The Fading Memory Polynomial Velocity Strategy
Applied To 1Min bar British Pound Futures from 1/2008 –12/2009

Working Paper February 2010

This is a mathematical technique that fits a n^{th} order polynomial to the last ***T*** price bars but calculates the coefficients of the polynomial such that the error between the current n^{th} order polynomial and the current bar is weighted much higher than the error between the price T bars ago and the value of the n^{th} order polynomial T bars ago. As an example, if the latest price is at time t and the price made a turn at time bar t-10 , then we do not want prices prior to t-10 effecting the current polynomial fit as much. As will be shown the most familiar case of this fading memory technique is the exponential moving average. The fading memory technique is in contrast to the Least Squares Polynomial fit, which weights all past errors between the polynomial and the price bar equally.

In previous working papers we showed how the application of a price curve generated by Nth Order Fixed Memory 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 cost of 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 a minimum threshold noise 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 British Pound futures contract is called the Nth Order ***Fading*** Memory Adaptive Polynomial Velocity Strategy. The word "Adaptive" is used because the polynomial inputs change over time, adapting to the changing trading patterns of the British Pound futures contract. The Nth Order Fading memory 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 to the Fading Memory Polynomial are the polynomial order, the optimum number of prices we need to determine the coefficients 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 Fading Memory Adaptive Polynomial Velocity System to trade the British Pound 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 British Pound futures contract(BP) traded on the CME/Globex from January 3, 2008 to December 31, 2009.

The n^{th} Order Fading Memory Adaptive Polynomial Velocity Defined

The adaptive n^{th} order Fading Memory Polynomial Velocity is constructed and plotted at each bar by solving for the coefficients $b_1, b_2, b_3, \ldots b_n$ for the discrete orthogonal Meixner polynomials at each bar using the exponential decay factor $\alpha=(1-\beta)$ and the equation for b_j shown in the **"Math"** appendix of this paper. Then **Velocity(T+1)** is constructed from the equation shown in the "Math" appendix and plotted under the price chart.

The velocity of a 2^{nd} , 3^{rd} and 4^{th} order polynomial should change faster than the straight line (1^{st} order velocity). As observed from the 2^{nd} order velocity equation in the "Math" section, there is an acceleration component in the calculation of the velocity. This means that the 2^{nd} order velocity will reflect a change in the price trend much faster than the straight line velocity which does not have an acceleration component. The same is true for 3^{rd} and 4^{th} order velocities. Whether higher order polynomial velocities is an advantage or not we will let the computer decide when we let the computer search for the "best" polynomial degree as described below.

At each bar we calculate the n^{th} order (1^{st} through 4^{th}) fading memory polynomial velocity from the formulas in the "Math" appendix. As we will show below, optimization will determine the order for nth order polynomial velocity that will be used. 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.

The strategy follows the velocity curve. When the velocity is greater than the threshold amount ***vup*** a buy signal is issued. The threshold **vup** serves as a noise filter. That is, price noise creates a lot of small back and forth velocity movement. Unless the velocity can break some threshold to the upside, no trade is issued and the move is considered price noise. The same logic holds for the sell threshold **vdn**.

Intraday Bars Exit Rule:
Close the position at 1350 before the BP 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 BP 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 BP 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 British Pound Prices

The British Pound (BP) is traded on Globex and on the trading floor at the CME. On Globex the BP 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 BP is done on Globex during the CME RTH hours. We have restricted our study to only trading the BP 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. **alpha =(1-β)** The exponential decay weight for the Nth Order Fading Memory Polynomial calculation**.**
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 British Pound (BP) futures contract traded on the CME/Globex and known by the symbol BP for the 105 weeks from Jan 2, 2008 to December 31, 2009.

We will test this strategy with the above BP 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 Fading Memory Polynomial Velocity Strategy on the BP 1min data from January 2, 2008 to Decemberr 31, 2009. The PWFO will breakup and create 100 calendar month test sections along with their corresponding one calendar week out-of-sample sections from the 105 weeks of BP (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 a 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 in the test section and printed out to the comma delimited file. These 30+ performance metrics allow for a superior and robust selection of input variables from the test section 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 105 weeks of BP one minute bar price data into 100 test/out-of sample files. The test sections will be 30 calendar days and the out-of-sample(oos) section will be the one week following the test section. The oos week will always end on a Friday as will the 30 day calendar test section. As an example the first test section would be from 1/3/2008 to 2/1/2008 and the out-of-sample section would be from 2/4/2008 to 2/8/2008.(our test and out-of-sample sections always end on a Friday). We would then move everything ahead a week and the 2nd test section would be from 1/9/2008 to 2/8/2008 and the out-of-sample section would be from 2/11/2008 to 2/15/2008. Etc.

The PWFO 100 test/out-of-sample section dates are shown in **Table 1** on page 8 below. We will then use another software product called the Walk Forward Performance Metric Explorer (WFPME) on each of the 100 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 100 weeks of BP 1 min bars with the following ranges for the nth order fading memory polynomial velocity strategy input variables.

1. degree from 1 to 4
2. N from 20 to 70 in steps of 10.
3. vup from 0.25 to 3 steps of 0.25
4. vdn from 0.25 to 3 in steps of 0.25

Note: I use **N** because it gives a better understanding of how many bars of past data are approximately being used. **N** and **α (α=1-β)** is approximately related by the formula **α=2/(1+N)**. N is converted to **α** by this formula in the Nth Order Fading Memory Polynomial calculation

This will produce 3456 different cases or combinations of the input parameters for each of the 100 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 metric 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 3456 different input variations or cases then the test section consists of 32 columns of performance metrics for each of the 3456 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*** that was used in this paper. We require that the number of trades(**NT**) in the test section be greater than 10 trades a month and less than 50. We require this so we can eliminate inputs that produce very few trades and inputs that generate more than 2.5 trades on average a day. One month is approximately 22 trading days so we are requiring at least a trade every other day on average. In addition we are trying to catch the intraday trend and not get whipsawed. Setting the number of trades per month to be less that 50 only chooses those inputs which have the tendency to catch the main intraday trend. After using a **NT** and **LR** filter as described there can still be 100's of rows left in the PWFO file. The PWFO generates the metric **mru-p.** This metric measures the difference between the maximum profit (trade runup) of each trade and the final profit of the trade. **mru-p** 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 metric, because we do not want the statistic distorted by a few outlier trades The smaller **mru-p** is, the more efficient the strategy is in getting the maximum profit from each trade. Let us choose the 20 rows that contain the **smallest(bottom) 20 mru-p** values from the rows that are left from the **NT-LR** screen. In other words we sort **mru-p** from low to high, eliminate the rows that have **NT**<10 and **NT**>50 and then choose the Bottom 20 Rows of whatever is left. This particular filter will now leave 20 cases or rows in the PWFO file that satisfy these filter conditions. We call this filter **b20mru-p|:10::50** where **b20mru-p** means the Bottom 20 **mru-p** Rows left ***after*** the NT-LR filter. Suppose for this filter, within the 20 PWFO rows that are left, we want the row that has the maximum PWFO **mp-rd** metric in the test section. This metric 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.. This would produce a filter named **b20mru-p|:10::50-mp-rd**. 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 **b20mru-p|:10::50-mp-rd** 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 as an example, we calculate the total out-of-sample net profits(tOnpNet) of 5000 different ***TopN-Metric- LR-NT*** filters. A mirror filter is created for each of the 5000 filters. However, instead of picking an out-of-sample net profit(OSNP) from a filter row, the mirror filter picks a ***random*** row's OSNP in each of the 100 PWFO files. Each of the 5000 mirror filters will choose a random row's OSNP of their own in each of the 100 PWFO files.. At the end, each of the 5000 mirror filters will have 100 ***random*** OSNP's picked from the rows of the 100 PWFO files. The sum of the 100 random OSNP picks for each mirror filter will generate a random total out-of-sample net profit(tOnpNet). The average and standard deviation of the 5000 mirror filter's different random tOnpNets will allow us to calculate the chance probability of each ***TopN-Metric- LR-NT*** 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 tOnpNet by pure chance alone. Since for this run there are 3844 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 (3854) 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 $20709 is 3844 x 2.30 10^{-7} = 0.00088 . 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>=10** and **#Trds<=50** and **Bottom 20 mru-p**

Where:

- **NT** = The number of trades for a particular set of inputs in the test section.

- **mru-p** = Each trade had a runup, **ru**, from the start of the trade to a maximum profit and then the trade closes at some point at a price **p**. **ru-p** is a metric which measures how close each trade comes to it's maximum profit. We take the medium of all trade **ru-p**'s and call it **mru-p**..

The first part of the filter chooses those rows or cases out of the 3456 rows in each PWFO file test(in-sample) section that satisfy the criteria **NT<=50 and NT>=10 .** 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 **mru-p** which is the **median ru-p** of all trades for a particular set of inputs in the test section. Let us choose the 20 rows that contain the **Bottom 20 mru-p** values from the rows that are left from the **NT-LR** screen.. This particular filter will now leave 20 cases or rows in the PWFO file that satisfy these filter conditions. We call this filter **b20mru-p|a:10::50** where b20mru-p means the Bottom 20 mru-p Rows left ***after*** the NT filter. Within the 20 PWFO rows that are left, we want the row that has the maximum PWFO **mp-rd** metric in the test section. 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 week** of one minute BP **out-of-sample** price bars **following** the test section.

Results

Table 1 on page 8 below presents a table of the 100 test and out-of-sample windows , the selected optimum parameters and the weekly out-of-sample results using the filter described above.

Figure 1 presents the out-of-sample 1 minute bar chart of BP for 12/30/2009 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 spreadsheet filter output are some statistics that are of interest for our filter. **BE** is the break even weeks. Assuming the trade average and standard deviation for this filter are from a normal distribution, this is how many weeks we need to trade this strategy so that we have a 99% probability that the equity after those number of weeks will be greater than zero. BE is 38 weeks for this filter. This means we would have to trade this strategy for at least 38 weeks to have a 99% probability that our equity would be positive. Another interesting statistic is **Blw**. Blw is the maximum number of weeks the OSNP equity curve failed to make a new high. Blw is 26 weeks for this filter. This means that 26 weeks 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 2^{nd}, 3^{rd} and 4^{th} order velocity with ***degree= 2, 3 and 4***. The 3^{rd} and 4^{th} 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 1^{st} or 2^{nd} order curve velocity. When the noise level is lower in the test section, it switches to the faster changing 3^{rd} or 4^{th} order curve velocity.

Using this filter, the strategy was able to generate $20,709 net equity after commissions and slippage trading one BP contract for 100 weeks. Note $17.50 roundtrip commission and slippage was subtracted from each trade and no positions were carried over night. The largest losing week was -$3039 and the largest drawdown was -$4401. The longest time between new equity highs was 26 weeks.

In observing Table 1 we can see that this strategy and filter made trades from a low of no trades/week to a high of 19 trades/week with an average of 4.4 trades/week. The strategy seemed to wait for really strong trends and then initiate a buy or sell. There were many weeks that have no trades. .In observing the chart from 12/30/2009 we can see the strategy trading mostly only when there is a big trend action.

Given 24 hour trading of the British Pound, 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 BP Nth Order Fading Memory Polynomial Velocity System

bp-1 min bars 1/2/2008 - 12/31/2009. The input values *degree(pw), N, vup, vdn* are the values found from applying the filter to the test section optimization runs.

Filter= NT<=50 and NT>=10 and Bottom 20 mru-p, maximum mp-rd

Test Dates		Out-Of_Sample Dates		osnp	NOnp$17.5	NetEq	ollt	odd	ont	avosnp	pw	len	vup	vdn
01/02/08 to	02/01/08	02/04/08 to	02/08/08	556	468	468	0	0	5	111.2	1	30	2.5	0.25
01/09/08 to	02/08/08	02/11/08 to	02/15/08	369	352	820	0	0	1	369	1	60	0.5	1
01/16/08 to	02/15/08	02/18/08 to	02/22/08	50	32	852	0	0	1	50	2	60	1	2
01/23/08 to	02/22/08	02/25/08 to	02/29/08	725	672	1525	-325	-325	3	241.7	2	50	1.25	1.75
01/30/08 to	02/29/08	03/03/08 to	03/07/08	431	326	1851	-288	-288	6	71.8	2	50	1.75	1
02/06/08 to	03/07/08	03/10/08 to	03/14/08	1950	1845	3696	-463	-463	6	325	2	50	1.75	1
02/13/08 to	03/14/08	03/17/08 to	03/21/08	1100	1048	4744	0	0	3	366.7	2	30	3	2
02/20/08 to	03/21/08	03/24/08 to	03/28/08	(163)	(216)	4528	-344	-344	3	-54.3	1	20	2	1.5
02/27/08 to	03/28/08	03/31/08 to	04/04/08	606	554	5082	0	0	3	202	2	60	2	1.25
03/05/08 to	04/04/08	04/07/08 to	04/11/08	206	171	5252	-119	-119	2	103	2	60	3	1.25
03/12/08 to	04/11/08	04/14/08 to	04/18/08	(44)	(62)	5191	0	0	1	-44	2	60	3	1.25
03/19/08 to	04/18/08	04/21/08 to	04/25/08	219	166	5358	-125	-125	3	73	1	30	2.25	1
03/26/08 to	04/25/08	04/28/08 to	05/02/08	925	855	6212	0	0	4	231.2	2	50	2.75	1.25
04/02/08 to	05/02/08	05/05/08 to	05/09/08			6212					3	60	2.25	1.75
04/09/08 to	05/09/08	05/12/08 to	05/16/08	(1288)	(1376)	4837	-681	-1288	5	-257.6	1	40	2	0.5
04/16/08 to	05/16/08	05/19/08 to	05/23/08			4837					4	70	2.75	2.25
04/23/08 to	05/23/08	05/26/08 to	05/30/08			4837					4	60	3	2.25
04/30/08 to	05/30/08	06/02/08 to	06/06/08	1244	1122	5958	-163	-163	7	177.7	3	60	0.5	2.25
05/07/08 to	06/06/08	06/09/08 to	06/13/08	(156)	(226)	5732	-256	-256	4	-39	4	60	2.5	2.25
05/14/08 to	06/13/08	06/16/08 to	06/20/08	444	392	6124	-263	-263	3	148	2	60	0.75	2.5
05/21/08 to	06/20/08	06/23/08 to	06/27/08	1025	938	7062	0	0	5	205	1	30	0.25	2.25
05/28/08 to	06/27/08	06/30/08 to	07/04/08	(381)	(468)	6593	-388	-438	5	-76.2	2	70	0.25	2.25
06/04/08 to	07/04/08	07/07/08 to	07/11/08	1406	1318	7912	-381	-381	5	281.2	4	70	0.25	2.75
06/11/08 to	07/11/08	07/14/08 to	07/18/08	113	43	7954	-319	-419	4	28.2	1	50	0.25	1.25
06/18/08 to	07/18/08	07/21/08 to	07/25/08	(300)	(318)	7637	0	0	1	-300	1	50	0.5	1.25
06/25/08 to	07/25/08	07/28/08 to	08/01/08	(81)	(151)	7486	-425	-581	4	-20.2	1	70	0.25	0.75
07/02/08 to	08/01/08	08/04/08 to	08/08/08	194	106	7592	-175	-175	5	38.8	3	30	2.75	3
07/09/08 to	08/08/08	08/11/08 to	08/15/08	(156)	(314)	7279	-306	-463	9	-17.3	4	40	3	3
07/16/08 to	08/15/08	08/18/08 to	08/22/08	(413)	(483)	6796	-513	-731	4	-103.2	1	50	1.75	0.25
07/23/08 to	08/22/08	08/25/08 to	08/29/08	1525	1402	8198	-313	-350	7	217.9	3	40	3	0.25
07/30/08 to	08/29/08	09/01/08 to	09/05/08	(275)	(608)	7591	-375	-963	19	-14.5	2	20	3	1.25
08/06/08 to	09/05/08	09/08/08 to	09/12/08	256	81	7672	-425	-431	10	25.6	3	70	2.75	0.5
08/13/08 to	09/12/08	09/15/08 to	09/19/08	306	184	7856	-800	-1406	7	43.7	1	40	1.75	0.25
08/20/08 to	09/19/08	09/22/08 to	09/26/08	200	182	8038	0	0	1	200	1	70	0.75	1.75
08/27/08 to	09/26/08	09/29/08 to	10/03/08	(1606)	(1746)	6292	-969	-2013	8	-200.8	1	50	1.25	1.25
09/03/08 to	10/03/08	10/06/08 to	10/10/08	1588	1500	7792	-475	-475	5	317.6	1	60	1.75	1
09/10/08 to	10/10/08	10/13/08 to	10/17/08	(563)	(580)	7212	0	0	1	-563	1	50	1.75	2
09/17/08 to	10/17/08	10/20/08 to	10/24/08	900	812	8024	-463	-1156	5	180	1	30	3	2.5
09/24/08 to	10/24/08	10/27/08 to	10/31/08	(250)	(355)	7670	-613	-869	6	-41.7	1	60	2	1
10/01/08 to	10/31/08	11/03/08 to	11/07/08	(819)	(872)	6798	-1113	-1900	3	-273	1	60	1.25	2
10/08/08 to	11/07/08	11/10/08 to	11/14/08	2356	2268	9066	-1181	-1181	5	471.2	1	70	3	0.25

Test Dates			Out-Of_Sample Dates		osnp	NOnp$17.5	NetEq	ollt	odd	ont	avosnp	pw	len	vup	vdn
10/15/08	to	11/14/08	11/17/08	to 11/21/08	994	906	9973	-456	-456	5	198.8	1	50	1.5	0.75
10/22/08	to	11/21/08	11/24/08	to 11/28/08	(3069)	(3174)	6799	-2213	-3069	6	-511.5	1	70	3	0.25
10/29/08	to	11/28/08	12/01/08	to 12/05/08	(694)	(782)	6018	-1019	-1819	5	-138.8	1	60	2.75	0.25
11/05/08	to	12/05/08	12/08/08	to 12/12/08	(638)	(708)	5310	-494	-638	4	-159.5	1	50	2.75	0.5
11/12/08	to	12/12/08	12/15/08	to 12/19/08	2919	2849	8158	0	0	4	729.8	1	70	0.5	2.5
11/19/08	to	12/19/08	12/22/08	to 12/26/08	88	53	8212	-275	-275	2	44	1	50	0.75	3
11/26/08	to	12/26/08	12/29/08	to 01/02/09	(300)	(422)	7789	-425	-663	7	-42.9	2	50	2.5	1
12/03/08	to	01/02/09	01/05/09	to 01/09/09	894	702	8490	-1369	-2075	11	81.3	1	20	2.75	0.25
12/10/08	to	01/09/09	01/12/09	to 01/16/09	1550	1462	9953	-388	-763	5	310	1	20	2.75	0.25
12/17/08	to	01/16/09	01/19/09	to 01/23/09	669	599	10552	-769	-769	4	167.2	1	50	0.75	3
12/24/08	to	01/23/09	01/26/09	to 01/30/09	488	418	10970	-369	-381	4	122	1	60	0.75	2.5
12/31/08	to	01/30/09	02/02/09	to 02/06/09	463	393	11363	-644	-644	4	115.8	1	70	1	1.5
01/07/09	to	02/06/09	02/09/09	to 02/13/09	1006	954	12316	-94	-94	3	335.3	1	40	2	1.25
01/14/09	to	02/13/09	02/16/09	to 02/20/09	256	221	12538	0	0	2	128	1	60	1.5	1
01/21/09	to	02/20/09	02/23/09	to 02/27/09	994	959	13496	0	0	2	497	1	30	2.5	2.25
01/28/09	to	02/27/09	03/02/09	to 03/06/09	338	320	13817	0	0	1	338	2	70	2.25	2
02/04/09	to	03/06/09	03/09/09	to 03/13/09	313	260	14078	-169	-169	3	104.3	1	70	1	1.25
02/11/09	to	03/13/09	03/16/09	to 03/20/09	656	551	14628	-575	-1044	6	109.3	1	60	0.75	1
02/18/09	to	03/20/09	03/23/09	to 03/27/09	206	188	14817	0	0	1	206	1	40	2	1.75
02/25/09	to	03/27/09	03/30/09	to 04/03/09			14817					1	60	1.5	1.25
03/04/09	to	04/03/09	04/06/09	to 04/10/09	(150)	(202)	14614	-344	-344	3	-50	2	60	1.25	2.75
03/11/09	to	04/10/09	04/13/09	to 04/17/09	13	(4)	14610	0	0	1	13	2	60	1.5	2.5
03/18/09	to	04/17/09	04/20/09	to 04/24/09	2031	1874	16484	-488	-606	9	225.7	2	40	0.25	2.5
03/25/09	to	04/24/09	04/27/09	to 05/01/09	(200)	(288)	16196	-194	-400	5	-40	2	40	2	2.75
04/01/09	to	05/01/09	05/04/09	to 05/08/09	1438	1316	17512	-656	-656	7	205.4	2	30	0.25	3
04/08/09	to	05/08/09	05/11/09	to 05/15/09	169	(58)	17453	-338	-913	13	13	2	20	0.75	3
04/15/09	to	05/15/09	05/18/09	to 05/22/09	3069	2946	20400	-256	-281	7	438.4	4	60	0.25	3
04/22/09	to	05/22/09	05/25/09	to 05/29/09	581	354	20753	-256	-419	13	44.7	3	40	0.5	3
04/29/09	to	05/29/09	06/01/09	to 06/05/09	(1881)	(2091)	18662	-1438	-2869	12	-156.8	3	70	0.25	3
05/06/09	to	06/05/09	06/08/09	to 06/12/09	244	122	18784	-1025	-1550	7	34.9	1	60	0.75	1
05/13/09	to	06/12/09	06/15/09	to 06/19/09	(1006)	(1128)	17655	-850	-1538	7	-143.7	2	60	1.75	2
05/20/09	to	06/19/09	06/22/09	to 06/26/09	400	382	18038	0	0	1	400	1	70	0.75	2.25
05/27/09	to	06/26/09	06/29/09	to 07/03/09	(81)	(98)	17939	0	0	1	-81	1	60	1	1.25
06/03/09	to	07/03/09	07/06/09	to 07/10/09	0	(70)	17869	-644	-1194	4	0	1	70	0.25	2.25
06/10/09	to	07/10/09	07/13/09	to 07/17/09	1088	1000	18870	-275	-275	5	217.6	1	70	0.25	1.5
06/17/09	to	07/17/09	07/20/09	to 07/24/09	69	34	18904	-88	-88	2	34.5	1	30	1	2.75
06/24/09	to	07/24/09	07/27/09	to 07/31/09	350	315	19218	-550	-550	2	175	1	60	0.5	1.75
07/01/09	to	07/31/09	08/03/09	to 08/07/09	(1306)	(1394)	17825	-988	-1531	5	-261.2	3	60	1.5	3
07/08/09	to	08/07/09	08/10/09	to 08/14/09	(825)	(948)	16878	-419	-1363	7	-117.9	1	70	0.25	0.75
07/15/09	to	08/14/09	08/17/09	to 08/21/09	906	854	17731	0	0	3	302	1	20	2.75	1.5
07/22/09	to	08/21/09	08/24/09	to 08/28/09	(231)	(301)	17430	-431	-475	4	-57.8	2	40	3	2
07/29/09	to	08/28/09	08/31/09	to 09/04/09	(288)	(323)	17107	-488	-488	2	-144	1	20	2.75	1.75
08/05/09	to	09/04/09	09/07/09	to 09/11/09	81	46	17153	0	0	2	40.5	1	20	2.75	1.75
08/12/09	to	09/11/09	09/14/09	to 09/18/09	381	346	17499	0	0	2	190.5	1	40	1.75	0.75
08/19/09	to	09/18/09	09/21/09	to 09/25/09	919	884	18383	0	0	2	459.5	2	50	2.75	1.25
08/26/09	to	09/25/09	09/28/09	to 10/02/09	(188)	(258)	18125	-263	-300	4	-47	1	40	1.75	0.75
09/02/09	to	10/02/09	10/05/09	to 10/09/09	644	626	18752	0	0	1	644	1	40	1.5	0.75
09/09/09	to	10/09/09	10/12/09	to 10/16/09	(194)	(212)	18540	0	0	1	-194	1	40	1.25	0.75
09/16/09	to	10/16/09	10/19/09	to 10/23/09	581	528	19068	-94	-94	3	193.7	1	60	0.75	0.5
09/23/09	to	10/23/09	10/26/09	to 10/30/09	(81)	(151)	18918	-394	-456	4	-20.2	1	70	1	0.5
09/30/09	to	10/30/09	11/02/09	to 11/06/09	(431)	(466)	18452	-419	-431	2	-215.5	1	50	2	0.5
10/07/09	to	11/06/09	11/09/09	to 11/13/09	(594)	(682)	17770	-550	-694	5	-118.8	1	50	2	0.5
10/14/09	to	11/13/09	11/16/09	to 11/20/09	500	448	18218	-131	-231	3	166.7	1	70	0.25	0.75

Test Dates			Out-Of_Sample Dates		osnp	NOnp$17.5	NetEq	ollt	odd	ont	avosnp	pw	len	vup	vdn
10/21/09	to	11/20/09	11/23/09	to 11/27/09	506	436	18654	-325	-325	4	126.5	2	70	0.5	1.5
10/28/09	to	11/27/09	11/30/09	to 12/04/09	900	865	19518	0	0	2	450	1	60	0.5	1
11/04/09	to	12/04/09	12/07/09	to 12/11/09	406	318	19837	-56	-63	5	81.2	1	40	0.5	1.25
11/11/09	to	12/11/09	12/14/09	to 12/18/09	350	298	20134	0	0	3	116.7	1	20	1	2.5
11/18/09	to	12/18/09	12/21/09	to 12/25/09	(138)	(156)	19979	0	0	1	-138	1	20	1	2.5
11/24/09	to	12/24/09	12/28/09	to 12/31/09	800	730	20709	-169	-306	4	200	1	20	1	2.5

osnp = Weekly Out-of-sample gross profit in $
NOnp$17.5 = Weekly Out-Of-Sample Net Profit in $ = **osnp-ont*25**.
NetEq = running sum of the weekly 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 week.
avosnp = The average out-of-sample profit per trade for that week $

Figure 2 Walk Forward Out-Of-Sample Performance Summary for BP Fading Memory Polynomial Velocity System 1 minute bar chart of es from 11/24/08-11/26/2008

Appendix 1: n^{th} Order Fading Memory Polynomial Next Bar's Forecast Math

What is The N^{th} Order Fading Memory Polynomial ?

This is a mathematical technique that fits a n^{th} order polynomial to the last ***T*** price bars but calculates the coefficients of the polynomial such that the error between the current n^{th} order polynomial and the current bar is weighted much higher than the error between the price T bars ago and the value of the n^{th} order polynomial T bars ago. As an example, if the latest price is at time t and the price made a turn at time bar t-10 , then we do not want prices prior to t-10 effecting the crrent polynomial fit as much. As will be shown the most familiar case of this fading memory technique is the exponential moving average. The fading memory technique is in contrast to the Least Squares Polynomial fit, which weights all past errors between the polynomial and the price bar equally.

Consider a time series x(t) where t is an integer value (a price bar number) like the number of days or minutes, etc from some starting time. Suppose we want to find at some given time some *n*th-degree polynomial that fits the data well at current and recent prices but ignores the fit as we move into the distant past. One way to construct this type of fit would be to weight the past data with a number that got smaller and smaller the further back in time we went. If we let the polynomial function be represented by the symbol **p(t-τ)** where **p(t-0)** is the current value of the polynomial, **p(t-1)** is the previous value of the polynomial, etc., then an error function can be formed that consists of the weighted sum of the squared difference between the price series **x(t-τ)** and the polynomial p(t-τ) given by

$$\mathbf{error = \Sigma\beta^{\tau}(x(t-\tau) - p(t-\tau))^2 \quad \tau=0 \text{ to } \infty \quad (1)}$$

where $\mathbf{0 < \beta < 1}$ **and** $\mathbf{\beta^{\tau}}$ is much much less than **1** for large $\boldsymbol{\tau}$**.**

It turns out that if we let the n^{th} degree polynomial **p(t-τ)** be constructed as a linear combination of orthogonal polynomials called Meixner polynomials then minimizing the error with respect to the coefficients of the orthogonal polynomials yields the best estimate of **x(t-τ)** as $\mathbf{x_{est}(t-\tau)}$ and given by the equation

$$x_{est}(t-\tau) = (1-\beta)\sum_{k=0}^{n}\beta^k b_{k,t}\Phi_k(t)|_{\tau} \qquad (2)$$

Where

$$\Phi_n(t) = \sum_{k=0}^{n}\binom{n}{k}\binom{t}{k}z^k$$

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

$$b_{j,t} = \sum_{k=0}^{\infty}\beta^k\Phi_j(k)x(t-k)$$

$$\mathbf{z = 1 - 1/\beta}$$

where n is the polynomial degree, $\mathbf{\Phi_k(\tau)}$ are the Meixner polynomials of degree k (k=0 to n), and $\mathbf{b_k(t)}$ are the coefficients that minimize the error of equation (1). Generally the summation for $\mathbf{b_j(t)}$ can be terminated when $\mathbf{\beta^k}$ <<1.

Appendix 1: n^{th} Order Fading Memory Polynomial Next Bar's Forecast Math

For the exact mathematical solutions that produce equation (2) and the mathematical descriptions of the Meixner polynomials refer to Reference 1.

To yield the 1 day ahead prediction the above equation becomes;

$$x_{est}(t+1)=(1-\beta)\sum \beta^k b_{k,t}\Phi_k(-1) \quad k=0 \text{ to } n \qquad (3)$$

After some algebraic manipulation with the Meixner polynomials the $b_{k,t}$ coefficients satisfy the following recursive relationship.(see Reference 1)

$$b_{k,t}=\beta\, b_{k,t-1}+b_{k-1,t}-b_{k-1,t-1}$$

One case is of immediate interest where the polynomial is a constant, that is ***n***=0.

For this case the solution to equation (3) can be found after some algebraic manipulation to be:

$$X0_{est}=\beta*X0_{est}[1]+(1-\beta)*x(t) \qquad (4)$$

Where $X0_{est}[1]$ is the previous estimated value, **x(t)** is the current bar's price and where the 0 in $X0_{est}$ indicates that we are estimating a polynomial of degree 0 or simply a constant. If a change of variables is made letting $\alpha=(1-\beta)$ then equation (4) becomes:

$$X0_{est}=(1-\alpha)*X0_{est}[1]+\alpha*x(t) \qquad (5)$$

This is the familiar formula for the exponential moving average.

Higher orders of ***n*** don't yield such compact solutions as the case where ***n***=0 .equations

$$P_F(T+1) = (1-\beta)*[b_{0,t}\,\phi_{0|t=-1} + \beta b_{1,t}*\phi_{1|t=-1} + \beta^2 b_{2,t}*\phi_{2|t=-1} +\ldots+ \beta^n b_{n,t}*\phi_{n|t=-1}]$$

$$\text{Velocity} = (dP_F/dt)_{(T=-1)} = (1-\beta)[\beta b_{1,t}*(d\phi_1/dt)_{|t=-1} + \beta^2 b_{2,t}*(d\phi_2/dt)_{|t=-1} +\ldots+ \beta^n b_{n,t}*(d\phi_n/dt)_{|t=-1)}]$$

$$\text{Accel} = (d^2P_F/d^2t)_{(T=-1)} = (1-\beta)[\beta^2 b_{2,t}*(d^2\phi_2/d^2t)_{|t=-1} + \beta^2 b_{2,t}*(d^2\phi_3/d^2t)_{|t=-1} +\ldots+ \beta^n b_{n,t}*(d^2\phi_n/d^2t)_{|t=-1}]$$

The n^{th} Order Fading Memory Forecast Next Bar's Price System Defined

The least squares forecast is constructed by solving for the coefficients $b_0, b_1, b_2, \ldots, b_n$ recursively at each bar using the last **T** bars of closing prices and the Discrete Orthogonal Meixner Polynomial equations above. Then $P_F(T+1)$ is constructed from the equation above and plotted under the price chart. In general what we will be doing is following the plotted curve of P_F which is calculated at each bar from the previous T bars. When the curve increases by a percentage amount ***pctup*** from the previous prior low of the curve we will go long. When the curve falls by the percentage amount ***pctdn*** from the previous prior high of the curve we will go short

Buy Rule:

- **IF P_f** has moved up by more than the percentage amount of ***pctup*** from the lowest low recorded in P_f while short then buy at the market.

Sell Rule:

- **IF P_f** has moved down by more than the percentage amount ***pctdn*** from the highest high recorded in P_f while long then sell at the market.

The n^{th} Order Fading Memory Forecast Next Bar's Velocity System Defined

The least squares forecast is constructed by solving for the coefficients $b_0, b_1, b_2, \ , \ , b_n$ recursively at each bar using the last **T** bars of closing prices and the Discrete Orthogonal Meixner Polynomial equations above. Then **Velocity** = $dP_F(T+1)/dt$ 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.

The n^{th} Order Fading Memory Forecast Next Bar's Acceleration System Defined

The least squares forecast is constructed by solving for the coefficients $b_0, b_1, b_2, \ , \ , b_n$ recursively at each bar using the last **T** bars of closing prices and the Discrete Orthogonal Meixner Polynomial equations above.. Then **Acceleration** = $d^2P_F(T+1)/d^2t$ is constructed from the acceleration equation above and plotted under the price chart. In general what we will be doing is following the plotted curve of **Acceleration** which is calculated at each bar from the previous T bars. When the acceleration is greater than a threshold amount ***aup*** we will go long. When the velocity is less than a threshold amount ***-adn*** we will go short.

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

Sell Rule:
IF acceleration is less than the threshold amount ***-adn*** 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 of All Trades Profit/Losses
tnp	Total Net Profits
nT	# of Trades in Test Section
pP	% Profitable Trades in Test Section
PF	Profit Factor
std	Standard Deviation of Trades
t	Student t Statistic for Trades
mlb	Median of all the losing bars in losing trades
lbr	Total Losing Bars
mwb	Median of all the winning bars in winning trades
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 2^{nd} Order Polynomial Line Where Equity 2^{nd} Order Line = b1*t + b2*t^2
dy	Velocity Of Equity 2^{nd} Order Polynomial Line evaluated on last trade i=nT.
b2	Acceleration of Equity 2^{nd} Order Polynomial Line evaluated on last trade
r22	Equity 2^{nd} 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 2^{nd} Order Polynomial Line/1000.

Appendix 2 Power Walk Forward Optimizer Performance Metrics

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-p 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 $Equity_{est} = a0+b0*t$. $Equity_{est}$ 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.
- **eq**10 - **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	BP1FadmV	d080208	d091231	tnp>0	nT<50	t100					a(40064)	s11831	f1922							c=$17.5	
2	Filter	Metric	tOnp	mOnp	aOnp	aanp	aOnT	B0	%P	t	std	llw	eqDD	lr	#	b00	Blw	BE	ndd	tOnpNet	Prob
3	b20mru-p|:10::50	mp-rd	28164	256	293	66.1	4.4	(2.0)	0.65	3.17	907.1	-3069	-4401	3	96	299	26	38.2	21.3	20709	1.40E-07
4	t20mp-rd|:10::50	mru-p	24300	306	253	56.0	4.5	0.3	0.62	2.42	1024.2	-3213	-4626	3	96	232	21	65.5	27.7	16705	8.00E-07
5	b20mru-p|:10::50	mtnp	24437	206	252	53.2	4.7	(1.4)	0.62	2.65	935.6	-3213	-6076	4	97	251	8	55.2	37	16404	9.08E-07
6	t20pP|:10::50	mru-p	23008	238	237	55.3	4.3	0.2	0.61	2.42	963.6	-2613	-5126	3	97	221	22	66	32.6	15728	1.20E-06
7	t20mp-rd|:10::50	llt	22493	256	232	51.8	4.5	0.0	0.63	2.57	889.5	-2725	-4544	3	97	217	22	58.9	30.5	14898	1.70E-06
8	t20mKr|:10::50	dev	17191	169	287	130.2	2.2	(2.4)	0.7	3.67	605.3	-1056	-1199	4	60	194	24	17.9	8.1	14881	1.71E-06

The Filter Output Columns are defined as follows

Row 1 is the PWFO Stub, File Start Date(2/8/08), File End Date(12/31/09), Number of weeks(t100) and Cost plus slippage of round trip trade (c=$17.5).

Filter = The filter that was run. For example, b20mru-p|:10::50 - mp-rd

Metric = The PWFO performance metric (defined on Appendix 2). For this filter, b20mru-p|:10::50 Metric=mp-rd (median of final trade profit/loss-maximum rundown of trade).

This b20mru-p|:10::50-mp-rd filter produced the following average 100 week statistics on this line(row 3).

tOnp = Total out-of-sample(oos) net profit for these 100 weeks.

mOsp = median oos net profit for the 100 weeks

aOsp = Average oss net profit for the 100 weeks

aanp = Average oos profit per trade

aOnT = Average number of oos trades per week

B0 = The 100 week trend of the out-of-sample weekly profits

%P = The percentage of oos weeks that were profitable

t = The student t statistic for the 100 weekly 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 100 weekly oos profits

llw = The largest losing oos week

eqDD = The oos equity drawdown

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

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

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

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

BE = Break even weeks. Assuming the average and standard deviation are from a normal distribution, this is the number of weeks 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.