Chapter 3 — Trading System Development

The trading system development procedure is an extensive in-sample data mining and (hopefully) learning process:

• Generate many alternative trading systems (ATS), each based on a specific model-data combination.
• Evaluate each ATS, giving each a score computed by the objective function.
• Rank the ATSs, preferring those with high scores, selecting one.

Followed by limited (ideally one time only) testing of the selected system using out-of-sample data to validate that the system has learned.
Figure 3.1 shows the flowchart of trading system development.

![Flowchart of Trading System Development](image)

**The Model**

Central to the development is the model. The model’s purpose is to identify patterns that precede profitable trades. First to learn to identify those patterns using historical data. Then to recognize those same patterns in real-time data.

Input to the model is price and volume data representing trades. Output is a series of signals telling the trader when to be long, flat, and short. Internally, the model is a set of mathematical relationships that transform input to output—price and volume to signals, then to trades.

The model typically includes formulas that manipulate the raw data, consolidating and aligning it as necessary, transforming it and creating indicators from it to increase its ability to detect patterns. Patterns are
filtered through a set of rules to generate the signals, then reported as trades. The scientific method of data-driven adjustment and validation is clearly present.

Models come in many varieties. Traditional development platforms have a limited number of model paradigms built in to them. The most common is decision tree, where a tree-like structure of comparisons and questions leads to a specific value of the signal for that data bar. Machine learning adds other model paradigms that can be chosen and used, such as linear regression, neural network, and support vector machines.

All models are, or at least can be expressed as, a set of mathematical operations. Generation of signals is essentially a linear algebra task involving solution of a set of simultaneous equations.

**Models and the Premises**

The underlying assumptions of technical analysis were listed in Chapter 1. Some commentary and a diagram might help understanding how the assumptions fit into the development process. Going point by point with each assumption, then explanation:

🌟 The markets we model are sufficiently inefficient for us to make a profit trading them.

Our success in profitable trading will tell us our answer. If we are able to develop a profitable trading system, that is evidence that the market is not efficient.

🌟 There is information—patterns—in the historical price series that can be used to identify profitable trading opportunities.

There are two parts to this—profitable trades and patterns. Begin with profitable trades.

Figure 3.2 shows the daily close of SPY for a period of six months. Assume we plan to trade SPY, taking a long position at the close of those days when tomorrow’s close will be higher than today’s close. The periods of rising closing prices are the profitable trades we want the model to identify. They are marked with the larger circles.
If patterns that precede those trades do exist, the model must find them.

Trading systems can be designed to recognize the patterns and give buy and sell signals.

The patterns might be found in or derived from the price data series, or they might come from the other auxiliary data series. This is the “data in”, “recognize pattern”, “signal out” part of the system. Some reasons why it is not easy are:

- The data being used might not contain an identifiable pattern that precedes the trades you want to identify.
- The data is noisy. We are searching for a weak signal in a noisy background. See the section “The Signal and the Noise” in Chapter 1.
- The data might change characteristics over the period we are searching. See the section “Stationarity” in Chapter 1.
- The data is ambiguous. Given a particular pattern, say the crossing of an RSI indicator with a given level, in some cases the next day the price is rising, and in others the next day it is falling. See the section “Indicators” below.
- The model might overfit the in-sample data used during development, memorizing the data but failing to learn, then be unable to identify profitable patterns in out-of-sample data.
- The model might need modification. It might need to be simpler, more complex, or use a different learning technique.

Patterns similar to those found in the historical data will continue to be found in future data.
No matter how accurately the model detects the signals that precede profitable trades during development, we cannot make profitable trades unless the future resembles the past. See the section “Stationarity” in Chapter 1.

What I Would Like to Know

A model is based on an idea. When I am leaving on a trip, I know the destination, and I have an estimate of the route, travel time, etc, before leaving the driveway. Similarly, my trading model begins with an idea and I use that to guide me.

Not all ideas are achievable. I might want a system where I can buy one stock or fund, hold it long enough to qualify for favorable tax benefits, minimize attention to the holding, have a high return, and have low drawdown. Much as I might want that, it is not achievable.

A more practical approach is to ask myself a question, the answer to which would help develop an achievable system, then answer the question and build the model accordingly.

For example.

Q. What would I like to know about tomorrow? What would I do if I knew?

A. I would like to know whether tomorrow’s closing price for the issue, yet to be selected, will be higher than today’s closing price. If I knew, I would trade at the close of the trading day. I would use daily data available at the close of trading, taking a long position when the signal indicated a rise, remaining flat when it did not. I would reevaluate each day, holding long when I received successive positive signals, going flat upon receiving the first non-positive signal.

My system development would be guided in an attempt to gather information needed to answer the question.

Unfortunately, some combinations of risk, reward, and trading style we might want are just not possible. The risk assessment will let us know.

Traditional Development Platforms

Until a few years ago, trading system models were developed nearly exclusively using development platforms—special purpose computer programs specifically created for trading. These platforms have features and functions built into them to handle data, display charts, compute indicators, build mathematical relationships, generate signals, create trades, and report performance.

- The model is decision tree.
The paradigm is “compute an indicator, then see what follows.” The signals they produce are impulses that identify categories—buy, sell, short, cover. See the section “Impulse and State Signals,” below.

Much of the early literature describing these models assumed:
- The patterns were fairly well defined and stood out from the noise.
- Successful indicators had long lookback periods and changed slowly.
- Holding periods for trades were relatively long.

Additionally, many analysts made unrealistic assumptions:
- Traditional wisdom does not require validation.
- A small number of anecdotal examples provide valuable forecasting.
- Relationships were, and would continue to be, stationary.
- Systems that worked once will continue to work indefinitely.
- Systems will always recover from drawdowns.
- Drawdowns are opportunities to increase position size and recover lost funds.

Figure 3.3, a repeat of Figure 1.4, is a schematic showing a trading system model of the type associated with traditional trading system development platforms. Examples of this type of platform are AmiBroker, NinjaTrader, and TradeStation.
Machine Learning Platforms

As markets are becoming more efficient, patterns are becoming harder to distinguish. Trading is more frequent, and holding periods shorter. Machine learning techniques are being adapted to trading. Machine learning gives new ways to identify profitable patterns. Ways that are more accurate and less prone to overfitting.

These are still the early days in machine learning platforms that include tools specific to trading. Data handling, charting, and functions related to analysis of trades are still somewhat rudimentary and the platforms lack the full set of features trading system developers want. These limitations are compensated for by the broad variety of models available, and the ability to perform multiple phase simulations in the same program run as pattern recognition.
• The model is supervised learning using any of the twenty or more machine learning techniques, including decision trees, neural networks, support vector machines, and random forests.
• The mathematical relationships are systems of simultaneous equations.
• The paradigm is “identify desirable trades, then see what happened earlier.”
• The signals are categories representing states—beLong, beFlat, beShort.

**Interchangeability**

Figure 3.4 shows how the two—indicator-based development using traditional platforms and machine learning development—relate to each other and fit into model development. Techniques used for issue selection and data preparation are the same. At the conclusion of validation, both produce a best estimate set of trades. The similarity should not be surprising, since decision tree is one of the machine learning techniques.

![Figure 3.4 Two paths for model development](image)

The choice of which development platform to use is yours. Whether you decide to use a traditional trading system development platform or machine learning, use the scientific method. Figure 3.5 is a repeat of Figure 1.1.
Impulse and State Signals

The system represented by the schematic in Figure 3.3 above uses impulse signals to identify entry into and exit from a trade. The alternative is state signals.

Impulse signals identify transitions, such as the beginning or end of a trade. They arise from an event that occurs on a single bar—such as
when an RSI indicator falls through a critical level. An impulse signal occurs once on the bar when the rules of the model are satisfied—one impulse for the entry, and one impulse for the exit. In general, there are no additional signals between the entry and the exit. (Although there could be additional entry signals.) Impulse signals are associated with actions—buy, sell, short, and cover.

Impulse signals define trades. In evaluation of the system’s trades, a data point is a *trade*—however many days that trade lasts.

State signals are also defined by the rules of the model. While impulse signals identify boundaries, state signals identify conditions—such as whether the position to be held for the next day is long, flat, or short. There is a state signal for every day.

Given the same indicators and patterns, trades signalled by state signals and marking to market daily hold positions for the same period as trades signalled by impulse signals. In the evaluation of the system’s trades when using state signals, a data point is a *day*—however many days that trade lasts.

The two are equivalent for many purposes. We can easily convert from impulse to state by changing state at the first occurrence of an impulse, then holding that same state until the first occurrence of an impulse that changes to a new position.

Your trading system development platform probably has functions that allow you to create state or impulse signals, as you wish, and to convert between them.

The bar-by-bar account equity is identical whether a trade is denoted by impulse or state signals. Refer to the section “Mark-to-Market Equivalence” in Chapter 2.

Figure 3.6 illustrates a multi-day trade with corresponding state and impulse signals.
Changing from impulse signals to state signals changes the way we view trades—from some number of bars between the buy and sell impulses, where there is no action and no opportunity for trade management, to a series of single-day marked-to-market trades.

When a sequence of days has the same state, the position is held over and appears as a single multi-day trade. With state signals, there will be as many trade-related data points as days the system holds an open position.

Using state signals, every day is evaluated on its own. With the exception of trades that are open across the boundaries of the test period, performance results are identical.

There are significant advantages to using state signals over impulse signals.

- A state signal is generated for each and every bar, giving the trader a clear indication of the correct position for the next bar, and coordinating measurement of the system with management of the system.
- During testing, a period of time is defined and trades that occur within that period are evaluated. Only those trades that are completely within the test period can be correctly and accurately evaluated.

The edges of the test period create distortions. Because indicators, parameters, and rules often change as a new test period is entered, the platform cannot determine that a trade was open in the previous period and should be continued. The first trade of each period begins with the first new signal of the period. Similarly, a trade that is open at the end of a test period can either be artificially closed or completely ignored.

When impulse signals are used, periods where a trade was held at either end of a test period are not accurately evaluated.

When state signals are used, there is at most one day lost at each end of the test period.

- Impulse signals work well with traditional development platforms because the platforms were developed with the expectation that trades would identify buy and sell points. State signals work equally well with traditional platforms.
- State signals are particularly well suited for use with machine learning platforms where it is important to have a target value for each bar to guide the learning.
- An advantage to state signals is increased opportunity for control of trade management throughout the trade. The health of the system is reevaluated every day at the same time it is
marked-to-market. This is also the frequency that position size is readjusted using the dynamic position sizing technique used to calculate safe-f for the next trade.

Learning

Learning is the process of examining data, recognizing some patterns, observing related patterns, resulting in generalization—the ability to recognize patterns in data not previously seen. As it applies to trading systems, we will be looking for patterns that provide either:

- **Classification.** Buy or Sell. Either to open a new position or to close an existing position. The issue of how much to buy or sell—position sizing—is addressed separately from pattern recognition.
- **Estimation.** Direction and magnitude of change anticipated. If an estimation technique is used in the trading system, we may apply a threshold filter to convert the estimation into a classification category. Estimation is used in the trading management system to specify the position size for the next trade.

Learning is not possible unless there is data to be examined and patterns to find. Preferably a lot of data and a lot of examples. This is definitely a data mining activity. The data mined is called the in-sample data. We are searching for patterns within the historical price data that precede profitable trades.

We cannot learn a feature that has not been seen. The in-sample data must include examples of the patterns to be learned.

Identifying patterns in the in-sample data is necessary for learning, but is not sufficient. There must be generalization. The test for generalization is validation. That is, testing previously unused data to estimate the success of detecting the patterns and defining the rules. The data tested for validation is called the out-of-sample data.

Validation is the step designed to provide the confidence requested in the goal.

Objective Function

Objective functions are extremely important both in trading system development and later in trading management. There are many millions of combinations of data series, rules, and parameters, each one of which defines a trading system. Which is best?

The trading system development process consists of generating many ATSs—combinations of data series, indicators, rules, and parameters—and choosing the best from among them.