
End;
```

Signal Inputs (Adv-Dec Divergence)

INPUT	DEFAULT	DESCRIPTION
LongTerm	30	The number of bars used to determine that a new high or low price has been reached
ShortTerm	5	The number of recent bars used to check that Volume has not made a long term high or low
ExAvgLength	9	The Length parameter used to calculate the exponential average
TradeAmount	2	The number of contracts that will be used to enter into a position

Signal Variables (Adv-Dec Divergence)

INPUT	DEFAULT	DESCRIPTION
MP	0	[Numeric] Used to store MarketPosition on a bar-by-bar basis
StopPrice	0	[Numeric] Used to calculate an exit value from the current position based on a Stop
ADLine	0	[Numeric] Stores the value of the advance-decline volume line
ADVolume	0	[Numeric] Stores the value of the exponential average of the advance-decline volume line
LowADVolume	0	[Numeric] Stores the lowest value of the exponential average of the advance-decline volume line over the long term
HighADVolume	0	[Numeric] Stores the highest value of the exponential average of the advance-decline volume line over the long term

Setup

The first step in the setup is to store the reserved word MarketPosition into the variable MP in order to allow references on a bar-by-bar basis. MarketPosition will return 1 for a Long position, -1 for a Short position and 0 for no position. When used alone the reserved word returns the current position and when using a parameter (ex. MarketPosition(n)) will return the position n number of bars ago.

```
MP = MarketPosition;
```

The Advance-Decline line and its exponential average need to be calculated according to their definition. Because division by zero will result in errors when applying any analysis technique, before calculating ADLine, there is a check that the value of the High minus the Low is not zero. As long as this is the case, first the ADLine is calculated as the net change of the day divided by the entire Range, multiplied by the Volume. The exponential average of ADLine is then taken using ExAvgLength (Input) as the length parameter.

```
If High - Low <> 0 Then Begin
        ADLine = (Close - Open) / (High - Low) * Volume;
        ADVolume = XAverage(ADLine, ExAvgLength);
End;
```

The final step in the setup section is determining the lowest and highest values of the ADVolume line over the last LongTerm (Input) days. These are calculated using the Lowest() and Highest() functions and stored in LowADVolume and HighADVolume, respectively.

```
LowADVolume = Lowest(ADVolume, LongTerm);
HighADVolume = Highest(ADVolume, LongTerm);
```

Long Entry

Both our Long and Short entries require a comparison to values calculated on the previous bar. For this reason, all EasyLanguage statements for entry are nested in a block If-Then-Else statement where the qualifier for this Block is that the current BarNumber is greater than 1. BarNumber is a function that counts the number of bars from where the analysis technique has begun. Checking that it is greater than one allows for proper comparisons when checking values of “one bar ago”, which on the first bar would reflect as the default value for any variable.

```
If BarNumber > 1 Then Begin
```

The Long entries are further restricted to when a new LongTerm Low is met. Comparing the value of the current Low with the lowest LongTerm Low of the previous bar can make this check.

```
    If Low < Lowest(Low, LongTerm)[1] Then Begin
```

A strong buy situation is defined by the ADVolume line being greater than its previous long term high. If this is the case, an order to buy TradeAmount (Input) contracts is placed at market. If the current ADVolume line is not greater, a test is made to verify that within the last ShortTerm (Input) number of bars the ADVolume line has not made a new LongTerm low. This test is made using the MRO() (Most Recent Occurrence) function, which takes a condition statement and returns the number of bars ago the specified condition occurred. If the condition did not occur within a certain length, -1 is returned. If the MRO() function used returns -1, then the condition of a new low ADVolume in the last ShortTerm bars did not occur, triggering a weak buy signal of only one-half of TradeAmount contracts at market.

```
If ADVolume > HighADVolume[1] Then
        Buy ("Strong Buy") TradeAmount contracts next bar at market
Else If MRO(ADVolume < LowADVolume[1], ShortTerm, 1) = -1 Then
        Buy ("Weak Buy") 1 contract next bar at market;
```

The block statement for the Long entries is cut off by the use of the word End. All of the statements between the Begin-End will be executed together based on the condition before the word Begin.

```
    End;
```

Short Entry

In addition to the overall entry restriction of BarNumber > 1, Short entries also are limited to when a new LongTerm High is met. Comparing the value of the current High with the highest LongTerm High of the previous bar makes this check.

```
    If High > Highest(High, LongTerm)[1] Then Begin
```

A strong sell situation is defined by the ADVolume line being less than its previous long term low. If this is the case, an order to sell TradeAmount contracts is placed at market. If not, the ADVolume line is tested to verify that within the last ShortTerm (Input) number of bars the line has not made a new LongTerm high. Again, this test is made using the MRO() function. If the MRO() function used returns -1, then the condition of a new how

ADVolume in the last ShortTerm bars did not occur, triggering a weak sell signal of only one-half of TradeAmount contracts at market.

```
If ADVolume < LowADVolume[1] Then
        Sell ("Strong Sell") 2 contracts next bar at market
Else If MRO(ADVolume > HighADVolume[1], ShortTerm, 1) = -1 Then
        Sell ("Weak Sell") 1 contract next bar at market;
```

The block statement for the Short entries is cut off by the use of the word End. There is a second End at this point, which ties to the original Begin which in turn is tied to the condition BarNumber > 1. All of the statements between these Begin-End statements will be executed together based on the conditions before the words Begin.

```
        End;
End;
```

Long Exits

Because the exit statements are dependent on a calculation, the ExitLong statements are restricted to when the strategy is in a Long position. This is done by checking that the variable MP is equal to one. More than one statement is dependent on this condition, so the Begin-End is used again in order to group them together.

```
If MP = 1 Then Begin
```

Because MP refers to MarketPosition on individual bars, a new stop value can be calculated on the first bar of the new position. This is determined when the MarketPosition of the previous bar was not 1. As defined, the Stop price is the Low of the entry bar minus the average Range of the last four bars.

```
        If MP[1] <> 1 Then
                StopPrice = Low - Average(Range, 4);
```

The ExitLong statement is generated at the current StopPrice value on a stop.

```
        ExitLong ("AD LX") next bar at StopPrice Stop;
```

StopPrice is recalculated for the next bar by adding the one third of the difference between the current bar's Low and the StopPrice to the current StopPrice. This action works similar to a parabolic stop, mixing between a Stop loss and a trailing profit stop.

```
        StopPrice = StopPrice + (Low - StopPrice) / 3;
```

The word End is used to terminate the block of statements that are dependent on a Long position.

```
End;
```

Short Exits

ExitShort statements are restricted to when the strategy is in a Short position, again due to calculations used by the Exit statement. Again, since more than one statement is dependent on this condition, the Begin-End is used again in order to group them together.

```
If MP = -1 Then Begin
```

On the first bar of the new position a new stop value is calculated. As defined, the Stop price for a Short position is the High of the entry bar plus the average Range of the last four bars.

```
If MP[1] <> -1 Then
        StopPrice = High + Average(Range, 4);
```

The ExitShort statement is generated at the current StopPrice value on a stop.

```
ExitShort ("AD SX") next bar at StopPrice stop;
```

StopPrice is recalculated for the next bar by subtracting one third of the difference between the StopPrice and the current bar's High to the current StopPrice. This action works similar to a parabolic stop, mixing between a Stop loss and a trailing profit stop.

```
StopPrice = StopPrice - (StopPrice - High) / 3;
```

The word End is used to terminate the block of statements that are dependent on a Short position.

```
End;
```

EasyLanguage Signal: Stop Loss

** See Common Stops Appendix

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing and Improving

We tested our *ADD* strategy on daily bars of General Electric (GE, long side only) from 4/96 to 4/00. We set the Max number of bars strategy will reference to 50 and didn't make any deductions for slippage and commission. The default values and testing protocol were as follows:

LongTerm = 30, testing 10-50 in increments of 10

ShortTerm = 5, testing 2-8 in increments of 2

ExAvgLength = 9, testing 5-13 in increments of 2

StopLoss = 500, testing 300-700 in increments of 50

Applied to GE, the *ADD* strategy earned $4,760 on 29 trades [Figure 2, GE Performance Summary]. Fifty-five percent of the trades were profitable, while the average winner ($420) was 2.79 times the amount of the average loser ($151). The largest winning trade ($2,144) was almost four times the size of the largest losing trade ($506). *ADD* held on to winners for an average of 13 bars, while exiting losers in an average of only three bars. Our strategy gained $3.43 for each $1.00 it lost.


TradeStation Strategy Performance Report

TradeStation Strategy Performance Report - STAD13: Adv-Dec Div GE-Daily

Performance Summary: All Trades

Total Net Profit	$4,760.40	Open position P/L	$0.00
Gross Profit	$6,721.10	Gross Loss	($1,960.70)
Total # of trades	29	Percent profitable	55.17%
Number winning trades	16	Number losing trades	13
Largest winning trade	$2,143.70	Largest losing trade	($506.30)
Average winning trade	$420.07	Average losing trade	($150.82)
Ratio avg win/avg loss	2.79	Avg trade (win & loss)	$164.15
Max consec. Winners	5	Max consec. losers	4
Avg # bars in winners	13	Avg # bars in losers	3
Max intraday drawdown	($987.60)		
Profit Factor	3.43	Max # contracts held	200
Account size required	$875,987.63	Return on account	0.70%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win/Loss | Time | Graphs | Settings

Figure 2. GE Performance Summary

Another encouraging performance measurement is that *ADD* posted gains in each of the seven years of the test period [Figure 3, GE Annual Performance Summary]. The Equity Curve continues to improve, with recent big gains accounting for most of the strategy's profits in GE [Figure 4, GE Equity Curve]. The Underwater Equity Curve gives the impression that the strategy has not performed well recently, but note that the three worst drawdowns were only about four percent of our equity in the stock [Figure 5, GE Underwater Equity Curve]. It's interesting to compare the Equity Curve and the Underwater Equity Curve. Both clearly reflect the major increase in volatility over the past few years, as *ADD* earned the most money while it also incurred its deepest drawdowns.

TradeStation Strategy Performance Report

Annual Trading Summary

Annual Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
YTD	$1,637.40	12.48%	4.23	2	50.00%
12 month	$2,949.90	24.98%	5.33	6	50.00%
99	$1,312.50	11.11%	11.50	3	66.67%
98	$999.60	9.25%	3.86	5	40.00%
97	$231.30	2.19%	1.46	3	66.67%
96	$6.20	0.06%	1.02	3	33.33%
95	$306.10	2.98%	N/A	4	100.00%
94	$71.40	0.70%	1.62	3	33.33%
93	$195.70	1.96%	3.39	6	50.00%

Annual Rolling Period Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
00	$1,637.40	12.48%	4.23	2	50.00%
99-00	$2,949.90	24.98%	5.67	5	60.00%
98-00	$3,949.70	36.54%	5.02	10	50.00%
97-00	$4,181.00	39.52%	3.82	13	53.85%
96-00	$4,187.20	39.60%	3.38	16	50.00%
95-00	$4,493.30	43.76%	3.55	20	60.00%
94-00	$4,564.70	44.77%	3.43	23	56.52%
93-00	$4,760.40	47.60%	3.43	29	55.17%

Figure 3. GE Annual Performance Summary

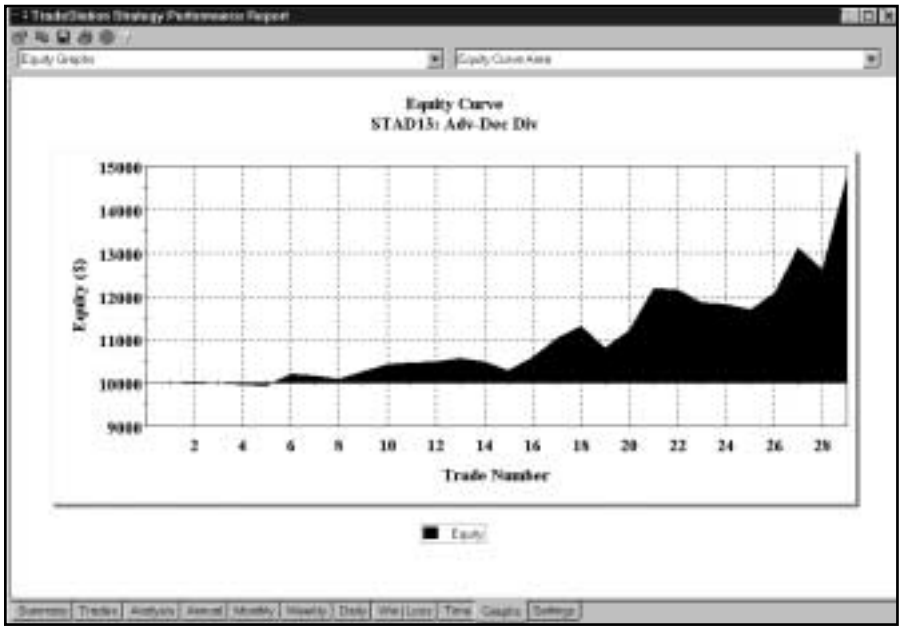


Figure 4. GE Equity Curve

Figure 5. GE Underwater Equity Curve

Summary

ADD is a countertrend strategy: it tries to buy low and sell high by identifying tops and bottoms that haven't been confirmed by accompanying increases in volume. Many traders believe that it's impossible to buy near bottoms and to sell near tops. We hope that our *ADD* strategy will at least encourage some research and discussion on the countertrend approach to trading.

CHAPTER 11

Triple Play

Our *Triple Play* strategy is a variation of Charles Lindsay's well-respected *Trident* Trading Strategy. The most significant change we made to this strategy for STAD Club is the addition of Bollinger Bands to mechanically locate the crucial chart points. *Triple Play* is based on three swing highs and lows (a swing high has at least one lower high immediately before and after it; a swing low has at least one higher low immediately before and after it).

Since the strategy's rules for long and short positions are analogous, we'll just introduce the long side here. For a long position, *Triple Play* first identifies a swing low (point 1), a swing high (point 2), and a swing low (point 3) between points 1 and 2 [Figure 1, British Pound Chart]. Bollinger Bands are overlaid on the price chart to help locate points 1 and 2. For both the long and the short setups, points 1 and 2 must be outside the bands; point 3 must be inside the bands. Also, point 3 must retrace between 38.2% and 61.8% (Fibonacci retracement levels) of the price swing between point 1 and point 2 (see Figure 1).

With the setup in place, our Triple Play strategy uses points 1, 2, and 3 to calculate an entry price, initial protective stop, trailing stop, and two price targets (see Figure 1). The strategy's setup, entries and exits are described next.

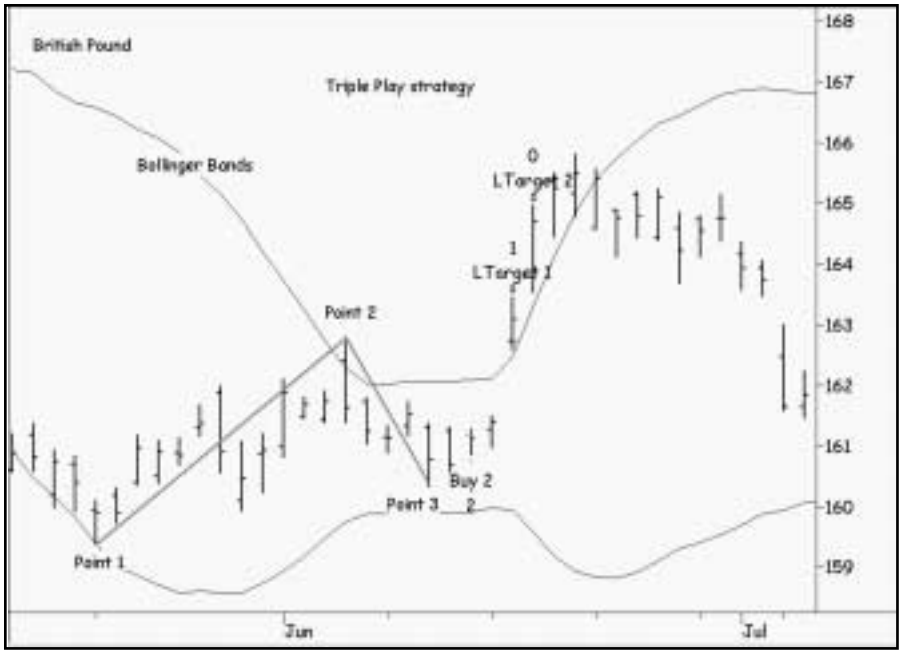


Figure 1. British Pound Chart

Defining Our Trading Rules

For the *Triple Play* strategy, we defined both long and short setup, entries and exits. We also calculated the Bollinger Bands indicator and the Fibonacci retracement levels. The setups, triggers, orders, and exits are described next.

Long and Short Setups

a) The setup to buy requires a price move from a swing low (point 1) to a swing high (point 2) and a retracement to a swing low (point 3).

b) The setup to sell requires a price move from a swing high (point 1) to a swing low (point 2) and a retracement to a swing high (point 3).

Long and Short Entries

a) The entry condition for a long position is a resumption of the up move to the following level: subtract point 1 from point 2, divide the difference by 4, and add the result to point 3.

b) To enter a long position, place an order to buy two units (200 shares of a stock or two contracts of a commodity) at the long entry trigger on a stop.

c) The entry condition for a short position is a resumption of the down move to the following level: subtract point 2 from point 1, divide the difference by 4, and subtract the result from point 3.

d) To enter a short position, place an order to sell short two units (200 shares of a stock or two contracts of a commodity) at the short entry trigger on a stop.

Long and Short Exits

a) If long two units, exit both at the initial protective stop or exit one unit at target 1 and one unit at either the trailing stop or target 2, whichever is hit first. (The initial protective stop is at point 3 minus one point. The trailing stop, which is activated after a close at point 2 or higher, is calculated as follows: Find the highest close since entry and subtract 25% of the difference between point 1 and point 2. Target 1 is calculated as follows: add point 3 to target 2 and divide the total by 4. Target 2 is calculated as follows: add point 2 to point 3 and subtract point 1 from the total.)

b) If short two units, exit both at the initial protective stop or exit one unit at target 1 and one unit at either the trailing stop or target 2, whichever is hit first. (The calculations for the initial protective stop, the trailing stop, target 1, and target 2 are analogous to the calculations for the long side.)

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Triple Play (STAD13: Triple Play)

Strategy Inputs (STAD13: Triple Play)

INPUT	DEFAULT	DESCRIPTION
PatternLength	25	The maximum number of bars that can be included in the formation of the Triple Play
Strength	7	Determines the strength of the swing points
BarsToEnter	10	Specifies the number of bars after a Triple Play formation occurs that an entry signal can last
Length	11	Determines the length used by the Bollinger Bands
Deviations	2	The number of standard deviations used by the Bollinger Bands
Trading	2	The number of contracts that will be used to enter into a position
EntryPriceDivisor	4	Used in calculating the entry stop value
Target1Divisor	2	Used to calculate the first profitable exit target
TrailingStopDivisor	4	Used to calculate the value of the trailing stop

Signal Components:

1. Enhanced Triple Play

EasyLanguage Signal: Enhanced Triple Play:

```
Inputs:
        PatternLength(25), Strength(7), BarsToEnter(10), Length(11), Deviations(2), Trading(2),
        EntryPriceDivisor(4), Target1Divisor(2), TrailingStopDivisor(4);
Variables:
        StrongLow(0), StrongHigh(0), WeakLow(0), WeakHigh(0),
        StrongLowBar(0), StrongHighBar(0), WeakLowBar(99999), WeakHighBar(99999),
        LongEntryPrice(99999), LongExitPrice(0), LongTarget1(0), LongTarget2(0),
        ShortEntryPrice(0), ShortExitPrice(0), ShortTarget1(0), ShortTarget2(0),
        RecentLow(0), RecentHigh(0), FoundAWeakLow(False), FoundAWeakHigh(False),
        Offset(Strength), TrailStopDrop(0), TrailStopRise(0),
        LongTrailingTrigger(0), ShortTrailingTrigger(0), HighClose(0), LowClose(99999),
        BuyCounter(0), SellCounter(0), MakeLongTrade(False), MakeShortTrade(False),
        LongTrailStop(False), ShortTrailStop(False),
        UpperBand(0), LowerBand(0);


UpperBand = BollingerBand(Close, Length, Deviations);
LowerBand = BollingerBand(Close, Length, -Deviations);

StrongHighBar = IFF(StrongHighBar <= PatternLength + Strength, StrongHighBar + 1, StrongHighBar);
StrongLowBar = IFF(StrongLowBar <= PatternLength + Strength, StrongLowBar + 1, StrongLowBar);

If SwingHigh(1, High, Strength, Strength + 1) <> -1 AND High[Strength] > UpperBand[Strength] Then Begin
        StrongHigh = SwingHigh(1, High, Strength, Strength + 1);
        StrongHighBar = Strength;
        RecentLow = MRO(SwingLow(1, Low, Strength, Strength + 1) <> -1, StrongLowBar - StrongHighBar, 1);
        If RecentLow <> -1 Then Begin
                WeakLow = SwingLow(1, Low, Strength, Strength + 1)[Value1];
                WeakLowBar = Value1 + Strength;
                FoundAWeakLow = True;
        End;
End;

If SwingLow(1, Low, Strength, Strength + 1) <> -1 AND Low[Strength] < LowerBand[Strength] Then Begin
StrongLow = SwingLow(1, Low, Strength, Strength + 1);
        StrongLowBar = Strength;
        RecentHigh = MRO(SwingHigh(1, High, Strength, Strength + 1) <> -1, StrongHighBar - StrongLowBar, 1);
        If RecentHigh <> -1 Then Begin
                WeakHigh = SwingHigh(1, High, Strength, Strength + 1)[Value1];
                WeakHighBar = Value1 + Strength;
                FoundAWeakHigh = True;
```

```
        End;
End;

Condition1 = StrongLowBar < PatternLength + Strength;
Condition2 = StrongHighBar < PatternLength + Strength;

If FoundAWeakLow AND Condition1 Then Begin
        If EntryPriceDivisor <> 0 Then
                LongEntryPrice = ( (StrongHigh - StrongLow) / EntryPriceDivisor ) + WeakLow;
        LongExitPrice = WeakLow - 1 point;
        LongTarget2 = WeakLow + (StrongHigh - StrongLow);
        If Target1Divisor <> 0 Then
                LongTarget1 = (WeakLow + LongTarget2) / Target1Divisor;
        LongTrailingTrigger = StrongHigh;
        If TrailingStopDivisor <> 0 Then
                TrailStopDrop = (StrongHigh - WeakLow) / TrailingStopDivisor;
        BuyCounter = 0;
        MakeLongTrade = True;
        MakeShortTrade = False;
End;

If FoundAWeakHigh AND Condition2 Then Begin
        If EntryPriceDivisor <> 0 Then
                ShortEntryPrice = WeakHigh - ( (StrongHigh - StrongLow) / EntryPriceDivisor ) ;
        ShortExitPrice = WeakHigh + 1 point;
        ShortTarget2 = WeakHigh - (StrongHigh - StrongLow);
        If Target1Divisor <> 0 Then
                ShortTarget1 = (WeakHigh + ShortTarget2) / Target1Divisor;
        ShortTrailingTrigger = StrongLow;
        If TrailingStopDivisor <> 0 Then
                TrailStopRise = (WeakHigh - StrongLow) / TrailingStopDivisor;
        SellCounter = 0;
        MakeShortTrade = True;
        MakeLongTrade = False;
End;

FoundAWeakLow = False;
FoundAWeakHigh = False;

If MarketPosition = 0 Then Begin
        If BuyCounter < BarsToEnter Then
                BuyCounter = BuyCounter + 1
        Else
                MakeLongTrade = False;
        If SellCounter < BarsToEnter Then
                SellCounter = SellCounter + 1
        Else
                MakeShortTrade = False;
        If MakeLongTrade Then
                Buy ("Buy 2") Trading Contracts Next Bar at LongEntryPrice Stop;
```

```
        If MakeShortTrade Then
                Sell ("Sell 2") Trading Contracts Next Bar at ShortEntryPrice Stop;
End;

If MarketPosition = 1 Then Begin
        MakeLongTrade = False;
        If CurrentContracts = Trading Then Begin
                LongTrailStop = False;
                HighClose = 0;
                ExitLong ("LDump 1") Next Bar at LongExitPrice Stop;
                ExitLong ("LTarget 1") (Trading / 2) Contracts Total Next Bar at LongTarget1 Limit;
        End
        Else Begin
                If Close > HighClose Then
                        HighClose = Close;
                ExitLong ("LProtect 2") Next Bar at EntryPrice Stop;
                If HighClose > LongTrailingTrigger Then
                        LongTrailStop = True;
                If LongTrailStop Then
                        ExitLong ("LTrail") Next Bar at HighClose - TrailStopDrop Stop;
                ExitLong ("LTarget 2") Next Bar at LongTarget2 Limit;
        End;
End;

If MarketPosition = -1 Then Begin
        MakeShortTrade = False;
        If CurrentContracts = Trading Then Begin
                ShortTrailStop = False;
                LowClose = 99999;
                ExitShort ("SDump 1") Next Bar at ShortExitPrice Stop;
                ExitShort ("STarget 1") (Trading / 2) Contracts Total Next Bar at ShortTarget1 Limit;
        End
        Else Begin
                If Close < LowClose Then
                        LowClose = Close;
                ExitShort ("SProtect 2") Next Bar at EntryPrice Stop;
                If LowClose < ShortTrailingTrigger Then
                        ShortTrailStop = True;
                If ShortTrailStop Then
                        ExitShort ("STrail") Next Bar at LowClose + TrailStopRise Stop;
                ExitShort ("STarget 2") Next bar at ShortTarget2 Limit;
        End;
End;
```

Signal Inputs (Enhanced Triple Play)

INPUT	DEFAULT	DESCRIPTION
PatternLength	25	The maximum number of bars that can be evaluated for the formation of the Trident pattern
Strength	7	Determines the strength of the swing points
BarsToEnter	10	Specifies the number of bars after a Triple Play formation occurs that an entry signal will last
Length	11	Used to determine the Length of the average calculation that the Bollinger Bands are based on.
Deviations	2	The number of standard deviations used by the Bollinger Bands
Trading	2	The number of contracts that will be used to enter into a position
EntryPriceDivisor	4	Used in calculating the entry stop value
Target1Divisor	2	Used to calculate the first profitable exit target
TrailingStopDivisor	4	Used to calculate the value of the trailing stop

Signal Variables (Enhanced Triple Play)

INPUT	DEFAULT	DESCRIPTION
StrongLow	0	[Numeric] Stores the value of the nearest Low swing point that was lower than the bottom Bollinger Band
StrongHigh	0	[Numeric] Stores the value of the nearest High swing point that was higher than the upper Bollinger Band
WeakLow	0	[Numeric] Stores the value of the nearest Low swing point that was within the Bollinger Bands
WeakHigh	0	[Numeric] Stores the value of the nearest High swing point that was within the Bollinger Bands
StrongLowBar	0	[Numeric] Stores the number of bars ago that a Low swing point that was lower than the bottom Bollinger Band occurred
StrongHighBar	0	[Numeric] Stores the number of bars ago that a High swing point that was higher than the upper Bollinger Band occurred
WeakLowBar	99999	[Numeric] Stores the number of bars ago that a Low swing point that was not lower than the bottom Bollinger Band occurred
WeakHighBar	99999	[Numeric] Stores the number of bars ago that a Low swing point that was not higher than the upper Bollinger Band occurred
LongEntryPrice	99999	[Numeric] Stores the value calculated for a Long Entry when the setup formation is completed
LongExitPrice	0	[Numeric] Stores the value calculated for a Long Entry when the setup formation is completed
LongTarget1	0	[Numeric] Used to Store the value for the exiting the first portion of a Long trade at a profit
LongTarget2	0	[Numeric] Used to Store the value for the exiting the final portion of a Long trade at a profit
ShortEntryPrice	0	[Numeric] Stores the value calculated for a Short Entry when the setup formation is completed
ShortExitPrice	0	[Numeric] Stores the value calculated for a Short Entry when the setup formation is completed
ShortTarget1	0	[Numeric] Used to Store the value for the exiting the first portion of a Short trade at a profit
ShortTarget2	0	[Numeric] Used to Store the value for the exiting the final portion of a Short trade at a profit
RecentLow	0	[Numeric] Stores the value of a Low swing point as the middle swing point of a Long side formation
RecentHigh	0	[Numeric] Stores the value of a High swing point as the middle swing point of a Short side formation
FoundAWeakLow	False	[True/False] Used to signal that a “weak” swing point has occurred between the high and low swing points to complete the formation
FoundAWeakHigh	False	[True/False] Used to signal that a “weak” swing point has occurred between the high and low swing points to complete the formation

Signal Variables (Enhanced Triple Play) *CONTINUED*

INPUT	DEFAULT	DESCRIPTION
TrailStopDrop	0	[Numeric] Stores the value to subtract from the highest Close since entry for use in calculating a Long trailing stop
TrailStopRise	0	[Numeric] Stores the value to add to the lowest Close since entry for use in calculating a Short trailing stop
LongTrailingTrigger	0	[Numeric] Stores the value to reach that will trigger the exit orders for the trailing stop, Long side only
ShortTrailingTrigger	0	[Numeric] Stores the value to reach that will trigger the exit orders for the trailing stop, Short side only
HighClose	0	[Numeric] Used to keep track of the highest Close since entry of the position
LowClose	99999	[Numeric] Used to keep track of the lowest Close since entry of the position
BuyCounter	0	[Numeric] Used to count the number of bars that pass since the Long formation is complete until the position is entered
SellCounter	0	[Numeric] Used to count the number of bars that pass since the Short formation is complete until the position is entered
MakeLongTrade	False	[True/False] Used to generate the Buy order
MakeShortTrade	False	[True/False] Used to generate the Sell order
LongTrailStop been triggered	False	[True/False] Used to indicate that the trailing stop for the Long side has
ShortTrailStop been triggered	False	[True/False] Used to indicate that the trailing stop for the Short side has
UpperBand	0	[Numeric] Holds the value calculated for the upper Bollinger Band
LowerBand	0	[Numeric] Holds the value calculated for the lower Bollinger Band

Setup

The major or “strong” swing points that are basis for the formation are those that break the upper and lower Bollinger Bands. The upper and lower band values are calculated with the BollingerBand() function. A negative Deviations (Input) is used in order to calculate the lower band.

```
UpperBand = BollingerBand(Close, Length, Deviations);
LowerBand = BollingerBand(Close, Length, -Deviations);
```

The number of bars ago that a strong swing point occurred is incremented using the IFF() function. IFF() evaluates a condition and if it is evaluated True, returns the second parameter entered in the function. If evaluated to False, the third parameter will be returned. The function requires that both the second and third parameters be numeric values. In this case, if StrongHighBar or StrongLowBar has not yet reached the limit of the PatternLength (Input) plus the Strength (Input) of the swing point, they are incremented by one.

```
StrongHighBar = IFF(StrongHighBar <= PatternLength + Strength, StrongHighBar + 1, StrongHighBar);
StrongLowBar = IFF(StrongLowBar <= PatternLength + Strength, StrongLowBar + 1, StrongLowBar);
```

If a swing High is found on a bar whose High is above the upper Bollinger Band, the value of the swing point is stored into StrongHigh and the number of bars ago it occurred into StrongHighBar. The occurrence of a "weak" swing Low between this strong high point and the most recent strong Low point can then be tested by using the MRO() function. The MRO() function will return the number of bars ago a condition occurred, and -1 if the condition does not evaluate to True within the specified range. If a weak swing Low is detected with the MRO() function, RecentLow is used to store the value of that Low point into WeakLow and the number of bars ago that it occurred into WeakLowBar. The FoundAWeakLow condition is then set to True in order to be used as a trigger for the Triple Play formation.

```
If SwingHigh(1, High, Strength, Strength + 1) <> -1 AND High[Strength] > UpperBand[Strength] Then Begin
        StrongHigh = SwingHigh(1, High, Strength, Strength + 1);
        StrongHighBar = Strength;
        RecentLow = MRO(SwingLow(1, Low, Strength, Strength + 1) <> -1, StrongLowBar - StrongHighBar, 1);
        If RecentLow <> -1 Then Begin
                WeakLow = SwingLow(1, Low, Strength, Strength + 1)[Value1];
                WeakLowBar = Value1 + Strength;
                FoundAWeakLow = True;
        End;
End;
```

On the opposite side, if a swing Low is found that falls below the lower Bollinger Band, the value of the swing point is stored into StrongLow and the number of bars ago it occurred into StrongLowBar. A "weak" swing High between this strong Low point and the most recent strong High point can be detected, again using the MRO() function. If a weak swing High is detected with the MRO() function, RecentHigh is used to store the value of the High point into WeakHigh and the number of bars ago that it occurred into WeakHighBar. The FoundAWeakHigh condition is set to True in order to be used as a trigger.

```
If SwingLow(1, Low, Strength, Strength + 1) <> -1 AND Low[Strength] < LowerBand[Strength] Then Begin
StrongLow = SwingLow(1, Low, Strength, Strength + 1);
        StrongLowBar = Strength;
        RecentHigh = MRO(SwingHigh(1, High, Strength, Strength + 1) <> -1, StrongHighBar - StrongLowBar, 1);
        If RecentHigh <> -1 Then Begin
                WeakHigh = SwingHigh(1, High, Strength, Strength + 1)[Value1];
                WeakHighBar = Value1 + Strength;
                FoundAWeakHigh = True;
        End;
End;
```

In addition to the formation of the swing points, a check that the further strong point has not occurred too far back in the past is made. This is done by checking the number of bars ago the swing point occurred to the value of PatternLength plus Strength.

```
Condition1 = StrongLowBar < PatternLength + Strength;
Condition2 = StrongHighBar < PatternLength + Strength;
```

By the manner in which the setup is evaluated, FoundAWeakLow will only be set to True on the bar that completes the formation of the Triple Play. If on that bar the range of the formation is within the PatternLength, all of the values to be used for the Long entry and its exits need to be calculated. This includes LongEntryPrice, LongExitPrice and the profit targets labeled LongTarget 1 and 2. These are respectively, the value of the weak Low plus the difference between the strong points divided by EntryPriceDivisor (Input), one point below the weak Low, and the weak Low plus the difference between the strong points and the weak Low plus the second target divided by Target1Divisor (Input). The trigger for the trailing stop is when the price reaches the current strong High value and the value used to determine the trailing stop is stored in TrailStopDrop, the difference between the strong High and weak Low, all divided by TrailingStopDivisor (Input). BuyCounter is set to zero in order to count the number of bars that pass after the formation of the pattern, and MakeLongTrade is set to True and MakeShortTrade set to False.

```
If FoundAWeakLow AND Condition1 Then Begin
        If EntryPriceDivisor <> 0 Then
                LongEntryPrice = ( (StrongHigh - StrongLow) / EntryPriceDivisor ) + WeakLow;
        LongExitPrice = WeakLow - 1 point;
        LongTarget2 = WeakLow + (StrongHigh - StrongLow);
        If Target1Divisor <> 0 Then
                LongTarget1 = (WeakLow + LongTarget2) / Target1Divisor;
        LongTrailingTrigger = StrongHigh;
        If TrailingStopDivisor <> 0 Then
                TrailStopDrop = (StrongHigh - WeakLow) / TrailingStopDivisor;
        BuyCounter = 0;
        MakeLongTrade = True;
        MakeShortTrade = False;
End;
```

Respectively, FoundAWeakHigh will only be True on the bar that completes the formation of the short side. If on that bar the range of the formation is within the PatternLength, all of the values are then set up to be used for the Short entry and its exits. This includes the calculation of ShortEntryPrice, ShortExitPrice and the profit targets labeled ShortTarget 1 and 2. These are respectively, the value of the weak High minus the difference between the strong points divided by EntryPriceDivisor, one point above the weak High, and the weak High minus the difference between the strong points and the weak High plus the previous value divided by Target1Divisor. The trigger for the trailing stop is when the price reaches the current strong Low value and the value used to determine the trailing stop is stored in TrailStopRise, the difference between the weak High and strong Low divided by TrailingStopDivisor. SellCounter is set to zero in order to count the number of bars that pass after the formation of the pattern, and MakeShortTrade is set to True and MakeLongTrade set to False.

```
If FoundAWeakHigh AND Condition2 Then Begin
        If EntryPriceDivisor <> 0 Then
                ShortEntryPrice = WeakHigh - ( (StrongHigh - StrongLow) / EntryPriceDivisor ) ;
        ShortExitPrice = WeakHigh + 1 point;
        ShortTarget2 = WeakHigh - (StrongHigh - StrongLow);
        If Target1Divisor <> 0 Then
                ShortTarget1 = (WeakHigh + ShortTarget2) / Target1Divisor;
        ShortTrailingTrigger = StrongLow;
        If TrailingStopDivisor <> 0 Then
                TrailStopRise = (WeakHigh - StrongLow) / TrailingStopDivisor;
        SellCounter = 0;
        MakeShortTrade = True;
        MakeLongTrade = False;
End;
```

After the Long or Short values are set the FoundAWeakLow and FoundAWeakHigh values are no longer needed. On every bar they are always reset to False.

```
FoundAWeakLow = False;
FoundAWeakHigh = False;
```

Long and Short Entry

If there is currently no position entered, the Buy and Sell counters are evaluated. If either has been reset to zero and is currently less than BarsToEnter (Input), then they are incremented. If they are equal to BarsToEnter, then the respective "Make Trade" trigger variables are set to False. If either of the "Make Trade" triggers are set to True, the respective Buy or Sell order will be placed for Trading (Input) contracts on the next bar, at the entry price determined when the formation of the Triple Play was complete.

```
If MarketPosition = 0 Then Begin
        If BuyCounter < BarsToEnter Then
                BuyCounter = BuyCounter + 1
        Else
                MakeLongTrade = False;
        If SellCounter < BarsToEnter Then
                SellCounter = SellCounter + 1
        Else
                MakeShortTrade = False;
        If MakeLongTrade Then
                Buy ("Buy 2") Trading Contracts Next Bar at LongEntryPrice Stop;
        If MakeShortTrade Then
                Sell ("Sell 2") Trading Contracts Next Bar at ShortEntryPrice Stop;
End;
```

Long Exits

When a Long position is taken, the first thing done is to turn off the trigger for MakeLongTrade. Then the number of current contracts is tested to determine what types of actions to take. If Trading contracts are still entered, LongTrailStop is set to False and HighClose is set to 0, they are to be used if the position is closed out to only one-half of the position. With two exits, we generate a protective exit that closes the entire position and a target exit which scales out of one-half of the position.

```
If MarketPosition = 1 Then Begin
        MakeLongTrade = False;
        If CurrentContracts = Trading Then Begin
                LongTrailStop = False;
                HighClose = 0;
                ExitLong ("LDump 1") Next Bar at LongExitPrice Stop;
                ExitLong ("LTarget 1") (Trading / 2) Contracts Total Next Bar at LongTarget1 Limit;
        End
```

If the target exit were hit, only one-half of Trading contracts would be left. We keep track of the highest Close in HighClose and generate a breakeven stop at the price of entry. The trailing stop trigger is set when the highest Close (of the scaled-back position) crosses above LongTrailingTrigger, which was calculated as the strong High value in the Triple Play formation. If the LongTrailStop is in effect, it will trigger an exit at a value of the highest Close of the scaled-back position minus the calculated TrailStopDrop. In addition, a second target stop for a profit is generated if the price continues to rise.

```
        Else Begin
                If Close > HighClose Then
                        HighClose = Close;
                ExitLong ("LProtect 2") Next Bar at EntryPrice Stop;
                If HighClose > LongTrailingTrigger Then
                        LongTrailStop = True;
                If LongTrailStop Then
                        ExitLong ("LTrail") Next Bar at HighClose - TrailStopDrop Stop;
                ExitLong ("LTarget 2") Next Bar at LongTarget2 Limit;
        End;
End;
```

Short Exits

When a Short position is taken, the trigger for MakeShortTrade is set to False. Again, the number of current contracts is tested to determine what types of actions to take. If Trading contracts are still entered, ShortTrailStop is set to False and LowClose is set to 0, they are to be used if the position is scaled back to one half. With two exits, a protective exit is generated that closes the entire position and a target exit, which scales out one half of the position.

```
If MarketPosition = -1 Then Begin
        MakeShortTrade = False;
        If CurrentContracts = Trading Then Begin
                ShortTrailStop = False;
                LowClose = 99999;
                ExitShort ("SDump 1") Next Bar at ShortExitPrice Stop;
                ExitShort ("STarget 1") (Trading / 2) Contracts Total Next Bar at ShortTarget1 Limit;
        End
```

If the target exit were hit, only one-half of the position would remain. The lowest Close is recorded in LowClose and a breakeven stop is generated at the price of entry. The trailing stop trigger is set when the lowest Close crosses below ShortTrailingTrigger, which was calculated as the strong Low value in the Triple Play formation. If the ShortTrailStop is in effect, it will trigger an exit at a value of the lowest Close of the scaled-back position plus TrailStopRise. In addition, a second target stop for a profit is generated if the price continues to fall.

```
        Else Begin
                If Close < LowClose Then
                        LowClose = Close;
                ExitShort ("SProtect 2") Next Bar at EntryPrice Stop;
                If LowClose < ShortTrailingTrigger Then
                        ShortTrailStop = True;
                If ShortTrailStop Then
                        ExitShort ("STrail") Next Bar at LowClose + TrailStopRise Stop;
                ExitShort ("STarget 2") Next bar at ShortTarget2 Limit;
        End;
End;
```

Testing and Improving

We tested the *Triple Play* strategy on daily data for IBM (long side only) from 6/92-4/00 and for British Pound futures (BP) from 1/90-4/00. We set the Max number of bars strategy will reference to 70 and did not make any deductions for slippage and commissions.

Let's see how *Triple Play* traded IBM. The optimized values are as follows:

PatternLength = 25

Strength = 7

BarstoEnter = 10

Length =11

Deviations = 2

EntryPriceDivisor = 4

Target1Divisor = 2

TrailingStopDivisor = 4

Applied to IBM, *Triple Play* captured profits of $12,187 on six trades (per 100 shares traded) [Figure 2, IBM Performance Summary]. Fifty percent of the trades were winners, and the average win ($4,756) was 6.86 times as large as the average loss ($1,519). The average trade earned an impressive $2,031. *Triple Play* let profits run on the winning trades for an average of 123 bars, while it cut losses short in only 35 bars. Our strategy won $6.86 for each $1.00 it lost. *Triple Play* was also a consistent strategy, producing a net profit in each of the five years of the test period [Figure 3, IBM Annual Trading Summary] and trading profitably in nine of twelve months when monthly returns are averaged for the test period [Figure 4, IBM Average Profit by Month].

Figure 2. IBM Performance Summary

TradeStation Strategy Performance Report

Annual Trading Summary

Annual Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
YTD	$887.60	4.17%	15.22	2	50.00%
12 month	$4,956.20	28.76%	3.73	8	50.00%
99	$6,549.80	44.41%	N/A	2	100.00%
98	$4,475.00	43.55%	3.95	2	50.00%
97	$68.80	0.67%	1.10	2	50.00%
96	$206.20	2.06%	N/A	1	100.00%

Annual Rolling Period Analysis (Mark-To-Market):

Period	Net Profit	% Gain	Profit Factor	# Trades	% Profitable
00	$887.60	4.17%	15.22	2	50.00%
99-00	$7,437.40	50.42%	120.19	4	75.00%
98-00	$11,912.40	115.94%	8.53	6	66.67%
97-00	$11,981.20	117.39%	6.24	8	62.50%
96-00	$12,187.40	121.87%	6.33	9	66.67%

Figure 3. IBM Annual Trading Summary

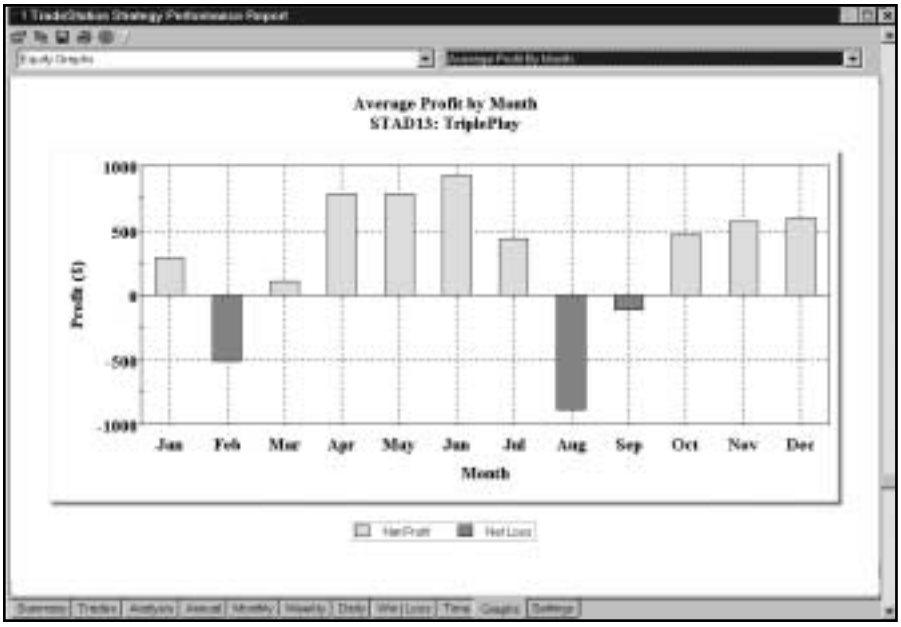


Figure 4. IBM Average Profit by Month

Let's look at *Triple Play*'s results in British Pound futures (BP). The optimized values are as follows:

PatternLength = 20

Strength = 5

BarstoEnter = 5

Length = 7

Deviations = 2

EntryPriceDivisor = 4

Target1Divisor = 2

TrailingStopDivisor = 2

Trading BP, *Triple Play* earned $9,563 on 12 trades, for an average trade of $797 [Figure 5, BP Performance Summary]. A very accurate 75% of the trades were profitable, which is exceptional for a strategy with an average winner 6.06 times the size of the average loser. The Profit Factor of 18.19 means that the strategy won $18.19 for each $1.00 it lost.

Figure 5. BP Performance Summary

Triple Play's Equity Curve in BP shows somewhat slow but very steady growth with minimal drawdowns [Figure 6, BP Equity Curve].

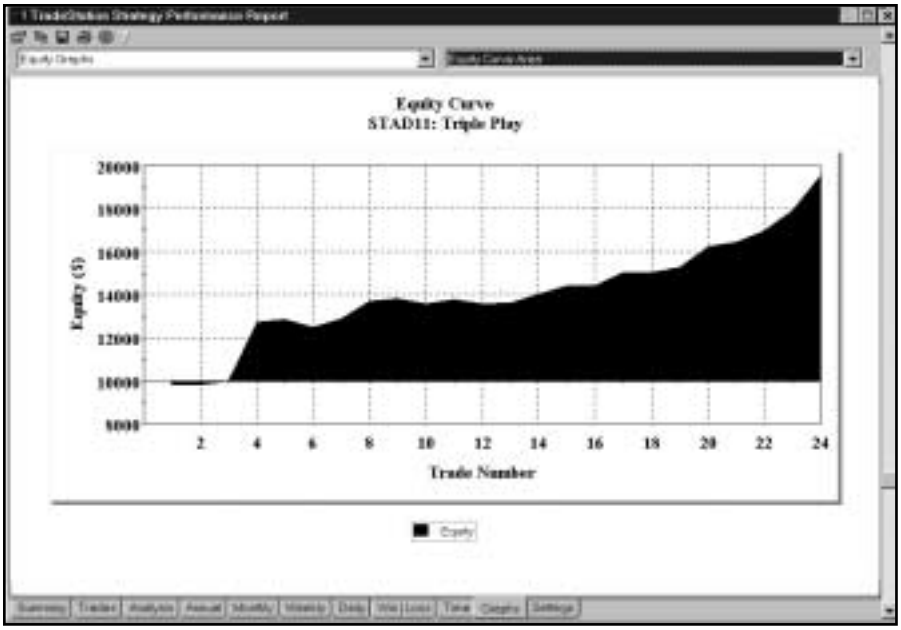


Figure 6. BP Equity Curve

Summary

We believe that adding Bollinger Bands to the *Trident* strategy improved the strategy significantly. With the bands, correctly identifying points 1 and 2 is objective and automatic, which had not been true with earlier versions of the strategy. We also found that *Triple Play* yields better results than most other strategies that employ profit targets as their main exit condition.

SECTION 4

Event Strategies

CHAPTER 12

VolEx

Bill Cruz , Larry Williams, and Charlie Wright were trading and teaching a version of *VolEx* many years ago, and the strategy has continued to be a favorite of many traders to the present day. The *VolEx* strategy attempts to find days with unusually volatile price activity. When it does so, it enters the market and uses the momentum produced by the sudden move to create a quick profit. Like other volatility expansion strategies, *VolEx* will stay in the market for very short periods of time — only the time necessary to produce a profit. This strategy was designed to trade the S&P contract using daily bars, but using its basic principles you can develop a strategy to trade any market.

As with any volatility expansion strategy, the leverage and volatility of the market is important. This type of strategy normally remains in the market for only a few bars as it attempts to make its profit out of numerous but relatively small winning trades. In other words, it is developed specifically for use in markets with high leverage and volatility.

The strategy will establish a buy and a sell price for every day (or for every bar). For the buy price, it multiplies the range of the previous day times a factor and adds it to the open price. For the sell price, it multiplies the range of the previous day times a factor and subtracts it from the open price.

Remember, this strategy is trying to find a significant move in either direction in an attempt to ride the momentum to a profit. Since markets usually fall quicker than they go up, we would need a bigger move down to consider it significant. Therefore, the short factor is normally larger than the buy factor.

This strategy incorporates several exits. First, we'll exit either our long or short position on the first profitable open. To customize this exit, we can consider adding commissions, slippage, and a minimum profit level required for each trade. Second, we will exit from all long positions if the market reaches the lowest low of the last four bars and exit all short positions when the market reaches the highest high of the last four bars.

The third exit technique is to exit the long or short position if neither of the first two exits has occurred, and the strategy has been in the market for more than four days without producing our minimum profit. In this case, we would exit from the trade, regardless of the profit or loss, in order to look for other trading opportunities. Again, the idea of this strategy is to give us many small profits, so there is no reason to remain in the same position without a quick profit. Finally, we'll place a money management stop to limit the losses when trades don't go our way.

Figure 1 shows a series of recent *VolEx* trades in S&P futures [Figure 1, SP Chart].

Figure 1. SP Chart

Defining Your Trading Rules

For this strategy, we defined both long and short entries and exits. These components are described next:

Long and Short Entries

a) Buy tomorrow at tomorrow's open plus 1.2 times the four-bar average range.

b) Sell tomorrow at tomorrow's open minus 1.8 times the four-bar average range.

Long and Short Exits

a) Exit at the first profitable open

b) Once in a long position, exit if prices fall to the lowest low of the last four bars.

c) Once in a short position, exit if prices rally to the highest high of the last four bars.

d) If four bars have elapsed without any of the exits being triggered, exit on the next open.

e) Exit on the next open if the StalePosition conditions are true.

f) Exit if prices reach the protective stop loss.

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: VolEx (STAD13: VOLEX)

Strategy Inputs (STAD13: VOLEX)

INPUT	DEFAULT	DESCRIPTION
LongFactor	1.2	The multiple of the average true range used to determine a long entry stop value
ShortFactor	1.8	The multiple of the average true range used to determine a short entry stop value
PositionBasis	True	Used to determine if the Stop Loss exit criteria should be based on a position basis or a per contract basis
StopLoss	1000	The strict amount of loss, in dollars, at which point the position will be closed
StalePositionTime	0	The number of bars that a position must last before the check for a minimum profitability is made
StaleMinimumProfit	0	The minimum amount of profit that will allow the position to stay open after the determined period of time
TrailingExitLength	4	The number of bars to use to determine the trailing exit
GoodOpenBase	200	The minimum amount of profit that needs to be achieved in order to generate an exit at the open of the next bar
GoodOpenDelay	0	The minimum number of bars that a trade must have in order to initiate the exit orders

Strategy Components:

1. Volatility Expansion
2. Price Trailing Exit
3. Stale Position Exit
4. First Good Open
5. Stop Loss
6. Last Bar Exit

EasyLanguage Signal: Volatility Expansion:

```
Inputs: LongFactor(1.2), ShortFactor(1.8);
Variables: NextBarOpen(0), BuyLevel(0), SellLevel(0);

NextBarOpen = Open of next bar;
BuyLevel = Average(Range, 4) * LongFactor;
SellLevel = Average(Range, 4) * ShortFactor;

{ Entry Orders }
Buy next bar at NextBarOpen + BuyLevel Stop;
Sell next bar at NextBarOpen - SellLevel Stop;
```

Signal Inputs (Volatility Expansion)

INPUT	DEFAULT	DESCRIPTION
LongFactor	1.2	The multiple of the average range used to determine a long entry stop value
ShortFactor	1.8	The multiple of the average range used to determine a short entry stop value

Signal Variables (Volatility Expansion)

INPUT	DEFAULT	DESCRIPTION
NextBarOpen	0	[Numeric] Used to store the value of the open of the next bar
BuyLevel	0	[Numeric] Used to determine the amount above the next bar's open that a long entry stop will be placed
SellLevel	0	[Numeric] Used to determine the amount above the next bar's open that a short entry stop will be placed

Setup

The setup portion of this signal uses the price value Open of next bar, and stores that into NextBarOpen. Open of next bar is a price value that can only be used in a Signal. Keep in mind there are special restrictions to using "Open of next bar" as a price, such as the inability to use any orders that are for the close of the bar in the same Signal. In addition, the value is of BuyLevel and SellLevel is taken by calculating the average range of the last four bars and multiplying by the respective LongFactor or ShortFactor.

```
NextBarOpen = Open of next bar;
BuyLevel = Average(Range, 4) * LongFactor;
SellLevel = Average(Range, 4) * ShortFactor;
```

Long and Short Entry

Orders are placed on every bar, thereby not requiring an If-Then statement for the Buy or Sell statement. They are written with the value, NextBarOpen plus or minus the BuyLevel / SellLevel on a Stop.

```
Buy next bar at NextBarOpen + BuyLevel Stop;
Sell next bar at NextBarOpen - SellLevel Stop;
```

EasyLanguage Signal: Price Trailing Exit:

```
Inputs: ExitLength(4);
Variables: MP(0);

MP = MarketPosition;

If MP = 1 Then
        ExitLong next bar at Lowest(Low, ExitLength) Stop;
If MP = -1 Then
        ExitShort next bar at Highest(High, ExitLength) Stop;
```

Signal Inputs (Price Trailing Exit)

INPUT	DEFAULT	DESCRIPTION
ExitLength	4	The number of bars to use to determine the trailing exit

Signal Variables (Price Trailing Exit)

INPUT	DEFAULT	DESCRIPTION
MP	0	[Numeric] Used to store the current MarketPosition (1 for Long, -1 for Short, 0 for no position)

Long and Short Exits

MarketPosition is stored in the variable MP to allow reference to MarketPosition on a bar by bar basis. If the current position is Long, an exit order is generated for the lowest Low of the last ExitLength (Input) number of bars. If the current position is Short, an exit order is generated for the highest High of the last ExitLength bars.

```
MP = MarketPosition;

If MP = 1 Then
        ExitLong next bar at Lowest(Low, ExitLength) Stop;
If MP = -1 Then
        ExitShort next bar at Highest(High, ExitLength) Stop;
```

EasyLanguage Signal: Stale Position Exit:

```
Inputs: TimetoDetermine(0), MinimumProfit(0);

If BarsSinceEntry > TimetoDetermine AND OpenPositionProfit < MinimumProfit Then Begin
        ExitLong ("Long Too Long") next bar at Market;
        ExitShort ("Short Too Long") next bar at Market;
End;
```

Signal Inputs (Stale Position Exit)

INPUT	DEFAULT	DESCRIPTION
TimetoDetermine	0	The number of bars that a position must last before the check for a minimum profitability is made
MinimumProfit	0	The minimum amount of profit that will allow the position to stay open after the determined period of time

This Signal does not contain any variables

Long and Short Exits

This Signal tests for a trade to have met a minimum duration period, represented in bars by TimetoDetermine (Input). If at any point beyond TimetoDetermine bars the OpenPositionProfit is less than the minimum profit, Exit orders are generated for the next bar at market.

```
If BarsSinceEntry > TimetoDetermine AND OpenPositionProfit < MinimumProfit Then Begin
        ExitLong ("Long Too Long") next bar at Market;
        ExitShort ("Short Too Long") next bar at Market;
End;
```

EasyLanguage Signal: First Good Open

** See Common Stops Appendix

EasyLanguage Signal: Stop Loss

** See Common Stops Appendix

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing and Improving

We tested our *VolEx* strategy on daily data for S&P futures (SP) from 1/99 - 4/00. We set the Max number of bars strategy will reference to 50 and did not make any deductions for slippage or commissions. The default values and testing protocol were as follows:

LongFactor = 1.2, testing .75-2 in increments of .25

ShortFactor = 1.8, testing .75-2 in increments of .25

StopLoss = $1,000, testing 1,000-3,000 in increments of 1,000

TrailingExitLength = 4, testing 1-5 in increments of 1

StalePositionTime = 0, testing 0-3 in increments of 1

StaleMinimumProfit = 0, testing 0-500 in increments of 100

GoodOpenBase = 200, testing 100-500 in increments of 100

GoodOpenDelay = 0, testing 0-5 in increments of 1

Let's see how well our *VolEx* strategy performed on SP. The optimized values are as follows:

LongFactor = .75

ShortFactor = .75

StopLoss = $2,000

TrailingExitLength = 2

StalePositionTime = 0

StaleMinimumProfit = 0

GoodOpenBase = 200

GoodOpenDelay = 0

Trading SP, *VolEx* earned $90,490 on 105 trades, for an average profit per trade of $862 [Figure 2, SP Performance Summary]. Forty-one percent of the trades were profitable, while the average winner ($5,1723) was 2.43 times the size of the average loser ($2,128). The strategy won $1.69 for each $1.00 it lost. The Equity Curve shows that *VolEx* traded only marginally profitably until the market's recent volatility increased and our profits soared [Figure 3, SP Equity Curve].

TradeStation Strategy Performance Report

TradeStation Strategy Performance Report - STAD13: VOLEX SP,LNG-Daily

Performance Summary: All Trades

Total Net Profit	$90,490.00	Open position P/L	$0.00
Gross Profit	$222,432.50	Gross Loss	($131,942.50)
Total # of trades	105	Percent profitable	40.95%
Number winning trades	43	Number losing trades	62
Largest winning trade	$19,567.50	Largest losing trade	($8,125.00)
Average winning trade	$5,172.85	Average losing trade	($2,128.10)
Ratio avg win/avg loss	2.43	Avg trade (win & loss)	$861.81
Max consec. Winners	6	Max consec. losers	7
Avg # bars in winners	2	Avg # bars in losers	1
Max intraday drawdown	($16,377.50)		
Profit Factor	1.69	Max # contracts held	1
Account size required	$19,752.50	Return on account	458.12%

Summary | Trades | Analysis | Annual | Monthly | Weekly | Daily | Win/Loss | Time | Graphs | Settings

Figure 2. SP Performance Summary

Figure 3. SP Equity Curve

Summary

If you prefer a trendfollowing or countertrend strategy to an event-based (i.e. volatility strategy), you may still want to include a volatility component in your strategy. For example, try testing your favorite setup with a volatility entry: with your setup in place, buy or sell on a volatility expansion from the previous bar's close or the current bar's open. A volatility component frequently reduces the number of trades a strategy takes, while it increases the winning percent and the average profit per trade.

CHAPTER 13

Volatility Clusters

One assumption that is frequently made when studying volatility breakouts is that on average there will be one bar that has high volatility per every x number of bars with normal volatility. This assumption is illustrated in a bar chart in Figure 1 [Figure 1, Commonly assumed volatility pattern].

In the above example, on average we will have one period of high volatility for every six normal periods.

However, in reality, high volatility periods tend to cluster as groups, and one period of high volatility will usually be followed by another of more or less the same magnitude. The reason is that the event that caused that one period of high volatility will not cease to have influence over the market simply because we have started drawing a new bar on our charts. In mathematical terms, we are taking a indiscrete event (like the influence of a news story over a stock price) and representing it in discrete terms (bars). We propose that high volatility periods will tend to look like the chart in Figure 2 [Figure 2, Realistic volatility pattern].

In our *Volatility Clusters* strategy, we are going to look for high volatility in two ways: 1) as periods in which the range increases significantly from its average or 2) as periods in which the distance from the current close to the previous close increases significantly from the average distance between two consecutive closes.

Because we are looking for an increase in volatility, we will only look for moves that are more than two standard deviations greater than the average move. When this happens, we will place stop orders both above and below the market to enter either long or short, depending on which way the market breaks out.

To determine the long and short entry prices, we will calculate the three-bar average of the range and both add it to the close and subtract it from the close. Because the range of the current bar is the one we specified as being greater than the average, we will use the three-bar average of the range of one bar ago; this way we will leave the bar with the volatility spike out of the entry price calculation.

We will exit at the first open that meets our profit target, and we will place a money management stop to protect ourselves against losing trades.

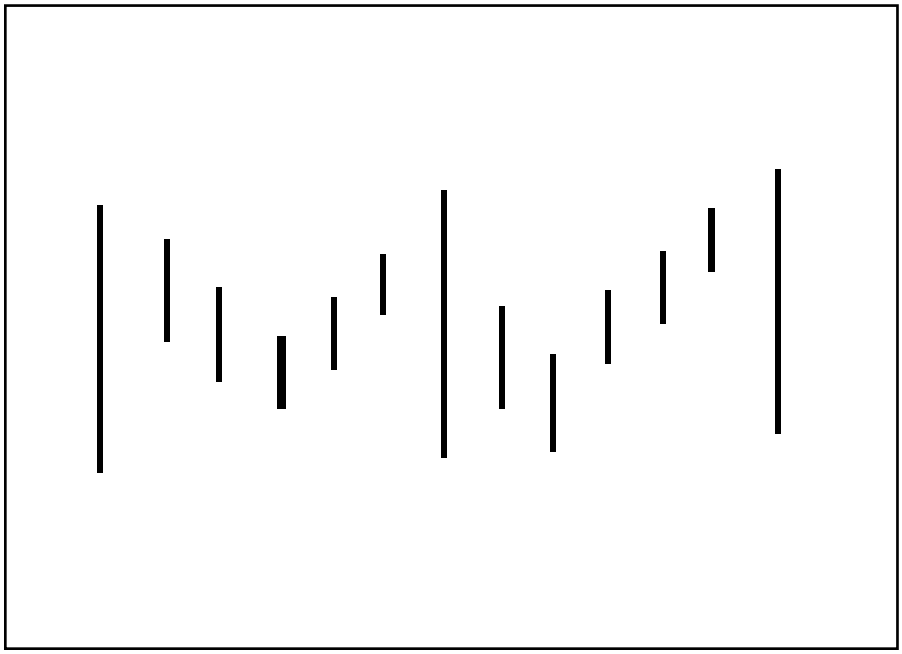


Figure 1. Commonly assumed volatility pattern

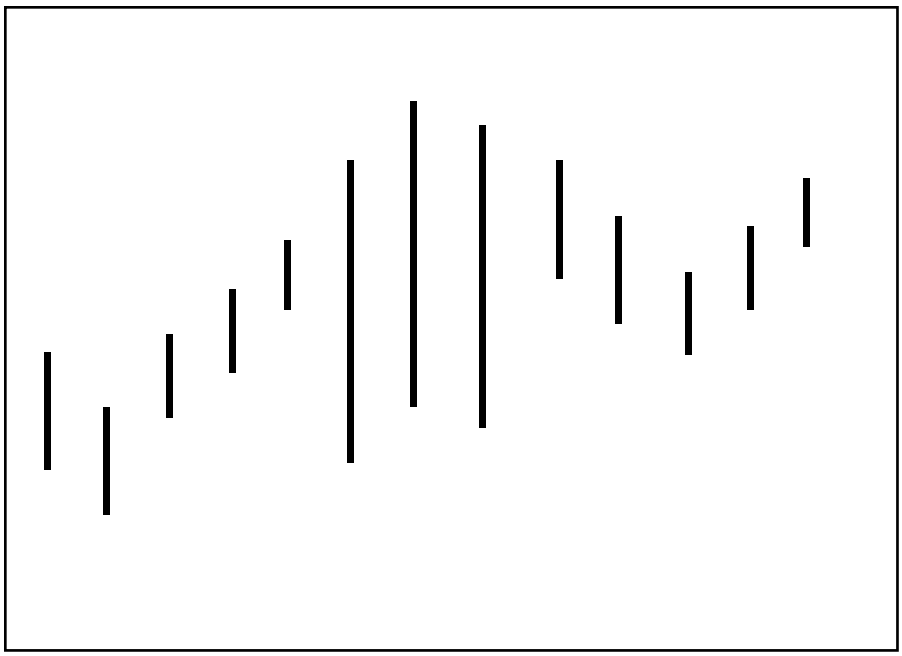


Figure 2. Realistic volatility pattern

Defining Our Trading Rules

In this strategy, we defined both long and short entries and exits. The long and short entries reverse the position, whereas the exits close out an existing position. The setup, entries, and exits are described next.

Long and Short Setup

a) Calculate the difference between the close of the current bar and the close of the previous bar. Find the 18-bar average of the difference between two consecutive closes. Add to this average two standard deviations of the difference of the closes over 18 bars.

b) Calculate the 18-bar average of the range and add to this average two standard deviations of the range over the last 18 bars.

Long Entries

a) Compare the difference between the two consecutive closes to the standard deviation calculation from the setup. If the difference is greater, place a buy stop at the close plus the three-bar average of the range (of one bar ago).

b) Compare the range to the standard deviation calculation from the setup. If the range is greater, place a buy stop at the close plus the three-bar average of the range (of one bar ago).

Short Entries

a) Compare the difference between the two consecutive closes to the standard deviation calculation from the setup. If the difference is greater, place a sell stop at the close minus the three-bar average of the range (of one bar ago).

b) Compare the range to the standard deviation calculation from the setup. If the range is greater, place a sell stop at the close minus the three-bar average of the range (of one bar ago).

Long and Short Exits

Exit at the first profitable open, which is calculated by comparing the open with the entry price (and accounting for a desired profit expressed in points and commission).

Designing & Formatting

This section presents the EasyLanguage instructions and formatting for the strategy, with the EasyLanguage instructions broken down and explained line by line.

EasyLanguage Strategy Components: Volatility Clusters (STAD13: VOL Clusters)

Strategy Inputs (STAD13: VOL Clusters)

INPUT	DEFAULT	DESCRIPTION
Deviations	2	The multiple of standard deviations to use in order to determine a volatility breakout
Length	15	The number of bars to use in calculating the averages and the standard deviation
PositionBasis	True	Used to determine if the Stop Loss exit criteria should be based on a position basis or a per contract basis
StopLoss	300	The strict amount of loss, in dollars, at which point the position will be closed
GoodOpenBase	0	The minimum amount of profit that needs to be achieved in order to generate an exit at the open of the next bar
GoodOpenDelay	0	The minimum number of bars that a trade must have in order to initiate the exit orders

Strategy Components:

1. *Volatility Clusters*
2. First Good Open
3. Stop Loss
4. Last Bar Exit

EasyLanguage Signal: Volatility Clusters:

```
Inputs: Deviations(2), Length(18);
Variables: CloseToClose(0), CTCDeviations(0), RangeDeviations(0), BigVolatility(False);

{Setup calculations}
CloseToClose = AbsValue(Close - Close[1]);
CTCDeviations = Average(CloseToClose, Length) + StdDev(CloseToClose, Length) * Deviations;
RangeDeviations = Average(Range, Length) + StdDev(Range, Length) * Deviations;

BigVolatility = CloseToClose > CTCDeviations[1] OR Range > RangeDeviations[1];

{Entry Orders}
```

```
If BigVolatility then Begin
        Buy next bar at Close + Average(Range, 3)[1] Stop;
        Sell next bar at Close - Average(Range, 3)[1] Stop;
End;
```

Signal Inputs (Volatility Clusters)

INPUT	DEFAULT	DESCRIPTION
Deviations	2	The multiple of standard deviations to use in order to determine a volatility breakout
Length	18	The number of bars to use in calculating the averages and the standard deviation

Signal Variables (Volatility Clusters)

INPUT	DEFAULT	DESCRIPTION
CloseToClose	0	[Numeric] Stores the difference between the Close prices of the last two bars
CTCDeviations	0	[Numeric] Stores the value used to compare the current Close difference to in order to determine a breakout
RangeDeviations	0	[Numeric] Stores the value used to compare the current bars Range to in order to determine a breakout
BigVolatility	False	[True/False] Used to evaluate that both the Close differ ences and the Range demonstrate a large move based on multiple standard deviations

Setup

For the *Volatility Clusters* setup, first the positive difference between Close prices is taken and stored into a variable. This can be done by taking the absolute value of the difference between the current and previous Close.

```
CloseToClose = AbsValue(Close - Close[1]);
```

An average of CloseToClose is taken, and added to it is Deviations (Input) times a standard deviation of the CloseToClose line. The standard deviation is calculated using the StdDev() function. The same is done for the Range values of the bar in order to make a comparison based on a breakout of more than Deviations times the standard deviation.

```
CTCDeviations = Average(CloseToClose, Length) + StdDev(CloseToClose, Length) * Deviations;
RangeDeviations = Average(Range, Length) + StdDev(Range, Length) * Deviations;
```

The condition of a breakout based on volatility is determined and stored in the True/False variable BigVolatility. BigVolatility is set to True if the current CloseToClose value is greater than the previous CTCDeviations calculation and the current Range is greater than the previous RangeDeviations calculation.

```
BigVolatility = CloseToClose > CTCDeviations[1] OR Range > RangeDeviations[1];
```

Long and Short Entry

Based on the evaluation of BigVolatility, Long and Short orders are placed on a Stop at the current Close plus or minus the average Range of the last three bars, respectively.

```
If BigVolatility then Begin
        Buy next bar at Close + Average(Range, 3)[1] Stop;
        Sell next bar at Close - Average(Range, 3)[1] Stop;
End;
```

EasyLanguage Signal: First Good Open:

** See Common Stops Appendix

EasyLanguage Signal: Stop Loss:

** See Common Stops Appendix

EasyLanguage Signal: Last Bar Exit

** See Common Stops Appendix

Testing and Improving

We tested our *Volatility Clusters* strategy (*VC*) on daily data for American Express (AXP) from 6/92 to 4/00 and Japanese Yen futures (JY) from 1/95 to 4/00. We set the Max number of bars strategy will reference to 50 and didn't deduct for slippage or commission. The default values and the testing protocol were as follows:

StopLoss = 300, testing 100-500 in increments of 100

Deviations = 2, testing 1-3 in increments of 1

Length = 15, testing 5-25 in increments of 5

GoodOpenBase = 0, testing 0-5 in increments of 1

GoodOpenDelay = 0, testing 0-5 in increments of 1

Let's take a look at our *Volatility Clusters* strategy results in AXP. Figure 3 shows *VC*'s most recent trade in AXP [Figure 3, AXP Chart]. The optimized values are as follows:

StopLoss = 300

Deviations = 1

Length = 5

GoodOpenBase = 4

GoodOpenDelay = 5

Figure 3. AXP Chart

Trading AXP, the *VC* strategy reaped a net profit of $15,335 on 62 trades [Figure 4, AXP Performance Summary]. Fifty-six percent of the trades were profitable, and the average winner ($683) was 2.15 times as large as the average loser ($317). The average trade earned $247, and the strategy won $2.79 for each $1.00 it lost.

The *VC* strategy traded profitably in all the years of the test period, except for 1994 in which it lost $175 (per 100 shares traded) [Figure 5, AXP Annual Trading Summary]. The Equity Curve is eerily consistent, rising at the same steady rate with only very minor equity dips throughout the test period [Figure 6, AXP Equity Curve].

The Underwater Equity Curve also shows that the *VC* strategy is good at limiting drawdowns; in fact, it's getting better. The worst drawdowns between 1994 and 1996 were about eight percent, while the worst drawdowns between 1998 and 2000 were only about four percent [Figure 7, AXP Underwater Equity Curve].

Figure 4. AXP Performance Summary

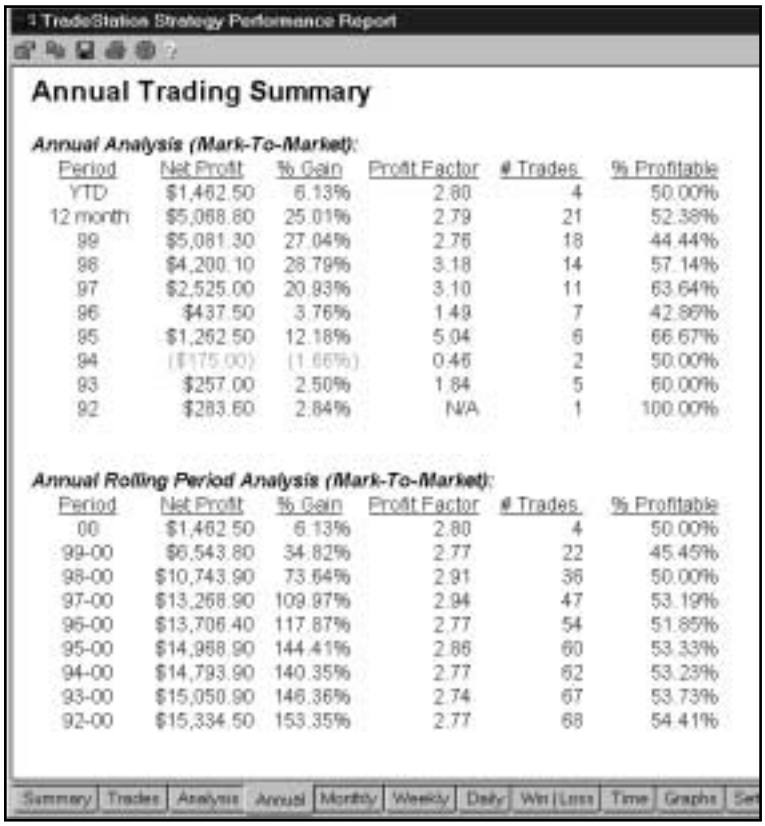


Figure 5. AXP Annual Trading Summary

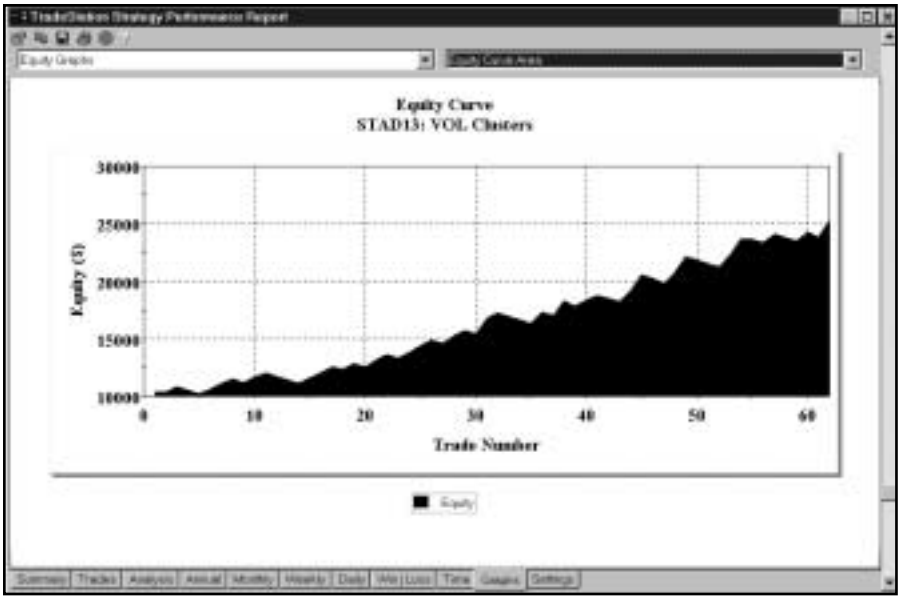


Figure 6. AXP Equity Curve

Figure 7. AXP Underwater Equity

Moving on to *VC*'s performance in the Yen, we see that our strategy earned $42,880 on 189 trades, of which 55% were profitable [Figure 8, JY Performance Summary]. The average trade made $227 per contract, and the strategy won $1.35 for each dollar it lost. The Equity Curve shows that the strategy earned about $50,000 in profits by trade 85, and that equity has been hovering around that level for quite a while [Figure 9, JY Equity Curve]. If the *VC* strategy doesn't make a new equity high in the near future, we'll want to reoptimize its inputs, beginning with data that corresponds to trade 85 and continues to the present, trying to get the strategy more in line with current price behavior in this market.

Figure 8. JY Performance Summary

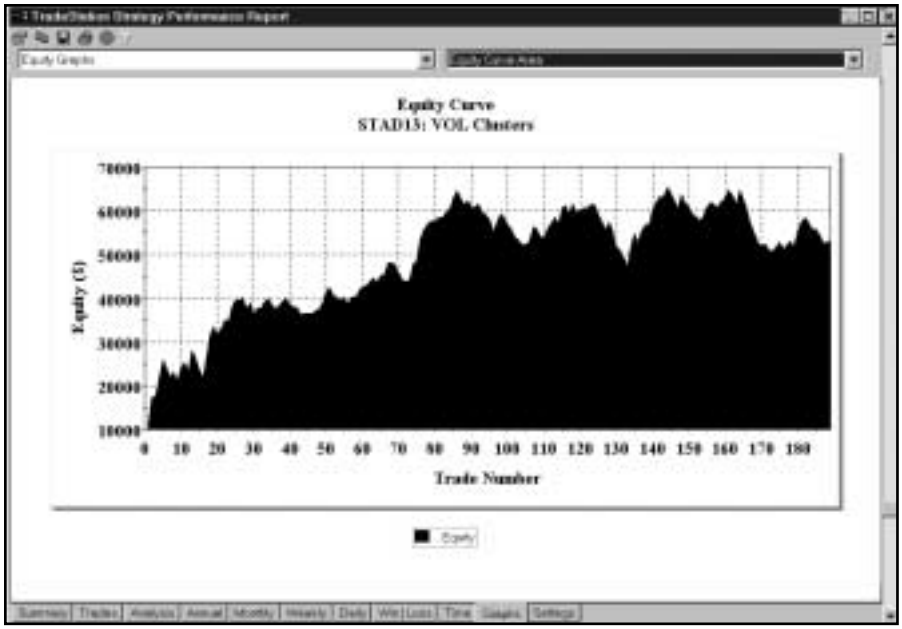


Figure 9. JY Equity Curve

Summary

Understanding that increased volatility tends to occur in clusters of bars rather than in isolated bars can prove useful even if we don't make *Volatility Clusters* the cornerstone of a strategy we actually trade. When we see a bar with significantly higher-than-normal volatility, we'll expect more of the same in the near future and plan our trades accordingly.

APPENDIX A

Common Stops

Common Exits

This section defines and explains the stops that are used more than once in the strategies presented in this issue. We hope that you will find this single reference chapter to be more convenient than repeated descriptions of each stop throughout the volume.

EasyLanguage Signal: ATR Big Profit Stop:

Applicable Strategies in this issue:

- **STAD13: LinReg - Mom**
- **STAD13: Mom-Retrace**

Signal EasyLanguage:

```
Inputs: BigProfitATRs(7), ATRLength(10), ExitBarLen(3);
Variables: ATRVal(0), PosHL(0);

ATRVal = AvgTrueRange(ATRLength) * BigProfitATRs;

If BarsSinceEntry = 0 Then
        PosHL = Close;

If MarketPosition = 1 Then Begin
        If Close > PosHL Then
                PosHL = Close;
        If PosHL > EntryPrice + ATRVal Then
                ExitLong Next Bar at Lowest(Low, ExitBarLen) Stop;
End;
```

```
If MarketPosition = -1 Then Begin
        If Close < PosHL Then
                PosHL = Close;
        If PosHL < EntryPrice - ATRVal Then
                ExitShort Next Bar at Highest(High, ExitBarLen) Stop;
End;
```

Signal Inputs (ATR Big Profit Stop)

INPUT	DEFAULT	DESCRIPTION
BigProfitATRs	7	The number of average true ranges used to determine the "Big Profit" level
ATRLength	10	The length parameter used to calculate the average true range
ExitBarLen	3	The length parameter used to determine the trailing stop after the "Big Profit" level has been achieved

Signal Variables (ATR Big Profit Stop)

INPUT	DEFAULT	DESCRIPTION
ATRVal	0	[Numeric] Used to store the price movement from EntryPrice that will determine that a "Big Profit" has been achieved
PosHL	0	[Numeric] Used to store the value of the highest / lowest Close during the position

Setup

The average true range is calculated and multiplied by BigProfitATRs (Input) in order to determine the "Big Profit" level. On the first bar of a position, the reserved word BarsSinceEntry will return zero. When this occurs, the variable PosHL is set to the Close of the current bar.

```
ATRVal = AvgTrueRange(ATRLength) * BigProfitATRs;

If BarsSinceEntry = 0 Then
        PosHL = Close;
```

Long Exit

As long as the market is in a Long position, there is a constant testing of PosHL compared to the current Close. If the current Close is greater than PosHL, then PosHL is reset to the value of the new Close, maintaining the highest Close value of the position in PosHL. The Long exit order is generated once the position has reached ATRVal points above EntryPrice. If the price activity has reached this level, a Long exit order at the lowest Low of the last ExitBarLen (Input) bars.

```
If MarketPosition = 1 Then Begin
        If Close > PosHL Then
                PosHL = Close;
        If PosHL > EntryPrice + ATRVal Then
                ExitLong Next Bar at Lowest(Low, ExitBarLen) Stop;
End;
```

Short Exit

When the market is in a Short position, there is a continual test of PosHL compared to the current Close. In contrast to the Long position, if the current Close is less than PosHL, then PosHL is reset to the value of the new Close, maintaining the lowest Close value of the position in PosHL. The Short exit order is generated once the position has reached ATRVal points below EntryPrice. If the price activity has reached this level, a Short exit order at the highest High of the last ExitBarLen bars.

```
If MarketPosition = -1 Then Begin
        If Close < PosHL Then
                PosHL = Close;
        If PosHL < EntryPrice - ATRVal Then
                ExitShort Next Bar at Highest(High, ExitBarLen) Stop;
End;
```

EasyLanguage Signal: ATR Breakeven Stop:

Applicable Strategies in this issue:

- **STAD13: LinReg - Mom**
- **STAD13: Mom-Retrace**

Signal EasyLanguage:

```
Inputs: ATRs(4), ATRLength(10);
Variable: ATRVal(0), PosHL(0);

ATRVal = AvgTrueRange(ATRLength) * ATRs;

If BarsSinceEntry = 0 Then
        PosHL = Close;
```

```
If MarketPosition = 1 Then Begin
        If Close > PosHL Then
                PosHL = Close;
        If PosHL > EntryPrice + ATRVal Then
                ExitLong ("1L") Next Bar at EntryPrice Stop;
End;

If MarketPosition = -1 Then Begin
        If Close < PosHL Then
                PosHL = Close;
        If PosHL < EntryPrice - ATRVal Then
                ExitShort ("1S") Next Bar at EntryPrice Stop;
End;
```

Signal Inputs (ATR Breakeven Stop)

INPUT	DEFAULT	DESCRIPTION
ATRs	4	The Floor value, the number of Average True Ranges above/below the Entry Price at which the Stop becomes active for the position
ATRLength	10	Length, expressed in bars, used to calculate the Average True Range

Signal Variables (ATR Breakeven Stop)

INPUT	DEFAULT	DESCRIPTION
ATRVal	0	[Numeric] Holds the value of the Average True Range multiplied by the number of Trailing ATRs
PosHL	0	[Numeric] Holds the value of the highest/lowest Close of the position

Setup

In the Setup portion of the signal, the Average True Range is calculated and multiplied by the number of 'ATRs' specified in the Inputs.

```
ATRVal = AvgTrueRange(ATRLength) * ATRs;
```

On the first bar of the position, when the 'BarsSinceEntry' is equal to 0, the 'PosHL' variable is assigned the Close value. This resets the tracking of the position highest/lowest Close of the position, based on the direction of the position.

```
If BarsSinceEntry = 0 Then
        PosHL = Close;
```

Long Exit

Once a Long position is taken, we must evaluate the highest closing price of the position and the Floor value established by the Average True Range. First, a comparison between the Close and the 'PosHL' Variable is made. During a Long position the 'PosHL' variable represents the highest Close of the position. Thus, if the Close is greater than the 'PosHL' value, the Close value is assigned to the 'PosHL' variable as the new highest Close. Next, If the highest Close of the position (PosHL) exceeds the sum of the 'EntryPrice' and the specified Average True Range (the Floor value), a Long Exit Stop order is placed at the entry price (breakeven price).

```
If MarketPosition = 1 Then Begin
        If Close > PosHL Then
                PosHL = Close;
        If PosHL > EntryPrice + ATRVal Then
                ExitLong ("1L") Next Bar at EntryPrice Stop;
End;
```

Short Exit

Once a Short position is taken, we must evaluate the lowest closing price of the position and the Floor value established by the Average True Range. First, a comparison between the Close and the 'PosHL' Variable is made. During a Short position the 'PosHL' variable represents the lowest Close of the position. Thus, if the Close is less than the 'PosHL' value, the Close value is assigned to the 'PosHL' variable as the new lowest Close. Next, If the lowest Close of the position (PosHL) falls below the difference between the 'EntryPrice' and the specified Average True Range (the Floor value), a Short Exit Stop order is placed at the entry price (breakeven price).

```
If MarketPosition = -1 Then Begin
        If Close < PosHL Then
                PosHL = Close;
        If PosHL < EntryPrice - ATRVal Then
                ExitShort ("1S") Next Bar at EntryPrice Stop;
End;
```

EasyLanguage Signal: ATR Protective Stop:

Applicable Strategies in this issue:

- **STAD13: LinReg - Mom**
- **STAD13: Mom-Retrace**

Signal EasyLanguage:

```
Inputs: ProtectiveATRs(3), ATRLength(10);
Variable: ATRVal(0);

ATRVal = AvgTrueRange(ATRLength) * ProtectiveATRs;

If MarketPosition = 1 Then
        ExitLong Next Bar at EntryPrice - ATRVal Stop;

If MarketPosition = -1 Then
        ExitShort Next Bar at EntryPrice + ATRVal Stop;
```

Signal Inputs (ATR Protective Stop):

INPUT	DEFAULT	DESCRIPTION
ProtectiveATRs	3	The number of Average True Ranges that are risked in the position
ATRLength	10	Length, expressed in bars, used to calculate the Average True Range

Signal Variables (ATR Protective Stop):

INPUT	DEFAULT	DESCRIPTION
ATRVal	0	[Numeric] Holds the value of the Average True Range, multiplied by the ProtectiveATRs

Setup

In the Setup portion of the signal, the Average True Range is calculated and multiplied by the number of ProtectiveATRs specified in the Inputs.

```
ATRVal = AvgTrueRange(ATRLength) * ProtectiveATRs;
```

Long Exit

When the market position is Long, a Long Exit is placed at the entry price minus the Protective Volatility Average True Range calculation (ATRVal).

```
If MarketPosition = 1 Then
        ExitLong Next Bar at EntryPrice - ATRVal Stop;
```

Short Exit

When the market position is Short, a Short Exit is placed at the entry price plus the Protective Volatility Average True Range calculation (ATRVal).

```
If MarketPosition = -1 Then
        ExitShort Next Bar at EntryPrice + ATRVal Stop;
```

EasyLanguage Signal: ATR Trailing Stop

Applicable Strategies in this issue:

- **STAD13: LinReg - Mom**
- **STAD13: Mom-Retrace**

Signal EasyLanguage:

```
Inputs: TrailingATRs(4), ATRLength(10);
Variables: PosHigh(0), PosLow(0), ATRVal(0);

ATRVal = AvgTrueRange(ATRLength) * TrailingATRs;

If MarketPosition = 1 Then Begin
        If BarsSinceEntry = 0 Then
                PosHigh = High;
        If High > PosHigh Then
                PosHigh = High;
        ExitLong Next Bar at PosHigh - ATRVal Stop;
End;

If MarketPosition = -1 Then Begin
        If BarsSinceEntry = 0 Then
                PosLow = Low;
        If Low < PosLow Then
                PosLow = Low;
        ExitShort Next Bar at PosLow + ATRVal Stop;
End;
```

Signal Inputs (ATR Trailing Stop)

INPUT	DEFAULT	DESCRIPTION
TrailingATRs	4	The number of average true ranges that are risked from the highest/lowest price of the position
ATRLength	10	The length parameter used to calculate the average true range

Signal Variables (ATR Trailing Stop)

VARIABLE	DEFAULT	DESCRIPTION
PosHigh	0	[Numeric] Used to store the highest value of the position
PosLow	0	[Numeric] Used to store the lowest value of the position
ATRVal	0	[Numeric] Used to store the trailing value from the high / low of the position

Setup

ATRVal stores TrailingATRs (Input) times the average true range of the last ATRLength (Input) number of bars.

```
ATRVal = AvgTrueRange(ATRLength) * TrailingATRs;
```

Long Exits

In a Long position, PosHigh is initialized on the first bar to the High of the bar. Each subsequent High is compared in order to keep track of the highest High of the position. On each bar, the Long exit order is placed at PosHigh minus ATRVal on a stop.

```
If MarketPosition = 1 Then Begin
        If BarsSinceEntry = 0 Then
                PosHigh = High;
        If High > PosHigh Then
                PosHigh = High;
        ExitLong Next Bar at PosHigh - ATRVal Stop;
End;
```

Short Exits

In a Short position, PosLow is initialized on the first bar to the Low of the bar. Each subsequent Low is compared in order to keep track of the lowest Low of the position. On each bar, the Short exit order is placed at PosLow plus ATRVal on a stop.

```
If MarketPosition = -1 Then Begin
        If BarsSinceEntry = 0 Then
                PosLow = Low;
        If Low < PosLow Then
                PosLow = Low;
        ExitShort Next Bar at PosLow + ATRVal Stop;
End;
```

EasyLanguage Signal: ATR Volatility Stop:

Applicable Strategies in this issue:

- **STAD13: LinReg - Mom**
- **STAD13: Mom-Retrace**

Signal EasyLanguage:

```
Inputs: VolatilityATRs(2), ATRLength(10);
Variable: ATRVal(0);

ATRVal = AvgTrueRange(ATRLength) * VolatilityATRs;

If MarketPosition = 1 Then
        ExitLong Next Bar at EntryPrice - ATRVal Stop;

If MarketPosition = -1 Then
        ExitShort Next Bar at EntryPrice + ATRVal Stop;
```

Signal Inputs (ATR Volatility Stop)

INPUT	DEFAULT	DESCRIPTION
VolatilityATRs	2	The number of average true ranges that are used to determine the volatility stop
ATRLength	10	The length parameter used to calculate the average true range

Signal Variables (ATR Volatility Stop)

INPUT	DEFAULT	DESCRIPTION
ATRVal	0	[Numeric] Used to store the value of the volatility stop

Setup

ATRVal stores VolatilityATRs (Input) times the average true range of the last ATRLength (Input) number of bars.

```
ATRVal = AvgTrueRange(ATRLength) * VolatilityATRs;
```

Long Exit

A Long exit order is placed at EntryPrice minus ATRVal from a Long position.

```
If MarketPosition = 1 Then
        ExitLong Next Bar at EntryPrice - ATRVal Stop;
```

Short Exit

A Short exit order is placed at EntryPrice minus ATRVal from a Short position.

```
If MarketPosition = -1 Then
        ExitShort Next Bar at EntryPrice + ATRVal Stop;
```

EasyLanguage Signal: First Good Open:

Applicable Strategies in this issue:

- **STAD13: VOL Clusters**
- **STAD13: VOLEX**

Signal EasyLanguage:

```
Inputs: BaseProfit(0), Delay(0);
Variables: NextBarOpen(0), MP(0);

NextBarOpen = Open of next bar;
MP = MarketPosition;

{ Exit at first Profitable Open }
If BarsSinceEntry > Delay Then Begin
        If MP = 1 AND NextBarOpen > EntryPrice + (BaseProfit / BigPointValue) Then
                ExitLong ("Long Profit") next bar at Market;
        If MP = -1 AND NextBarOpen < EntryPrice - (BaseProfit / BigPointValue) Then
                ExitShort ("Short Profit") next bar at Market;
End;
```

Signal Inputs (First Good Open)

INPUT	DEFAULT	DESCRIPTION
BaseProfit	0	The minimum amount of profit that needs to be achieved in order to generate an exit at the open of the next bar
Delay	0	The minimum number of bars that a trade must have in order to initiate the exit orders

Signal Variables (First Good Open)

INPUT	DEFAULT	DESCRIPTION
NextBarOpen	0	[Numeric] Used to store the value of the open of the next bar
MP	0	[Numeric] Used to store the current MarketPosition (1 for Long, -1 for Short, 0 for no position)

Setup

The open of the next bar and MarketPosition are stored into variables to allow for fewer function calls and to allow for reference on a bar by bar basis.

```
NextBarOpen = Open of next bar;
MP = MarketPosition;
```

Long and Short Exits

The exits in this signal wait for a position to be maintained for Delay (Input) number of bars. Once the number of bars have been passed, the exit should occur when the strategy is Long and the next bar opens high enough to generate a base profit amount. If the position is Short, the next bar needs to open low enough to generate the desired profit. In EasyLanguage, MP reflects the position of the market, and the required price of the Open can be determined by dividing the dollar amount of desired profit by the BigPointValue and adding or subtracting this value from the EntryPrice. If the Open of the next bar meets the requirements, the position is exited.

```
{ Exit at first Profitable Open }
If BarsSinceEntry > Delay Then Begin
        If MP = 1 AND NextBarOpen > EntryPrice + (BaseProfit / BigPointValue) Then
                ExitLong ("Long Profit") next bar at Market;
        If MP = -1 AND NextBarOpen < EntryPrice - (BaseProfit / BigPointValue) Then
                ExitShort ("Short Profit") next bar at Market;
End;
```

EasyLanguage Signal: Last Bar Exit:

Applicable Strategies in this issue:

• STAD13: Adv-Dec Div

• STAD13: BondCurrency

• STAD13: LinReg - Mom

• STAD13: LUXOR

• STAD13: Mom-Retrace

• STAD13: OC Histogram

• STAD13: RangeLeaders

• STAD13: Vol Clusters

• STAD13: VOLEX

Signal EasyLanguage:

```
If LastBarOnChart Then Begin
ExitLong This Bar on Close;
        ExitShort This Bar on Close;
End;
```

This Signal does not contain any Inputs or Variables.

Long/Short Exits

On the last bar of the chart, any Long or Short positions are closed out in order to insure that all trades are included in the System Report.

```
If LastBarOnChart Then Begin
        ExitLong This Bar on Close;
        ExitShort This Bar on Close;
End;
```

EasyLanguage Signal: Stop Loss:

Applicable Strategies in this issue:

• STAD13: Adv-Dec Div

• STAD13: RangeLeaders

• STAD13: VOL Clusters

• STAD13: VOLEX

Signal EasyLanguage:

```
Inputs: PositionBasis(True), Amount(0);

If PositionBasis Then
        SetStopPosition
Else
        SetStopContract;

SetStopLoss(Amount);
```

Signal Inputs (Stop Loss)

INPUT	DEFAULT	DESCRIPTION
PositionBasis	True	Used to determine if the Stop Loss exit criteria should be based on a position basis or a per contract basis
Amount	0	The strict amount of loss, in dollars, at which point the position will be closed

This Signal does not contain any Variables.

Setup

The Input PositionBasis will determine if the exit will be based on the value of the entire position, or on a per contract basis.

```
If PositionBasis Then
        SetStopPosition
Else
        SetStopContract;
```

Long and Short Exits

The SetStopLoss() function will generate an exit when the position reaches a loss equal to or greater than Amount (Input).

```
SetStopLoss(Amount);
```

APPENDIX B

Volume In Review

Here's a question about optimizing from a STAD Club member:
I see the STAD club studies very often will come up with different optimized values for each equity illustrated. Isn't that "Curve-fitting." Shouldn't the system to be valuable be profitable among many equities using the same variables?

STAD Club does optimize for the best values for each stock or commodity. If we tested several stocks or commodities for each strategy and published the optimized values, we'd add the step of Universalization. As many of you know, Universalization deletes the highest and lowest values for each optimized parameter and averages the remaining values to obtain a robust value that can be applied to all the stocks or commodities on which the strategy was tested. Since we usually just test and optimize one or two stocks per strategy as examples, the Universalization strategy can't be implemented. I'd recommend that you apply any STAD strategies that catch your attention to several of the markets you like to trade and Universalize the resulting values. You can find an article about Universalization in STAD Club volume 13. Thanks for your question.

Another club member sent this question about volatility stops:
Why is the Close used to anchor the Volatility Stop, described in the "First Prize" strategy in STAD v.12? Subtracting 3 ATRs from the Close makes the stop dependent on where in the bar the Close occurred, instead of on the Range of that bar. Is a Close that is well above the Low of the bar supposed to represent a dramatic upswing off the Low? If two bars had identical Ranges, one with a Close well above the Low could trigger the stop while the bar with a Close near the Low would not. Thank you.

Anchoring the volatility stop on the previous bar's close is one logical choice out of several good possibilities. With the volatility stop, we are just making sure we exit a position if there's an abnormally large price move against us on a single bar. Other stops take care of other circumstances, such as a gradual erosion of our open profits. The volatility stop could just as well be calculated from the previous bar's high, low, or median price as from the close. In fact, that's a great idea for testing. What volatility stop works best with your favorite strategy — high minus n ATRs, low - n ATRs, close - n ATRs, or median price - n ATRs (for a long position)? Come to think of it, the current bar's OPEN - n ATRs might even make a good volatility stop for a long position. Let's check it out.

INDEX