Out Of Sample Analysis Applied To The n^{th} Order Adaptive Polynomial Velocity Strategy And Eurodollar Futures

Working Paper April 2008

In previous working papers we showed how the application of a price curve generated by second, third and forth degree polynomial velocity could be used to develop a strategy to buy and sell futures intraday. The reasoning behind this type of system was to only trade when the price trend velocity was above a certain threshold. Many times prices meander around without any notable trend and this is considered noise. During these times we do not wish to trade because of the whipsaw losses that would occur from this type of price action. When a price trend finally starts, the velocity of that price trend moves above some minimum threshold value. Thus the velocity system would only issue a trade when certain velocity thresholds above "noise" levels are crossed.

The velocity system that we will use here to trade the Eurodollar futures contract is called the nth Order Adaptive Polynomial Velocity Strategy. The word "Adaptive" is used because the polynomial inputs change over time, adapting to the changing trading patterns of the Eurodollar futures contract. The nth Order Adaptive Polynomial Velocity Strategy has a number of unknown inputs that we have to determine before we can use this strategy to trade. These unknown inputs are the polynomial order, the optimum number of prices we need to determine the co-efficients of the polynomial and finally the velocity thresholds.   Here we will use Walk Forward Optimization and out-of-sample performance to determine the "best" polynomial inputs as well as how these inputs should change over time.   We will use the nth Order Adaptive Polynomial Velocity System to trade the Eurodollar futures contract on an intraday basis using one minute bar price data To test this strategy we will use one minute bar prices of the Eurodollar futures contract(EC) traded on the CME/Globex from February 1, 2005 to March 31, 2008.

The n^{th} Order Fixed Memory System Defined

The least squares forecast n^{th} order fixed memory polynomial velocity is constructed by solving for the coefficients $\beta_0, \beta_1, \beta_2, \beta_3 \ldots \beta_n$ for the discrete orthogonal Legendre polynomials each day using the last **N bars** of closing prices and the equation for β_j shown in the **"Math"** appendix at the end of this working paper. Then **Velocity(T+1)** is constructed from the equation shown in the "Math" appendix and plotted under the price chart.

Due to polynomial mathematics, the Velocity of the 2nd , 3rd and 4th order polynomial curve changes faster than it's corresponding first order polynomial curve velocity. Whether higher order polynomial velocities are an advantage or not, will be determined by the computer when we use a walk forward optimization technique described below.

At each bar we calculate the n^{th} order (1st through 4th) fixed memory polynomial velocity from the formulas in the "Math" appendix. As will be shown below, walk forward optimization will determine the order for nth order polynomial velocity, the number of prices, N, needed to compute the polynomial coefficients and the threshold amounts vup and vdn. When the velocity is greater than the threshold amount ***vup*** we will go long. When the velocity is less than the threshold amount ***-vdn*** we will go short.

Buy Rule:

IF Velocity is greater than the threshold amount ***vup*** then buy at the market.

Sell Rule:

IF Velocity is less than the threshold amount ***-vdn*** then sell at the market.

Intraday Bars Exit Rule:

Close the position at 1350 before the EC close (no trades will be carried overnight).

Intraday Bars First Trade of Day Entry Rule:

Ignore all trade signals before **7:00**am . For the Buy and Sell rules above we have included a first trade of the day entry rule. Trading in the EC futures has changed a lot in the last 4 years because of 24hr Globex trading. In particular trading starts a lot earlier in the morning when Asia and then Europe opens and then dies down. The EC volume starts to pick up again around 7am CST before the CME floor opens at 7:20am so we will have the strategy resume trading at 7am.

Discussion of Eurodollar Prices

The Eurodollar (EC) is traded on Globex and on the trading floor at the CME. On Globex the EC is traded on a 23hour basis . The CME hours for floor trading (RTH) are 7:20 to 14:00 CST. Over 50% of the volume in the EC is done on Globex during the CME RTH hours. We have restricted our study to only trading the EC during the 7:00 to 1350 hours.

Testing The Polynomial Velocity System Using Walk Forward Optimization

There will be four strategy parameters to determine:

1. **degree**, degree=1 for straight line velocity, degree=2 for 2nd order velocity, etc.
2. ***N***, is the number of lookback bars of prices to calculate the **velocity.**
3. ***vup***, the threshold amount that velocity has to be greater than to issue a buy signal
4. ***vdn***, the threshold amount that velocity has to be less than to issue a sell signal

To test this system we will use one minute bar prices of the Eurodollar (EC) futures contract traded on the CME/Globex and known by the symbol EC from February 1, 2005 to March 31, 2008.

We will test this strategy with the above EC 1 min bars on a walk forward basis, as will be described below. To create our walk forward files we will use the ***add-in*** software product called the Power Walk Forward Optimizer (PWFO). In TradeStation (TS), we will run the PWFO strategy ***add-in*** along with the nth Order Polynomial Velocity Strategy on the EC 1min data from February 1, 2005 to March 31, 2008. The PWFO will breakup and create thirty-four 4 calendar month test sections along with their corresponding one calendar month out-of-sample sections from the 3 years of EC (see Walk forward Testing below).

What Is A Test Section and Out-Of-Sample Section?

Whenever we do a TS optimization on a number of different strategy inputs, TS generates a report of performance metrics (total net profits, number of losing trades, etc) vs these different inputs. If the report is sorted on say the total net profits(tnp) performance metric column then the highest tnp would correspond to a certain set of inputs. This is called a test section. If we choose a set of strategy inputs from this report based upon some performance metric we have no idea whether these strategy inputs will produce the same results on future price data or data they have not been tested on. Price data that is not in the test section is defined as out-of-sample data. Since the performance metrics generated in the test section are mostly due to "curve fitting" (see Walk Forward Out-of-Sample Testing section below) it is important to see how the strategy inputs chosen from the test section perform on out-of-sample data.

What Does The Power Walk Forward Optimizer (PWFO) Do?

The PWFO is an TS ***add-in*** that breaks up the TS optimization run into a number of user selectable test and out-of-sample sections. The PWFO prints out the test sample performance **and the out-of-sample performance results**, on one line, for each case or input variable combination that is run by the TradeStation(TS) optimization module to a user selected spreadsheet comma delimited file. The PWFO can generate up to 500 different test and out-of-sample date optimization files in one TS run, saving the user from having to generate optimization runs one at a time. The PWFO output allows you to quickly determine whether your procedure for selecting input parameters for your strategy just curve fits the price and noise, or produces statistically valid out-of-sample results. In addition to the out-of-sample performance results presented for each case, 30+ superior and robust performance metrics (many are new and never presented before) are added to each case line and printed out to the comma delimited file. These 30+ performance metrics allow for a superior and robust selection of input variables that have a higher probability of performing well on out-of-sample data (Please see Appendix 2 for a listing of these performance metrics).

For our computer run we will have the PWFO breakup the 3+ year of EC one minute bar price data into 34 test/out-of sample files. The test sections will be 4 months and the out-of-sample section will be the one month following the test section.

The PWFO 34 test/out-of-sample section dates are shown in **Table 1** on page 7 below. We will then use another software product called the Walk Forward Performance Metric Explorer (WFPME) on each of the thirty-four test and out-of-sample(oos) sections generated by the PWFO to find the best test section performance ***filter*** that determines the system input parameters ***(degree, N, vup, vdn)*** that will be used on the out-of-sample data. Detailed information about the PWFO and the WFPME can be found at www.meyersanalytics.com

For the test data we will run the TradeStation optimization engine on the three years of EC 1 min bars with the following ranges for the nth order polynomial velocity strategy input variables.

1. degree from 1 to 4
2. N from 20 to 70 in steps of 10.
3. vup from 0.2 to 3 steps of 0.20
4. vdn from 0.2 to 3 in steps of 0.20

This will produce 5400 different cases or combinations of the input parameters for each of the 50 PWFO output files.

Walk Forward Out-of-Sample Testing

Walk forward analysis attempts to minimize the curve fitting of price noise by using the law of averages from the Central Limit Theorem on the out-of-sample performance. In walk forward analysis the data is broken up into many test and out-of-sample sections. Usually for any system one has some performance parameter selection procedure which we will call a ***filter*** used to select the input parameters from the optimization run. For instance, a ***filter*** might be all cases that have a profit factor (PF) greater than 1 and less than 3. For the number of cases left, we might select the cases that had the best percent profit. This procedure would leave you with one case in the test section output and it's associated strategy input parameters. Now suppose we ran our optimization on each of our many test sections and applied our filter to each test section output. We would then use the strategy input parameters found by the ***filter*** in each test section on the out-of-sample section immediately following that test section. The input parameters found in each test section and applied to each out-of-sample section would produce independent net profits and losses for each of the out-of-sample sections. Using this method we now have "x" number of independent out-of-sample section profit and losses from our filter. If we take the average of these out-of-sample section net profits and losses, then we will have an estimate of how our system will perform on average. Due to the Central Limit Theorem, as our sample size increases, the spurious noise results in the out-of-sample section performance tend to average out to zero in the limit leaving us with what to expect from our system and filter. Mathematical note: This assumption assumes that the out-of-sample returns are from probability distributions that have a finite variance.

Why use the walk forward technique? Why not just perform an optimization on the whole price series and choose the input parameters that give the best total net profits or profit factor? Surely the price noise cancels itself out with such a large number of test prices and trades. Unfortunately, nothing could be farther from the truth! Optimization is a misnomer and should really be called combinatorial search. As stated above, whenever we run a combinatorial search over many different combinations of input parameters on noisy data on a fixed number of prices, ***no matter how many***, the best performance parameters found are guaranteed to be due to ***"curve fitting"*** the noise and signal. What do we mean by ***"curve fitting"***? The price series that we trade consists of random spurious price movements, which we call noise, and repeatable price patterns (*if they exist*). When we run, for example, 5000 different input parameter combinations , the best performance parameters will be from those system input variables that are able to produce profits from the price pattern ***and*** the random spurious movements While the price patterns will repeat, the same spurious price movements will not. If the spurious movements that were captured by a certain set of input parameters were a large part of the total net profits, then choosing these input parameters will produce losses when traded on future data. These losses occur because the spurious movements will not be repeated in the same way. This is why system optimization or combinatorial searches with no out-of-sample testing cause loses when traded in real time from something that looked great in the test section. Unfortunately it is human nature to extrapolate past performance to project future trading results and thus results from curve fitting give the illusion, a modern "siren call" so to speak, of future trading profits.

In order to gain confidence that our input parameter selection method using the optimization output of the test data will produce profits, we must test the input parameters we found in the test section on out-of-sample data. In addition, we must perform the test/out-of-sample analysis many times. Why not just do the out-of-sample analysis once? Well just as in Poker or any card game, where there is considerable variation in luck from hand to hand, walk forward out-of-sample analysis give considerable variation in month to month out-of-sample profit "luck". That is, by pure chance we may have chosen some input parameter set that did well in the test section data ***and*** the out-of-sample section data. In order to minimize this type of "luck", statistically we must repeat the walk forward out-of-sample (oos) analysis over many test/oos sections and take an average of our monthly results over all out-of-sample sections. This average gives us an expected monthly return and a standard deviation of monthly returns which allows us to statistically estimate the expected equity and it's range for N months in the future.

Finding The Strategy Input Parameters in The Walk Forward Test Sections

The PWFO generates a number of performance metrics in the test section. (Please see appendix II for a listing of these performance metrics). The question we are attempting to answer statistically, is which performance metric or combination of performance metrics (which we will call a ***filter***) in the test section will produce strategy inputs that produce statistically valid profits in the out-of-sample section. In other words we wish to find a metric ***filter*** that we can apply to the test section that can give us strategy inputs that will produce, on average, good trading results in the future. The PWFO produces a total of 32 different performance metrics in the test section. If we have 5400 different input variations or cases then the test section consists of 32 columns of performance metrics for each of the 5400 input cases or rows.

An example of a simple filter would be to choose the row in the test section that had the highest net profit or perhaps a row that had one the best performance metric from one of the other 32 PWFO metrics. Unfortunately it was found that this type of simple filter very rarely produces good out-of-sample results. More complicated metric filters can produce good out-of-sample results minimizing spurious price movement biases in the selection of strategy inputs.

Here is an *example* of a better more complicated ***filter***. First, because in real time it is difficult to sustain more than five losses in a row and still keep trading, we will eliminate all those test sample case rows in the test section that have more than 5 losses in a row(LR). We will further require that the number of trades(NT) in the test section be greater than 40. We require this so we can eliminate inputs that produce very few trades. Four months is approximately 84 trading days so we are requiring at least a trade every other day on average. After using a NT and LR filter as just described there can still be 100's of rows left in the PWFO file. The PWFO generates the metric **LBR** which is the **Total Losing Bars** of all trades for a particular set of inputs in the test section. Let us choose the five rows that contain the **Bottom 5 LBR** values from the rows that are left from the **NT-LR** screen. In other words we sort LBR from high to low, eliminate the rows that have NT<40 and LR>5 and then choose the Bottom 5 Rows of whatever is left. This particular filter will now leave 5 cases or rows in the PWFO file that satisfy these filter conditions. We call this filter **b5LBR|5:40** where b5LBR means the Bottom 5 LBR Rows left ***after*** the NT-LR filter. Suppose for this filter, within the 5 PWFO rows that are left, we want the row that has the maximum PWFO **b2** metric in the test section where **b2** is the equity curve acceleration(see appendix 2 for definition). This would produce a filter named **b5LBR|5:40-b2**. This filter leaves only one row in the PWFO test section with its associated strategy inputs and out-of-sample net profit in the out-of-sample section. This particular **b5LBR|5:40-b2** filter is then calculated for each of the PWFO files and the average out-of-sample performance is calculated. In addition many other important out-of-sample performance statistics for this filter are calculated and summarized. **Appendix 3** shows such a computer run along with a small sample of other filter combinations that are constructed in a similar manner. Row 3 of the sample output in Appendix 3 shows the results of the filter discussed above.

Bootstrap Probability of Filter Results. Using modern "Bootstrap" techniques, we can calculate the probability of obtaining each filter's total out-of-sample ***net*** profits by chance. By ***net*** we mean subtracting the cost and slippage of all round trip trades from the total out-of-sample profits. Here is how the bootstrap technique is applied. Suppose for this example, we calculate the total out-of-sample net profits(tOnpNet) of 5400 different TopN-Metric- LR-NT filters. A mirror filter is created for each of the 5400 filters. However, instead of picking an out-of-sample net profit(OSNP) from a filtered row, the mirror filter picks a ***random*** row's OSNP in each PWFO file. Each of the 5400 mirror filters will choose a random row's OSNP of their own in each of the PWFO files. Thus if there are 34 PWFO files, each of the 5400 mirror filters will pick a random row's OSNP in each of the 34 PWFO files. At the end, each mirror filter will have 34 ***random*** OSNP's picked from the rows of the 34 PWFO files. The sum of the 34 random OSNP picks for each mirror filter will generate a random total out-of-sample net profit(tOnpNet) for each of the 5400 mirror filters. The average and standard deviation of the 5400 mirror filter's random tOnpNets will allow us to calculate the chance probability of each filter's tOnpNet. Thus given the mirror filter's bootstrap random tOnpNet average and standard deviation, we can calculate the probability of obtaining the TopN-Metric-PF-LR-NT filter's tONet by pure chance alone. In addition since for this run there are 5400 different filters, we can calculate the expected number of cases that we could obtain by pure chance that would match or exceed the tOnpNet of the filter we have chosen or 5400 X tOnpNet Probability. For our filter in row 3 in Appendix 3 the expected number of cases that we could obtain by pure chance that would match or exceed the $23180 is 5400 x 4.58 10^{-7} = 0.00247 . This is much less than one case so it is improbable that our result was due to pure chance.

The run shown in appendix 3 reveals that the following filter will produce the most consistent and reliable out-of-sample results.

Filter: **#Trds>=40** and **LR<=5** and **Bottom 5 LBR**

Where:

- **LR** = Maximum consecutive loses in a row in test optimization section. Since in real time it is tough to sustain more that five losses in a row and still keep trading, we will eliminate all those cases that have more than five losses in a row
- **NT** = The number of trades for a particular set of inputs in the test section.
- **LBR** = the **Total Losing Bars** of all trades for a particular set of inputs in the test section.

The first part of the filter chooses those rows or cases out of the 5400 rows in each PWFO file test(in-sample) section that satisfy the criteria **LR<=5 and NT>=40 .** After using a NT and LR filter, there can still be 100's of rows left in the PWFO file. The PWFO generates the metric **LBR** which is the **Total Losing Bars** of all trades for a particular set of inputs in the test section. Let us choose the five rows that contain the **Bottom 5 LBR** values from the rows that are left from the **NT-LR** screen.. This particular filter will now leave 5 cases or rows in the PWFO file that satisfy these filter conditions. We call this filter **b5LBR|5:40** where b5LBR means the Bottom 5 LBR Rows left ***after*** the NT-LR filter. Within the 5 PWFO rows that are left, we want the row that has the maximum PWFO **b2** metric in the test section where **b2** is the equity curve acceleration(see appendix 2 for definition). This ***Filter*** or selection procedure will leave only one choice for the system input values of ***degree, N, vup, vdn***. We then use these input values found in the test section by the **Filter** on the **next month** of one minute bar EC **out-of-sample** price bars **following** the test section.

Results

Table 1 on page 8 below presents a table of the thirty-four test and out-of-sample windows , the selected optimum parameters and the monthly out-of-sample results using the filter described above.

Figure 1 presents the out-of-sample 1 minute bar chart of EC for 2/27/2007 with the Velocity Indicator and all the buy and sell signals for that date.

Discussion of System Performance

In Appendix 3 Row 3 of the filter output are some statistics that are of interest for our filter. **BE** is the break even months. Assuming the trade average and standard deviation for this filter are from a normal distribution, this is how many months we need to trade this strategy so that we have a 99% probability that the equity after those number of months will be greater than zero. BE is 6 months for this filter. This means we would have to trade this strategy for at least six months to have a 99% probability that our equity would be positive. Another interesting statistic is **Blw**. Blw is the maximum number of months the OSNP equity curve failed to make a new high. Blw is 4 months for this filter. This means that 4 months was the longest time that the equity for this strategy failed to make a new equity high.

To see the effect of walk forward analysis, take a look at **Table 1**. Notice how the input parameters ***pw, N, vup and vdn*** take sudden jumps from high to low and back . This is the walk forward process quickly adapting to changing volatility conditions in the test sample. In addition, notice how often ***degree*** changes from a straight line velocity with ***degree=1*** to a 2nd, 3rd and 4th order velocity with ***degree= 2, 3 and 4***. The 3rd and 4th order velocities, due to the higher order components, change much faster than the straight line velocity. When the data gets very noisy with a lot of spurious price movements, it's better to have the velocity change slower filtering out the noisy data. During other times when the noise level is not as much it is better to have the velocity break its ***vup*** and ***vdn*** barriers faster to get onboard a trend faster. This is what the filter is doing. When there is a lot of noise in the test section it switches to the 1st order curve velocity. When the noise level is lower in the test section, it switches to the faster changing 3rd or 4th order curve velocity.

Using this filter, the strategy was able to generate $23,180 net equity after commissions and slippage trading one EC contract for 34 months. Note $30 roundtrip commission and slippage was subtracted from each trade and no positions were carried over night. The largest losing month was -$1538 and the largest drawdown was -$2163. The longest time between new equity highs was 4 months.

In observing Table 1 we can see that this strategy and filter made trades from a low of 6 times a month to a high of 29 times a month with an average of 13.6 trades a month. The strategy seemed to wait for really strong trends and then initiate a buy or sell. There were many months with just 7 trades. .In observing the chart for 12/27/07 we can see the startegy trading only when there is a big trend action.

Given 24 hour trading of the Eurodollar, restricting the strategy to trade only from 7am to 1:50pm caused the strategy to miss many profitable trends opportunities when Asia and then Europe opened trading in the early morning. Further research will include the A.M. time zones.

Disclaimer
The strategies, methods and indicators presented here are given for educational purposes only and should not be construed as investment advice. Be aware that the profitable performance presented here is based upon hypothetical trading with the benefit of hindsight and can in no way be assumed nor can it be claimed that the strategy and methods presented here will be profitable in the future or that they will not result in losses.

References

1. Efron, B., Tibshirani, R.J., (1993), "An Introduction to the Bootstrap", New York, Chapman & Hall/CRC.
2. Morrison, Norman "Introduction to Sequential Smoothing and Prediction", McGraw-Hill Book Company, New York, 1969.

Table 1 Walk Forward Out-Of-Sample Performance Summary for EC Polynomial Velocity System

EC-1 min bars 2/1/2005 - 3/31/2008. The input values *degree(pw), N, vup, vdn* are the values found from applying the filter to the test Sample optimization runs.

Filter= LR<=5 and NT>=10 and Bottom 5 LBR

Test Dates			Out-Of-Sample Dates			osnp	NOnp$30	NetEq	ollt	odd	ont	avosnp	pw	N	vup	vdn
02/01/05	To	05/31/05	06/01/05	To	06/30/05	3475	2965	2965	-675	-675	17	204.4	2	70	3	3
03/01/05	To	06/30/05	07/01/05	To	07/31/05	1825	1195	4160	-725	-1600	21	86.9	2	70	3	3
04/01/05	To	07/31/05	08/01/05	To	08/31/05	-138	-498	3662	-725	-1538	12	-11.5	1	30	3	3
05/01/05	To	08/31/05	09/01/05	To	09/30/05	388	58	3720	-625	-738	11	35.3	1	40	2.4	3
06/01/05	To	09/30/05	10/01/05	To	10/31/05	2175	1785	5505	-463	-763	13	167.3	1	40	2.4	3
07/01/05	To	10/31/05	11/01/05	To	11/30/05	2875	2425	7930	-738	-1600	15	191.7	1	70	2.8	3
08/01/05	To	11/30/05	12/01/05	To	12/31/05	-163	-433	7497	-413	-425	9	-18.1	1	70	3	2.8
09/01/05	To	12/31/05	01/01/06	To	01/31/06	1613	1313	8810	-638	-900	10	161.3	1	60	3	2.8
10/01/05	To	01/31/06	02/01/06	To	02/28/06	1413	1143	9953	-263	-325	9	157	3	60	3	3
11/01/05	To	02/28/06	03/01/06	To	03/31/06	3275	2735	12688	-363	-588	18	181.9	2	50	2.6	3
12/01/05	To	03/31/06	04/01/06	To	04/30/06	2650	2230	14918	-563	-563	14	189.3	1	60	2.8	3
01/01/06	To	04/30/06	05/01/06	To	05/31/06	2813	2273	17191	-625	-1238	18	156.3	1	70	2.8	3
02/01/06	To	05/31/06	06/01/06	To	06/30/06	988	598	17789	-463	-1850	13	76	1	70	2.6	2.8
03/01/06	To	06/30/06	07/01/06	To	07/31/06	213	-87	17702	-963	-963	10	21.3	1	70	2.6	2.8
04/01/06	To	07/31/06	08/01/06	To	08/31/06	-625	-1075	16627	-763	-1588	15	-41.7	1	60	2.6	2.8
05/01/06	To	08/31/06	09/01/06	To	09/30/06	-1538	-1898	14729	-788	-1538	12	-128.2	1	70	2.4	2.8
06/01/06	To	09/30/06	10/01/06	To	10/31/06	1050	780	15509	-125	-125	9	116.7	2	70	2.2	2.6
07/01/06	To	10/31/06	11/01/06	To	11/30/06	-250	-610	14899	-700	-1100	12	-20.8	2	70	2.2	2.8
08/01/06	To	11/30/06	12/01/06	To	12/31/06	2550	1980	16879	-550	-1125	19	134.2	3	70	3	3
09/01/06	To	12/31/06	01/01/07	To	01/31/07	800	560	17439	-238	-238	8	100	3	70	3	2.8
10/01/06	To	01/31/07	02/01/07	To	02/28/07	-138	-348	17091	-588	-588	7	-19.7	3	70	3	3
11/01/06	To	02/28/07	03/01/07	To	03/31/07	1500	1320	18411	0	0	6	250	3	70	3	3
12/01/06	To	03/31/07	04/01/07	To	04/30/07	-50	-260	18151	-563	-600	7	-7.1	4	70	2.8	3
01/01/07	To	04/30/07	05/01/07	To	05/31/07	338	128	18279	-225	-313	7	48.3	3	60	2.6	3
02/01/07	To	05/31/07	06/01/07	To	06/30/07	488	308	18587	-163	-163	6	81.3	3	60	2.2	3
03/01/07	To	06/30/07	07/01/07	To	07/31/07	300	60	18647	-325	-425	8	37.5	4	60	2.4	3
04/01/07	To	07/31/07	08/01/07	To	08/31/07	1438	808	19455	-313	-575	21	68.5	4	70	2.2	2.8
05/01/07	To	08/31/07	09/01/07	To	09/30/07	1850	1340	20795	-175	-288	17	108.8	3	50	2.2	3
06/01/07	To	09/30/07	10/01/07	To	10/31/07	663	-117	20678	-475	-1300	26	25.5	4	60	2.6	2.8
07/01/07	To	10/31/07	11/01/07	To	11/30/07	-1463	-1943	18735	-1250	-2188	16	-91.4	3	70	3	2.6
08/01/07	To	11/30/07	12/01/07	To	12/31/07	3575	3245	21980	-463	-663	11	325	2	70	3	2.6
09/01/07	To	12/31/07	01/01/08	To	01/31/08	725	-145	21835	-1125	-1500	29	25	2	70	3	2.8
10/01/07	To	01/31/08	02/01/08	To	02/29/08	2025	1665	23500	-613	-613	12	168.8	1	70	3	2.6
11/01/07	To	02/29/08	03/01/08	To	03/31/08	400	-320	23180	-638	-2063	24	16.7	1	60	3	3

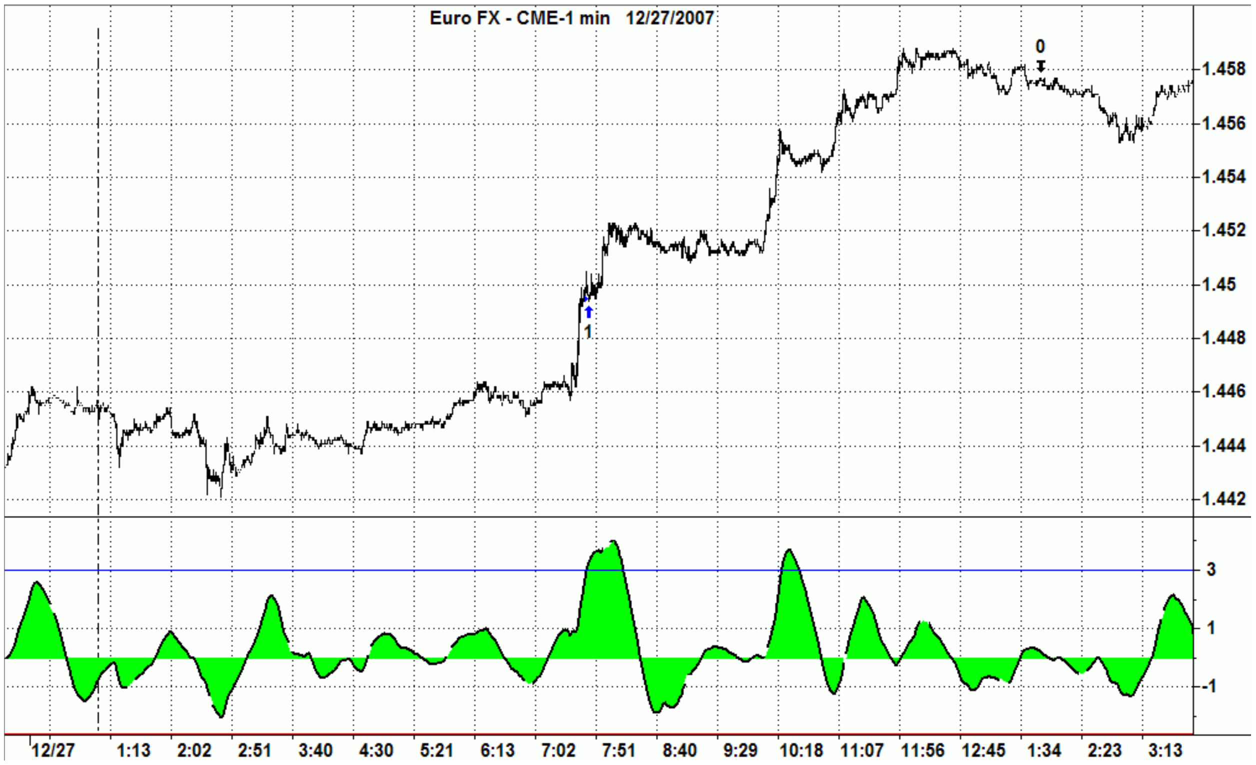
LR= Maximum consecutive loses in a row in test optimization section
NT = Minimum number of trades in the test section.
LBR = the **Total Losing Bars** of all trades for a particular set of inputs in the test section.

osnp = Monthly Out-of-sample gross profit in $
Nonp$30 = Monthly Out-Of-Sample Net Profit in $ = **osnp-ont*30**.
NetEq = running sum of the monthly out-of-sample net profits in $
ollt = The largest losing trade in the out-of-sample section in $.

odd = The close drawdown in the out-of-sample section in $.
ont = The number of trades in the out-of-sample month.
avosnp = The average out-of-sample profit per trade for that month $

pw = polynomial degree. degree=2 for parabolic curve velocity, degree=3 for 3rd order velocity, etc.
N = is the number of bars lookback period to calculate the polynomial **velocity.**
vup =, the threshold amount that velocity has to be greater than to issue a buy signal
vdn =, the threshold amount that velocity has to be less than to issue a sell signal

Figure 2 Walk Forward Out-Of-Sample Performance Summary for EC Fixed Memory Polynomial Velocity System 1 minute bar chart of EC from 12/27/07



Appendix I: *n*th Order Polynomial Next Bar's Forecast Math

What is the nth Order Polynomial ?

The nth Order Polynomial, also called the nth Order Fixed Memory Polynomial, is simply the least square fit of a polynomial of the form $b_0+b_1*t+b_2*t^2+b_3*t^3+\ldots b_n*t^n$ to a ***fixed*** number of past data points. Where **t** is discrete time bars. Time could be daily bars or one minute bars. We use the term "Fixed Memory" to designate that only a fixed number of data points are used to calculate the polynomial coefficients. It is assumed that the time bars occur at fixed intervals of time so tic bars would not be appropriate for this analysis. Least squares is a mathematical technique where the squared vertical distance between the data and the curve that is being fit to the data is minimized. When the net squared distance (also called the sum of the squared errors) is minimized, a unique set of coefficients $b_0,b_1,b_2,\ldots,b_n$ for the polynomial is determined. This type of error minimization is mathematically solvable and is widely used in science and mathematics.

For a 4th order polynomial equation, the least squares coefficients are obtained from the solution of the following matrix equation.

$$\begin{vmatrix} T & \Sigma t & \Sigma t^2 & \Sigma t^3 & \Sigma t^4 \\ \Sigma t & \Sigma t^2 & \Sigma t^3 & \Sigma t^4 & \Sigma t^5 \\ \Sigma t^2 & \Sigma t^3 & \Sigma t^4 & \Sigma t^5 & \Sigma t^6 \\ \Sigma t^3 & \Sigma t^4 & \Sigma t^5 & \Sigma t^6 & \Sigma t^7 \\ \Sigma t^4 & \Sigma t^5 & \Sigma t^6 & \Sigma t^7 & \Sigma t^8 \end{vmatrix} \begin{vmatrix} a_0 \\ b_0 \\ c_0 \\ d_0 \\ e_0 \end{vmatrix} = \begin{vmatrix} \Sigma p(t) \\ \Sigma(p(t)*t) \\ \Sigma(p(t)*t^2) \\ \Sigma(p(t)*t^3) \\ \Sigma(p(t)*t^4) \end{vmatrix}$$

where

p(T) is the current bar's price, **p(T-1)** is the previous bar's price and **p(1)** is the price **T** bars ago.
T is the number of Bars in the Least Squares estimation
Σp(t) is the summation of prices from **t=1 to T** bars
Σp(t)*t is the summation of prices times t from **t=1 to T** bars
Σt is the summation of the integer **t** from **t=1 to T** bars
Σt^2 is the summation of the integer **t** squared from **t=1 to T** bars
etc.

Once the coefficients to the polynomial have been solved for we generate the forecast for the next bar's price which is given for the equation by:

$$P_f = a_0 + b_0*(T+1) + c_0*(T+1)^2 + d_0*(T+1)^3 + e_0*(T+1)^4$$

Where P_f stands for price forecast.

With these coefficients, we can also generate the forecast for the next bar's ***velocity*** and ***velocity*** by the equations:

$$\text{Velocity}(T+1) = dP_f /dt = b_0 + 2c_0*(T+1) + 3d_0*(T+1)^2 + 4e_0*(T+1)^3$$

$$\text{Velocity}(t+1) = d^2P_f /d^2t = 2\, c_0 + 6d_0*(T+1) + 12e_0*(T+1)^2$$

We use the next bar forecast because changes in the trend are more quickly reflected in the forecast price, velocity and velocity than in the end point price, velocity and velocity.

Programs that solve for the solution to matrix equations can be found in the book "Numerical Recipes" by W. Press, et. al. However these type of matrix equation solvers are very slow and for these type of problems are unstable. They cause numerical errors and floating point overflows due to matrix inversion ill conditioning which produces false results.

Fortunately these type of problems can be solved by a fast, efficient and accurate algorithm using Discrete

Appendix I: *n*th Order Polynomial Next Bar's Forecast Math

Orthogona1 Legendre Polynomials. This method is explained in detail in Norman Morrison' book entitled "Introduction to Sequential Smoothing and Prediction", Chapter 7 page 223., referenced at the end of this section.

Without going into detail here (see Morrison reference), the polynomial filter can now be represented by:

$$P_e(t) = \sum_{j=0}^{n} \beta_j * \phi_j(t) \quad j=0 \text{ to } n$$

Where **n** is the polynomial order, **T** is the total number of Bars of data used in the Least Squares fit and

$$\beta_j = \sum_{k=0}^{T-1} p(t-T+k) * \phi_j(k)$$

$\phi_j(t)$ = the *normalized* discrete Legendre polynomial. t = an integer from 0 to T

The coefficients, $\beta_0, \beta_1, \beta_2, \beta_3, , , ,\beta_n$ for a **n**th order polynomial can now be solved for by the equation above and we can generate the forecast for the next bar's close, velocity and velocity which are given by the equations

$$P_F(T+1) = \beta_0*\phi_0(T+1) + \beta_1*\phi_1(T+1) + \beta_2*\phi_2(T+1) + \beta_3*\phi_3(T+1) + \ldots + \beta_n*\phi_n(T+1)$$

$$\text{Velocity} = (dP_F/dt)_{(T+1)} = \beta_1*(d\phi_1/dt)_{(T+1)} + \beta_2*(d\phi_2/dt)_{(T+1)} + \beta_3*(d\phi_3/dt)_{(T+1)} ++ \beta_n*(d\phi_n/dt)_{(T+1)}$$

$$\text{Velocity} = (d^2P_F/d^2t)_{(T+1)} = \beta_2*(d^2\phi_2/d^2t)_{(T+1)} + \beta_3*(d^2\phi_3/d^2t)_{(T+1)} +\ldots+ \beta_n*(d^2\phi_n/d^2t)_{(T+1)}$$

The nth Order Fixed Memory Forecast Next Bar's Velocity System Defined

The least squares forecast is constructed by solving for the least squares coefficients $\beta_1, \beta_2, , , \beta_n$ at each bar using the last **T** bars of closing prices and the Discrete Orthogonal Legendre Polynomial equations for β_j above. Then **Velocity** = $\mathbf{d^2P_F(T+1)/d^2t}$ is constructed from the velocity equation above and plotted under the price chart. In general what we will be doing is following the plotted curve of **Velocity** which is calculated at each bar from the previous T bars. When the velocity is greater than a threshold amount ***vup*** we will go long. When the velocity is less than a threshold amount ***-vdn*** we will go short.

Buy Rule:
IF Velocity is greater than the threshold amount ***vup*** then buy at the market.

Sell Rule:
IF Velocity is less than the threshold amount ***-vdn*** then sell at the market.

References

1. Morrison, Norman "Introduction to Sequential Smoothing and Prediction", McGraw-Hill Book Company, New York, 1969.

Appendix 2 Power Walk Forward Optimizer Performance Metrics

Symbol	Performance Metrics For The Test Section:
mtnp	Median Trade Profit
tnp	Total Net Profits
nT	# of Trades in Test Section
pP	% Profitable Trades in Test Section
PF	Profit Factor
std	Standard Deviation of Trades
tnp	Student t Statistic for Trades
mlb	Median losing bars
lbr	Total Losing Bars
mwb	Median of winning bars
wbr	Total Winning Bars
awl	mwb/mlb
wlb	wbr/lbr
mru-p	Median of all Trades(Maximum Trade Runup - Trade Profit)
mp-rd	Median of all Trades(Trade Profit - Maximum Trade Rundown)
wr	Max winners in a row
lr	Max losers in a row
mpft	Median of winning trades
mlos	Median of losing trades
p/l	mpft/mlos
dd	Test Section Drawdown
llt	Largest losing trade in Test Section
b0	Slope Of Trade Section Equity Regression Line
r2	Trade Equity Regression Trend Line Coefficient Of Correlation r^2
dev	Median of the absolute deviations of equity from straight line fit to equity curve
b1	Slope Of Equity 2nd Order Polynomial Line Where Equity 2nd Order Line = b1*t + b2*t^2
dy	Velocity Of Equity 2nd Order Polynomial Line evaluated on last trade i=nT.
b2	Acceleration of Equity 2nd Order Polynomial Line evaluated on last trade
r22	Equity 2nd Order Polynomial Line Coefficient Of Correlation r^2
mKr	Modified k-ratio = B0/Dev.
e-3	End Equity minus Equity 3 Trades before
eq10	Projected Equity 10 Trades in Future Using 2nd Order Polynomial Line/1000.

Explanation of The logic Behind The New Performance Metrics.

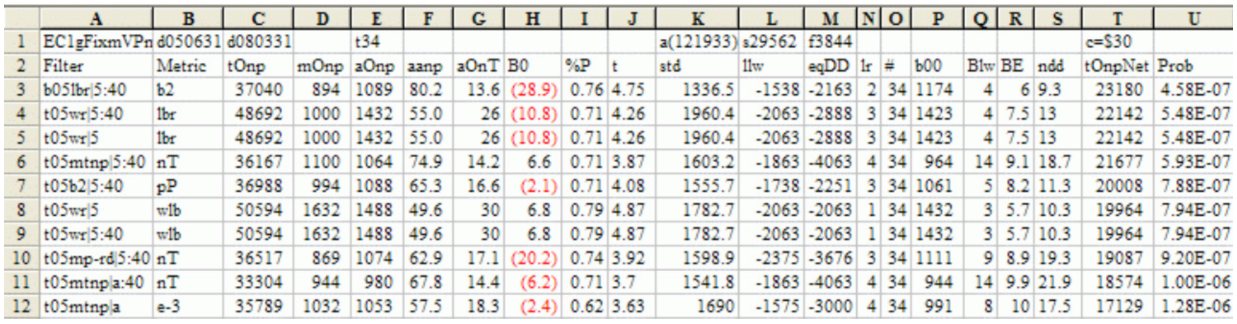
A number of performance statistics are new. This section will explain the logic behind the selection of these new statistics.

- **std** - **Trade Standard Deviation.**
  This is the standard deviation of the trade profits/losses in dollars.
- **t -** Student t-statistic. Used to determine the probability that the Average Trade Profit (tnp/nT)<>0 . Statistically, high values of t indicate that there is a very small probability that the sample average trade profit on the spreadsheet is <= zero and a random number. In real life, I find that if I screen out the top 5% of t values in Excel, I screen out a lot of curve fit input parameters. Same goes for mkr.
- **mlb** - **Median Of The Trade Losing Bars.**
  Each losing trade takes a certain number of bars. If we order the number of bars each losing trade takes then the median of all the losing trade bars is a robust statistic. We take the median of the losing trades bars to minimize the effect of large and small losing trade bars that may be outliers that distort this statistic.
- **mwb -Median Of The Trade Winning Bars**
  Each winning trade takes a certain number of bars. If we order the number of bars each winning trade takes then the median of all the winning trade bars is a robust statistic. We take the median of the winning trades bars to minimize the effect of large and small winning trade bars that may be outliers that that distort this statistic.
- **mru-p – Median of All Trades(Maximum Trade Runup Minus Final Trade Profit)**
  This statistic measures the difference between the maximum profit (trade runup) of each trade and the final profit of the trade. mru is the median of this difference for all trades for the given input variables. The closer the final trade profit is to the maximum trade profit, the better the performance of the input variable. Thus we would want the median to be as small as possible. We use the median for this statistic, because we do not want the statistic distorted by a few outlier trades
- **mp-rd – Median of All Trades(Final Trade Profit Minus Maximum rundown of Trade).**
  This statistic measures the difference between the final profit of each trade and the maximum trade loss (rundown) of the trade. The farther the final trade profit is from the maximum trade drawdown, the better the performance of the input variable. Thus, we would want the median to be as large as possible. We use the median for this statistic, because we do not want the statistic distorted by a few outlier trades
- **mpft** - **Median Of The Winning Trades.**
  This is the median of the winning trade profits. We take the median of the winning trades to minimize the effect of large winning trades that may be outliers that are not repeatable.
- **mlos -Median Of The Losing Trades**
  This is the median of the losing trade losses. We take the median of the losing trades to minimize the effect of large losing trades that may be outliers that are not repeatable.
- **p/l** – **mpft/|mlos|.**
  This is the ratio of mpft to absolute value of mlos. A high ratio indicates robustness indicating that the statistic of the median of the winning trades to losing trades is not caused by outliers.
- **b0 – Slope Of Trade Equity Regression Line.**
  The equity curve is fitted by a straight line where $\text{Equity}_{est} = a0+b0*t$. Equity_{est} is the straight line estimate of the Equity curve and **t** is the trade number. **b0** is the slope of the straight line. The dimensions of **b0** are dollars per trade.
- **r2 - Trade Equity Regression Trend Line Coefficient Of Correlation r2**
  r2 is a measure of how well a straight line fits the equity curve. R2 goes from 0 to 100. An R2 of 100 means the equity curve fits a straight line exactly. In general, a high value of r2 is desirable because it indicates trade profit consistency.
- **dev - Median Of The Absolute Values Of (The Equity At Each Trade Minus The Equity Regression Trend Line)**
  Dev is the median of all the absolute values of the deviations of the equity curve from the Equity straight line regression. This measure is similar to the standard deviation. However, the standard deviation squares

each error(deviation of the equity curve from the Equity straight line regression) in it's sum it weights large deviations much more strongly and is highly distorted by outliers. I prefer to the robust median statistic.

- **mkr - Modified K-Ratio = 100*b0/dev.**
  Lars Kestner developed the K-ratio in *Technical Analysis of Stocks and Commodities*, March 1996. the K-ratio compares reward to risk. Lars K-ratio used the Standard deviation from the trend line. I felt this gave to high a weight to the large profits and large losses so I modified the K-ratio by using the **dev** described above.
- **b1 - Slope Of Equity Curve Least Squares 2nd Order Polynomial Line Where Equity_{est}= b1*i +b2*i^2.**
  The equity curve is fitted by a 2nd order polynomial where Equity_{est} = b1*i+b2*i^2. Equity_{est} is the estimate of the Equity curve and **i** is the trade number. **b1** is the slope of the parabolic line. The dimensions of **b1** are dollars per trade. Many times the Equity curve doesn't fit a straight line very well. This is especially true if the equity curve is increasing or decreasing faster near the last trades than it was near the first trades. In this case, b1 gives a better representation of the equity curve trend.
- **dy - Velocity Of Equity Curve Least Squares Parabola Line.**
  This is a measure of how fast the equity curve is moving on the last trade. If dy is negative and b1 is positive this indicates that the equity curve is trending down on the last trades. Since after the last test sample trade comes trades in real time, it's important to know what direction the equity curve was headed before you trade.
- **b2 - Acceleration Of Equity Curve Least Squares 2nd Order Polynomial Line Where Equity= b1*i +b2*i^2.**
  The equity curve is fitted by a 2nd order polynomial where Equity_{est} = b1*i+b2*i^2. Equity_{est} is the estimate of the Equity curve and **i** is the trade number. **b2** is the acceleration of the parabolic line. The dimensions of **b2** are dollars per trade^2. Many times the Equity curve doesn't fit a straight line very well. This is especially true if the equity curve is increasing or decreasing faster near the last trades than it was near the first trades. In this case, b2 is a measure of how fast the slope of the equity curve is changing. If b2 is positive the equity curve's slope is increasing upward meaning the equity is increasing faster than a straight line. If b2 is negative, the equity curve's slope is decreasing meaning the equity will start decreasing a some point in time if the equity continues to follow this parabolic curve.
- **r22 - Equity Least Squares 2nd Order Polynomial Line Coefficient Of Correlation r^2**
  r22 is a measure of how well a least squares 2nd order polynomial line fits the equity curve. Note: for a 2nd order polynomial r22 can be negative. This is because the coefficient of correlation is for straight lines. 2nd Order and higher polynomials would be considered nonlinear. An r^2 of 100 means the equity curve fits a parabolic exactly. In general, a high value of r^2 is desirable because it indicates trade profit consistency.
- **e-3 – End Equity Minus Equity 3 Trades before.**
  This is another measure of the equity curve at the end. This shows what the equity trend is doing on the last three trades.
- eq10 - **Projected Equity 10 Trades in Future Using Curve Least Squares 2nd Order Polynomial Line/1000.**
  This is another measure of the equity curve 10 trades in the future. The projected equity is divided by 1000. This shows what the best estimate of the equity would be if it followed the least squares parabolic line for ten trades into the future. This performance variable can also serve as a curve fit alert. If eq10 is very high, this would indicate that the input parameters for this case have curve fit the noise and will not work well in the future. Using Excel you could exclude the Top 10% of eq10 cases.

Appendix 3 Power Walk Forward Optimizer Filters Output



	A	B	C	D	E	F	G	H	I	J	K	L	M	N	O	P	Q	R	S	T	U
1	EC1gFixmVPn	d050631	d080331		t34						a(121933)	s29562	f3844							c=$30	
2	Filter	Metric	tOnp	mOnp	aOnp	aanp	aOnT	B0	%P	t	std	llw	eqDD	lr	#	b00	Blw	BE	ndd	tOnpNet	Prob
3	b05lbr|5:40	b2	37040	894	1089	80.2	13.6	(28.9)	0.76	4.75	1336.5	-1538	-2163	2	34	1174	4	6	9.3	23180	4.58E-07
4	t05wr|5:40	lbr	48692	1000	1432	55.0	26	(10.8)	0.71	4.26	1960.4	-2063	-2888	3	34	1423	4	7.5	13	22142	5.48E-07
5	t05wr|5	lbr	48692	1000	1432	55.0	26	(10.8)	0.71	4.26	1960.4	-2063	-2888	3	34	1423	4	7.5	13	22142	5.48E-07
6	t05mtnp|5:40	nT	36167	1100	1064	74.9	14.2	6.6	0.71	3.87	1603.2	-1863	-4063	4	34	964	14	9.1	18.7	21677	5.93E-07
7	t05b2|5:40	pP	36988	994	1088	65.3	16.6	(2.1)	0.71	4.08	1555.7	-1738	-2251	3	34	1061	5	8.2	11.3	20008	7.88E-07
8	t05wr|5	wlb	50594	1632	1488	49.6	30	6.8	0.79	4.87	1782.7	-2063	-2063	1	34	1432	3	5.7	10.3	19964	7.94E-07
9	t05wr|5:40	wlb	50594	1632	1488	49.6	30	6.8	0.79	4.87	1782.7	-2063	-2063	1	34	1432	3	5.7	10.3	19964	7.94E-07
10	t05mp-rd|5:40	nT	36517	869	1074	62.9	17.1	(20.2)	0.74	3.92	1598.9	-2375	-3676	3	34	1111	9	8.9	19.3	19087	9.20E-07
11	t05mtnp|a:40	nT	33304	944	980	67.8	14.4	(6.2)	0.71	3.7	1541.8	-1863	-4063	4	34	944	14	9.9	21.9	18574	1.00E-06
12	t05mtnp|a	e-3	35789	1032	1053	57.5	18.3	(2.4)	0.62	3.63	1690	-1575	-3000	4	34	991	8	10	17.5	17129	1.28E-06

The Filter Output Columns are defined as follows

Row 1 is the PWFO Stub, File Start Date(050630), File End Date(080331), Number of Months(t50) and Cost plus slippage of round trip trade (c=$30).

Filter = The filter that was run. For example, b05lbr|5:40 is: PF<=2.5, LR<=4, R2>=60, Number of trades/test section>=12. pfxx|4|r2>60:10 is: any PF, LR<=4, R2>=60, Number of trades/test section>=10

Metric = The PWFO performance metric (defined on Appendix 2). For this filter, b05lbr|5:40 Metric=b2 (Equity curve acceleration).

This b05lbr|5:40-b2 filter produced the following average 50 month statistics on this line(row 3).

tOnp = Total out-of-sample(oos) net profit for these 50 months.

mOsp = median oos net profit for the 50 months

aOsp = Average oss net profit for the 50 months

aanp = Average oos profit per trade

aOnT = Average number of oos trades per month

B0 = The 34 month trend of the out-of-sample monthly profits

%P = The percentage of oos months that were profitable

t = The student t statistic for the 34 monthly oos profits. The higher the t statistic the higher the probability that this result was not due to pure chance

std = The standard deviation of the 34 monthly oos profits

llw = The largest losing oos month

eqDD = The oos equity drawdown

lr = The largest number of losing oos months in a row

= The number of months this filter produced a monthly result. Note for some months there can be no strategy inputs that satisfy a given filter's criteria.

b00 = The straight line trend of the oos equity curve in $/month.

Blw = The maximum number of months the oos equity curve failed to make a new high.

BE = Break even months. Assuming the average and standard deviation are from a normal distribution, this is the number of months you would have to trade to have a 99% probability that your oos equity is above zero.

ndd = The normalized equity drawdown = 100*eqDD/tONet

tONet = Total out-of-sample net profit(tOnp) minus the total trade cost. tONet=tOnp **-** #*aOnT*Cost.

Prob = The probability that the filter's tOnpNet was due to pure chance. For example, for row 3 where tOptNet=23180 there is a 1 in 2.18million (1/0.000000458) chance that the tOnpNet obtained by this filter was due to chance.