International Journal of Computer Information Systems,
Vol. 2, No. 3, 2011

GARCH -Monte-Carlo Simulation Models with Wavelets Decomposition Algorithm for Stock Returns Prediction

Eleftherios Giovanis
Department of Economics
Royal Holloway University of London
Egham, England
Eleftherios.Giovanis.2010@live.rhul.ac.uk

***Abstract*— In this paper we examine four different approaches in trading rules for stock returns. More specifically we examine the popular procedures in technical analysis, which are the moving average and the Moving Average Convergence-Divergence (MACD) oscillator. The third approach is the simple random walk autoregressive model and the fourth model we propose is a Generalized Autoregressive Conditional Heteroskedasticity (GARCH) regression with wavelets decomposition and Monte-Carlo simulations algorithm developed in MATLAB. We examine five major stock market index returns for a testing forecasting period of 10 days ahead. We conclude that moving average and MACD might lead to net profits, but not in all cases, therefore are not consistent procedures. Furthermore, moving average 1-30 provides the best results. On the other hand random walk autoregressive model leads in all cases to net losses. Finally, the model we propose not only leads always to net profits, but also to significant higher profits in three stock indices than the respective conventional technical analysis tools.**

Keywords- Forecasting; MACD; MATLAB; Moving Average; Stock Returns; Random Walk

I. INTRODUCTION

Two of the most used and popular trading rules used by financial traders and fund managers are the moving average and the MACD oscillator. Moving average is one method of technical analysis among others, which during the last years has gained a significant increase of interest in the academic world too. Some empirical studies like among others [1]-[3] provide evidences of profitability by using technical trading rules. Many other applications have been developed in technical analysis and trading rules modeling, as Fibonacci retracement, candlesticks, various oscillators and momentum indicators among others [4].

Since the decade of 1990 and especially the last years new approaches have been applied with superior results than the traditional technical analysis tools and traditional statistic and econometric approaches [5]-[11]. Some of them are neural networks, fuzzy logic, and genetic algorithms and wavelets analysis. In this paper we propose a computation model where the full programming routine written in MATLAB software is available and provided in the appendix, which combines the well know GARCH model with Monte-Carlo simulation and wavelet decomposition.

The structure of this paper is: The second section presents a brief description on the methodology of the four approaches we examine. In the third section the data sample is provided. In the fourth section the empirical results are reported and finally, in section five we discuss the main conclusions of our findings.

II. METHODOLOGY

A. Moving average

The simple moving average is defined as:

$$SMA_t = (1/L)\sum_{i=0}^{L-1} p_{t-i} \quad (1)$$

Eq. (1) can be used as a trading rule, which generates a buy signal when the current stock price is higher or above the moving average and a sell signal when it is below. Eq. (1) is the simple moving average, while there are also additional modifications in moving average, as the exponential, the square root weighted, the weighted or the linear moving average. We examine all the mentioned possible modifications of moving average, but we present only the results of the simple moving average as the conclusions are the same, as also many financial traders claim that. We examine three possible moving averages. The 1-30, 1-50 and 1-200, where the short period is 1 day and the long periods are respectively 30, 50 and 200 days. We should mention that we refer to days,

because the data used in analysis of this paper are on daily frequency.

B Moving average convergence / divergence (MACD)

The MACD oscillator is computed by subtraction of the 26-period exponential moving average with the 12-period exponential moving average. In the first case the smoothing factor *a* is 0.075 and in the second case is 0.15 Then the 9-period exponential moving average of MACD, with smoothing factor α equal with 0.20, is used as the signal line. So if the MACD and 9-period moving average lines are crossed and if the MACD line falls below the other, then a sell signal is generated, while in the opposite situation we have a buy signal. More specifically is defined as:

$$MACD_t = EMA_{t\text{-}12} - EMA_{t\text{-}26} \quad (2)$$

If MACD line rises above the 9-period EMA line then buy

Else If MACD line falls below the 9-period EMA line then sell

, where EMA denotes the Exponential Moving Average and the formula is:

$$S_t = a \cdot Y_t + (1-a) \cdot S_{t-1} \quad (3)$$

, where the coefficient α represents the degree of weighting decrease, a constant smoothing factor between 0 and 1. A higher α discounts older observations faster. Alternatively, α may be expressed in terms of *N* time periods, where $\alpha = 2/(N+1)$. Y_t is the observation at a time period *t* and S_t is the value of the EMA at period *t-1*.

C. *Random walk autoregressive model*

The autoregressive (AR) models are widely used for time series analysis by economists, which is a random process. We examine an autoregressive model of order one AR(1) and is defined as:

$$R_t = c + \phi R_{t-1} + \varepsilon_t \quad (4)$$

, where R_t and R_{t-1} denote the current stock returns and stock returns with one lag respectively, *c* is the constant, φ is the estimated parameter and ε_t is the disturbance term or the white noise. Also the stock returns R_t are defined as:

$$R_t = \log(P_t - P_{t-1}) \quad (5)$$

, where P_t and P_{t-1} is the current price and the price of the previous period respectively. In the following we predict the value for the next period and we form the simple trading rules:

If the predicted value is positive then we buy

Else If the predicted value is negative then we sell

We should mention that in the above rules we do not incorporate the comparison of predicted values with actual stock returns or moving average, as the disadvantage of AR models is that generate very low forecasts, which are not close to actual stock returns, but are in many case close to zero. In addition the computational algorithm we propose generates in the most case high values and very close to the actual stock returns.

D. GARCH with Monte Carlo simulations and wavelets decomposition computational algorithm.

In this part we present the computational algorithm we propose in a simple manner. There are two alternatives approaches and we describe the first one, while the second is almost the same. The steps of the computation are:

a) First we decompose data applying discrete wavelet transformation (DWT) with Daubechies wavelets on the initial stock returns [12]. More specifically the DWT of a signal *X* for 1 level is defined as:

$$y_{highpass}[n] = \sum_{k=-\infty}^{\infty} X[k] g[2n - k] \quad (6)$$

$$y_{lowpass}[n] = \sum_{k=-\infty}^{\infty} X[k] h[2n - k] \quad (7)$$

,where *g* and *h* denote the impulse response and the filter outputs are sub-sampled by two. The final convolution is:

$$y_{highpass} = X \cdot g \downarrow 2 \quad (8)$$

$$y_{lowpass} = X \cdot h \downarrow 2 \quad (9)$$

The DWT is computed by successive lowpass and highpass filtering of the discrete-time domain signal. For the cases of DAX, CAC-40 and NIKKEI-225 indices we used five-level decomposition and for S&P 500 and FTSE-100 indices we used six-level decomposition for the optimum forecasting performance. Because in all cases the data we use are too long, there is need to take high levels of decomposition. Specifically for data longer than 4,000 levels above 4 are necessary to obtain the optimum results. In fig. 1 a 5-level decomposition tree is presented. Similarly, a 6-level can be drawn.

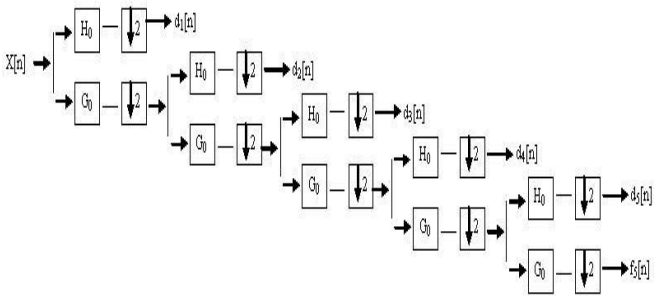

Fig. 1 Five-level wavelet decomposition tree

Daubechies wavelets are defined by the polynomial:

$$P_N(\mathrm{y}) = \sum_{k=0}^{N-1} \binom{\kappa + N - 1}{\kappa} y^k = 1 + N_y + \frac{N(N+1)}{2} y^2 + \ldots\ldots + \binom{2N-2}{N-1} y^{N-1} \quad (10)$$

$$\binom{\kappa + N - 1}{\kappa} \quad (11)$$

, where denotes the binomial coefficients. Because the further description of Daubechies wavelets and their properties are too long, more details can be found in [13].

b) The second step is to estimate a GARCH (1,1) process where the mean and variance equations are presented in Eq. (12)- (14) respectively.

$$R_t = c + \varepsilon_t \quad (12)$$

$$\varepsilon_t \sim (0, \sigma^2) \quad (13)$$

$$\sigma_t^{\,2} = \omega + a_0 u_{t-1}^{\,2} + a_1 \sigma_{t-1}^{\,2} \quad (14)$$

, where R_t is defined as previously, c is the constant and ε_t is the disturbance term which follows the distribution in (13). Parameter ω is the constant of the variance equation, α_0 is the coefficient of ARCH effects and similarly α_1 is the coefficient of ARCH effects.

c) The third step is to simulate the residuals or the innovations of GARCH (1,1) estimation with Monte-Carlo. First we should set up the parameters for the simulation process. First we should define the random number generator seed for reproducibility, which is set up at this way that every time that we run the algorithm the generator is reset to its initial state. Because we have long sample we set up the simulated samples roughly 4 times greater than the initial data. So for all stock indices, except from S&P 500, we have data ranging 4,600 through 6,500 we obtain a simulated sample equal with 20,000. Similarly, for the S&P 500 where the initial data are roughly 15,000 we take a sample equal with 60,000.

d) The final step is to take a random permutation sample with length equal with the initial data sample and then this sample is selected from the residuals or innovations after a number of replications. The forecast period is defined as 10 days and the replication number is set up at 100. The final selected sample denotes the predicted values. The trading rule is defined exactly as in the case of random walk autoregressive model:

If the predicted value is positive then we buy

Else If the predicted value is negative then we sell

We should mention that with this algorithm we have the ability to forecast for a period than one ahead, as we do with the other traditional technical analysis approaches. So we have the opportunity to observe the market for a10 period ahead, as we define this period as the forecasting test sample and to take the appropriate measures and trading strategic decisions in order to maximize our portfolio. Furthermore, this procedure can be applied to forecast longer periods as 100 days with net profits, while for this period of 100 days MACD and moving averages leads in the most cases to net losses, expect from moving average 1-200. Also, as we mentioned above, we cannot forecast a long ahead period as 10 or 100 samples with the traditional approaches in technical analysis, so each time we have to wait for the new actual price to be formulated. In addition there are technical analysis tools which show the trend process in the long period ahead, but usually these approaches are more appropriate in the short-term periods. Also, the second approach which is not presented here can be split in to two samples. The first sample contains the first half data and the second sample the remained data. We follow the above steps we described for the first sample and the same process is applied for the second sample. Then we test both samples in order to investigate which one gives the best results.

E. Estimation of Net Profits-Losses

In order to test which of the approaches we examine is the most profitable we apply the following formula procedure to compute the net profits or losses. We discriminate three cases

in stock trading. The first one is the case when the stock returns are positive, the second one when are zero and finally when stock returns are negative. We define a dummy variable $D_{t+1.}$

If $r_{t+1} > 0$ then

$$D_{t+1} = 1 \quad (15)$$

If $r_{t+1} = 0$ then

$$D_{t+1} = D_t \quad (16)$$

If $r_{t+1} < 0$ then

$$D_{t+1} = -1 \quad (17)$$

The net profits-losses are given by relation:

$$R_{t+1} = D_{t+1} \cdot (P_{t+1} - P_t) - c(\left|D_{t+1} - D_t\right| \cdot P_t) \quad (18)$$

Variables D_{t+1} and D_t are defined as previously which is the forecasting and the current signal respectively, variables P_{t+1} and P_t are the future and current stock prices respectively and c is the commission rate which is charged for the services of trading and depending on whether it's an intraday trading or electronic trading or depending on the different rates offered by various companies and financial trading institutions in different countries. Usually the commissions rates can be varied between 0.008 and 0.01, but nowadays in the *Forex* trading the most companies do mot charge any commission rate or this rate can be very low. Besides that we set up a very low rate equal with 0.001. Next we present the most used statistical test for the investment measure [14], which is the *Sharpe Ratio* and can be computed as:

$$SR = \frac{\mu(r_i) - r_f}{\sigma_i} \quad (19)$$

, where r_i is the stock returns in the period i, r_f is the risk free interest rate, μ is the average and σ_i is the standard deviation of the net profits-losses in the forecasting period. As long as the average is higher and the standard deviation is lower the higher the Sharpe Ratio is, which indicates a higher and more efficient investment performance. The risk free rate is omitted as it can be very low and insignificant.

III. DATA

We examine four major stock market indices based on daily data of closed price returns. These are the S&P 500 for U.S.A., FTSE-100 index for UK, DAX index for Germany, CAC-40 for France and NIKKEI-225 for Japan. The estimation starting period is 3, January 1950 for S&P 500, 2, April 1984 for FTSE-100, 26, November 1990 for DAX, 1, March 1990 for CAC-40 and 4, January 1984 for NIKKEI-225. The ending time period for the estimation is common for all estimations and it is 30, October 2009. The remaining period from 2, November through 13, November 2009 is obtained as the forecasting test period, which is actually 10 trading days. We should mention that the forecasting is done each time for one period ahead with the traditional approaches. So, each day we find the forecasting value and the new actual value is added each time in our sample. But in order to investigate the forecasting power of each procedure we examine a 10 days period. On the contrary with the computational algorithm we propose we apply forecasts for 10 periods ahead, as we mentioned in the previous section.

IV. EMPIRICAL RESULTS

In Table 1 we present the correct percentage concerning the prediction of the correct sign of stock returns. We observe that the model we propose does not outperform the other approaches, but as we mentioned above it is not always enough to predict the correct sign. The important is to predict the correct sign of significant changes. For example there is a great difference of predict correctly the sign of a change of 0.023 percentage decrease than a 0.23 per cent. More specifically, in table 2 we observe that the moving average 1-200 and the simulated GARCH model have the same percentage of correct prediction, but the net profits generated by the last model are much more superior.

Also, the random walk model in all cases presents net losses, while we get net profits only in DAX and NIKKEI-225 indices for MACD oscillator and CAC-40 and NIKKEI-225 indices for moving average 1-200 respectively. Net profits are reported in three and four stock indices with moving average 1-50 and 1-30 respectively. So based on the traditional technical analysis approaches we observe that moving average 1-30 presents the best forecasting performance. On the other hand the model we propose leads always to net profits, which are higher than the other procedures in four stock indices, where in CAC-40 and NIKKEI-225 the profits are impressively higher. This indicates that wavelets and GARCH Monte-Carlo simulations are able to capture the significant changes, increased or decreased values, in stock returns.

In Table III we present the estimated results of Sharpe ratios, which are positive in all cases with the wavelets GARCH model. As we mentioned, this model can be used for the prediction for much longer periods, as 100 sample data, with net profits. We recommend that this algorithm can be improved by many ways, as taking lag values of stock returns as independent variables, or setting up fuzzy rules with genetic algorithms optimization, which can be trained also with neural networks.

TABLE I. Correct predicted percentage of stock returns sign

Index	***Moving Average 1-200***	***Moving Average 1-50***	***Moving Average 1-30***	***Random Walk***	***MACD***	***GARCH-MONTE-CARLO***
S&P 500	40.00%	40.00%	50.00%	40.00%	50.00%	50.00%
FTSE-100	10.00%	80.00%	90.00%	60.00%	40.00%	70.00%
CAC-40	60.00%	30.00%	50.00%	40.00%	30.00%	70.00%
DAX	40.00%	60.00%	60.00%	50.00%	50.00%	40.00%
NIKKEI 225	80.00%	70.00%	50.00%	30.00%	50.00%	80.00%

TABLE II. Net Profits (Losses)

Index	***Moving Average 1-200***	***Moving Average 1-50***	***Moving Average 1-30***	***Random Walk***	***MACD***	***GARCH-MONTE-CARLO***
S&P 500	-47.99	13.49	35.81	-53.63	-38.34	44.46
FTSE-100	-536.51	236.53	457.30	-156.08	-16.78	212.15
CAC-40	491.34	-712.04	233.26	-63.35	-730.43	631.08
DAX	-362.24	176.27	176.27	-334.27	217.70	191.85
NIKKEI 225	204.18	-23.14	-314.86	-550.20	201.43	735.79

TABLE III. Shape Ratios of forecasts

Index	*Moving Average 1-200*	*Moving Average 1-50*	*Moving Average 1-30*	*Random Walk*	*MACD*	***GARCH-MONTE-CARLO***
S&P 500	-0.0940	0.0270	0.0719	-0.1060	-0.0722	0.0882
FTSE-100	-0.2313	0.1242	0.2586	-0.0763	-0.0075	0.1040
CAC-40	0.1098	-0.1497	0.0485	-0.0132	-0.1534	0.1329
DAX	-0.1221	0.0625	0.0625	-0.3683	0.0729	0.0673
NIKKEI 225	0.0597	-0.0065	-0.0876	-0.1268	0.0584	0.2306

V. CONCLUSIONS

In the paper we proposed an alternative tool as a trading rule strategy, which is a discrete wavelet transformation of stock returns and univariate GARCH with Monte-Carlo simulated regressions. Additionally we applied two popular technical analysis approaches, the moving average and the MACD as also we examined a random walk autoregressive model and we have shown that theses procedures do not always lead to net profits. The model we propose, might not present always superior profits than the traditional technical analysis trading rules, but we should notice that presents always net profits, so it is more consistent, as also in two stock indices we examined, the profits are much more superior to the profits generated by the other procedures.

REFERENCES

[1] W. Brock, J. Lakonish, and B. LeBaron, "Simple technical rules and the stochastic properties of stock returns", Journal of Finance, vol. 47, pp. 1731–1764, 1992

[2] K. Chang and C. L. Osler, "Methodical madness: technical analysis and the irrationality of exchange-rate forecasts", Economic Journal, vol. 109, pp. 636–661, 1999

[3] R. Levich and R. T., "The significance of technical trading-rules profits in the foreign exchange market: a bootstrap approach", Journal of International Money and Finance, vol. 12, pp. 451–474, 1993

[4] B. S. Achelis, Technical Analysis from A to Z, 3rd ed. McGrawHill, New York, 2003

[5] H. White, "Economic prediction using neural networks: the case of the IBM daily stock returns", Proceedings IEEE International Conference on Neural Networks San Diego, California, USA, July 24-27, 1988

[6] T. Kimoto, K. Asakawa, M. Yoda and T. Masakazu, "Stock market prediction system with modular neural networks", Proceedings IEEE International Joint Conference on Neural Networks, Piscataway, NJ, USA, June 17-21, 1990

[7] Y. Yoon and G. Swales, "Predicting stock price performance: a neural network approach", Proceedings IEEE 24th Annual Hawaii International Conference of Systems Sciences, January 8-11, 1991

[8] A. Skabar and I. Cloete, "Neural networks, financial trading and the efficient markets hypothesis", Proceedings of the twenty-fifth Australasian conference on Computer science, **4**, pp. 241-249, 2002

[9] X. Guo, L. Xun and N. Li, "Automatically Recognizing Stock Patterns Using RPCL Neural Networks", Proceedings ISKE International Conference on Intelligent Systems and Knowledge Engineering. Advances in Intelligent Systems Research, Chengdu, China, October 15, 2007

[10] A. Ghandar, Z. Michalewicz, M. Schmidt and Z. Ralf, "Computational Intelligence for Evolving Trading Rules Export Evolutionary Computation", IEEE Transactions on In Evolutionary Computation, vol. 13, no. 1, pp. 71-86, 2009

[11] U. A. Khan, T. K. Bandopadhyaya and S. Sharma, "Classification of Stocks Using Self Organizing Map. International Journal of Soft Computing Applications", vol. 4, pp. 19-24, 2009

[12] I. Daubechies, "Wavelets and other phase space localization methods", The Proceedings of the International Congress of Mathematicians, Zürich, Switzerland, August 3-11, 1994

[13] I. Daubechies, "Ten Lectures on Wavelets", CBMS-NSF Regional Conference Series in Applied Mathematics Lecture Notes, vol. 61, Philadelphia, Pennsylvania, Society for Industrial and Applied Mathematics, 1992

[14] W.F. Sharpe, "Mutual Fund Performance". Journal of Business, vol. 39, no. S1, pp. 119–138, 1966

APPENDIX

MATLAB routine (script file)

```
clear all;
% Load input data
load file.mat
method=3 % 1 for moving average, 2 for MACD, 3 for GARCH and wavelets where
% gives a 'buy' signal for positive predicted values and 'sell' signal for
```

```
% negative values, 4 for the same with 3 but use moving average of
% predicted values, 5,6,7 and 8 the same with 1,2,3 and 4 respectively but
%for real application and not for testing or backtesting.
lag=50 % Define the length of moving average. Usually 10,20,30,50, 70,100 and
% 200 are used.
factor='e' % 0 for simple, 0.5 for square root weighted moving average,
% 1 for linear moving average, 2 for square weighted moving average
% end 'e' for exponential
risk_free=0.001
length_test=100 % length of sample for testing. This script is used for testing. If you
% wish to apply for future purposes set up the value of length_test=0
decomposition_tree=1 % lenght of decomposition tree
M=length_test; % length of predicted data
if method==1
train_data=data(1:end-length_test-1,:)
[Short,Long]=movavg(train_data,1,lag,factor)
test_sample=data(end-length_test:end,:)
test_long=Long(end-length_test-1:end,:)
stock=data(end-length_test-1:end,:)
[nk1,ni]=size(test_sample)
for kk=1:nk1

if test_sample(kk,:)>test_long(kk,:)
s(kk,:)=1 % buy
elseif test_sample(kk,:)<test_long(kk,:)
   s(kk,:)=-1 %sell
end
end
for jj=2:nk1
total(jj,:)=s(jj)*(stock(jj)-stock(jj-1))-0.001*(abs(s(jj)-s(jj-1))*stock(jj))
end
profit=sum(total)
average=mean(total)
standard_deviation=std(total)
sharpe_ratio=(average)/standard_deviation
elseif method==2
train_data=data(1:end-length_test-1,:)
[macdvec, nineperma] = macd(train_data)
test_sample=macdvec(end-length_test:end,:)
test_macd=nineperma(end-length_test:end,:)
stock=data(end-length_test-1:end,:)
[nk1,ni]=size(test_sample)
for kk=1:nk1
if test_sample(kk,:)>test_macd(kk,:)
s(kk,:)=1 % buy
elseif test_sample(kk,:)<test_macd(kk,:)
s(kk,:)=-1 % sell
end
end
for jj=2:nk1
total(jj,:)=s(jj)*(stock(jj)-stock(jj-1))-0.001*(abs(s(jj)-s(jj-1))*stock(jj))
end
profit=sum(total)
average=mean(total)
standard_deviation=std(total)
sharpe_ratio=(average)/standard_deviation
elseif method==3
y=price2ret(data)
N=length(y)
% We decompose our data with function db3
[XX,l] = wavedec(y,decomposition_tree,'db3');
% We define GARCH (1,1) process
[Kappa, Alpha, Beta] = ugarch(XX, 1, 1)
% We set the random number generator seed for reproducability
randn('state', 0)
NumSamples = 20000;
firstpoint=length_test
% We simulate the process with Monte Carlo
[U , H] = ugarchsim(Kappa, Alpha, Beta, NumSamples);
% Length of vector
%V=1%length(data);
% From current day we extract firstpoint data randomly selected
currentprice = randperm(N-M);
currentprice= currentprice+N;
for j=1:firstpoint
Y1 = currentprice(j);
Y0 = Y1-N+1;
p = U(Y0:Y1);
p = p(:);
Y1(1,:) = p(1,:);
prediction = U(Y1+1:Y1+M);
end
[nk1,ni]=size(prediction)
for kk=1:nk1
if prediction(kk,:)>0
s(kk,:)=1 % buy
elseif prediction(kk,:)<0
s(kk,:)=-1 % sell
end
end
stock=data(end-length_test:end,:)
for jj=2:nk1
total(jj,:)=s(jj)*(stock(jj)-stock(jj-1))-0.001*(abs(s(jj)-s(jj-1))*stock(jj))
end
profit=sum(total)
average=mean(total)
standard_deviation=std(total)
sharpe_ratio=(average)/standard_deviation
elseif method==4
N=length(data)
% We decompose our data with function db3
[XX,l] = wavedec(data,decomposition_tree,'db3');
train_data=XX(1:end-length_test-1,:)
[Short,Long]=movavg(train_data,1,lag,factor)
test_sample=data(end-length_test:end,:)
test_long=Long(end-length_test-1:end,:)
stock=data(end-length_test-1:end,:)
[nk1,ni]=size(test_sample)
for kk=1:nk1
if test_sample(kk,:)>test_long(kk,:)
s(kk,:)=1 % buy
elseif test_sample(kk,:)<test_long(kk,:)
s(kk,:)=-1 % sell
end
end
for jj=2:nk1
total(jj,:)=s(jj)*(stock(jj)-stock(jj-1))-0.001*(abs(s(jj)-s(jj-1))*stock(jj))
```

```
end
profit=sum(total)
average=mean(total)
standard_deviation=std(total)
sharpe_ratio=(average)/standard_deviation
elseif method==5
[Short,Long]=movavg(data,1,lag,factor)
if data (end,:)> Long (end,:)
s=1 % buy
elseif data (end,:)< Long (end,:)
s=-1 % sell
end
elseif method==6
[macdvec, nineperma] = macd(data)
if macdvec (end,:)> nineperma (end,:)
s=1 % buy
elseif macdvec (end,:)< nineperma (end,:)
s=-1 % sell
end
elseif method==7
y=price2ret(data)
N=length(y)
% We decompose our data with function db3
[XX,l] = wavedec(y,decomposition_tree,'db3');
% We define GARCH (1,1) process
[Kappa, Alpha, Beta] = ugarch(XX, 1, 1)
% We set the random number generator seed for reproducability
randn('state', 0)
NumSamples = 20000;
firstpoint=length_test
% We simulate the process with Monte Carlo
[U , H] = ugarchsim(Kappa, Alpha, Beta, NumSamples);
% Length of vector
%V=1%length(data);
% From current day we extract firstpoint data randomly selected
currentprice = randperm(N-M);
currentprice= currentprice+N;
for j=1:firstpoint
Y1 = currentprice(j);
Y0 = Y1-N+1;
p = U(Y0:Y1);
p = p(:);
Y1(1,:) = p(1,:);
prediction = U(Y1+1:Y1+M);
end
if prediction>0
s=1
elseif prediction<0
s=-1 % sell
end
elseif method==8
N=length(data)
% We decompose our data with function db3
[XX,l] = wavedec(data,decomposition_tree,'db3');
[Short,Long]=movavg(XX,1,lag,factor)
if data(end,:)> Long (end,:)
s=1 % buy
elseif data (end,:)< Long (end,:)
s=-1 % sell
end
end
```

AUTHORS PROFILE

Mr.Eleftherios Giovanis is a first year Ph.D student at the Department of Economics at Royal Holloway University of London. He has an undergraduate degree in Economics. He studied his first Msc in Applied Economics and Finance at University of Macedonia in Greece and then he studied Msc in Quality Assurance with specialization in Reliability and Maintenance Analysis in Engineering at Hellenic Open University. He was working as statistician-econometrician analyst before he starts his Ph.D studies.

.