TIME SERIES FORECASTING
BY USING
WAVELET KERNEL SUPPORT VECTOR MACHINES

By Ali Habibnia
(a.habibnia@lse.ac.uk)
31 Oct 2012
Outline

- Introduction to Statistical Learning and SVM
- SVM & SVR Formula
- Wavelet as a Kernel Function
- **Study 1:** Forecasting volatility based on wavelet support vector machine, *Written by Ling-Bing Tang, Ling-Xiao Tang, Huan-Ye Sheng*
- **Study 2:** Forecasting Volatility in Financial Markets By Introducing a GA-Assisted SVR-Garch Model, *Written by Ali Habibnia*
- Suggestion for further research + Q&A
SVMs History

- The Study on Statistical Learning Theory was started in the 1960s by Vladimir Vapnik. He is well-known as a founder (together with Professor Alexey Chervonenkis) of this theory.

- He has also developed the theory of the support vector machines (for linear and nonlinear input–output knowledge discovery) in the framework of statistical learning theory in 1992.

- Prof. Vapnik has been awarded the 2012 Benjamin Franklin medal in Computer and Cognitive Science from the Franklin Institute.
History and motivation

- SVMs (a novel ANN) is a supervised learning algorithm for
  - Pattern Recognition
  - Regression Estimation – Non Parametric (Applications for function estimation started ~ 1995 called Support Vector Regression)

- Remarkable characteristics of SVMs
  - Good generalization performance: SVMs implement the Structural Risk Minimization Principle which seeks to minimize the upper bound of the generalization error rather than only minimize the training error.
  - Absence of local minima: Training SVM is equivalent to solving a linearly constrained quadratic programming problem. Hence the solution of SVMs is unique and globally optimal.

- It has a simple geometrical interpretation in a high-dimensional feature space that is nonlinearly related to input space
The Advantages of SVM(R)

- Based on a strong and nice Theory:
  In contrast to previous “black box” learning approaches, SVMs allow for some intuition and human understanding.

- Training is relatively easy:
  No local optimal, unlike in neural network
  Training time does not depend on dimensionality of feature space, only on fixed input space thanks to the kernel trick.

- Generally avoids over-fitting:
  Trade-off between complexity and error can be controlled explicitly.

- Generalize well even in high dimensional spaces under small training set conditions. Also it is robust to noise.
Linear Classifiers

- $g(x)$ is a linear function:
  \[ g(x) = w^T x + b \]
  
  A hyper-plane in the feature space

- (Unit-length) normal vector of the hyper-plane:
  \[ n = \frac{w}{||w||} \]
How would you classify these points using a linear discriminant function in order to minimize the error rate?

Infinite number of answers!
Linear Classifiers

- How would you classify these points using a linear discriminant function in order to minimize the error rate?
- Infinite number of answers!
How would you classify these points using a linear discriminant function in order to minimize the error rate?

Infinite number of answers!
Which one is the best?
Large Margin Linear Classifier

- The linear discriminant function (classifier) with the maximum margin is the best.
- Margin is defined as the width that the boundary could be increased by before hitting a data point.
- Why it is the best?
  - Robust to outliers and thus strong generalization ability.
Large Margin Linear Classifier

- Given a set of data points:
  \{(x_i, y_i)\}, i = 1, 2, \ldots, n
  
  For \(y_i = +1\), \(w^T x_i + b > 0\)
  
  For \(y_i = -1\), \(w^T x_i + b < 0\)

- With a scale transformation on both \(w\) and \(b\), the above is equivalent to
  
  For \(y_i = +1\), \(w^T x_i + b \geq 1\)
  
  For \(y_i = -1\), \(w^T x_i + b \leq -1\)
We know that:
\[ w^T x^+ + b = 1 \]
\[ w^T x^- + b = -1 \]

The margin width is:
\[
M = (x^+ - x^-) \cdot n \\
= (x^+ - x^-) \cdot \frac{w}{\|w\|} = \frac{2}{\|w\|}
\]
Formulation:

\[
\begin{align*}
&\text{maximize} & & \frac{2}{\|w\|} \\
&\text{such that} & & \\
&\text{For } y_i = +1, & & w^T x_i + b \geq 1 \\
&\text{For } y_i = -1, & & w^T x_i + b \leq -1
\end{align*}
\]
This is the simplest kind of SVM (Called an LSVM)

- Formulation:
  \[
  \text{minimize} \quad \frac{1}{2} \|w\|^2 \\
  \text{such that}
  \begin{align*}
  &\text{For } y_i = +1, \quad w^T x_i + b \geq 1 \\
  &\text{For } y_i = -1, \quad w^T x_i + b \leq -1 \\
  &\text{such that} \\
  &y_i (w^T x_i + b) \geq 1
  \end{align*}
\]
The Optimization Problem Solution

Quadratic programming with linear constraints

\[
\begin{align*}
& \text{minimize} \quad \frac{1}{2} \| \mathbf{w} \|^2 \\
& \text{s.t.} \quad y_i (\mathbf{w}^T \mathbf{x}_i + b) \geq 1
\end{align*}
\]

Lagrangian Function

\[
\begin{align*}
& \text{minimize} \quad L_p (\mathbf{w}, b, \alpha_i) = \frac{1}{2} \| \mathbf{w} \|^2 - \sum_{i=1}^{n} \alpha_i \left( y_i (\mathbf{w}^T \mathbf{x}_i + b) - 1 \right) \\
& \text{s.t.} \quad \alpha_i \geq 0
\end{align*}
\]
The Optimization Problem Solution

minimize \( L_p(w, b, \alpha_i) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i \left( y_i (w^T x_i + b) - 1 \right) \)

s.t. \( \alpha_i \geq 0 \)

\[ \frac{\partial L_p}{\partial w} = 0 \quad \Rightarrow \quad w = \sum_{i=1}^{n} \alpha_i y_i x_i \]

\[ \frac{\partial L_p}{\partial b} = 0 \quad \Rightarrow \quad \sum_{i=1}^{n} \alpha_i y_i = 0 \]
The Optimization Problem Solution

\[
\begin{align*}
\text{minimize} & \quad L_p(w, b, \alpha_i) = \frac{1}{2} \|w\|^2 - \sum_{i=1}^{n} \alpha_i \left( y_i (w^T x_i + b) - 1 \right) \\
\text{subject to} & \quad \alpha_i \geq 0
\end{align*}
\]

Lagrangian Dual Problem

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j x_i^T x_j \\
\text{subject to} & \quad \alpha_i \geq 0, \text{ and } \sum_{i=1}^{n} \alpha_i y_i = 0
\end{align*}
\]
The Optimization Problem Solution

- From KKT condition, we know:

\[ \alpha_i \left( y_i (w^T x_i + b) - 1 \right) = 0 \]

- Thus, only support vectors have \( \alpha_i \neq 0 \)

- The solution has the form:

\[ w = \sum_{i=1}^{n} \alpha_i y_i x_i = \sum_{i \in SV} \alpha_i y_i x_i \]  
get \( b \) from \( y_i (w^T x_i + b) - 1 = 0 \),  
where \( x_i \) is support vector
Slack variables $\xi_i$ can be added to allow misclassification of difficult or noisy examples.

What should our quadratic optimization criterion be?

Minimize

$$\frac{1}{2} \mathbf{w} \cdot \mathbf{w} + C \sum_{k=1}^{R} \varepsilon_k$$
Hard Margin v.s. Soft Margin

- **The old formulation:**
  
  Find \( w \) and \( b \) such that
  
  \[
  \Phi(w) = \frac{1}{2} w^T w \text{ is minimized and for all } \{(x_i, y_i)\}
  \]
  
  \[y_i (w^T x_i + b) \geq 1\]

- **The new formulation incorporating slack variables:**

  Find \( w \) and \( b \) such that
  
  \[
  \Phi(w) = \frac{1}{2} w^T w + C \sum \xi_i \text{ is minimized and for all } \{(x_i, y_i)\}
  \]
  
  \[y_i (w^T x_i + b) \geq 1 - \xi_i \text{ and } \xi_i \geq 0 \text{ for all } i\]

- **Parameter C can be viewed as a way to control overfitting.**
Non-linear SVMs: Feature spaces

- General idea: the original input space can always be mapped to some higher-dimensional feature space where the training set is separable:

\[ \Phi: \mathbf{x} \rightarrow \varphi(\mathbf{x}) \]
The “Kernel Trick”

- The linear classifier relies on dot product between vectors $K(x_i,x_j) = x_i^T x_j$

- If every data point is mapped into high-dimensional space via some transformation $\Phi: x \rightarrow \phi(x)$, the dot product becomes:
  $$K(x_i,x_j) = \phi(x_i)^T \phi(x_j)$$

- A kernel function is some function that corresponds to an inner product in some expanded feature space.

- Kernel methods map the data into higher dimensional spaces in the hope that in this higher-dimensional space the data could become more easily separated or better structured.
The “Kernel Trick”

- This mapping function, however, hardly needs to be computed because of a tool called the kernel trick.

- The kernel trick is a mathematical tool which can be applied to any algorithm which solely depends on the dot product between two vectors. Wherever a dot product is used, it is replaced by a kernel function.

<p>| | |</p>
<table>
<thead>
<tr>
<th></th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td>1.</td>
<td>Linear Kernel</td>
</tr>
<tr>
<td>2.</td>
<td>Polynomial Kernel</td>
</tr>
<tr>
<td>3.</td>
<td>Gaussian Kernel</td>
</tr>
<tr>
<td>4.</td>
<td>Exponential Kernel</td>
</tr>
<tr>
<td>5.</td>
<td>Laplacian Kernel</td>
</tr>
<tr>
<td>6.</td>
<td>ANOVA Kernel</td>
</tr>
<tr>
<td>7.</td>
<td>Hyperbolic Tangent (Sigmoid) Kernel</td>
</tr>
<tr>
<td>8.</td>
<td>Rational Quadratic Kernel</td>
</tr>
<tr>
<td>9.</td>
<td>Multiquadric Kernel</td>
</tr>
<tr>
<td>10.</td>
<td>Inverse Multiquadric Kernel</td>
</tr>
<tr>
<td>11.</td>
<td>Circular Kernel</td>
</tr>
<tr>
<td>12.</td>
<td>Spherical Kernel</td>
</tr>
<tr>
<td>13.</td>
<td>Wave Kernel</td>
</tr>
<tr>
<td>14.</td>
<td>Power Kernel</td>
</tr>
<tr>
<td>15.</td>
<td>Log Kernel</td>
</tr>
<tr>
<td>16.</td>
<td>Spline Kernel</td>
</tr>
<tr>
<td>17.</td>
<td>B-Spline Kernel</td>
</tr>
<tr>
<td>18.</td>
<td>Bessel Kernel</td>
</tr>
<tr>
<td>19.</td>
<td>Cauchy Kernel</td>
</tr>
<tr>
<td>20.</td>
<td>Chi-Square Kernel</td>
</tr>
<tr>
<td>21.</td>
<td>Histogram Intersection Kernel</td>
</tr>
<tr>
<td>22.</td>
<td>Generalized Histogram Intersection Kernel</td>
</tr>
<tr>
<td>23.</td>
<td>Generalized T-Student Kernel</td>
</tr>
<tr>
<td>24.</td>
<td>Bayesian Kernel</td>
</tr>
<tr>
<td>25.</td>
<td>Wavelet Kernel</td>
</tr>
</tbody>
</table>
What Functions are Kernels?

- For some functions $K(x_i, x_j)$ checking that
  \[ K(x_i, x_j) = \phi(x_i)^T \phi(x_j) \]
can be cumbersome.

- Mercer’s theorem:

  Every semi-positive definite symmetric function is a kernel

- Semi-positive definite symmetric functions correspond to a semi-positive definite symmetric Gram matrix:

\[
\begin{array}{cccc}
K(x_1, x_1) & K(x_1, x_2) & K(x_1, x_3) & \ldots & K(x_1, x_N) \\
K(x_2, x_1) & K(x_2, x_2) & K(x_2, x_3) & \ldots & K(x_2, x_N) \\
\vdots & \vdots & \vdots & \ddots & \vdots \\
K(x_N, x_1) & K(x_N, x_2) & K(x_N, x_3) & \ldots & K(x_N, x_N)
\end{array}
\]
Examples of Kernel Functions

- Linear kernel: \[ K(x_i, x_j) = x_i^T x_j \]

- Polynomial kernel: \[ K(x_i, x_j) = (1 + x_i^T x_j)^p \]

- Gaussian (Radial-Basis Function (RBF)) kernel:
  \[ K(x_i, x_j) = \exp(-\frac{\|x_i - x_j\|^2}{2\sigma^2}) \]

- Sigmoid:
  \[ K(x_i, x_j) = \tanh(\beta_0 x_i^T x_j + \beta_1) \]
Nonlinear SVM: Optimization

Formulation: (Lagrangian Dual Problem)

\[
\begin{align*}
\text{maximize} & \quad \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j) \\
\text{such that} & \quad 0 \leq \alpha_i \leq C \\
& \quad \sum_{i=1}^{n} \alpha_i y_i = 0 
\end{align*}
\]

The solution of the discriminant function is

\[
g(x) = \sum_{i \in SV} \alpha_i K(x_i, x) + b
\]

The optimization technique is the same.
Some Issues

- Choice of kernel
  - Gaussian or polynomial kernel is default
  - if ineffective, more elaborate kernels are needed

- Choice of kernel parameters
  - e.g. $\sigma$ in Gaussian kernel
  - $\sigma$ is the distance between closest points with different classifications
  - In the absence of reliable criteria, applications rely on the use of a validation set or cross-validation to set such parameters.

- Optimization criterion – Hard margin v.s. Soft margin
  - a lengthy series of experiments in which various parameters are tested
Support vector regression

- Maximum margin hyperplane only applies to classification
- However, idea of support vectors and kernel functions can be used for regression
- Basic method is the same as in linear regression: minimize error
  - Difference A: ignore errors smaller than e and use absolute error instead of squared error
  - Difference B: simultaneously aim to maximize flatness of function
- User-specified parameter e defines “tube”
Ordinary Least Squares (OLS)

\[ f(x) = wx + b \]

**Solution:**
\[ \text{Loss} = \|(wx + b) - Y\|_2 \]
\[ \frac{d\text{Loss}}{dw} = 0 \Rightarrow (X^T X)w = X^T Y \]
Support Vector Regression (SVR)

- **Solution:**
  \[
  \text{Min} \frac{1}{2} w^T w
  \]

- **Constraints:**
  \[
  y_i - w^T x_i - b \leq \varepsilon \\
  w^T x_i + b - y_i \leq \varepsilon
  \]

\[
f(x) = wx + b
\]
Support Vector Regression (SVR)

Minimise:
\[
\frac{1}{2} w^T w + C \sum_{i=1}^{N} (\xi_i + \xi_i^*)
\]

Constraints:
\[
y_i - w^T x_i - b \leq \varepsilon + \xi_i
\]
\[
w^T x_i + b - y_i \leq \varepsilon + \xi_i^*
\]
\[
\xi_i, \xi_i^* \geq 0
\]

\[
f(x) = wx + b
\]
Lagrange Optimisation

\[ L = \frac{1}{2} w^T w + C \sum_{i=1}^{N} (\xi_i + \xi_i^*) \]

\[ \sum_{i=1}^{N} \alpha_i (\varepsilon + \xi_i - y_i + w^T x_i + b) \]

\[ \sum_{i=1}^{N} \alpha_i^* (\varepsilon + \xi_i^* - y_i + w^T x_i + b) \]

\[ \sum_{i=1}^{N} (\eta_i \xi_i + \eta_i^* \xi_i^*) \]

Target

Constraints

Regression:

\[ y(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle x_i, x \rangle + b \]
Nonlinear Regression

\[ f(x) = \xi + \epsilon - \epsilon_0 \]

\[ \varphi(x) = \xi + \epsilon + \epsilon_0 \]
Regression Formulas

- **Linear:**
  \[ y(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle x_i, x \rangle + b \]

- **Nonlinear:**
  \[ y(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot \langle \varphi(x_i), \varphi(x) \rangle + b \]

- **General:**
  \[ y(x) = \sum_{i=1}^{N} (\alpha_i - \alpha_i^*) \cdot K(x_i, x) + b \]
Kernel Types

- **Linear:** \[ K(x, x_i) = \langle x, x_i \rangle \]
- **Polynomial:** \[ K(x, x_i) = \langle x, x_i \rangle^d \]
- **Radial basis function:** \[ K(x, x_i) = \exp \left( - \frac{\|x - x_i\|^2}{2\sigma^2} \right) \]
- **Exponential RBF:** \[ K(x, x_i) = \exp \left( - \frac{\|x - x_i\|^2}{2\sigma^2} \right) \]
Wavelet Kernel

The Wavelet kernel (Zhang et al, 2004) comes from Wavelet theory and is given as:

\[ k(x, y) = \prod_{i=1}^{N} h\left(\frac{x_i - c}{a}\right) h\left(\frac{y_i - c}{a}\right) \]

Where \( a \) and \( c \) are the wavelet dilation and translation coefficients, respectively (the form presented above is a simplification). A translation-invariant version of this kernel can be given as:

\[ k(x, y) = \prod_{i=1}^{N} h\left(\frac{x_i - y_i}{a}\right) \]

Where in both \( h(x) \) denotes a mother wavelet function. i.e:

\[ h(x) = \cos(1.75x)\exp\left(-\frac{x^2}{2}\right) \]
A simple View of Wavelet Theory

Fourier Analysis

- Breaks down a signal into constituent sinusoids of different frequencies.

In other words: Transform the view of the signal from time-base to frequency-base.
What’s wrong with Fourier?

- By using Fourier Transform, we lose the time information: WHEN did a particular event take place?
- FT cannot locate drift, trends, abrupt changes, beginning and ends of events, etc.
- Calculating use complex numbers.
Wavelets vs. Fourier Transform

- In Fourier transform (FT) we represent a signal in terms of sinusoids.
- FT provides a signal which is localized only in the frequency domain.
- It does not give any information of the signal in the time domain.
Wavelets vs. Fourier Transform

- Basis functions of the wavelet transform (WT) are small waves located in different times.
- They are obtained using scaling and translation of a scaling function and wavelet function.
- Therefore, the WT is localized in both time and frequency.
Wavelets vs. Fourier Transform

- If a signal has a discontinuity, FT produces many coefficients with large magnitude (*significant coefficients*)
- But WT generates a few significant coefficients around the discontinuity
- Nonlinear approximation is a method to benchmark the approximation power of a transform
Wavelets vs. Fourier Transform

- In nonlinear approximation we keep only a few significant coefficients of a signal and set the rest to zero.
- Then we reconstruct the signal using the significant coefficients.
- WT produces a few significant coefficients for the signals with discontinuities.
- Thus, we obtain better results for WT nonlinear approximation when compared with the FT.
Wavelets vs. Fourier Transform

- Most natural signals are smooth with a few discontinuities (are piece-wise smooth)
- Speech and natural images are such signals
- Hence, WT has better capability for representing these signal when compared with the FT
- Good nonlinear approximation results in efficiency in several applications such as compression and denoising
Study 1: Forecasting volatility based on wavelet support vector machine, Ling-Bing Tang, Ling-Xiao Tang, Huan-Ye Sheng

- combine SVM with wavelet theory to construct a multidimensional wavelet kernel function to predict the conditional volatility of stock market returns based on GARCH model.
- General Kernel function in SVM cannot capture the cluster feature of volatility accurately.
- Wavelet function yields features that describe the volatility time series both at various locations and at varying time granularities.
- The prediction performance of SVM is greatly dependent upon the selection of kernel functions.

\[
k(s, t) = \sum_{1 \leq j < 2^N, k \in \mathbb{Z}} \psi_{j,k}(s)\psi_{j,k}(t)
\]

\[j \text{ and } k \text{ denote the dilation and translation}
\]

\[
\text{maximize } \sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i=1}^{n} \sum_{j=1}^{n} \alpha_i \alpha_j y_i y_j K(x_i, x_j)
\]
Study 1: Forecasting volatility based on wavelet support vector machine, Ling-Bing Tang, Ling-Xiao Tang, Huan-Ye Sheng

Descriptive statistics of the daily returns

<table>
<thead>
<tr>
<th>DAXINDX</th>
<th>FRCAC40</th>
<th>FTSE100</th>
<th>JAPDOWA</th>
<th>SPCOMP</th>
</tr>
</thead>
<tbody>
<tr>
<td>Mean</td>
<td>0.00439</td>
<td>0.00212</td>
<td>0.00719</td>
<td>0.00374</td>
</tr>
<tr>
<td>S.D.</td>
<td>0.10893</td>
<td>0.17430</td>
<td>0.13595</td>
<td>0.16883</td>
</tr>
<tr>
<td>Skewness</td>
<td>−0.15341</td>
<td>−0.02455</td>
<td>0.04861</td>
<td>0.39928</td>
</tr>
<tr>
<td>Kurtosis</td>
<td>4.54437</td>
<td>3.77320</td>
<td>5.4952</td>
<td>6.74197</td>
</tr>
<tr>
<td>r(1)</td>
<td>0.0370</td>
<td>0.0307</td>
<td>0.0573</td>
<td>0.0025</td>
</tr>
<tr>
<td>Q(20)</td>
<td>34.0088</td>
<td>19.8654</td>
<td>35.0099</td>
<td>13.7866</td>
</tr>
<tr>
<td>r2(1)</td>
<td>0.0108</td>
<td>0.0547</td>
<td>0.1755</td>
<td>0.1088</td>
</tr>
<tr>
<td>Q2(20)</td>
<td>62.4672</td>
<td>82.2997</td>
<td>299.6639</td>
<td>183.4667</td>
</tr>
</tbody>
</table>

r(1), autocorrelation of order 1 of the original observations $y_t$; r2(1), autocorrelation of order 1 of the squared observations $y_t^2$; Q(20), Box–Ljung statistic for $y_t$ (31.4 is the 5% critical value); Q2(20), Box–Ljung statistic for $y_t^2$ (31.4 is the 5% critical value).
Study 1: Forecasting volatility based on wavelet support vector machine, Ling-Bing Tang, Ling-Xiao Tang, Huan-Ye Sheng

Table 6

Results on the test set

<table>
<thead>
<tr>
<th>Stock indices</th>
<th>Gaussian kernel</th>
<th></th>
<th>Wavelet kernel</th>
<th></th>
</tr>
</thead>
<tbody>
<tr>
<td></td>
<td>NMSE</td>
<td>NMAE</td>
<td>HR</td>
<td>NMSE</td>
</tr>
<tr>
<td>DAXINDX</td>
<td>0.7627</td>
<td>0.8284</td>
<td>0.7179</td>
<td>0.7614</td>
</tr>
<tr>
<td>FRCAC40</td>
<td>0.7292</td>
<td>0.7958</td>
<td>0.7239</td>
<td>0.7268</td>
</tr>
<tr>
<td>FTSE100</td>
<td>0.6690</td>
<td>0.7284</td>
<td>0.7211</td>
<td>0.6703</td>
</tr>
<tr>
<td>JAPDOWA</td>
<td>0.7949</td>
<td>0.8891</td>
<td>0.6936</td>
<td>0.7603</td>
</tr>
<tr>
<td>SCOMP</td>
<td>0.7476</td>
<td>0.8436</td>
<td>0.7077</td>
<td>0.7014</td>
</tr>
</tbody>
</table>

\[ t\text{-value} \quad 1.6825 > t_{0.1,4} = 1.5332 \]

\[ \text{NMSE} = \sqrt{\frac{\sum_{t=1}^{N} (\hat{\sigma}_t^2 - y_t^2)^2}{\sum_{t=1}^{N} (y_{t-1}^2 - y_t^2)^2}} \quad (16) \]

\[ \text{NMAE} = \frac{\sum_{t=1}^{N} |\hat{\sigma}_t^2 - y_t^2|}{\sum_{t=1}^{N} |y_{t-1}^2 - y_t^2|} \quad (17) \]

\[ \text{HR} = \frac{1}{N} \sum_{t=1}^{N} q_t, \quad q_t = \begin{cases} 1 : (\hat{\sigma}_t^2 - y_{t-1}^2)(y_t^2 - y_{t-1}^2) \geq 0 \\ 0 : \text{else} \end{cases} \quad (18) \]
Study 1: Forecasting volatility based on wavelet support vector machine, Ling-Bing Tang, Ling-Xiao Tang, Huan-Ye Sheng
Study 2: Forecasting Volatility in Financial Markets By Introducing a GA-Assisted SVR-Garch Model, Ali Habibnia

- No structured way being available to choose the free parameters of SVR and kernel function, these parameters are usually set by researcher in trial and error (Grid Search and Cross Validation) procedure which is not optimal.

- In this study a novel method, named as GA assisted SVR has been introduced, which a genetic algorithm simultaneously searches for SVR’s optimal parameters and kernel parameter (in this study: a radial basis function (RBF)).

- The SVM(R) tries to get the best fit with the data, not relying on any prior knowledge, and it only concentrates on minimizing the prediction error with a given machine complexity.

Summary statistics and diagnostics for the total sample

<table>
<thead>
<tr>
<th>Mean</th>
<th>St.Dev.</th>
<th>Skewness</th>
<th>Kurtosis</th>
<th>JB</th>
<th>Q(20)</th>
<th>Q(20)</th>
<th>ARCH-LM</th>
</tr>
</thead>
<tbody>
<tr>
<td>0.0074</td>
<td>1.3378</td>
<td>-0.1295</td>
<td>10.3662</td>
<td>4278.3</td>
<td>6.5539</td>
<td>60.0503</td>
<td>426.2948</td>
</tr>
</tbody>
</table>

- FTSE 100 index from 04 Jan 2005 to 29 Jun 2012.
**Study 2: Forecasting Volatility in Financial Markets By Introducing a GA-Assisted SVR-Garch Model,** **Ali Habibnia**

### Table 4.4 forecasting performance for traditional and proposed model

<table>
<thead>
<tr>
<th>Parameter</th>
<th>MAD</th>
<th>MSE</th>
<th>RMSE</th>
</tr>
</thead>
<tbody>
<tr>
<td>GARCH(1,1)</td>
<td>0.2376</td>
<td>0.1414</td>
<td>0.3760</td>
</tr>
<tr>
<td>GA-SVR-GARCH</td>
<td>0.0781</td>
<td>0.00057</td>
<td>0.0239</td>
</tr>
</tbody>
</table>

---

**Fig 4.5** One step a head forecast of FTSE100 volatility in out of sample period by GARCH(1,1)

**Fig 4.6** One step a head forecast of FTSE100 volatility in out of sample period by GA-SVR GARCH(1,1)
Thanks for your patience ;)

By Ali Habibnia
a.habibnia@lse.ac.uk
31 Oct 2012