1 Parameter Tuning in Trading Algorithms Using ASTA

Thomas Hellström[1]
Department of Computing Science
Umeå University
SE-901 87 Umeå, Sweden

Kenneth Holmström[2]
Center for Mathematical Modeling (CMM)
Department of Mathematics and Physics
Mälardalen University
P.O. Box 883
SE-721 23 Västerås, Sweden

This paper describes ASTA, an Artificial Stock Trading Agent, in the Matlab programming environment. The primary purpose of the project is to supply a stable and realistic test bench for the development of multi-stock trading algorithms. The behavior of the agent is controlled by a high-level language, which is easily extendable with user-defined functions. The buy and sell rules can be composed interactively and various types of data screening can be easily performed, all within the Matlab m-file language syntax.

Apart from being a Windows-based test bench for trading algorithms, the system can be also run in batch mode, where a supplied objective function maps a trading strategy to a profit measure. This can be used to tune parameters or to automate the development of trading strategies, for example with genetic methods. Examples of tuning parameters in standard technical indicators are presented using new techniques for global optimization available in the optimization environment TOMLAB. A modified Sharpe Ratio is used as a performance measure.

To improve the performance of a given algorithm, the data from the simulated trades can be output and post-processed by classification methods such as artificial neural networks or fuzzy rule bases.

The ASTA system has been applied successfully to historical stock data, and results covering 11 years of the Swedish stock market are presented.

Keywords: trading, artificial trader, technical analysis, data mining, prediction, stock returns, Sharpe ratio, parameter tuning, global optimization.

---

1. thomash@cs.umu.se; http://www.cs.umu.se/ thomas. Research financed by the Applied Optimization and Modeling project, Mälardalen University, Västerås, Sweden.

2. hkh@mdh.se; http://www.ima.mdh.se/tom.

1.1 Introduction

The idea of expressing stock prediction algorithms in the form of trading rules has gained considerable attention in academic research in the last years. The international conference NNCM-96 devoted a whole section in the proceedings to "Decision Technologies". [Bengio 1997] writes about the importance of training artificial neural networks with a financial criterion rather than a prediction criterion. [Moody and Wu 1997] use reinforcement learning to train a trading system with objective functions such as profit, economic utility and Sharpe ratio. [Atiya 1997] describes a trading system based on time-variable stop-losses and profit objectives.

This paper describes the system ASTA, which is an implementation of an Artificial Stock Trading Agent. With ASTA, trading-rule-based prediction algorithms are easily evaluated using historical data. ASTA performs a simulation of multi-stock trading, where trading rules are executed for a large number of available stocks every day in the simulation period. The situation is fundamentally different from the single-stock prediction case.

Besides being an evaluation tool, ASTA is also a development tool where trading rules can be combined and tuned. Examples of tuning parameters in the technical indicator Stochastics are presented.

The program is developed in the Matlab programming language and is used either as an ordinary objective function called from a user's program, or as an interactive tool for making benchmarks and development of trading algorithms. The ASTA system is thoroughly described in [Hellström 1998a] and in the present paper we only give a short introduction to the system.

1.1.1 The Usage of ASTA

The development of ASTA was instigated by a need for good working tools for the following research tasks:

1. *A test bench for trading algorithms.*
Many "technical indicators" for stock prediction are accepted and widely used without having ever been subject to an objective scientific analysis with historical stock data. It is true that many commercial software packages for technical analysis offer both a comprehensive programming language and a simulation mode where the performance can be computed. However, most available products do not take this task very seriously and real trading simulations with a multi-stock portfolio are seldom possible.

2. *An interactive development tool for trading rules.*

There are reasons to believe that a successful trading system consists of many disjunct parts where a buy signal may be, for example, "screened" by looking at the traded volume. A buy signal issued with a low traded volume may then be rejected. Other composite rules include looking at the general trend of the stock before accepting a signal from the system. ASTA provides the possibility to test such composite rules easily.

3. *A non-interactive development tool for trading rules.*
Furthermore, it is possible and maybe also fruitful to automate the development of trading rules. Since ASTA defines the trading strategy as symbolic Buy rules and Sell rules given as arguments to the system, it would be perfectly possible to construct buy and sell rules in a genetic framework, for example.
Even if the general look of the algorithm is fixed, there are often a lot of tunable parameters that affect the trading performance. Examples are filter coefficients, order of polynomials and levels above or below which an entity should pass to generate a trading signal. Since we believe that the actual behavior during a realistic trading situation is essential for proper selection and optimization of an algorithm, there is a need for an objective function that can be included in an optimization phase for parameter tuning.

4. *A data generating tool for post processing.*
The comprehensive and user-friendly macro language in ASTA makes it a very suitable tool for extracting data for further analysis, such as classification with neural networks or fuzzy rule bases.

1.2 Basic Approaches to Stock Predictions

Prediction algorithms for stock prices can be categorized in a number of ways. One categorization focuses on the way the points to predict are selected. Two broad classes can be identified; "The Time Series Approach" and "The Trading Simulation Approach"

1.2.1 The Time Series Approach

The traditional way to define a stock prediction problem is to view the stock returns as a time series $y(t)$. For example, one-day stock returns are defined as

$$y(t) = \frac{Close(t) - Close(t-1)}{Close(t-1)}. \tag{1.1}$$

To predict the return h days in the future, $y(t+h)$ is assumed to be a function g of the previous (lagged) values

$$y(t+h) = g(y(t), y(t-1), ..., y(t-k)). \tag{1.2}$$

The task for the learning or modeling process is to find the function g that best approximates a given set of measured data. The unknown function g can be chosen in many ways, e.g. as linear autoregressive (AR) models or feed-forward neural networks. The unknown parameters in the model are normally computed by a learning (identification) algorithm that minimizes the root mean square prediction error

$$RMSE = \sqrt{\frac{1}{N}\sum_{t=1}^{N} (g(t) - y(t+h))^2}. \tag{1.3}$$

It is most common to let the minimized RMSE measure (1.3) be the end point in the prediction task. However, to utilize the predictions, a decision-taking rule has to be created. A simple rule often used is

$$D(t) = \left\{ \begin{array}{ll} \text{Buy} & : \text{ if g(t)} > \alpha \\ \text{Sell} & : \text{ if g(t)} < -\beta \\ \text{Do nothing} & : \text{ otherwise} \end{array} \right\}, \tag{1.4}$$

where α and β are positive valued threshold parameters for buy and sell actions depending on the predicted change in the stock price.

The time series formulation based on the minimized RMSE measure (1.3) is not always ideal for useful predictions of financial time series. Some reasons are:

1. The fixed prediction horizon h does not reflect the way in which financial predictions are being used. The ability of a model to predict should not be evaluated at one single fixed point in the future. A big increase in a stock value 14 days into the future is as good as the same increase 15 days into the future!

2. The equation (1.3) treats all predictions, small and large, as equal. This is not always appropriate. Prediction points that would never be used for actual trading (i.e. price changes too small to be interesting) may cause higher residuals at the other points of more interest, to minimize the global RMSE.

3. A small predicted change in price, followed by a large real change in the same direction, is penalized by the RMSE measure. A trader is normally happy in this case, at least if, say, the small positive prediction was large enough to give a buy signal.

4. Several papers report a poor correlation between the RMSE measure and the profit made by applying a prediction algorithm, e.g. [Leitch and Tanner 1991] and [Bengio 1997]. A strategy that separates the modeling from the decision-taking rule, such as the one in 1.4, is less optimal than modeling the decision taking directly [Moody 1992]. Arguments 2 and 3 both give some explanations to these results.

1.2.2 The Trading Simulation Approach

Instead of separating the prediction task and the decision task as was done in the "Time Series Approach", algorithms can be constructed to recognize situations where one should buy and sell stocks respectively (The approach in the ASTA system). A trading rule can be described as a time series $T(t)$ defined as

$$T(t) = \left\{ \begin{array}{lll} \text{Buy} & : & \text{if } g(\mathbf{X}(t)) > 0 \\ \text{Sell} & : & \text{if } g(\mathbf{X}(t)) < 0 \\ \text{Do nothing} & : & \text{otherwise} \end{array} \right\}. \tag{1.5}$$

The unspecified function g determines the type of trading rule. The argument $\mathbf{X}(t)$ has the form

$$\mathbf{X}(t) = (R_1(t), ..., R_N(t)), \tag{1.6}$$

where each $R_k(t)$ is an observable feature at time t. In the case of stock predictions it may be for example the k-day returns defined as

$$R_k(t) = 100 \cdot \frac{Close(t) - Close(t-k)}{Close(t-k)} \tag{1.7}$$

or standard technical indicators such as the Stochastic Oscillator, the Relative Strength Index (RSI) or the Moving Average Convergence/Divergence (MACD) [Achelis 1995]. $R_k(t)$ can of course also simply be lagged values of the returns, i.e. $R_k(t) = y(t-k)$.

The task for the learning process in The Trading Simulation Approach is to find the function g to maximize the profit, when applying the rule on real data. Various ways to measure the profit are discussed in [Hellström 1998a]. Note the difference between this and The Time Series Approach, where the learning task is to find a function g that minimizes the $RMSE$ error (1.3) for the entire time series.

The Trading Simulation Approach avoids many of the problems previously described of the Time Series Approach but does indeed have problems of its own, primarily that of statistical significance. The trading rule $T(t)$ normally issues Buy or Sell signals only for a minor part of the points in the time series.

1.3 Design of the Artificial Trader

In this section we discuss the design and implementation of the ASTA system. The task of the Artificial Trader is to act on an artificial market with a large number of available stocks that vary in prices over time. The Trader has to execute the trading rule $T(t)$ at every time step and decide whether to buy or sell stocks.

1.3.1 Performance Evaluation

The result of the artificial trader is presented as annual profits together with the increase in index. The mean difference between these two figures constitutes the net performance for the trader. The performance is displayed in both tabular and graphical formats as shown in the table part of Figure 1.2 and in Figure 1.3. Various considerations when measuring the profit are discussed in more detail in the thesis by [Hellström 1998b]. We now turn to the general architecture of the developed system.

1.3.2 Basic Architecture

The architecture of ASTA is based on an object-oriented approach with two major objects; the Market and the Trader. The basic layout is presented in Figure 1.1.

The Market Object The Market Object consists essentially of the total number of stocks participating in the trading simulation. A stock is defined by four time series; Close, High, Low and Volume. The basic operation on the Market Object is the simulation of changing prices as the date moves from start date to end date.

The Trader Object The Trader Object is more complex than the Market Object, as far as both attributes and allowed operations are concerned. The Buy rule and Sell rule are the main attributes that affect the behavior of the Trader. They are expressed in a high-level language and may include calls to a large number of predefined Matlab functions that access the stock data in the Market Object. User-defined functions can also be called directly.

Other Parts of the System The Market and Trader Objects have to be controlled by a support system that takes care of the following "meta" operations:

- Simulation.

The Step in time operation has to be applied to the Market Object in a loop for the selected time period. For each time step T, the Trader Object should also be activated. The following pseudo code describes the full ASTA system:

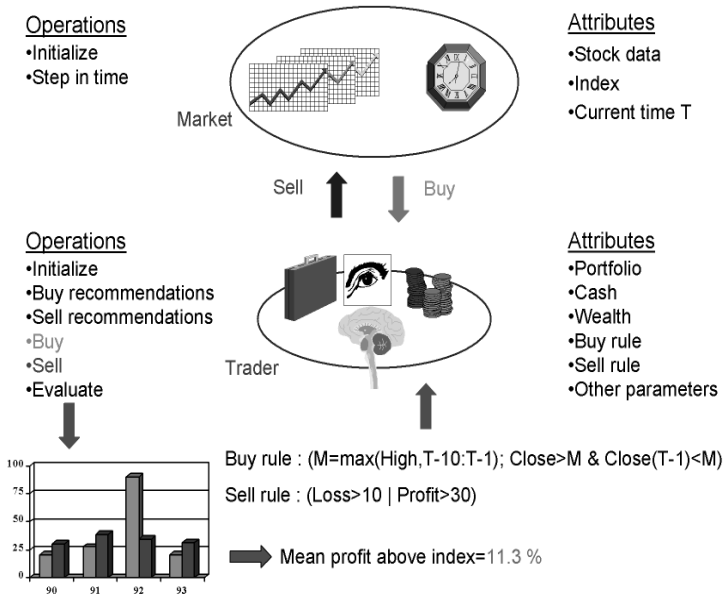

Figure 1.1
Basic Components of the ASTA System.

```
Trader.Initialize
Market.Initialize
loop until Market.T≥EndDate
    s = Trader.Sell_Recommendations
    Trader.Sell(s) % Sell all stocks of type s
    s = Trader.Buy_Recommendations
    n = T.available_cash / length(s)
    Trader.Buy(s, n) % Buy n stocks of type s
   Market.Step in time
end loop
Trader.Evalute
```

- User interface.

The end user assigns values to parameters such as the Buy rule and Sell rule of the Trader and the chosen time period for simulation. After the simulation the computed performance of suggested trades is presented.

The "silent mode" makes it possible to use the system as an objective function in a parameter optimization. The Artificial Trader is then called as a standard Matlab function from the optimization program code, returning a function value for each set of input parameters.

1.3.3 Predefined ASTA Functions

ASTA has a large number of predefined functions that make it possible to express compound trading rules interactively (as Buy and Sell rules). They also provide the developer of new algorithms with basic database access functions as well as some useful high-level functions. A complete description of the predefined functions in ASTA can be found in [Hellström 1998a].

1.4 An Example

In this section one example from the Windows version of ASTA is presented. 32 major stocks with active trading from the Swedish stock market for the years 1987-1997 have been selected for analysis. The main ASTA screen is shown in Figure 1.2. The most interesting items are the lines "Buy rule" and "Sell Rule". This is where the trading algorithm is defined. The rules follow the Matlab syntax and can include the predefined functions or the users' own functions with new algorithms. The example shows a test of the Stochastics indicator, here defined as

$$Stochastics(t, K, KS, D) \;=\; mav(100 * (Close(t) - L)/(H - L), D), \tag{1.8}$$

where $L = mav(\min(Low(t-K:t)), KS)$ and $H = mav(\max(High(t-K:t)), KS)$. The parameter K is the length of the window, and KS and D the length of the moving average function mav. A *Buy* signal is issued when $Stochastics(t, K, KS, D) > Buylevel$ and a Sell signal when $Stochastics(t, K, KS, D) < Sellevel$. The parameters K, KS, D, $Sellevel$ and $Buylevel$ control the performance of a trading strategy based on the indicator. Define $Stoch(K, KS, D, Buylevel, Sellevel)$ as the ASTA trading function, which will be subject to analysis in the next section.

In Figure 1.2, the "standard" values $30, 3, 3, 20, 80$ common in technical trading are used for the parameters. The results shown in Figure 1.2 and Figure 1.3 are quite stunning, with an average annual profit of 53.4% compared to the 16.3% achieved by the index. It is noteworthy that the only negative result is for the year 1997, where the strategy only made 1.7% whereas the index increased by 23.8%.

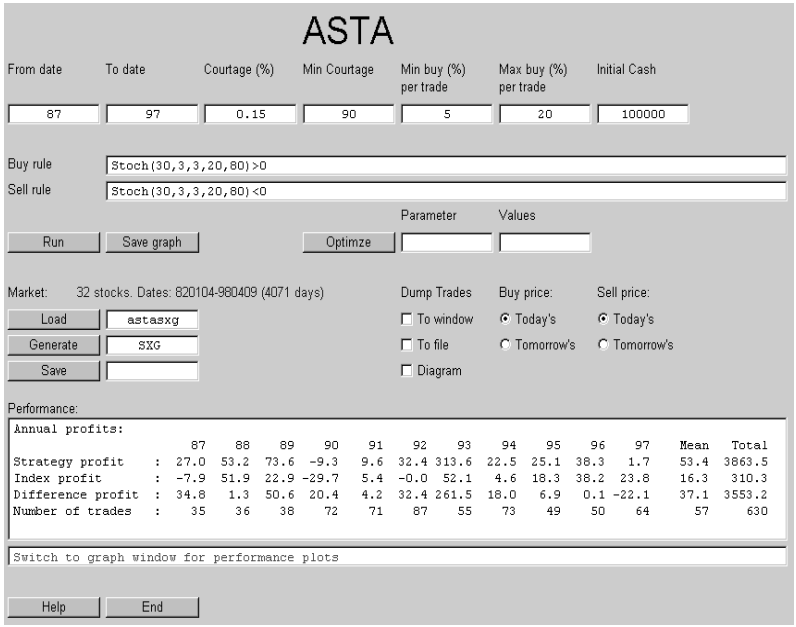

Figure 1.2
ASTA command window with buy and sell rules using the *Stoch* trading function based on the Stochastics indicator.

1.5 Viewing the Trader as an Objective Function

The obvious wish to maximize the profit p can be tackled in two ways:

1. Parameterizing the Buy rule and Sell rule, i.e. introducing parameters within the rules. Example:
Buy rule $= 'Close(T) > Maxx('High', N_{high}, T-1)'$
Sell rule $= 'Loss > L \mid (Profit > P \ \& \ Close(T) < Close(T-1))'$
The function $Maxx$ determines the maximum time series value in the interval $[T - N_{high}, T - 1]$. It is now possible to optimize the profit P with respect to the parameters N_{high}, L and P.

2. Viewing the Buy rule and Sell rule as symbolic expressions. The optimization then turns into a search problem, most naturally implemented in a genetic framework or using Inductive Logic Programming.

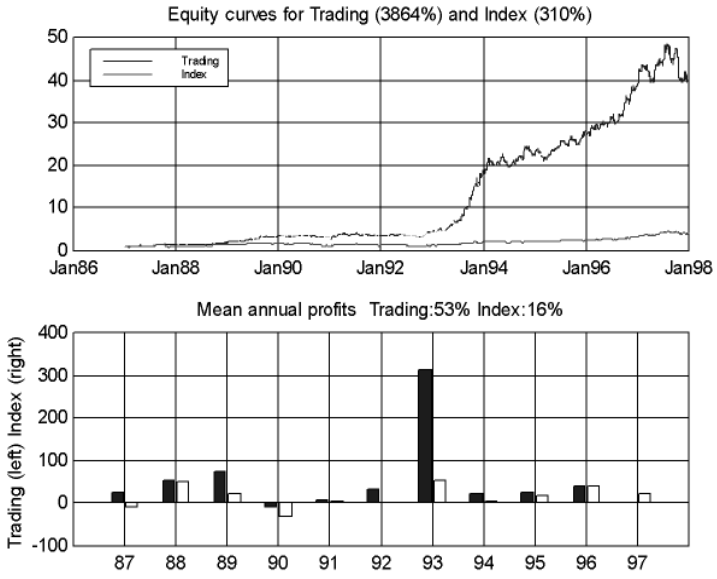

Figure 1.3
Performance of the trading function $Stoch(30, 3, 3, 20, 80)$ based on the Stochastics indicator.

An optimization of one of the parameters in the Stochastics trade function serves as an example for approach 1. We set up an optimization with buy and sell rules according to:

Buy rule = '$Stoch(30, 3, 3, Sellevel, 80)$'

Sell rule = '$Stoch(30, 3, 3, Sellevel, 80)$'.

The excess profit P can now be optimized with respect to $Sellevel$. The graphs in Figure 1.4 are automatically generated by the "Sweep" function in the ASTA system. The name of the parameter is given in the "Parameter" text box and the range in the "Values" text box.

It must be emphasized that optimization of the performance is a multi-dimensional parameter estimation problem. The graphs presented show the profit P as a function of *one* of the involved parameters whereas the rest of the parameters are fixed. The main purpose is to illustrate the possibilities and problems involved, even in a one-dimensional optimization.

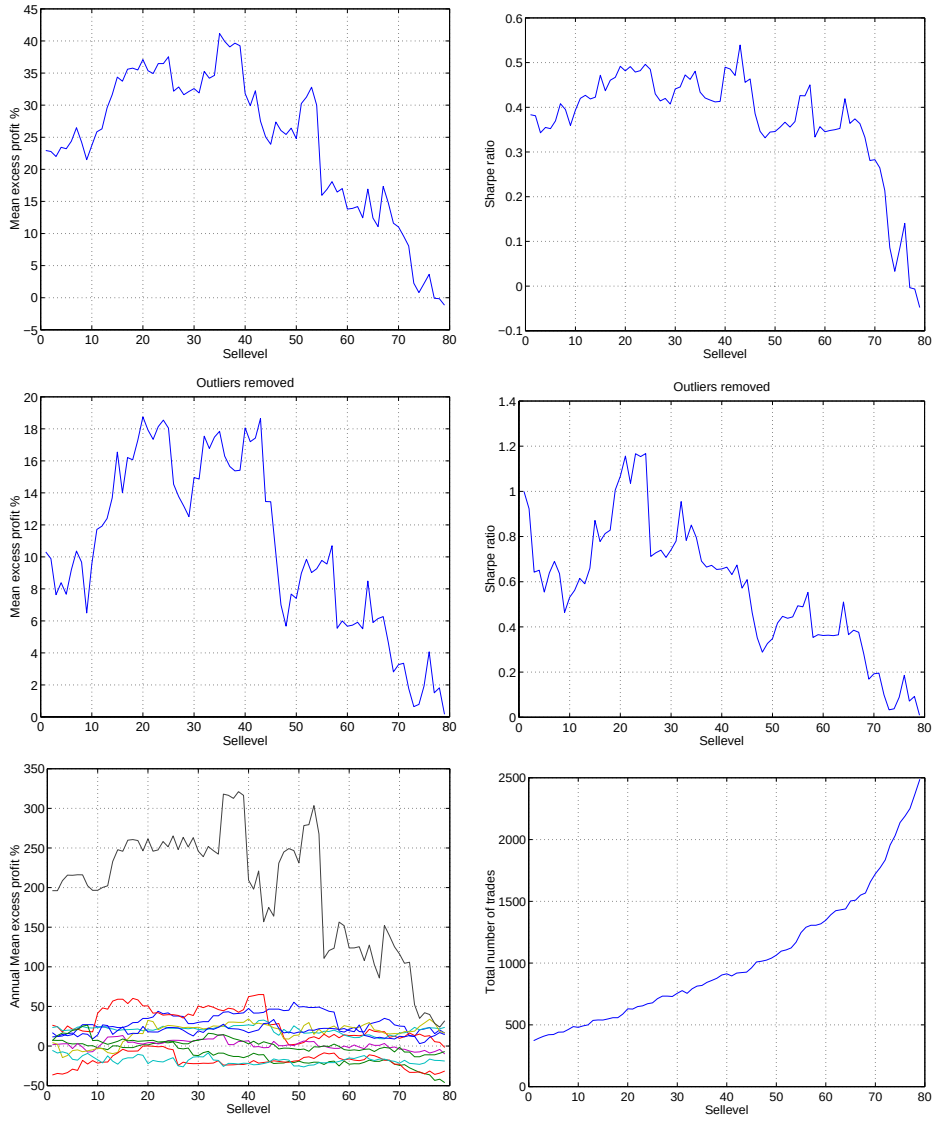

Figure 1.4
Trading results as a function of the *Sellevel* parameter. Stocks: SXG. Years: 87-97. Buy rule: $Stoch(30, 3, 3, Sellevel, 80)$. Sell rule: $Stoch(30, 3, 3, Sellevel, 80)$.

From the summary graph (top left) of the entire training period we can deduce that the highest profit is achieved for a *Sellevel* somewhere around 35. However, viewing the data at a higher resolution reveals a more complicated situation. In the bottom left diagram the same relation is plotted with one curve for each year in the training data set. It is clear that the mean profit is totally dominated by the results from *one* of the years (1993).

From these curves we can learn at least two important points:

1. The spread between individual years is very high.
2. The location of the maximum is not obvious.

This behavior of data is typical for most variables. The profit cannot be easily described as a function of measurable variables without introducing a dominant noise term in the function. Let us view the annual excess profit as a stochastic variable $P(\theta)$, where θ stands for one particular setting of the parameters that affect the profit. In the shown example, θ is the *Sellevel* parameter. $P(\theta)$ has been sampled once per year during the eleven years in the data set

$$\{P_1(\theta), P_2(\theta), P_3(\theta), P_4(\theta), P_5(\theta), P_6(\theta), P_7(\theta), P_8(\theta), P_9(\theta), P_{10}(\theta), P_{11}(\theta)\}. \quad (1.9)$$

Viewed this way, the task of tuning the parameter θ to find the "maximum" profit P is not well defined. P is a stochastic function and consequently has no "maximum". It has a probability distribution with an expected value E and a variance V. It's important to realize that tuning θ to maximize $E[P(\theta)]$ is just one of the available options. Maximizing $E[P(\theta)]$ provides the highest *mean* performance. Another possibility is to maximize the lower limit of a confidence interval. Since the risk factor is always a major concern in investments, and since the spread between individual years obviously can be very high, this sounds like a promising idea. A lower limit P_{low} for a confidence interval could be defined as

$$P_{low} = E[P(\theta)] - \sqrt{V[P(\theta)]}, \quad (1.10)$$

where $V[P(\theta)]$ is the variance of the stochastic variable $P(\theta)$. Yet another possibility is to use the Sharpe Ratio SR, which expresses the excess return in units of its standard deviation as

$$SR = \frac{E[P(\theta)]}{\sqrt{V[P(\theta)]}}. \quad (1.11)$$

The Sharpe Ratio is normally used to *evaluate* the performance of a trading strategy ([Sharpe 1966, Sharpe 1994]). However, [Choey and Weigend 1997] suggest to use the Sharpe Ratio as an objective function in portfolio optimization and derive a

learning algorithm for artificial neural networks. The Sharpe Ratio should be as high as possible for an optimal trading algorithm.

Due to the high noise level in the data we have added a modified Sharpe Ratio where the outliers are removed. The largest and smallest $P_i(\theta)$ in the set 1.9 are removed before taking the expected value and standard deviation for each θ. The unmodified Sharpe Ratio is shown in the top-right diagram and the one with outliers removed in the mid-right diagram. As we can see, the one with the outliers removed clearly reveals a maximum at around 20-25 followed by a clear decline. The unmodified Sharpe Ratio shows no such pattern.

1.5.1 Global Optimization

It is evident from the previous discussion that whatever performance measure we are using, the resulting objective function is very noisy. Therefore, new techniques for global optimization are needed to find the correct extreme point and optimal parameter values for a multi-variable problem.

We have made some preliminary tests using the DIRECT algorithm ([Jones et al 1993]) as implemented in the optimization environment TOMLAB ([Holmström 1999]). In Figure 1.5 we see the points sampled when trying to find the optimal buy and sell rules in the Stochastics Indicator. They cluster around $(40, 78)$, which seems to be the global optimum. In Figure 1.6 one-dimensional views of the *Net profit* (with reversed sign) versus the $Buylevel$ and the $Sellevel$ are shown. The optimum is more well-determined and distinct in the $Buylevel$. The global optimum is in fact very close to the standard values used in technical analysis. Further testing and analysis are needed to establish robustness properties of the parameters found.

1.6 Results and Further Development

The presented system provides a powerful tool for the development and evaluation of trading algorithms. Parameter settings can be tested and data screening can be easily performed interactively. As was mentioned in section 1.1.1, one of the reasons for the development of the ASTA system was to use it as an objective function when tuning model parameters and to find the general structure of trading rules, for example within a generic framework. The preliminary results show both possibilities and difficulties. The track can be examined considerably.

The dangers with "data snooping" got highlighted by breaking down the performance measures into shorter intervals. However, the inherent uncertainty result-

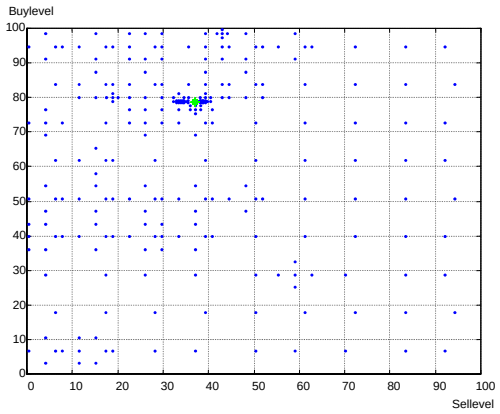

Figure 1.5
Sampled points in the parameter space by the TOMLAB global optimization solver when optimizing the buy and sell levels for the trading function $Stoch(30, 3, 3, Sellevel, Buylevel)$.

ing from the noisy processes involved calls for more computer-intensive simulation schemes to achieve statistically significant performance measures.

References

Achelis, S. B. 1995. *Technical Analysis from A to Z.* Irwin Professional Publishing, Chicago, 2nd edition.

Atiya, A. 1997. Design of time-variable stop losses and profit objectives using neural networks. In Weigend, A. S., Y. S. Abu-Mostafa, and A.-P. N. Refenes, editors, *Decision Technologies for Financial Engineering (Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets, NNCM-96)*, pages 76–83, Singapore. World Scientific.

Bengio, Y. 1997. Training a neural network with a financial criterion rather than a prediction criterion. In Weigend, A. S., Y. S. Abu-Mostafa, and A.-P. N. Refenes, editors, *Decision Technologies for Financial Engineering (Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets, NNCM-96)*, pages 36–48, Singapore. World Scientific.

Choey, M. and A. S. Weigend. 1997. Nonlinear trading models through Sharpe Ratio maximization. In Weigend, A. S., Y. S. Abu-Mostafa, and A.-P. N. Refenes, editors, *Decision Technologies for Financial Engineering (Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets, NNCM-96)*, pages 3–22, Singapore. World Scientific.

Hellström, T.. 1998a. Asta - a test bench and development tool for trading algorithms. Technical Report UMINF-98.12, ISSN-0348-0542, Department of Computing Science Umeå University, Umeå Sweden.

Hellström, T.. 1998b. *A Random Walk through the Stock Market*. Licentiate Thesis, UMINF 98.16 ISSN-0348-0542, Department of Computing Science,, Umeå University, Sweden.

Holmström, K.. 1999. The TOMLAB Optimization Environment in Matlab. *Advanced Modeling and Optimization*, 1. To be published.

Jones, D. R., C. D. Perttunen, and B. E. Stuckman. 1993. Lipschitzian optimization without the lipschitz constant. *Journal of Optimization Theory and Applications*, 79(1):157–181.

Leitch, G. and J. Tanner. 1991. Economic forecast evaluation: Profit versus the conventional error measures. *The American Economic Review*, pages 580–590.

Moody, J. E. 1992. Shooting craps in search of an optimal strategy for training connectionist pattern classifiers. In Moody, J. E., S. J. Hanson, and R. P. Lippmann, editors, *Advances in Neural Information Processing Systems 4, Proceedings of the 1991 NIPS Conference*, pages 847–854, San Mateo , CA. Morgan Kaufmann Publishers.

Moody, J. E. and L. Z. Wu. 1997. Optimization of trading systems and portfolios. In Weigend, A. S., Y. S. Abu-Mostafa, and A.-P. N. Refenes, editors, *Decision Technologies for Financial Engineering (Proceedings of the Fourth International Conference on Neural Networks in the Capital Markets, NNCM-96)*, pages 23–35, Singapore. World Scientific.

Sharpe, W. F. 1966. Mutual fund performance. *Journal of Business*, pages 119–138.

Sharpe, W. F. 1994. The Sharpe Ratio. *Journal of Portfolio Management*, 21:49–58.

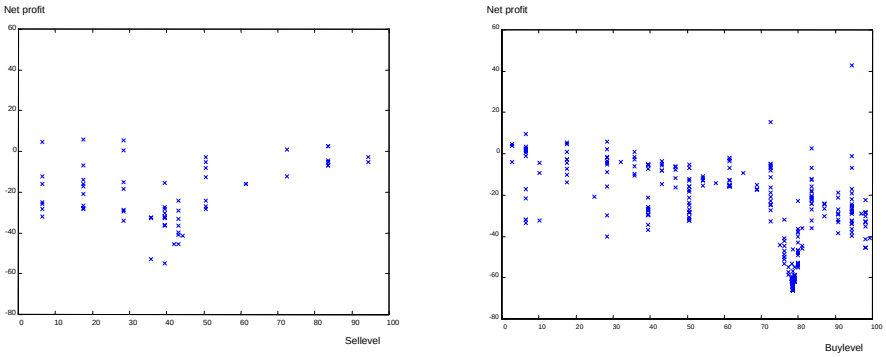

Figure 1.6
One-dimensional views of the global optimization of the parameters in the trading function $Stoch(30, 3, 3, Sellevel, Buylevel)$. The left graph shows the *Net profit* versus the *Buylevel* for an equidistant grid of values of the *Sellevel*. The right graph shows the *Net profit* versus the *Sellevel* for an equidistant grid of values of the *Buylevel*. Note that the sign of the *Net profit* is reversed, making it a minimization problem.