Revista de Matemática: Teoría y Aplicaciones 2000 **7**(1-2) : 107–116
cimpa – ucr – ccss issn: 1409-2433

# UN ALGORITMO PARA EL ENTRENAMIENTO DE MÁQUINAS DE VECTOR SOPORTE PARA REGRESIÓN

John Goddard[*] – Sergio Gerardo de los Cobos Silva[**]
Blanca Rosa Pérez Salvador[***] – Miguel Ángel Gutiérrez Andrade[****]

*Recibido: 16 Mayo 2000*

**Abstract**

The aim of the present paper is twofold. Firstly an introduction to the ideas of Support Vector regression is given. Then a new and simple algorithm, suggested by the work of Campbell y Cristianini in [16], is proposed which solves the corresponding quadratic programming problem in an easy fashion. The algorithm is illustrated by example and compared with classical regression.

**Keywords:** Support vector machines, $\varepsilon-$support vector regression.

**Resumen**

El propósito del presente artículo es doble. Primero se proporciona una introducción a las ideas básicas de la Máquinas de Vector Soporte para regresión. Posteriormente, se presenta un algoritmo novedoso y sencillo, basado en el trabajo de Campbell y Cristianini [16], que resuelve de manera fácil el correspondiente problema de programación cuadrática. Se ilustra el algoritmo con un ejemplo, y se compara con el método de regresión clásico.

**Palabras clave:** Máquinas de vector soporte, regresión $\varepsilon$-vector soporte.

**Mathematics Subject Classification:** 62J99

[*]Depto. de Ingeniería Eléctrica, Universidad Autónoma Metropolitana-Iztapalapa, Av. Michoacán y La Purísima, Col. Vicentina, CP 09340 México D.F., México; E-Mail: jgc@xanum.uam.mx
[**]Misma dirección que J. Goddard; E-Mail: cobos@xanum.uam.mx
[***]Depto. de Matemáticas, Universidad Autónoma Metropolitana-Iztapalapa, Av. Michoacán y La Purísima, Col. Vicentina, CP 09340 México D.F., México.
[****]Depto. de Sistemas, Universidad Autónoma Metropolitana-Azcapotzalco, Av. San Pablo 180, Colonia Reynosa Tamaulipas, 02200 México, D.F. México; E-Mail: gama@hp9000a1.uam.mx

# 1. Introducción

Las Máquinas de Vector Soporte (MVS) es una área prometedora del aprendizaje maquinal desarrollada inicialmente por Vapnik [1,2] para construir clasificadores. Se funda en la teoría estadística del aprendizaje, o teoría de VC. Esta teoría permite escoger un clasificador que minimiza una cota superior sobre el riesgo (o error de prueba), y proporciona una buena medida para obtener clasificadores que generalizan bien sobre datos no previamente vistos. Esto le da una ventaja a MVS sobre otros clasificadores, como redes neuronales artificiales que pueden producir modelos que sobreajustan a los datos. Además, las MVS tienen la habilidad de construir regiones de decisión no lineales de una manera discriminativa mediante la introducción de una función núcleo.

Las MVS han sido estudiadas intensamente en años recientes y aplicadas exitosamente en gran variedad de temas como: reconocimiento del habla [3], estimación de densidades probabilísticas [4], y la predicción de series de tiempo [5]. El lector puede encontrar un buen tutorial sobre los aspectos de MVS para clasificación en [6].

En este artículo se considera la aplicación de las MVS en la aproximación de funciones. En este caso el método se llama regresión $\varepsilon - SV$ [7,8,9,10], y el objetivo es: dado un conjunto finito de datos $\{(x_i, y_i)\}, x_i \in \mathbb{R}^n, y_i \in \mathbb{R}^1$, $i = 1, ..., N$, encontrar una función $f(x)$ que esté a lo más $\varepsilon$ desviaciones de los objetivos $y_i$ para todos los datos de entrenamiento $x_i$ y que a la vez sea tan "plana" como sea posible.

En el caso más sencillo, dado los datos, queremos encontrar una función lineal $f$ de la forma:

$$f(x) = \langle w, x \rangle + b \tag{1}$$

donde $\langle \cdot, \cdot \rangle$ es el producto interior. En este contexto, "plana" significa que queremos encontrar un $w$ "pequeña". Para lograr esto, podemos minimizar $||w||^2$, la norma cuadrada, y transformar el problema en el siguiente de optimización convexa (c.f. [8]):

$$\text{Minimizar} \qquad \frac{1}{2} \|w\|^2 \tag{2}$$

$$\text{sujeto a :} \qquad yi - \langle w, x_i \rangle - b \leq \varepsilon \;\text{ y }\; \langle w, x_i \rangle + b - y_i \leq \varepsilon$$

Esto supone que el problema es factible. Cuando no es el caso, se reformula el problema (2) introduciendo variables de holgura $\xi_i$, $\xi_i^*$ de la siguiente manera:

$$\text{Minimizar} \qquad \frac{1}{2} \|w\|^2 + C \sum_i (\xi_i + \xi_i^*) \tag{3}$$

$$\text{sujeto a:} \qquad yi - \langle w, x_i \rangle - b \leq \varepsilon + \xi_i, \;\; \langle w, x_i \rangle + b - y_i \leq \varepsilon + \xi_i^*, \xi_i, \xi_i^* \geq 0$$

y donde $C$ determina el compromiso entre lo plano de $f$ y la cantidad de desviaciones mayores a $\varepsilon$ que están permitidas. De aquí, el problema dual se deriva dando:

$$\text{Maximizar} \qquad -\frac{1}{2} \sum_{i,j} (\alpha_i^* - \alpha_i)(\alpha_j^* - \alpha_j) \langle x_i, x_j \rangle + \sum_i y_i (\alpha_i^* - \alpha_i) - \varepsilon \sum_i (\alpha_i^* + \alpha_i) \tag{4}$$

$$\text{sujeto a:} \qquad \sum_i (\alpha_i^* - \alpha_i) = 0 \text{ y } \ \alpha_i^*, \alpha_i \in [0, C]$$

donde todas las sumatorias recorren desde 1 a $N$. Por lo que la solución del problema (4) está dada por:

$$f(x) = \sum_i (\alpha_i^* - \alpha_i) \langle x_i, x \rangle + b \tag{5}$$

Frecuentemente el número de $\alpha_i^*, \alpha_i$ distintos a cero (cuyos vectores correspondientes se llaman de **soporte**) es pequeño, y la representación requiere de pocos miembros. Más aún, la formulación se generaliza al caso no lineal, que generalmente se requiere para modelar adecuadamente los datos, sustituyendo el producto interior por una función núcleo,$k$, de tal forma que el problema a resolver queda como:

$$\text{Maximizar} \qquad -\frac{1}{2} \sum_{i,j} (\alpha_i^* - \alpha_i) \left(\alpha_j^* - \alpha_j\right) k(x_i, x_j) + \sum_i y_i (\alpha_i^* - \alpha_i) - \varepsilon \sum_i (\alpha_i^* - \alpha_i) \tag{6}$$

$$\text{sujeto a:} \qquad \sum_i (\alpha_i^* - \alpha_i) = 0 \text{y } \alpha_i^*, \alpha_i \in [0, C]$$

por lo que la solución está dada por:

$$f(x) = \sum_i (\alpha_i^* - \alpha_i) k(x_i, x) + b \tag{7}$$

Algunos de los núcleos que han sido usados son:

a) Polinomio: $k(x, y) = (\langle x, y \rangle + 1)^d, d = 1, 2, ...$

b) Funciones de base radial: $k(x, y) = exp\{-|x - y|^2/\sigma^2\}$

c) Redes neuronales de dos capas: $k(x, y) = tanh(b \langle x, y \rangle - c)$ , (para ciertos valores de $b$ y $c$).

Se puede mostrar [8,11] que:

1. Los multiplicadores Lagrangianos $\alpha_i^*, \alpha_i$ no pueden ser simultáneamente diferentes de cero, por lo tanto la restricción $\alpha_i^* \alpha_i = 0$ se cumple.

2. Los vectores soporte son aquellos puntos $x_i$ en los cuales el error de interpolación es mayor o igual a $\varepsilon$. Los puntos en los que el error de interpolación es menor que $\varepsilon$ nunca son vectores de soporte, y no forman parte de la solución. Una vez que se han encontrado, podrían ser eliminados del conjunto de datos, y si se resuelve el problema de programación nuevamente sobre el conjunto reducido se encuentra la misma solución.

3. Si $\alpha_i^* = C$ o $\ \alpha_i = C$, esto implica que el vector correspondiente, $(x_i, y_i)$, está afuera del $\varepsilon-$ tubo.

Mientras que teóricamente el problema de optimización tiene una solución única, para problemas con muchos datos, es imposible encontrar una solución exacta. Además, se ha observado [8,12] que aún en problemas con pocos datos, paquetes comerciales de programación cuadrática no siempre convergen, y cuando lo hacen no siempre dan las mismas soluciones. Esto, y las dificultades involucradas en implementar algoritmos para resolver un problema de programación cuadrática, han sido una desventaja para que el método de MVS sea usado más ampliamente. Para remediar esta situación, se han introducido distintos algoritmos como: algoritmos de punto interior [13], algoritmos de elección de subconjuntos [14], y optimización minimal secuencial [15].

En este artículo, se presenta un algoritmo sencillo basado en el método de gradiente ascendente y sugerido por Campbell y Cristianini en [16], para encontrar los coeficientes $\alpha_i^*, \alpha_i$ en el caso no lineal. En la siguiente sección se dan detalles de la derivación del algoritmo y después se muestra que la función objetivo es convexa. En la sección 4 se especifica la implementación del algoritmo. Posteriormente se ilustra el algoritmo con un ejemplo. Finalmente se mencionan algunas conclusiones derivadas del trabajo.

## 2. El método

Una manera sencilla de resolver el problema (1), siguiendo a [16], es aplicar el método de gradiente ascendente a la función objetivo con el Lagrangiano correspondiente definido por:

$$L(\alpha^*,\alpha) = -\frac{1}{2}\sum_{i,j}(\alpha_i^*-\alpha_i)(\alpha_j^*-\alpha_j)k(x_i,x_j)+\sum_i y_i(\alpha_i^*-\alpha_i)-\varepsilon\sum_i(\alpha_i^*+\alpha_i)-\lambda\sum_i(\alpha_i^*-\alpha_i) \tag{8}$$

donde $\alpha = (\alpha_1,\ldots,\alpha_N)^T, \alpha^* = (\alpha_1^*,\ldots,\alpha_N^*)^T \epsilon \mathbb{R}^N$ y $0 \leq \alpha_i,\alpha_i^* \leq C \quad i = 1,\ldots,N$
Tomando las derivadas parciales, se obtiene:

$$\begin{aligned} \frac{\partial L}{\partial \alpha_m^*} &= y_m - \varepsilon - \lambda - \sum_i(\alpha_i^* - \alpha_i)k(x_i,x_m) \\ \frac{\partial L}{\partial \alpha_m} &= -y_m - \varepsilon + \lambda + \sum_i(\alpha_i^* - \alpha_i)k(x_i,x_m) \end{aligned} \tag{9}$$

Introduciendo los pasos para el método de gradiente ascendente $\delta\alpha_m^*$, $\delta\alpha_m$, se obtiene:

$$\delta\alpha_m^* = \eta\frac{\partial L}{\partial \alpha_m^*}, \delta\alpha_m = \eta\frac{\partial L}{\partial \alpha_m} \tag{10}$$

y sustituyéndolos en (8) se obtiene:

$$L(\alpha^* + \delta\alpha_m^*, \alpha) = \delta\alpha_m^*\frac{\partial L}{\partial \alpha_m^*} - \frac{1}{2}\delta\alpha_m^{*^2}k(x_m,x_m) \tag{11}$$

$$L(\alpha^* + \delta\alpha_m^*, \alpha) = \delta\alpha_m^{*^2}\left[\frac{1}{\eta} - \frac{1}{2}k(x_m,x_m)\right] \geq 0 \tag{12}$$

si $\left[\frac{1}{\eta} - \frac{1}{2}k(x_m, x_m)\right] \geq 0$ o $2 \geq \eta\ k(x_m, x_m) \geq 0$ suponiendo que $\eta \geq 0$.

De manera análoga,

$$L(\alpha^*, \alpha + \delta\alpha_m) = \delta\alpha_m^2 \left[\frac{1}{\eta} - \frac{1}{2}k(x_m, x_m)\right] \geq 0 \tag{13}$$

si $2 \geq \eta k(x_m, x_m) \geq 0$ suponiendo que $\eta \geq 0$.

En caso de un núcleo de función de base radial,$k(x_m, x_m) = 1$, con la condición $2 \geq \eta \geq 0$. En la siguiente sección se muestra que $L(\alpha^*, \alpha)$ es una función convexa.

## 3. Convexidad de la función $L(\alpha^*, \alpha)$

En esta sección se muestra que la función objetivo de (6) y $L(\alpha^*, \alpha)$ son convexas, lo cual es importante para garantizar la existencia y unicidad del óptimo global. Sustituyendo $a = \alpha^* - \alpha, y = (y_1, \ldots, y_N)^T \in \mathbb{R}^n$ se puede escribir la función objetivo de (6) como:

$$F(a) = -\frac{1}{2}a^T K a + y^T a - \varepsilon||a||_1 \tag{14}$$

$$\text{sujeta a} \qquad \sum_i a_i = 0$$

donde $K$ es la matriz $N \times N$ con entradas $k(x_i, x_j)$, y donde $||a||_1 = \sum_i |\alpha_i^* + \alpha_i| = \sum_i (\alpha_i^* + \alpha_i)$ puesto que $\alpha^*, \alpha \in [0, C]$.

Con esta notación, se puede escribir el Lagrangiano $L$ como:

$$L(a) = -\frac{1}{2}a^T K a + y^T a - \varepsilon||a||_1 - \lambda a^T \mathbf{1} \tag{15}$$

donde $\mathbf{1} = (1, ..., 1)^T$. Primero se mostrará que la función $F(a)$ es convexa, y posteriormente que $F(\alpha^*, \alpha)$ también lo es.

**Teorema 1** *La función $F(a)$ de (15) es convexa.*

DEMOSTRACIÓN: Se tiene por definición que una función $f$ es convexa si para dos vectores $a_1$ y $a_2$ se tiene que:

$$f(\gamma_1 a_1 + \gamma_2 a_2) \geq \gamma_1 f(a_1) + \gamma_2 f(a_2)$$

para $\gamma_1, \gamma_2 > 0$ y $\gamma_1 + \gamma_2 = 1$.
Nótese que:

$$\begin{aligned} F(\gamma_1 a_1 + \gamma_2 a_2) &= -\frac{1}{2}(\gamma_1 a_1 + \gamma_2 a_2)^T K(\gamma_1 a_1 + \gamma_2 a_2) + y^T(\gamma_1 a_1 + \gamma_2 a_2) - \varepsilon||\gamma_1 a_1 + \gamma_2 a_2||_1 \\ &= -\frac{1}{2}\gamma_1^2 a_1^T K a_1 - \gamma_1\gamma_2 a_1^T k a_2 - \frac{1}{2}\gamma_2^2 a_2^T K a_2 + \gamma_1 y^T a_1 + \gamma_2 y^T a_2 \\ &\quad -\varepsilon \left\|\gamma_1 a_1 + \gamma_2 a_2\right\|_1 \end{aligned}$$

y que:

$$\gamma_1 F(a_1) + \gamma_2 F(a_2) = -\frac{1}{2}\gamma_1 a_1^T K a_1 + \gamma_1 y^T a_1 - \varepsilon||\gamma_1 a_1||_1 - \frac{1}{2}\gamma_2 a_2^T K a_2 + \gamma_2 y^T a_2 - \varepsilon||\gamma_2 a_2||_1$$

Entonces:

$$\begin{aligned} F(\gamma_1 a_1 + \gamma_2 a_2) - \gamma_1 F(a_1) - \gamma_2 F(a_2) &= \frac{1}{2}\gamma_1(1-\gamma_1)a_1^T K a_1 + \frac{1}{2}\gamma_2(1-\gamma_2)a_2^T K a_2 \\ &\quad -\gamma_1\gamma_2 a_1^T K a_2 - \varepsilon(||\gamma_1 a_1 + \gamma_2 a_2||_1 \\ &\quad -||\gamma_1 a_1||_1 - ||\gamma_2 a_2||_1) \end{aligned} \tag{16}$$

Como $\gamma_1 + \gamma_2 = 1$, sustituyendo en (17) se obtiene:

$$\begin{aligned} F(\gamma_1 a_1 + \gamma_2 a_2) - \gamma_1 F(a_1) - \gamma_2 F(a_2) &= \frac{1}{2}\gamma_1\gamma_2(a_1^T K a_1 + a_2^T K a_2 - 2a_1^T K a_2) \\ &\quad +\varepsilon(\|\gamma_1 a_1\|_1 + \|\gamma_2 a_2\|_1 - ||\gamma_1 a_1 + \gamma_2 a_2||_1) \\ &= \frac{1}{2}\gamma_1\gamma_2(a_1 - a_2)^T K(a_1 - a_2) \\ &\quad +\varepsilon(\|\gamma_1 a_1\|_1 + \|\gamma_2 a_2\|_1 - \|\gamma_1 a_1 + \gamma_2 a_2\|_1) \\ &\geq 0 \end{aligned}$$

Por lo que $F(a)$ es convexa.

Por un procedimiento semejante se tiene que $L(a)$ es convexa también. Para ver que $F(\alpha^*, \alpha)$ es convexa, basta ver que: como $a = \alpha^* - \alpha = (I, -I)\begin{pmatrix} \alpha^* \\ \alpha \end{pmatrix}$

$$\begin{aligned} F(\alpha^*, \alpha) &= -\frac{1}{2}(\alpha^{*T}, \alpha^T)\begin{pmatrix} I \\ -I \end{pmatrix} K(I, -I)\begin{pmatrix} \alpha^* \\ \alpha \end{pmatrix} + y^T(I, -I)\begin{pmatrix} \alpha^* \\ \alpha \end{pmatrix} \\ &\quad -\varepsilon\left\|(I. - I)\begin{pmatrix} \alpha^* \\ \alpha \end{pmatrix}\right\|_1 \end{aligned} \tag{17}$$

y como $\gamma_1\begin{pmatrix} \alpha_1^* \\ \alpha_1 \end{pmatrix} + \gamma_2\begin{pmatrix} \alpha_2^* \\ \alpha_2 \end{pmatrix}$ al transformarse da:

$$(I, -I)\left[\gamma_1\begin{pmatrix} \alpha_1^* \\ \alpha_1 \end{pmatrix} + \gamma_2\begin{pmatrix} \alpha_2^* \\ \alpha_2 \end{pmatrix}\right] = \gamma_1 a_1 + \gamma_2 a_2 \tag{18}$$

los cálculos para probar que $F(\alpha^*, \alpha)$ es convexa son los mismos.

## 4. Implementación del algoritmo

Siguiendo las ideas expuestas en [16], donde se desarrolla el caso de clasificación, en este trabajo se desarrolla la implementación de un algoritmo para el caso de regresión, el cual se proporciona en la Figura 1. De manera análoga que en [16], es importante observar que en relación a los pasos del algoritmo:

1. En el caso que $\alpha_i^{*0}$ o $\alpha_i^0 < 0$, se asigna el valor de cero a la variable correspondiente.

2. Se aplica el método de secante para calcular $\lambda$

3. El sesgo $b$, de $f(x)$, es tomado como el valor final de $\lambda$

4. Habrá que incluir una condición dentro del algoritmo para asegurar que el factor $(\varpi^{t-1} - \varpi^{t-2})$ sea distinto a cero.

5. Para asegurar que $\alpha_i^{*0}$, $\alpha_i^0 \leq C$ , se asigna el valor $C$ a la variable correspondiente.

Inicializar $\alpha_i^{*0}, \alpha_i^0 = 0$
Para $t = 0$ to genmax
Si $(t == 0)$ Entonces
$\lambda^0 = \mu$
Otro Si $(t == 1)$ Entonces
$\lambda^1 = -\mu$
Otro
$\lambda^t = \lambda^{t-1} - \omega^{t-1}(\lambda^{t-1} - \lambda^{t-2})/(\omega^{t-1} - \omega^{t-2})$
Fin_Si
Para $i = 1$ to $N$
$z_i = \sum_{j=1}^{N} \left(\alpha_j^* - \alpha_j\right) k\left(x_i, x_j\right)$
$\delta\alpha_i^t = \eta\left(-y_i - \varepsilon + z_i + \lambda^{t^t}\right)$
$\delta\alpha_i^{*t} = \eta\left(y_i - \varepsilon - z_i - \lambda^{t^t}\right)$ :
Si $\left(\alpha_i^t + \delta\alpha_i^t\right) \leq 0$ Entonces $\alpha_i^t = 0$
Otro Si $\left(\alpha_i^t + \delta\alpha_i^t\right) < C$ Entonces $\alpha_i^t = \alpha_i^t + \delta\alpha_i^t$
Otro $\alpha_i^t = C$
Fin_Si
Si $\left(\alpha_i^{*t} + \delta\alpha_i^{*t}\right) \leq 0$ Entonces $\alpha_i^{*t} = 0$
OtroSi $\left(\alpha_i^{*t} + \delta\alpha_i^{*t}\right) < C$ Entonces $\alpha_i^{*t} = \alpha_i^{*t} + \delta\alpha_i^{*t}$
Otro $\alpha_i^{*t} = C$
Fin_Si
Fin_Para
$\varpi^t = \sum_j \left(\alpha_j^{*t} - \alpha_j^t\right)$
Fin_Para

Fig.1 Algoritmo para el caso de la regresión.

## 5. Resultados

En esta sección se presenta un ejemplo de aproximación ilustrando el uso del algoritmo descrito anteriormente. Se tomó los valores de la función:

$$f(x) = \frac{sen\left(10\pi\ \frac{x}{3}\right)}{x} \tag{19}$$

en 100 puntos igualmente espaciados en el intervalo $[-1, 1]$, de manera similar al ejemplo dado en Vapnik [2]. Se tomó como núcleo la función de base radial. Después de experimentar con los parámetros del núcleo y del algoritmo, se escogieron los siguientes valores: $\delta = 0{,}1$, $\mu = 0{,}1$, $\eta = 1$, y el número máximo de generaciones igual a 3000. Se presentan las gráficas para los casos de $\varepsilon = 0{,}05$ y $0{,}1$, en las figuras 2 y 3. Para $\varepsilon = 0{,}05$ se obtuvieron 11 vectores soporte, y para $\varepsilon = 0{,}1$, 13 vectores soporte. En la Fig.4, se presenta el modelo polinomial de regresión de grado 6 que mejor ajusta a los datos, y que utiliza el criterio de mínimos cuadrados.

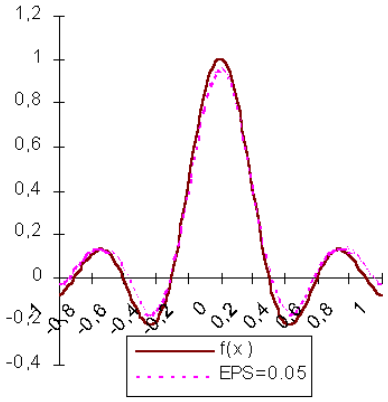


Fig. 2: Caso $\varepsilon = 0{,}05$.

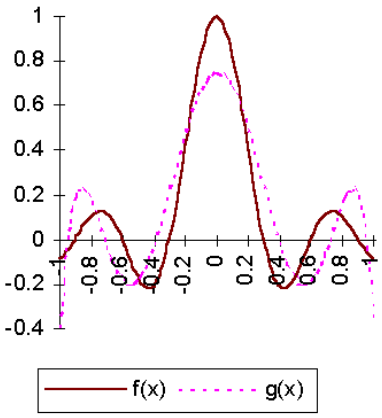


Fig. 3: Caso $\varepsilon = 0{,}1$.

Los resultados que se observan en las Fig.2 y 3 son comparables con aquellos dados por Vapnik [2]. Además, la suma de los cuadrados de los residuales para $\varepsilon = 0{,}05, 0{,}1$ y el modelo polinomial de regresión fueron de $0{,}15, 0{,}62, 2{,}65$ respectivamente, por lo que se obtiene una mejor aproximación de la función con el algoritmo propuesto. Esto se debe principalmente al criterio usado en el método de regresión $\varepsilon - SV$, el cual trata de obtener una aproximación de a lo más $\varepsilon$-desviaciones respecto a los datos originales. Lo que se consideraría como factor crítico para muchas aplicaciones.

## 6. Conclusiones

En este artículo se ha presentado una introducción al tema de MVS para regresión, y se ha propuesto un algoritmo sencillo y novedoso, basado en el trabajo de Campbell y Cristianini, para el caso de regresión.

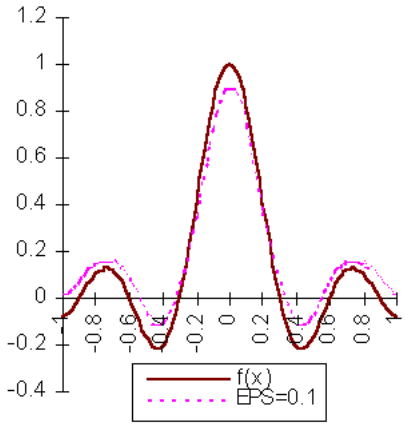


Fig. 4: Modelo polinomial de regresión de grado 6.

Mediante el uso del método de gradiente ascendente dentro del algoritmo, se evita la necesidad de utilizar un paquete de programación cuadrática, lo que hace que la aplicación de la regresión $\varepsilon - SV$ sea más accesible. Para una comparación de la regresión $\varepsilon - SV$ para estimar funciones respecto a otras técnicas Vapnik [2] ha presentado varios ejemplos. Así, para el caso de estimación de funciones de regresión lineal, se puede observar de las tablas de comparaciones que presenta Vapnik que la regresión $\varepsilon - SV$ proporciona resultados que son mejores en todos los casos que los obtenidos con el método de mínimos cuadrados ordinarios.

El ejemplo desarrollado en este trabajo es similar a uno dado en [2], y se puede observar que los resultados obtenidos utilizando el algoritmo propuesto son comparables con aquellos de Vapnik.

También la regresión $\varepsilon - SV$ tiene ventajas sobre el método de regresión tradicional en varios sentidos: no es necesario experimentar con varios modelos a priori y después identificar cuál es el de mejor ajuste, lo que ahorra tiempo. El modelo producido por la regresión $\varepsilon - SV$ se obtiene a partir del núcleo escogido y de los datos de entrenamiento. Sería importante en futuros trabajos explorar el impacto de los diferentes parámetros en el algoritmo, en particular la elección de la función núcleo.

## Referencias

[1] Vapnik, V. (1995) *The Nature of Statistical Learning Theory*. Springer-Verlag, New York.

[2] Vapnik, V. (1998) *Statistical Learning Theory*. John Wiley and Sons, New York.

[3] Ganapathiraju, A.; Hamaker, J.; Picone, J. (1998) "Support vector machines for speech recognition", Proc. *ICSLP*, Australia.

[4] Weston, J.; Gammerman, A.; Stitson, M.; Vapnik, V.; Vovk, V.; Watkins. (1998) "Density estimation using support vector machines", Technical Report CSD-TR-97-23.

[5] Müller, K.; Smola, A.; Rätsch, G.; Schölkopf, B.; Kohlmorgan, J.; Vapnik, V. (1997) "Predicting time series with support vector machines", Proc. of the *ICANN Conference*.

[6] Burges, C. (1998) "A Tutorial on support vector machines for pattern recognition", *Data Mining and Knowledge Discovery*, **2**(2).

[7] Vapnik, V.; Golowich, S.; y Smola, A. (1997) "Support vector method for function approximation, regression estimation, and signal processing", *Advances in Neural Information Processing Systems*, Vol. 9, MIT Press, Cambridge Mass.: 281–287.

[8] Smola, A.; Schölkopf, B. (1998) "A tutorial on support vector regression", NeuroCOLT2 Technical Report Series, NC2-TR-030.

[9] Gunn, S. (1998) "Support vector machines for classification and regression", ISIS Technical Report, University of Southampton.

[10] Stitson, M.; Weston, J.; Gammerman, A.; Vovk V.; Vapnik, V. (1996) "Theory of support vector machines", Tech. Report CSD-TR-96-17, RHUL.

[11] Girosi, F. (1998) "An equivalence between sparse approximation and support vector machines", *Neural Computation*, **10**(6): 1455–1480.

[12] Chin, K. (1998) "*Support Vector Machines applied to Speech Pattern Classification*". M.Phil. Thesis in Computer Speech and Language Processing, Cambridge University, Engineering Department.

[13] Vanderbei, R.J. (1997) "LOQO user's manual 3.10", Technical Report SOR-97-08, Statistics and Operations Research, Princeton University.

[14] Osuna, E.; Freund, R.; Girosi, F. (1997) "An improved training algorithm for support vector machines", *Neural Networks for Signal Processing VII - Proc. of the 1997 IEEE Workshop,* J. Principe, L. Gile, N. Morgan, y E. Wilson (Eds.): 276–285.

[15] Platt, J. (1998) "Sequential minimal optimization: A fast algorithm for training support vector machines", *Advances in Kernel Methods- Support Vector Learning*, B. Schölkopf, C. Burges, A. Smola (Eds.) MIT Press, Cambridge Mass.

[16] Campbell, C.; Cristianini, N. (1999) "Simple learning algorithms for training support vector machines", Department of Engineering Mathematics, University of Bristol.