ESTUDIO E IMPLEMENTACIÓN DE UNA HERRAMIENTA BASADA EN MÁQUINAS DE SOPORTE VECTORIAL APLICADA A LA LOCALIZACIÓN DE FALLAS EN SISTEMAS DE DISTRIBUCIÓN

GERMÁN ANDRÉS MORALES ESPAÑA
ALVARO GÓMEZ RUIZ

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA
2005

ESTUDIO E IMPLEMENTACIÓN DE UNA HERRAMIENTA BASADA EN MÁQUINAS DE SOPORTE VECTORIAL APLICADA A LA LOCALIZACIÓN DE FALLAS EN SISTEMAS DE DISTRIBUCIÓN

GERMÁN ANDRÉS MORALES ESPAÑA
ALVARO GÓMEZ RUIZ

Trabajo de grado para optar al título de Ingeniero Electricista

Director
HERMANN RAÚL VARGAS TORRES
Doctor Ingeniero Electricista

Codirector
JUAN CARLOS RODRÍGUEZ SUÁREZ
Magister(c) Ingeniero Electricista

ESCUELA DE INGENIERÍAS ELÉCTRICA, ELECTRÓNICA Y TELECOMUNICACIONES
UNIVERSIDAD INDUSTRIAL DE SANTANDER
BUCARAMANGA
2005

A mi padre German, por ser el ejemplo de mi vida.
A mi madre Nancy por ser guía en mi vida.
A Laura por brindarme su amistad, cariño y compañía.

Germán Andrés Morales España

A mis padres
Alvaro Gómez García
Luisa Ruiz de Gómez

Alvaro Gómez Ruiz

AGRADECIMIENTOS

Los autores expresan sus agradecimientos a:

Al Dr. Hermann Raúl Vargas, director del proyecto, por su dirección y asesoría las cuales hicieron posible la realización de este trabajo de grado.

Al Ing. Juan Carlos Rodríguez, codirector del proyecto, por su constante colaboración en el transcurso del desarrollo del proyecto.

Al Dr. Gabriel Ordoñez y el Ing. Jorge Cormane, por sus importantes aportes para mejorar la calidad de este trabajo de grado.

RESUMEN

TÍTULO:

ESTUDIO E IMPLEMENTACIÓN DE UNA HERRAMIENTA BASADA EN MÁQUINAS DE SOPORTE VECTORIAL PARA LA LOCALIZACIÓN DE FALLAS EN SISTEMAS DE DISTRIBUCIÓN*

AUTORES:

GERMÁN ANDRÉS MORALES ESPAÑA

ALVARO GÓMEZ RUIZ**

PALABRAS CLAVE:

Huecos de tensión, inteligencia artificial, localización de fallas, máquinas de soporte vectorial, sistemas de distribución.

DESCRIPCIÓN:

Este trabajo de grado propone un método para estimar la zona más probable de falla en sistemas de distribución con ayuda de inteligencia artificial. Se estudia una técnica de inteligencia artificial para clasificación llamada máquinas de soporte vectorial (SVM), esta técnica se aplica al problema de la localización de fallas en sistemas de distribución.

Para estimar la ubicación de la zona fallada es necesario conocer la topología del sistema de distribución que se va a analizar. Se divide el circuito en zonas, se simula el sistema en estado de falla variando la ubicación, tipo y resistencias de falla, adquiriendo así las señales de tensión en la cabecera del circuito. Teniendo una base de datos de posibles fallas, se extraen descriptores de los valores eficaces de las señales de tensión de fase, línea y secuencia cero. Se entrena la SVM con el fin de obtener los parámetros adecuados que proporcionen una clasificación satisfactoria tanto del tipo como la zona de falla. Para estimar un porcentaje de rendimiento de las SVM, se realiza una prueba con datos desconocidos.

Se realizó una prueba con un circuito de distribución, donde se obtienen resultados satisfactorios utilizando únicamente las señales de tensión. No se ve comprometida la precisión con valores de resistencias de falla altos (alrededor de 40Ω).

---

*Proyecto de Grado

**Facultad de Ingenierías Físico-Mecánicas. Escuela de Ingenierías Eléctrica, Electrónica y de Telecomunicaciones. Hermann Raúl Vargas Torres.

ABSTRACT

TITLE:

STUDY AND IMPLEMENTATION OF A TOOL BASED ON SUPPORT VECTOR MACHINES APPLIED TO THE LOCATION OF FAULTS IN DISTRIBUTION SYSTEMS*

AUTHORS:

GERMÁN ANDRÉS MORALES ESPAÑA

ALVARO GÓMEZ RUIZ**

KEY WORDS:

Location of faults, distribution systems, sags, artificial intelligence, support vector machines.

DESCRIPTION:

A methodology for estimate the most probable faulted area in distribution systems is presented. An artificial intelligence technique called support vector machines (SVM) is studied for classification; this technique is applied to the problem of location faults in distribution systems.

To estimate the location of the faulted area applying the methodology propose in this document is necessary know the distribution system topology. The distribution system is divided in areas, the system is simulated varying the location of the fault (nodes), type of the fault (single-phase-to-ground, phase-to-phase, two-phase-to-ground, and balanced three-phase and three-phase-to-ground) and fault resistances, for this way the voltage signs seen in the substation are acquiring for the distribution system. Having a database of possible faults in the distribution system, the describers from the voltages of phase, line and sequence zero are extracted. The SVM trains with the purpose of obtaining the appropriate parameters that provide a satisfactory classification for the type and the area location of the fault. To estimate a error percentage for the SVM, it is tested with unknown data.

The methodology is tested in a distribution system (SaskPower) and satisfactory results are obtained only using the voltage signs. The results indicated that the proposed methodology works well for low and high fault resistance (around 40 ohms).

---

*Degree Project

**School of Electrical Engineering. Hermann Raúl Vargas Torres.

Índice general

1. Introducción **1**

1.1. Objetivos 1

1.2. Estructura del documento. 2

2. Localización de fallas **3**

2.1. Introducción 3

2.2. Sistemas de distribución 4

2.3. Tipos de falla 4

2.4. Localización de fallas en sistemas de distribución 5

2.4.1. Método de Novosel [11]. 6

2.4.2. Método de Ratan Das [12]. 8

2.4.3. Algoritmo de Saha [20]. 10

2.4.4. Algoritmo de Takagi [22]. 13

3. Método propuesto **15**

3.1. Introducción 15

3.2. Historia 15

3.2.1. Metodología de aprendizaje 16

3.2.2. Máquinas de aprendizaje lineales 16

3.2.3. Funciones kernel 16

3.2.4. Teoría de generalización 17

3.2.5. Teoría de optimización 17

3.2.6. SVM 17

3.3. Máquina de soporte vectorial 20

3.3.1. Aprendizaje a través de ejemplos 20

3.3.2. El hiperplano clasificador óptimo 20

3.3.3. Solución al problema de optimización con restricciones . . . 22
3.3.4. Caso linealmente no separable . . . 24
3.3.5. SVM no lineales . . . 25
3.3.6. Multiclasificación con SVM . . . 27
3.3.6.1. Máquinas multiclasificadoras SV . . . 27
3.3.6.2. Máquinas biclasificadoras SV generalizadas . . . 28
3.4. Caracterización del sistema en falla. . . . 31
3.4.1. Detección de la falla . . . 32
3.4.2. Caracterización de los huecos de tensión . . . 32
3.4.2.1. Descriptores . . . 32
3.5. Metodología . . . 36
3.5.1. Zonificación de la red. . . . 36
3.5.2. Adquisición de datos de falla . . . 37
3.5.3. Preprocesamiento de la señal . . . 37
3.5.4. Entrenamiento de la SVM . . . 37
3.5.5. Prueba . . . 38

4. Resultados **39**
4.1. Introducción . . . 39
4.2. Sistema de prueba . . . 39
4.3. Pruebas realizadas . . . 39
4.4. Resultados . . . 40
4.4.1. Localizador independiente del tipo de falla - LIF . . . 40
4.4.1.1. Clasificador 1 . . . 41
4.4.1.2. Clasificador 2 . . . 42
4.4.2. Localizador dependiente del tipo de falla - LDF . . . 43
4.4.2.1. Etapa 1 . . . 43
4.4.2.2. Etapa 2 . . . 44
4.4.3. Modelo seleccionado . . . 48

5. Conclusiones, aportes y trabajos futuros **49**
5.1. Conclusiones . . . 49
5.2. Aportes . . . 50
5.3. Trabajos futuros . . . 50

Bibliografía **52**

A. Especificaciones del sistema de prueba. **54**

B. Entrenamiento de la máquina de soporte vectorial **57**

B.1. Método de descomposición 57

B.2. Función kernel 57

B.3. Selección de parámetros 57

B.3.1. Validación cruzada [16]. 58

B.3.2. Búsqueda en malla [16]. 59

C. Selección del modelo **60**

D. Toolbox SVM **64**

D.1. Introducción 64

D.2. Instalación 64

D.3. Formato de datos 64

D.4. Funciones 65

D.4.1. svm_2class 68

D.4.2. svm_2decis 68

D.4.3. svm_kernel 68

D.4.4. k_clases 69

D.4.5. k_decis 69

D.4.6. gen_cod 69

D.4.7. exact_decis 69

D.4.8. valcros 70

D.4.9. aleatorio 70

D.5. Ejemplos 70

D.5.1. Ejemplo de multiclasificación 71

D.5.2. Ejemplo de validación cruzada 71

D.6. Interfaz gráfica 74

Lista de tablas

3.1. Historia de la SVM. 19

4.1. Datos de entrenamiento y prueba del clasificador 1 41
4.2. Parámetros escogidos en el entrenamiento para el clasificador 1 41
4.3. Datos de entrenamiento y prueba del clasificador 2 42
4.4. Parámetros escogidos en el entrenamiento para el clasificador 2 43
4.5. Datos de entrenamiento y prueba de la SVM tipo 1 44
4.6. Parámetros escogidos en el entrenamiento para SVM tipo 1 44
4.7. Datos de entrenamiento y prueba de la SVM tipo 2 45
4.8. Parámetros escogidos en el entrenamiento para SVM tipo 2 45
4.9. Datos de entrenamiento y prueba de la SVM tipo 3 46
4.10. Parámetros escogidos en el entrenamiento para SVM tipo 3 46
4.11. Datos de entrenamiento y prueba de la SVM tipo 4 47
4.12. Parámetros escogidos en el entrenamiento para SVM tipo 4 47

A.1. Parámetros del generador. 54
A.2. Datos de cargas del sistema. 55
A.3. Factor de potencia. 55
A.4. Parámetros de las lineas. 56

D.1. Formato de descriptores 65
D.2. Estructura que configura la SVM 66
D.3. Estructura de la SVM entrenada 67

Lista de figuras

2.1. Falla monofásica a tierra. 4
2.2. Falla bifásica. 5
2.3. Falla bifásica a tierra. 5
2.4. Falla trifásica. 6
2.5. Esquema equivalente de una impedancia de carga. 6
2.6. Esquema de un circuito alimentador de distribución. 8
2.7. Circuito para el cálculo de la distancia a la falla (s). 9
2.8. Medición de las variables del sistema en el alimentador. 11
2.9. Diagrama de secuencia positiva del alimentador cuando se presenta una falla. 12
2.10. Esquema del circuito alimentador durante una falla 13
2.11. Equivalente de Thevenin del circuito alimentador durante una falla. 14

3.1. Hiperplanos que separan correctamente los datos. El OSH de la derecha tiene un mayor margen de separación entre clases, por lo tanto se espera una mejor generalización. . 21
3.2. Hiperplano lineal clasificador para el caso no separable 24
3.3. Transformación del espacio de entrada al espacio característico. 25
3.4. Hueco de tensión medido en un subestación en presencia de una falla. 32
3.5. Descriptores base. 33
3.6. Comportamiento de las tensiones de fase ante una falla monofásica en la fase A en dos lugares diferentes. 34
3.7. Comportamiento de las tensiones de línea y secuencia cero ante una falla monofásica en la fase A en dos lugares diferentes. 34
3.8. Falla monofásica en la misma fase a diferente ángulo. 35
3.9. Descriptores basado en los intervalos de la fase fallada. 36

4.1. Esquema LIF. 40

4.2. Resultados para el Clasificador 1. 42
4.3. Resultados para el Clasificador 2. 43
4.4. Esquema LDF. 43
4.5. Resultados para SVM tipo 1. 45
4.6. Resultados para SVM tipo 2 46
4.7. Resultados para SVM tipo 3 47
4.8. Resultados para SVM tipo 4 48

A.1. Diagrama unifilar del modelo de sistema de distribución. 54

B.1. *a.* SVM Sobre-entrenada y datos de entrenamiento. *b.* SVM Sobre-entrenada y datos de prueba. *c.* SVM sin Sobre-entrenamiento y datos de entrenamiento. *d.* SVM sin Sobre-entrenamiento y datos de prueba. 58

C.1. Modelo LIF con kernel polinomial 60
C.2. Modelo LDF con kernel polinomial 61
C.3. Modelo LIF con kernel sigmoide 61
C.4. Modelo LDF con kernel sigmoide 62
C.5. Modelo LIF con kernel RBF 62
C.6. Modelo LDF con kernel RBF 63

D.1. Derecha: multiclasificación uno contra el resto. Izquierda: multiclasificación uno contra uno. 70
D.2. Ejemplo validación cruzada. 72
D.3. Interfaz gráfica 74

Capítulo 1

Introducción

La energía eléctrica constituye uno de los elementos fundamentales para el desarrollo económico y social de una región, dado que su disponibilidad determina en gran medida los niveles de productividad, las posibilidades de desarrollo agroindustrial y la calidad de vida de los pobladores. Por ello el estudio de la calidad y continuidad del suministro de energía eléctrica ha tomado mucha fuerza y dentro de esta temática, el problema de la localización de fallas se posiciona como uno de los más importantes. Las interrupciones indeseadas significan pérdidas de dinero básicamente por dos razones:

- Pérdidas a nivel industrial[1]
- Multas impuestas a empresas de energía eléctrica por incumplimiento de las regulaciones establecidas (sobrepaso de los índices DES y FES[2]).

Así, es necesario conocer de manera confiable y rápida, el lugar donde ha ocurrido la falla, para tomar medidas que solucionen o mitiguen su impacto en la calidad y continuidad del servicio, cumpliendo con las regulaciones especificadas por la CREG[3].

Este trabajo de grado analiza mediante una técnica de inteligencia artificial, el problema de la localización de fallas[4] en sistemas de distribución e implementar un método que sea económicamente viable para los operadores de red, con el objeto de disminuir el tiempo de atención a fallas.

1.1. Objetivos

El objetivo general de este trabajo de grado es implementar un método basado en Máquinas de Soporte Vectorial (SVM) para determinar la zona más probable de la ubicación de una falla en sistemas de distribución de energía eléctrica.

[1] Tiempo de no operación o suspensión de procesos, pérdidas de información e insumos, daño en maquinarias, etc,.

[2] DES: Duración Equivalente de las interrupciones del Servicio y FES: Frecuencia Equivalente de las interrupciones del Servicio.

[3] CREG: Comisión de Regulación de Energía y Gas

[4] No se da una distancia al lugar de la falla como lo hacen los métodos algorítmicos. Con esta técnica basada en conocimiento, se determina la zona más probable de ubicación de la falla.

Por ello, para el cumplimiento de este objetivo, se trazaron los siguientes objetivos específicos:

- Analizar las máquinas de soporte vectorial como herramienta de clasificación.
- Adaptar la herramienta de Inteligencia Artificial para la localización de fallas.
- Escoger de manera comparativa entre las funciones kernel de base radial, sigmoide y polinomial, la que se ajusta a la localización de fallas.
- Determinar los parámetros de la función kernel y de la SVM que se ajusten a la localización de fallas.

1.2. Estructura del documento.

Este documento está dividido en cinco capítulos, cuyo contenido se resume a continuación:

En el capítulo 2 se contextualiza el problema de la localización de fallas y se presentan algunos métodos existentes que buscan solucionar este problema.

El capitulo 3 muestra la historia y teoría básica de las SVMs, la caracterización del sistema para ser utilizado por las SVMs y la metodología para el localizador de fallas.

El capítulo 4 contiene las pruebas y los resultados obtenidos al usar el método propuesto en un sistema de distribución típico, simulado en EMTP/ATP.

Finalmente el capítulo 5 presenta las conclusiones de este trabajo, así como algunas recomendaciones para trabajos futuros.

El documento completa cuatro anexos. El anexo A, muestra el modelo del sistema de distribución utilizado en las pruebas. El anexo B, presenta el método de selección de parámetros para la SVM. El anexo C se muestra la selección del modelo adecuado (arquitectura y kernel de la SVM) para la localización de fallas. El anexo D, describe de manera general la toolbox sobre SVM, realizada por los autores de este trabajo de grado.

Capítulo 2

Localización de fallas

2.1. Introducción

Los sistemas de transmisión y distribución de energía eléctrica están expuestos a fallas asociadas a causas externas tales como fenómenos atmosféricos, sobrecargas por cambios bruscos de demanda, acercamientos entre conductores y fallas relacionadas con elementos extraños (árboles, animales, etc.). En la mayoría de los casos, las fallas provocan daños físicos o alteraciones en la configuración del sistema. La atención de las fallas puede mejorar si se establece su naturaleza y localización.

La energía eléctrica hoy en día es uno de los productos más importantes y necesarios. Su interrupción permanente o transitoria trae consigo efectos para las industrias, el comercio y los usuarios residenciales. Aún en los casos más triviales, la existencia de fallas genera consecuencias en la salud, el ambiente, la economía, etc., siendo este último uno de los efectos de mayor preocupación. Por ello, se están desarrollando estrategias que minimicen los efectos de las fallas en los sistemas de energía eléctrica.

Las estrategias desarrolladas utilizan métodos basados en el modelo de la red (métodos algorítmicos) que buscan disminuir el tiempo para la identificación y localización del punto de falla. Los métodos tienen buena eficiencia, pero debido a las características propias de los algoritmos implementados, presentan limitantes y errores en circunstancias especiales. Por otra parte, muchos de estos algoritmos necesitan equipos especiales o múltiples para su implementación haciéndolos costosos. Pero dado el servicio y la infraestructura que poseen las empresas de transmisión, estas cuentan con mayores recursos para invertir e implementar métodos eficientes para localizar fallas. Caso contrario se presenta en los operadores de red, donde implementar las mismas medidas no resulta viable económicamente, debido a la cantidad de ramificaciones y nodos con que cuenta una red.

Es por esto que se explora una técnica de inteligencia artificial, para aprovechar mejor la información disponible y la experiencia adquirida, con un mínimo de equipos y personal en las subestaciones de energía eléctrica.

2.2. Sistemas de distribución

Se entiende por sistema de distribución de energía eléctrica a la disposición adoptada por los conductores, transformadores, consumidores y demás elementos del sistema, para lograr que la energía generada en las centrales pueda ser utilizada en los sitios de consumo [12].

Los principales inconvenientes en la identificación y localización de fallas en redes eléctricas están relacionados con aspectos de infraestructura. Los sistemas de distribución, por ejemplo, no disponen de localizadores de fallas dedicados en cada uno de sus circuitos (de hecho, en subestaciones de distribución sólo existen equipos enfocados a la monitorización del comportamiento global de las mismas). De otro lado, la configuración de sus redes no es homogénea, ya que generalmente contienen diferentes secciones de conductor y presentan múltiples derivaciones de cargas a lo largo de cada circuito alimentador.

Debido a lo anterior, la atención a fallas en los sistemas de distribución puede tornarse ineficiente por parte de los operadores de red y las empresas distribuidoras, al no contar con sistemas que permitan una rápida identificación y localización de las mismas.

2.3. Tipos de falla

Las fallas en los sistemas de energía se pueden dividir en dos categorías: falla serie y falla paralelo. Una falla paralelo es un desbalance entre fases o entre fase y neutro, mientras que la falla serie es un desbalance entre las impedancias de línea, que no incluye interconexiones entre líneas. Se analizarán las fallas paralelo por ser las más comunes en los sistemas de distribución [9].

Para un circuito trifásico, las fallas paralelo se pueden clasificar dentro de los siguientes grupos:

- Falla monofásica (FLT): Se presenta cuando una línea hace contacto con la tierra a través de un elemento conductor (Figura 2.1). Muchas de estas fallas ocurren por deterioro en el aislamiento de la línea. De todos los tipos de fallas que ocurren, las fallas monofásicas a tierra corresponden al 85 % [8].

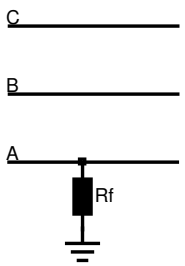


Figura 2.1: Falla monofásica a tierra.

- Falla bifásica (FLL): Se presenta cuando dos líneas se conectan entre sí (Figura 2.2). Es el segundo tipo de fallas más común entre los sistemas de distribución considerándose un 8 % de todas las fallas [8].

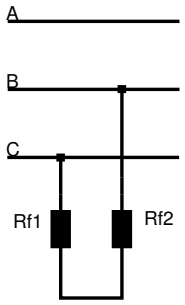


Figura 2.2: Falla bifásica.

- Falla bifásica a tierra (FLLT): Se presenta cuando dos líneas hacen contacto con tierra (Figura 2.3). Hasta un 5 % de las fallas son de este tipo [8].

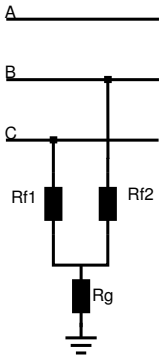


Figura 2.3: Falla bifásica a tierra.

- Falla trifásica (FLLL): Se presenta cuando las tres líneas se interconectan entre sí (Figura 2.4). Se considera que este tipo de fallas son balanceadas[1]. Aunque puede ser también conducida a tierra se considera su efecto no significativo. Estas fallas representan aproximadamente el 2 % de las fallas ocurridas [8].

2.4. Localización de fallas en sistemas de distribución

Los métodos algorítmicos son los más utilizados en las empresas de energía debido a su fácil implementación. Estos métodos utilizan el modelo del sistema, así como los valores de la componente fundamental de tensión y corriente de prefalla y postfalla. Tienen una alta precisión, pero se ven

[1] La resistencia de falla entre cada una de las líneas es igual.

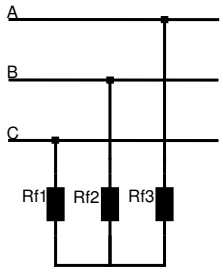


Figura 2.4: Falla trifásica.

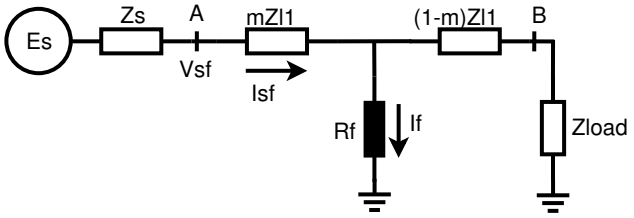


Figura 2.5: Esquema equivalente de una impedancia de carga.

afectados por las características no homogéneas de las redes, la alta ramificación de los circuitos y la existencia de cargas a lo largo de los mismas. Adicionalmente presentan problemas de múltiple estimación. A continuación se presentan algunos métodos utilizados para la localización de fallas en sistemas de distribución de energía, basados principalmente en el cálculo de la impedancia.

2.4.1. Método de Novosel [11].

Este método se aplica principalmente a líneas de transmisión cortas pudiéndose extender a las de distribución. Las cargas a lo largo de la línea se representan como una carga concentrada detrás de la falla (Figura 2.5). Esta aproximación es válida debido a que las impedancias de carga son mucho mayores que la impedancia del alimentador.

El método se fundamenta en el cálculo de la impedancia de falla vista desde ambos extremos de la línea, basado en la información en estado estable de prefalla y postfalla y las corrientes medidas en la subestación. En el esquema de la Figura 2.5, la impedancia de carga y la impedancia vista desde la fuente está dada por:

$$Z_{LOAD} = \frac{V_{PS}}{I_{PS}} - Z_{L1} \tag{2.1}$$

$$Z_S = \frac{\Delta V_S}{\Delta I_S} \tag{2.2}$$

donde V_{PS} e I_{PS} son los valores de prefalla de tensión y corriente medidos en la subestación, Z_{L1} impedancia desde la subestación hasta la carga y

$$\Delta V_S = V_{SF} - V_{PS} \tag{2.3}$$

$$\Delta I_S = I_{SF} - I_{PS} \tag{2.4}$$

La impedancia medida desde la subestación se obtiene de la siguiente relación,

$$Z_{MED} = \frac{V_{SF}}{I_{SF}} = mZ_{L1} + R_F \frac{I_F}{I_{SF}} \tag{2.5}$$

De la ecuación (2.5) se puede obtener la siguiente relación cuadrática:

$$m^2 - m * k_1 + k_2 - k_3 * R_F = 0 \tag{2.6}$$

donde,

$$k_1 = \frac{V_{SF}}{I_{SF} * Z_{L1}} + \frac{Z_{LOAD}}{Z_{L1}} + 1 \tag{2.7}$$

$$k_2 = \frac{V_{SF}}{I_{SF} * Z_{L1}} * \left(\frac{Z_{LOAD}}{Z_{L1}} + 1 \right) \tag{2.8}$$

$$k_3 = \frac{\Delta I_S}{I_{SF} * Z_{L1}} * \left(\frac{Z_S + Z_{LOAD}}{Z_{L1}} + 1 \right) \tag{2.9}$$

La solución para la ecuación cuadrática se obtiene separando las partes real e imaginaria. Se puede obtener el valor de m después de eliminar el término R_F. El tipo de falla se determina con una adecuada selección de las corrientes y tensiones dentro del cálculo.

Para una mayor exactitud, el método puede incluir compensación debido a las cargas distribuidas; sin embargo, en un sistema con muchas cargas distribuidas, la exactitud puede reducirse en la localización de fallas hacia el final del alimentador.

- Ventajas
  - Considera la resistencia de falla y las corrientes de carga.
  - Es un método de análisis sencillo para determinar el tipo de falla, basado en las medidas de prefalla y falla de tensiones y corrientes en la subestación.
  - Las cargas presentes en el sistema pueden ser polifásicas.
  - Tiene en cuenta la no homogeneidad del sistema.

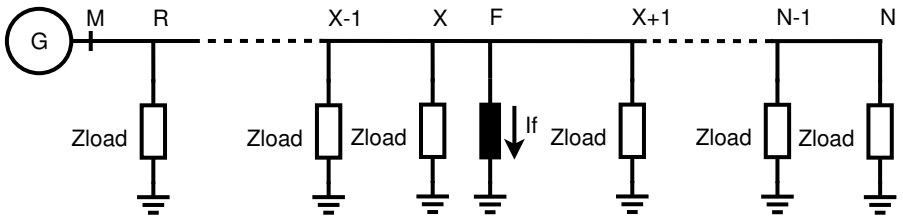


Figura 2.6: Esquema de un circuito alimentador de distribución.

- Desventajas

  - La impedancia de carga se considera constante, ésta no depende de los cambios de tensión presentes en la carga cuando ocurre la falla. La impedancia de carga se calcula con los valores de tensión y corriente de prefalla.
  - El método se hace inexacto si la impedancia del alimentador es grande o si la impedancia de la carga es muy pequeña.
  - Puede presentar múltiple estimación.

2.4.2. Método de Ratan Das [12].

Este método utiliza la información de tensiones y corrientes en estado estable de prefalla y postfalla a frecuencia fundamental.

En la Figura 2.6 se encuentra el esquema de un circuito alimentador de distribución con una fuente equivalente, y una representación de las cargas laterales y diferentes secciones de conductor a lo largo del alimentador.

En la primera parte del método se realiza una estimación inicial de la localización de la falla entre dos nodos del sistema (nodos X y $X+1$). La estimación se realiza teniendo en cuenta los parámetros de las líneas, el tipo de falla y los fasores de secuencia de tensión y corriente. Con base en esta estimación, todas las cargas pertenecientes a los laterales desde la fuente hasta la posible localización de la falla se consideran conectadas al nodo en que está conectado el lateral. Los efectos de estas cargas se representan por la compensación de sus corrientes. Se emplean modelos de carga de tipo estático para todas las cargas hasta el nodo X, e igualmente para las demás cargas representadas como una carga concentrada en el extremo remoto. Este modelo está descrito por la siguiente ecuación, para un nodo R:

$$Y_R = G_R \left|V_R\right|^{np-2} + jB_R \left|V_R\right|^{nq-2} \tag{2.10}$$

Siendo V_R la tensión en el nodo R, Y_R la admitancia de carga, G_R y B_R constantes proporcionales a la conductancia y susceptancia respectivamente (estimadas de los valores de prefalla), y np y nq las

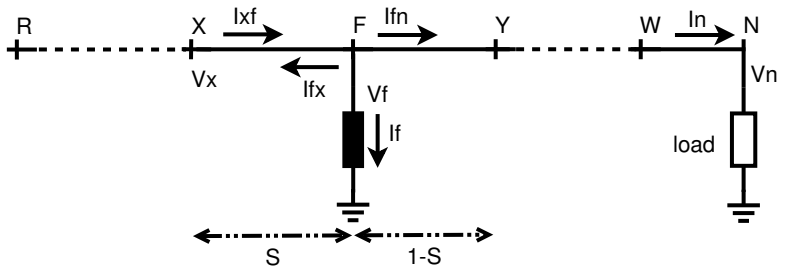


Figura 2.7: Circuito para el cálculo de la distancia a la falla (s).

constantes para las componentes activa y reactiva de la carga. Se calculan las tensiones y corrientes en el nodo F durante la falla asumiendo que todas las cargas posteriores a la estimación de la falla se encuentran concentradas en el extremo remoto (nodo N)

Las tensiones y corrientes en el nodo F y X se relacionan por la ecuación,

$$\begin{bmatrix} V_F \\ I_{FX} \end{bmatrix} = \begin{bmatrix} 1 & -s * B_{XY} \\ s * C_{XY} & -1 \end{bmatrix} \begin{bmatrix} V_X \\ I_{XF} \end{bmatrix} \tag{2.11}$$

donde s es la distancia en por unidad ($p.u.$) hasta el nodo F desde el nodo X (Figura 2.7).

Las tensiones y corrientes de secuencia en los nodos N y F durante la falla están relacionadas por la siguiente ecuación:

$$\begin{bmatrix} V_N \\ -I_N \end{bmatrix} = \begin{bmatrix} D_E & -B_E \\ C_E & -A_E \end{bmatrix} \begin{bmatrix} 1 & -(1-s) * B_{XY} \\ -(1-s) * C_{XY} & 1 \end{bmatrix} \begin{bmatrix} V_F \\ I_{FN} \end{bmatrix} \tag{2.12}$$

donde A_E, B_E, C_E y D_E son las constantes equivalentes de las secciones en cascada entre los nodos $X+1$ y N.

Las corrientes en el nodo F están dadas por la siguiente expresión:

$$I_{FN} = -I_{FX} - I_F \tag{2.13}$$

Mediante las sustituciones adecuadas, se puede llegar a una expresión de V_N e I_F, en términos de V_K e I_{XF}, truncando los elementos de orden superior en s, así:

$$\begin{bmatrix} V_N \\ I_F \end{bmatrix} = \frac{1}{K_V + s * K_W} \begin{bmatrix} K_M + s * K_N & s * K_P \\ K_Q + s * K_R & K_V + s * K_U \end{bmatrix} \begin{bmatrix} V_X \\ I_{XF} \end{bmatrix} \tag{2.14}$$

donde los términos K son parámetros complejos calculados empleando las constantes Y_N, B_{XY}, C_{XY}, A_E, B_E, C_E y D_E.

Finalmente, la estimación de la localización de la falla, punto F desde el nodo X, expresada como una fracción de la distancia desde el punto X y el punto $X+1$, se calcula a partir de la relación

tensión corriente. Para una falla fase-tierra, esta relación está dada por:

$$\frac{V_{AF}}{I_{AF}} = \frac{V_{0F} + V_{1F} + V_{2F}}{I_{0F} + I_{1F} + I_{2F}} = Z_F \tag{2.15}$$

Igualando los términos imaginarios en la anterior ecuación, y realizando las sustituciones adecuadas para las tensiones y corrientes de secuencia resulta:

$$Im\left[\frac{K_A + s * K_B}{K_C + s * K_D}\right] = 0 \tag{2.16}$$

donde cada parámetro Kes un número complejo. La solución para s, eliminando los términos de orden superior es:

$$s = \frac{K_{AR} * K_{CI} - K_{AI} * K_{CR}}{(K_{CR} * K_{BI} - K_{CI} * K_{BR}) + (K_{DR} * K_{AI} - K_{DI} * K_{AR})} \tag{2.17}$$

- Ventajas
  - El método propuesto considera que una línea de distribución puede tener conductores diferentes por tramo.
  - Toma en cuenta que la admitancia de la carga varía en función de la tensión.
  - Considera la compensación de corrientes debido a las cargas laterales.
- Desventajas
  - Se tiene que caracterizar la carga en cada uno de los nodos para poder obtener valores reales de las constantes de carga np y nq.
  - No es claro el proceso para concentrar las cargas que tienen valores diferentes de np y nq.

2.4.3. Algoritmo de Saha [20].

Este algoritmo utiliza las mediciones de tensión y corriente del sistema de distribución a frecuencia fundamental antes y durante la falla. La estimación de la distancia a la falla se calcula con base en principios topológicos. Este método fue propuesto para su aplicación en sistemas de media tensión, los cuales pueden incluir derivaciones de carga laterales. Igualmente tiene en cuenta la no homogeneidad de la red.

En la primera etapa del algoritmo se calcula un lazo de impedancia de falla con base en las mediciones de tensión y corriente antes y durante la falla. Seguidamente, se calcula la impedancia a lo largo del alimentador asumiendo la ocurrencia de la falla sucesivamente en cada sección del alimentador. La distancia a la falla es el resultado de la comparación de la impedancia medida con la calculada.

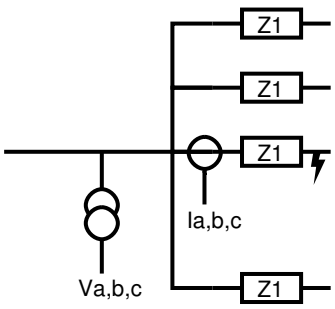


Figura 2.8: Medición de las variables del sistema en el alimentador.

Debido a que sólo son conocidas las mediciones en un extremo del alimentador (Figura 2.8), el lazo de impedancia de falla en secuencia positiva se calcula con el tipo de falla.

En el ejemplo de la Figura 2.8 se supone una falla en el alimentador k, el cual tiene una impedancia equivalente de prefalla Z_{LK}, los demás alimentadores se representan como una rama equivalente en paralelo con una impedancia equivalente Z_L. Ambas impedancias se asumen de secuencia positiva, y éstas se calculan con base en las matrices de transformación tradicionales.

Asumiendo que la impedancia Z_L permanece invariante durante la falla, para el estado de prefalla se establece la siguiente ecuación:

$$Z_{PRE} = \frac{V_{PRE}}{I_{PRE}} = \frac{Z_L * Z_{LK}}{Z_L + Z_{LK}} \tag{2.18}$$

donde V_{PRE}e I_{PRE} son los valores de tensión y corriente de prefalla respectivamente.

Dos posibles casos de prefalla se pueden considerar:

- Lazo de falla Fase-Fase: La impedancia del alimentador en falla, vista desde la cabecera del alimentador, se obtiene de la ecuación:

$$Z_K = \frac{V_{PP}}{I_{PP} - (1 - K_{ZK})\frac{V_{PP}}{Z_{PRE}}} \tag{2.19}$$

donde:

$$K_{ZK} = \frac{Z_L}{Z_L + Z_{LK}} \tag{2.20}$$

- Lazo de falla Fase-Tierra: La impedancia de secuencia positiva se calcula en forma tradicional. Debido a que el lazo de falla considera únicamente un circuito fase-tierra, la corriente de secuencia cero medida en la cabecera del alimentador contiene a la corriente de falla I_{KN}, y ésta fluye a través de las capacitancias del sistema.

$$Z_K = \frac{Z_G * Z_{PRE}}{Z_{PRE} - Z_G\,(1 - K_{ZK}) * \left(1 - \frac{V_0}{V_{FASE}}\right)} \tag{2.21}$$

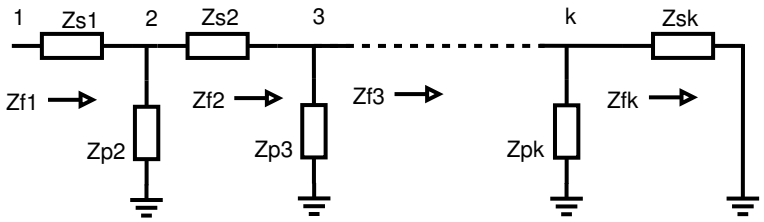


Figura 2.9: Diagrama de secuencia positiva del alimentador cuando se presenta una falla.

donde

$$Z_G = \frac{V_{FASE}}{I_{FASE} + K_{KN} * I_{KN}} \tag{2.22}$$

$$V_0 = \frac{V_A + V_B + V_C}{Z_{PRE}} \tag{2.23}$$

En esta expresión, la impedancia del lazo de falla se define en términos de la impedancia de secuencia positiva.

Basados en el cálculo de la impedancia del lazo de falla, Z_K, y en los parámetros conocidos de la red, es posible determinar la distancia a la falla.

En el esquema de la Figura 2.9, se encuentran representadas las cargas en cada nodo a lo largo del alimentador. La impedancia vista en cada nodo hacia el punto de falla, Z_{FI} está definida en la siguiente ecuación:

$$Z_{FI} = \frac{Z_{PI} * (Z_{FI-1} - Z_{SI-1})}{Z_{PI} - Z_{FI-1} - Z_{SI-1}} \tag{2.24}$$

donde Z_{SI-1} representa la impedancia del segmento de cable mientras que Z_{PI} representa la impedancia equivalente de las ramas conectadas al nodo I. El valor de esta impedancia se calcula con base en las mediciones en estado estable de prefalla.

En la anterior relación, puede inferirse de forma directa que la impedancia equivalente Z_{FI} se aproxima a cero cuanto más cerca se esté del punto de falla, mientras que la impedancia de la sección fallada es:

$$Z_{FK} = \lambda_{FK-1} - Z_{SK-1} + R_F \tag{2.25}$$

donde λ_{FK-1} representa la distancia en pu desde el nodo K hasta el punto de falla (asumiendo que la longitud total del alimentador es 1). Z_{SK-1} es la impedancia del segmento de cable entre los nodos $K-1$ y K y R_F es la impedancia de falla.

- Ventajas

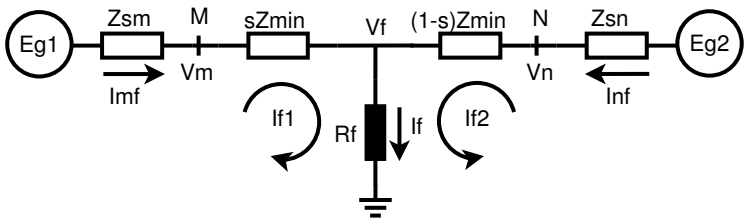


Figura 2.10: Esquema del circuito alimentador durante una falla

- Realiza una estimación del posible lugar de falla y después trata de encontrar el punto exacto de la misma.
- Considera cargas intermedias.
- Tiene en cuenta la no homogeneidad del sistema.

- Desventajas
  - Puede ubicar erróneamente la falla si la resistencia de falla es muy alta.
  - Se requiere una completa base de datos de los parámetros de la línea, carga y conectividad de la red.
  - El método no considera la variabilidad de la admitancia de carga con la tensión.

2.4.4. Algoritmo de Takagi [22].

Está basado en la medición de las componentes de frecuencia fundamental de tensiones y corrientes en el extremo de fuente del circuito, antes y durante la falla. La estimación de la localización de la falla se basa en el equivalente de Thevenin del sistema fallado (Figura 2.10).

$$V_F = (I_{F1} + I_{F2}) * R_F \tag{2.26}$$

Sin embargo, el circuito alimentador se puede representar por su equivalente de Thevenin desde el punto de falla (Figura 2.11).

En este esquema, los valores de V_F y $I_{F1}^{"}$se pueden expresar en términos de los valores de tensión y corriente en el extremo M, y la constante de propagación de la línea γ, con las ecuaciones tradicionales de líneas de transmisión:

$$V_F = V_M * \cosh(\gamma_{MN} * I_{MF}) - Z_{MN} * I_{MF} * \sinh(\gamma_{MN} * I_{MF}) \tag{2.27}$$

$$I_{F1}^{"} = \frac{V_M^{"}}{Z_{MN}} * \sinh(\gamma_{MN} * I_{MF}) - I_{MF}^{"} * \cosh(\gamma_{MN} * I_{MF}) \tag{2.28}$$

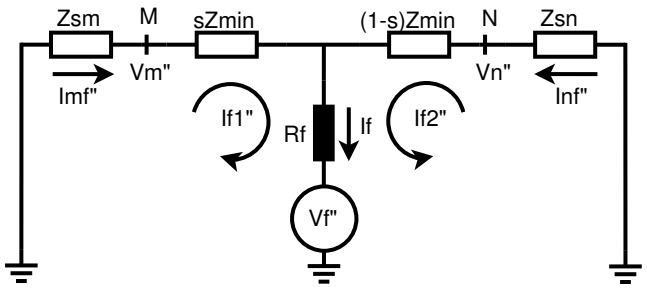


Figura 2.11: Equivalente de Thevenin del circuito alimentador durante una falla.

Si la corriente de falla I_F se expresa como un múltiplo de $I_{F1}^{"}$:

$$I_{F1} = \eta * I_{F1}^{"} \tag{2.29}$$

donde

$$\eta = \frac{I_F}{I_{F1}^{"}} \tag{2.30}$$

$$\theta = \arg(\eta) \tag{2.31}$$

Al combinar las anteriores ecuaciones, y asumiendo que $\theta = 0$, se puede obtener la siguiente expresión para la distancia a la falla l_{MF}:

$$l_{MF} = \frac{Im\left(V_M * \left(I_{MF}^{"}\right)^*\right)}{Im\left(\gamma_{MN} * Z_{MN} * I_{MF} * \left(I_{MF}^{"}\right)^*\right)} \tag{2.32}$$

Este algoritmo sin embargo presenta varias debilidades:

- La definición de corrientes que definen al parámetro θ, dependen de las impedancias de fuente que no son conocidas directamente, y que de hecho varían constantemente con la configuración del sistema.
- El algoritmo utiliza una comparación entre valores de prefalla y postfalla sobre el mismo circuito, cuando las características de impedancia del mismo varían en condiciones de falla.

Capítulo 3

Método propuesto

3.1. Introducción

La propuesta aquí presentada se fundamenta en los métodos de inteligencia artificial y no en métodos algorítmicos.

Los métodos algorítmicos utilizan la información del modelo de la red y el valor eficaz de la componente fundamental de tensión y corriente en estado estable, tanto de falla como de prefalla. Estos métodos permiten obtener la distancia eléctrica desde el sitio de medida hasta la ubicación de la falla, pero dada la naturaleza topológica de la red, tienen el inconveniente de presentar múltiple estimación de la localización de la falla.

Los métodos inteligencia artificial son aquellos que utilizan características significativas tanto en estado permanente como en estado transitorio de las señales de tensión y corriente, registrados durante una falla. Usando este tipo de métodos es posible detectar la sección del sistema afectada por el evento. Estos métodos aplicados a problemas simples, pueden identificar con mucha precisión los factores a tener en cuenta para un aprendizaje exitoso con solo la teoría estadística de aprendizaje, pero las aplicaciones del mundo real demandan el uso de modelos y algoritmos mas complejos (ej. redes neuronales, técnicas Bayesianas, etc), que son difíciles de analizar.

Este trabajo utiliza la técnica de inteligencia artificial llamada *"máquinas de soporte vectorial"*, pues a diferencia del método Bayesiano presenta la ventaja de no requerir ningún tipo de hipótesis sobre la densidad de probabilidad de los rasgos, mientras que sobre las redes neuronales ofrecen la ventaja de ser convenientes en términos de la dimensionalidad del problema[1] [14].

3.2. Historia

La historia aquí plasmada está basada en las referencias bibliográficas [2, 1, 23].

[1] Como se verá más adelante, la arquitectura de las SVM, solo depende del parámetro C, la función kernel (incluyendo sus parámetros). Para el caso del kernel RBF [7] solo se requiere el parámetro σ evitando así requerimientos sobre parámetros exclusivos de arquitectura, tales como número de nodos y capas, tipo de conexión entre capas, etc.

3.2.1. Metodología de aprendizaje

Ver sección 3.3.1.

El problema del aprendizaje a través de ejemplos, ha sido investigado por los filósofos a lo largo de la historia bajo el nombre de inferencia inductiva, pero la formulación clásica de este problema se realizó en el siglo XVIII por el filosofo escoses David Hume. Sólo hasta el siglo XX se reconoció como imposible la inducción pura a menos que se asuma algún conocimiento a priori. Este logro conceptual es debido principalmente al trabajo de Karl Popper.

Dentro del estudio del problema de aprendizaje desde un marco estadístico, existe la idea de la regresión de mínimos cuadrados propuesta en el siglo XVIII por Gauss y el procedimiento para la clasificación realizado por Ronald Fisher en 1930, que proveen el punto de partida para la mayoría de los métodos y análisis.

En el campo de la inteligencia artificial, Alan Turing en 1950 propuso la idea de las máquinas de aprendizaje, contradiciendo la creencia *"una máquina solo puede hacer las cosas que sabemos decirle como debe hacerlo"* y mostrando una importante característica de las máquinas de aprendizaje *"el maestro frecuentemente desconocerá lo que sucede en el interior de su alumno pero será capaz en alguna medida de predecir su comportamiento"*. En particular la idea de modelar los problemas de aprendizaje como problemas de búsqueda en espacios hipotéticos es característica de la metodología de inteligencia artificial. El desarrollo de algoritmos de aprendizaje se convirtió en un importante sub-campo de la inteligencia artificial y con el tiempo se separó en un área llamada máquinas de aprendizaje.

3.2.2. Máquinas de aprendizaje lineales

Ver sección 3.3.2.

En 1956 Frank Rosenblatt introdujo las reglas de aprendizaje para el perceptrón, basándose en el trabajo realizado por Fisher en 1930 sobre clasificación.

La idea del hiperplano de margen máximo ha sido redescubierta varias veces y es nuevamente discutido por Vapnik y Lerner, Duda y Hart. Un algoritmo iterativo conocido como Adatrón para el aprendizaje con hiperplanos de margen máximo fue investigado por Anlauf y Biehl (en la literatura de la mecánica estadística).

3.2.3. Funciones kernel

Ver sección 3.3.5.

La teoría de los kernels es antigua. El teorema de Mercer es anterior a 1909, y el estudio de kernels del espacio Hilbert fue desarrollada por Aronszajn en la década de 1940.

El uso del teorema de Mercer para interpretar los kernels como el producto punto en un espacio característico, se introdujo a las máquinas de aprendizaje en 1964 por el trabajo de Aizermann, Bravermann y Rozoener.

El primer uso del kernel polinomial fue echo por Poggio en 1975. Los kernels fueron ampliamente empleados en máquinas de aprendizaje y redes neuronales por Poggio y Girosi desde el comienzo de los 90. Pero sus posibilidades no fueron completamente entendidas hasta que fue usada por Boser, Guyon y Vapnik en la introducción al método de los vectores de soporte.

3.2.4. Teoría de generalización

El análisis de la generalización basada en la dimensión VC fue desarrollada por Vapnik y Chervonenkis a mediados de los años 60. En 1984 Valiant coloca las bases de lo que años más tarde se conoció como teoría del aprendizaje computacional, describiendo un gran número de modelos, por ejemplo, el aprendizaje en línea, el aprendizaje supervisado y el no supervisado, el aprendizaje reforzado, etc. La teoría VC ha sido usada hace tiempo para analizar el desempeño de diversos sistemas de aprendizaje como, árboles de decisión, redes neuronales, etc., además, muchos aprendizajes heurísticos y principios usados en aplicaciones prácticas de máquinas de aprendizaje son explicados en términos de la teoría VC. La teoría VC también es conocida como la teoría de aprendizaje estadístico.

3.2.5. Teoría de optimización

Ver sección 3.3.3.

La teoría de optimización data desde antes del trabajo de Fermat, quien formuló los resultados para problemas sin restricciones en el siglo XVII. El paso al caso con restricciones de igualdad, lo realizó Lagrange en 1788, y solo hasta 1951 esta teoría fue generalizada al caso con restricciones de desigualdad por Kuhn y Tucker, dando lugar a la teoría moderna de optimización convexa. Karush ya había descrito las condiciones para la optimización en 1939, razón por la cual las condiciones extraídas del teorema de Kuhn Tucker son comúnmente conocidas como las condiciones Karush Kuhn Tucker (KKT).

En los años siguientes, grandes trabajos sobre el caso de programación convexo fueron hechos por Wolfe, Mangasarian, Duran, y otros, además, la difusión de computadoras en los años sesenta aumentó el interés en la programación matemática, que estudia la solución de problemas (normalmente lineal o cuadrático) por métodos de optimización.

3.2.6. SVM

Son una clase específica de algoritmos caracterizados por:

- Uso de funciones kernel

- Ausencia de mínimos locales
- Capacidad de control obtenida al manejar el margen.

Desarrollado por Vladimir Vapnik y sus colaboradores, su primera presentación fue en 1992. Aunque sus propiedades ya existían y habían sido usadas en las máquinas de aprendizaje desde 1960:

- Los hiperplanos de margen grande en el espacio de entrada fueron tratados por Duda y Venado, Cubra, Vapnik, y otros.
- El uso de kernels se propuso por Aronszajn, Wahba, Poggio, y otros, pero fue Aizermann en 1964 quien introdujo la interpretación geométrica de los kernels como los productos internos en un espacio característico.
- El uso de variables de relajación para superar el problema de ruido y la no separabilidad, se introdujo en los años sesenta por Smith y perfeccionado por Bennett y Mangasarian.

Sin embargo, no fue hasta 1992 que todos estas características se reunieron para formar el clasificador del margen fuerte (ver sección 3.3.2), la Máquina Soporte Vectorial básica, y hasta 1995 la versión del margen débil (ver sección 3.3.4) se introdujo por Cortés y Vapnik: *"Es sorprendente como natural y elegantemente todas las piezas se ajustan entre si y se complementan unas con otras"* [23].

Las publicaciones de Shawe-Taylor y Bartlett dieron el primer límite estadístico riguroso en la generalización de las SVMs de margen fuerte. Luego en trabajos Shawe-Taylor y Cristianini dan límites similares para los algoritmos de margen débil.

Después de la introducción de las SVMs, un creciente número de investigadores han trabajado en el análisis algorítmico y teórico de estos sistemas, creando en unos pocos años una nueva línea de investigación, fusionando conceptos de disciplinas tan distantes como estadística, análisis funcional, optimización y maquinas de aprendizaje.

Se ha trabajado sobre la generalización del método y su extensión al caso de multiclasificación por parte de Weston y Watkins; Platt, Cristianini y Shawe-Taylor; Vapnik.

Jaakkola y Haussler, Vapnik y Chapelle, Weston y Herbrich, Wahba, Lin, Zhang, Opper y Winther en sus trabajos proporcionan un análisis del error esperado de la validación cruzada y Vapnik presenta el límite del error esperado en términos del margen y el radio de la esfera más pequeña que contiene un vector de soporte.

Tabla 3.1: Historia de la SVM.

Algunos de los grandes acontecimientos	
Antes del siglo XVIII.	Filósofos. Problema de Inferencia Inductiva
Siglo XVIII.	Carl Gauss. Mínimos cuadrados
1788.	Joseph Louis Lagrange. Optimización con restricciones (igualdades)
1909.	J. Mercer. Teorema de Mercer (producto punto en un espacio característico)
1930.	Ronald Fisher. Procedimiento para clasificación Proporciona puntos de partida (Análisis y métodos).
1940.	N. Aronszajn. Espacios Hilbert
1950.	Alan Turin. Propuso la idea de las máquinas de aprendizaje (Las máquinas podrían pensar)
1951.	Kuhn y Tucker. Optimización con restricciones (desigualdades)
1957.	Frank Rosenblatt. Perceptrón (primera máquina de aprendizaje)
1960.	Vapnik y Chervonenkis. Teoría de generalización
1963.	Vapnik, V. y Lerner, A. Máximo margen, generalización
1964.	M. Aizerman, E. Braverman, y L. Rozonoer. Interpretación geométrica del Kernel. Introdujeron las funciones Kernel en las máquinas de aprendizaje.
1968.	F. W. Smith. Problemas no separables (con ruido) Introdujo variables slack (de relajación)
1975.	T. Poggio. Kernel polinomial
1986.	Rumelhart, Hilton y Williams. Back - propagation. (Segundo nacimiento del Perceptrón) Método para resolver el problema Perceptrón, hallando el peso de todas las neuronas simultáneamente
1989.	J. K. Anlauf y M. Biehl. El adatrón: Adaptación del perceptrón maximizando el margen (máximos locales)
1990.	T. Poggio y F. Girosi. Utilizar funciones Kernel en las redes neuronales.
1991.	Bennett y Mangasarian. Perfeccionaron la técnica de variables slack y margen débil
1992.	Boser, Guyon y Vapnik. Introdujo las SVM
1995.	Cortes Vapnik. Se introdujo en las SVM el margen débil
1996.	Shawe-Taylor y Cristianini. Margen débil
1998.	Weston y Watkins; Platt, Cristianini y Shawe-Taylor; Vapnik. Multiclasificación para las SVMs
1998.	Vapnik, Generalización de funciones Kernel (buen uso).
Después de 1998.	Un creciente número de investigadores trabaja en la evolución de las SVMs. Jaakkola y Haussler, Vapnik y Chapelle, Weston y Herbrich, Wahba, Lin, Zhang, Opper y Winther. Proporcionan el análisis de validación cruzada para el error esperado.

3.3. Máquina de soporte vectorial

Las máquinas de vectores de soporte o SVM son una consecuencia practica de la teoría de aprendizaje y su estudio es útil básicamente por dos razones:

1. Desde el punto de vista teórico satisface totalmente la teoría de vectores de soporte para aprendizaje, basado en ideas maravillosamente simples y que proveen una clara intuición de que se esta aprendiendo de los ejemplos.

2. Ofrece grandes resultados en aplicaciones prácticas. En este sentido se considera que el algoritmo de soporte vectorial (SV) es la intercepción entre la teoría y la práctica.

3.3.1. Aprendizaje a través de ejemplos

Considérese n datos de entrenamiento $\mathbb{N}$ dimensional $(\vec{x}_i)$ con su respectiva etiqueta (y_i),

$$\vec{x}_i \in \mathbb{R}^{\mathbb{N}} \qquad \text{y} \qquad y_i \in \{+1, -1\} \tag{3.1}$$

Con los cuales se busca estimar una función f tal que para una entrada en $\mathbb{R}^{\mathbb{N}}$ produzca una salida en $\{\pm 1\}$,

$$f \colon \mathbb{R}^{\mathbb{N}} \to \{+1, -1\} \tag{3.2}$$

para que así pueda clasificar correctamente un nuevo dato $(\vec{x}, y)$ (téngase en cuenta que $y = f(\vec{x})$ para este nuevo dato, y es generado con la misma distribución de probabilidad $P(\vec{x}_i, y_i)$ de los datos de entrenamiento). Si ninguna restricción es impuesta en la clase de función que se escoge, se pueden cometer errores en el estimado, pues aunque trabaje bien en los datos de entrenamiento, no necesariamente tiene una buena generalización para datos desconocidos, luego el aprendizaje es imposible y la minimización del error de entrenamiento no implica que se deba esperar un pequeño error de prueba.

3.3.2. El hiperplano clasificador óptimo

Los clasificadores de soporte vectorial están basados en hiperplanos que separan los datos de entrenamiento en dos subgrupos que poseen cada uno una etiqueta propia. En medio de todos los posibles planos de separación entre las dos clases, etiquetadas $y \in \{-1, +1\}$, existe un único *hiperplano de separación óptimo*(OSH), de forma que la distancia entre el hiperplano óptimo y el patrón de entrenamiento más cercano sea máxima, con la intención de forzar la generalización de la máquina de aprendizaje [7, 21].

El *hiperplano de separación óptimo* (OSH) se expresa de la forma[2]:

[2]Un plano $\mathbb{N}$ dimensional es de la forma $a_n x_n + a_{n-1} x_{n-1} + ... + a_1 x_1 + a_0 = 0$, donde se puede volver a escribir

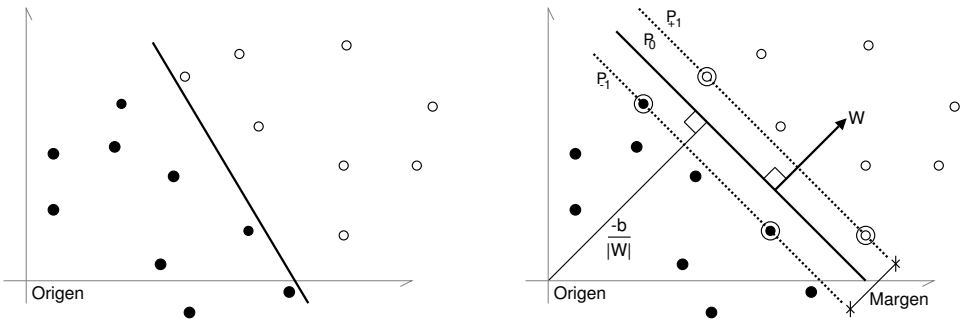


Figura 3.1: Hiperplanos que separan correctamente los datos. El OSH de la derecha tiene un mayor margen de separación entre clases, por lo tanto se espera una mejor generalización.

$$g(\vec{x}) = (\vec{w} \cdot \vec{x}) + b = 0 \tag{3.3}$$

donde se desea maximizar el margen (Figura 3.1). Se proponen dos planos paralelos a (3.3) que contienen los puntos más cercanos al OSH, definiendo como $1/\|\vec{w}\|$ la distancia entre el punto más cercano al OSH se obtienen las ecuaciones de dichos planos[3] :

$$P_{+1} \quad : \quad (\vec{w} \cdot \vec{x_i}) + b = +1 \tag{3.4}$$

$$P_{-1} \quad : \quad (\vec{w} \cdot \vec{x_i}) + b = -1 \tag{3.5}$$

Siendo el *margen* la distancia $\perp$ entre (3.4) y (3.5):

$$\begin{aligned} \left[\left(\vec{w} \cdot \vec{x_{+1}}\right) + b\right] - \left[\left(\vec{w} \cdot \vec{x_{-1}}\right) + b\right] &= (+1) - (-1) \\ \vec{w} \cdot \left(\vec{x_{+1}} - \vec{x_{-1}}\right) &= 2 \\ \frac{\vec{w}}{\|\vec{w}\|} \cdot \left(\vec{x_{+1}} - \vec{x_{-1}}\right) &= \frac{2}{\|\vec{w}\|} \end{aligned} \tag{3.6}$$

y la distancia del OSH al origen:

$$\begin{aligned} (\vec{w} \cdot \vec{x_i}) + b &= 0 \\ \vec{x_i} &= 0 \\ \frac{(\vec{w} \cdot \vec{x_i}) + b}{\|\vec{w}\|} &= \frac{b}{\|\vec{w}\|} \end{aligned} \tag{3.7}$$

donde (3.4), (3.5), (3.6) y (3.7) se muestran en la Figura 3.1.

Debe notarse que entre (3.4) y (3.5) no existen datos de entrenamiento, todos los datos deben

de la forma $(\vec{w} \cdot \vec{x_i}) + b = 0$, siendo $\vec{w} = [a_n, a_{n-1}, ..., a_1]$un vector perpendicular al hiperplano es $\vec{x_i} = [x_n, x, ..., x_1]$, y $b = a_0$

[3]Recordando la distancia perpendicular que hay de cualquier punto a un plano: $\frac{(\vec{w} \cdot \vec{x_i}) + b}{||w||}$

cumplir:

$$(\vec{w} \cdot \vec{x_i}) + b \geq +1 \quad \text{para} \quad y_i = +1 \tag{3.8}$$

$$(\vec{w} \cdot \vec{x_i}) + b \leq -1 \quad \text{para} \quad y_i = -1 \tag{3.9}$$

luego la función decisión $f_{w,b}(\vec{x_i}) = y_i$, puede definirse como el signo que resulta de evaluar un dato en la ecuación del OSH (3.3) :

$$f_{w,b}\left(\vec{x_i}\right) = \text{sign}\left[g\left(\vec{x_i}\right)\right] = \text{sign}\left[(\vec{w} \cdot \vec{x_i}) + b\right] \tag{3.10}$$

combinando (3.8) y (3.9) se obtiene:

$$y_i\left(\vec{w} \cdot \vec{x_i} + b\right) \geq 1 \tag{3.11}$$

Si existe un hiperplano que satisfaga (3.11), se dice que los datos son *linealmente separables*.

Para encontrar el OSH se debe maximizar el margen (3.6), teniendo en cuenta la restricción (3.11), equivalente a resolver el siguiente problema:

$$\min_{w} \frac{1}{2}\left(\vec{w} \cdot \vec{w}\right) \tag{3.12}$$

$$\text{Sujeto a} \quad y_i\left(\vec{w} \cdot \vec{x_i} + b\right) \geq 1, \ \forall i \tag{3.13}$$

La función (3.12) es llamada función objetivo, y junto con (3.13) es llamado problema de optimización cuadrático con restricciones. Los problemas de este tipo son tratados introduciendo el *método de multiplicadores de Lagrange*.

3.3.3. Solución al problema de optimización con restricciones

Para esta clase de problemas con restricciones, se introducen los *multiplicadores de Lagrange* $\alpha_i \geq 0$ (uno por cada restricción). Para las restricciones de la forma $R_i \geq 0$, cada restricción es multiplicada por α_i (un multiplicador de Lagrange positivo) y se restan de la función objetivo, para así formar la *función de Lagrange* [7]:

$$L\left(\vec{w}, b, \vec{\alpha}\right) = \frac{1}{2}\left\|\vec{w}\right\|^2 - \sum_{i=1}^{n} \alpha_i\left[y_i\left(\vec{w} \cdot \vec{x_i} + b\right) - 1\right] \tag{3.14}$$

La función de Lagrange (3.14) debe ser minimizada con respecto a las variables primarias w y b, y maximizada sobre los α_i (en otras palabras encontrar el punto de silla) [21]. Para el caso de $y_i\left(\vec{w} \cdot \vec{x_i} + b\right) - 1 > 0$, el correspondiente α_i debe ser cero, debido a que este es el valor de α_i que maximiza (3.14), los $\alpha_i \neq 0$ son para el caso en que $y_i\left(\vec{w} \cdot \vec{x_i} + b\right) - 1 = 0$ (estos son los patrones de entrenamiento que quedan sobre los planos paralelos (3.4) y (3.5) al OSH). Este último enunciado

son las condiciones de *Karush-Kuhn-Tucker* (condiciones complementarias de optimización) [15]:

$$\alpha_i \left[y_i \left(\vec{w} \cdot \vec{x_i} + b \right) - 1 \right] = 0, \ \forall i \tag{3.15}$$

$$\frac{\partial}{\partial w} L \left(\vec{w}, b, \vec{\alpha} \right) = 0 \qquad \Rightarrow \qquad \vec{w} = \sum_{i=1}^{n} \alpha_i y_i \vec{x_i} \tag{3.16}$$

$$\frac{\partial}{\partial b} L \left(\vec{w}, b, \vec{\alpha} \right) = 0 \qquad \Rightarrow \qquad \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3.17}$$

La solución de $\vec{w}$ (3.16) queda en función de un subconjunto de patrones de entrenamiento, aquellos cuyo multiplicador de Lagrange es diferente de cero, es decir el soporte de $\vec{w}$ está en los patrones de entrenamiento más cercanos al OSH. De aquí el nombre de *Máquinas de Soporte Vectorial.*

Reemplazando (3.16) y (3.17) en (3.14), se eliminan las variables primarias $\vec{w}$ y b llegando así al problema de *optimización dual de Wolfe*, el cual es el problema que se resuelve en la práctica:

$$\max_{\alpha} \left[\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j \left(\vec{x_i} \cdot \vec{x_j} \right) \right] \tag{3.18}$$

$$\text{sujeto a} \quad \alpha_i \geq 0, \ \forall i \text{ y } \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3.19}$$

Este problema puede ser resuelto con métodos de programación cuadrática estándar [7]. Una vez obtenido el vector $\vec{\alpha}$,se pueden obtener los parámetros $\vec{w}$ y b con (3.16) y (3.15) respectivamente:

$$\begin{aligned} \vec{w} &= \sum_{i=1}^{n} \alpha_i y_i \vec{x_i} \\ b &= y_i - \vec{w} \cdot \vec{x_i} \end{aligned}$$

Ahora la ecuación del OSH y la función de decisión pueden ser escritas como:

$$g(\vec{x}) = \sum_{i=1}^{n} \left[\alpha_i y_i \left(\vec{x_i} \cdot \vec{x} \right) \right] + b \tag{3.20}$$

$$f(\vec{x}) = \text{sign} \left(\sum_{i=1}^{n} \left[\alpha_i y_i \left(\vec{x_i} \cdot \vec{x} \right) \right] + b \right) \tag{3.21}$$

Se trabajó con teoría de Lagrange por dos razones fundamentales [7]:

- Las restricciones (3.13) fueron remplazadas y quedaron en términos de α_i (3.19), lo cual es mucho más fácil de manejar.

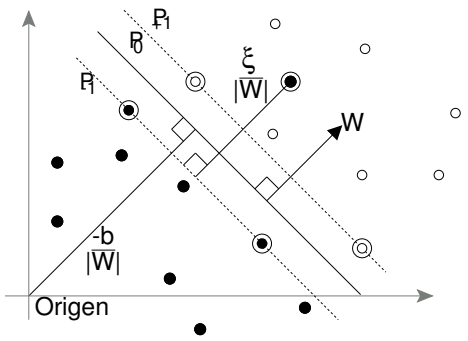


Figura 3.2: Hiperplano lineal clasificador para el caso no separable

- En la reformulación del problema, los datos de entrenamiento $\vec{x_i}$ solo aparecen en forma de productos punto entre ellos mismos (ecuaciones (3.18), (3.20) y (3.21)), lo cual es conveniente como se verá en la sección 3.3.5.

3.3.4. Caso linealmente no separable

Hasta el momento se ha descrito la manera en la cual funcionan las SVMs, pero su implementación puede crear clasificadores erróneos, debido a que en la práctica no necesariamente existe un hiperplano separador, y si existe, no siempre es la mejor solución para el problema de clasificación. Esto es, si existen datos erróneos, ruido o alto solapamiento de clases en los datos de entrenamiento, puede afectar el hiperplano clasificador óptimo. Por esta razón se cambia un poco la perspectiva y se busca el mejor hiperplano clasificador que pueda tolerar el ruido en los datos de entrenamiento. Una solución que seria obvia es encontrar el hiperplano que conduzca al menor número de errores de entrenamiento, pero desafortunadamente, esto se convierte en un problema combinacional el cual es difícil de aproximar.

Cortes y Vapnik [10], toman un acercamiento diferente para las SVMs, basándose en [4], para permitir la posibilidad de ejemplos que violen la restricción (3.13), ellos introducen las variables slack (de relajación), ver Figura 3.2.

$$\xi_i \quad \geq \quad 0, \; \forall i \tag{3.22}$$

para formular una nueva restricción:

$$y_i \left(\vec{w} \cdot \vec{x_i} + b\right) \geq 1 - \xi_i, \; \forall i \tag{3.23}$$

luego el clasificador que generaliza bien es hallado controlando tanto su capacidad de clasificación (con $\|\vec{w}\|$), como el limite superior del numero de errores de entrenamiento ($\sum_{i=1}^{n} \xi_i$). La forma de

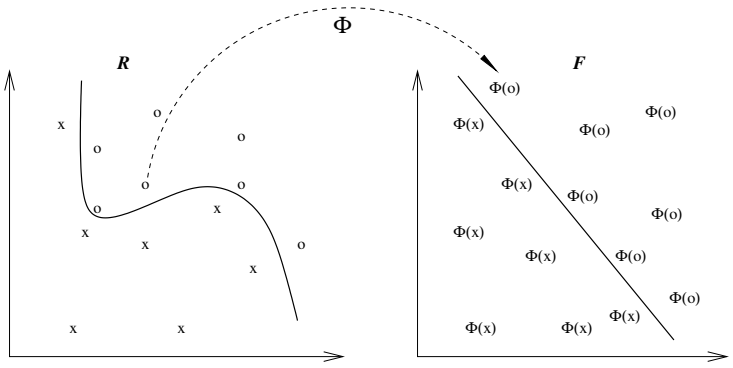


Figura 3.3: Transformación del espacio de entrada al espacio característico.

obtener el hiperplano clasificador óptimo con margen débil es minimizando la función:

$$\min_{w} \left[\frac{1}{2} (\vec{w} \cdot \vec{w}) + C \sum_{i=1}^{n} \xi_i \right] \tag{3.24}$$

$$\text{Sujeto a} \quad y_i (\vec{w} \cdot \vec{x_i} + b) \geq 1 - \xi_i, \ \forall i \tag{3.25}$$

El parámetro C es elegido a priori por el usuario de tal manera que un valor grande es una alta penalización a lo errores. Si se usan los multiplicadores de Lagrange el problema se transforma en:

$$\max_{\alpha} \left[\sum_{i=1}^{n} \alpha_i - \frac{1}{2} \sum_{i,j=1}^{n} \alpha_i \alpha_j y_i y_j (\vec{x_i} \cdot \vec{x_j}) \right] \tag{3.26}$$

$$\text{sujeto a} \quad 0 \leq \alpha_i \leq C, \ \forall i \text{ y } \sum_{i=1}^{n} \alpha_i y_i = 0 \tag{3.27}$$

Cuya solución da como resultado (3.16), luego el hiperplano separador solución puede ser escrito como (3.20), y la función de decisión (3.21).

3.3.5. SVM no lineales

El principio de las SVM no lineales consiste en mapear el espacio de entrada a un espacio de representación de dimensión alta a través de una función no lineal elegida a priori [6], ver Figura 3.3, donde por medio de una función (Φ) se trazan los datos de entrada ($\vec{x_i} \in \mathbb{R}^{\mathbb{N}}$), a algún espacio de mayor dimensión y con producto punto, este espacio es llamado espacio característico (F).

$$\Phi : \quad \mathbb{R}^{\mathbb{N}} \longrightarrow F \tag{3.28}$$

Así la función (3.20) que depende del producto punto de los vectores en el espacio de entrada, pasa a una función que depende del producto punto de los vectores en el espacio característico:

$$g(\vec{x}) = \sum_{i=1}^{n} \left[\alpha_i y_i \left(\Phi\left(\vec{x_i}\right) \cdot \Phi\left(\vec{x}\right)\right)\right] + b \tag{3.29}$$

entonces se define una función que sea el producto punto de los vectores en el espacio característico:

$$k\left(\vec{x_i},\vec{x}\right) = \Phi\left(\vec{x_i}\right) \cdot \Phi\left(\vec{x}\right) \tag{3.30}$$

debido a que (F) es de alta dimensión, el lado derecho de la ecuación (3.30) es costosa en términos computacionales, sin embargo existe una función kernel (k), que puede evaluarse eficazmente y demostrar que corresponde un trazado de Φ en un espacio que abarca todos los productos punto. Por ejemplo el kernel polinomial:

$$k\left(\vec{x},\vec{y}\right) = \left(\left(\vec{x}\right) \cdot \left(\vec{y}\right)\right)^d \quad \text{con } \left(\vec{x},\vec{y}\right) \in \mathbb{R}^2 \quad \text{y} \quad d = 2 \tag{3.31}$$

se tiene que:

$$\begin{aligned} k\left(\vec{x},\vec{y}\right) &= \left[\begin{pmatrix} x_1 \\ x_2 \end{pmatrix} \cdot \begin{pmatrix} y_1 \\ y_2 \end{pmatrix}\right]^2 \\ k\left(\vec{x},\vec{y}\right) &= \left[x_1 y_1 + x_2 y_2\right]^2 \\ k\left(\vec{x},\vec{y}\right) &= x_1^2 y_1^2 + 2 x_1 y_1 x_2 y_2 + x_2^2 y_2^2 \end{aligned} \tag{3.32}$$

esta ecuación se puede escribir de la forma:

$$k\left(\vec{x},\vec{y}\right) = \left[\begin{pmatrix} x_1^2 \\ \sqrt{2} x_1 x_2 \\ x_2^2 \end{pmatrix} \cdot \begin{pmatrix} y_1^2 \\ \sqrt{2} y_1 y_2 \\ y_2^2 \end{pmatrix}\right] = \Phi\left(\vec{x}\right) \cdot \Phi\left(\vec{y}\right) \tag{3.33}$$

Se observa que la función kernel nos ahorra buscar explícitamente la función Φ, y nos lleva directamente al resultado del producto punto, que es lo que realmente interesa. Generalizando, se puede probar que por cada función kernel que dé una matriz definida positiva, puede construir una función Φ la cual cumpla (3.30). Los kernel más utilizados son:

- Polinomial:

$$k\left(\vec{x},\vec{y}\right) = \left(\left(\vec{x}\right) \cdot \left(\vec{y}\right) + c\right)^d \text{ para } c > 0 \tag{3.34}$$

- Función de base radial (RBF):

$$k\left(\vec{x},\vec{y}\right) = e^{\left(-\frac{|\vec{x}-\vec{y}|^2}{2\sigma^2}\right)} \tag{3.35}$$

- Sigmoide:

$$k\left(\vec{x},\vec{y}\right) = \tanh\left(\kappa\left(\vec{x}\cdot\vec{y}\right)+\Theta\right) \tag{3.36}$$

Ahora se pueden reescribir (3.26) sujeta a (3.27):

$$\begin{gathered}\max_{\alpha}\left[\sum_{i=1}^{n}\alpha_i - \frac{1}{2}\sum_{i,j=1}^{n}\alpha_i\alpha_j y_i y_j k\left(\vec{x_i},\vec{x_j}\right)\right] \\ \text{sujeto a} \quad 0 \leq \alpha_i \leq C,\ \forall i \text{ y } \sum_{i=1}^{n}\alpha_i y_i = 0\end{gathered} \tag{3.37}$$

y las ecuaciones del OSH (3.20) y función decisión (3.21) también cambiarán a:

$$g(\vec{x}) = \sum_{i=1}^{n}\left[\alpha_i y_i k\left(\vec{x_i},\vec{x}\right)\right] + b \tag{3.38}$$

$$f(\vec{x}) = \text{sign}\left(\sum_{i=1}^{n}\left[\alpha_i y_i k\left(\vec{x_i},\vec{x}\right)\right] + b\right) \tag{3.39}$$

3.3.6. Multiclasificación con SVM

Hasta el momento solo se ha tratado el problema de la biclasificación (en los cuales las clases solo pueden tomar valores: ± 1), pero en la vida real mucho problemas son de más de dos clases ($y_i \in \{1,\ldots,l\},\ l > 2$). Para resolver el problema de multiclasificación con máquinas de vectores de soporte se admiten dos tipos de arquitectura.

3.3.6.1. Máquinas multiclasificadoras SV

Esta máquina construye una función clasificadora global directamente considerando todas las posibles clases a la vez. Se puede intentar resolver el problema, modificando la función objetivo de la SVM de tal manera que permita calcular una única maquina capaz de trabajar con todas las clases a la vez. Así pues por [25, 5], podemos usar la siguiente función objetivo

$$\min_{w_r,b_r,\xi^r}\left[\frac{1}{2}\sum_{r=1}^{l}\|\vec{w}\|^2 + \frac{C}{m}\sum_{i=1}^{m}\sum_{r\neq y_i}\xi_i^r\right] \tag{3.40}$$

$$\begin{gathered}\text{Sujeto a} \quad \left(\vec{w_{y_i}}\cdot\vec{x_i}+b_{y_i}\right) \geq \vec{w_r}\cdot x_i + b_r \vec{+} 2 - \xi_i^r, \quad \xi_i^r \geq 0, \\ m \in \{1,\ldots,l\}\setminus y_i, \quad \text{y} \quad y_i \in \{1,\ldots,l\}\end{gathered} \tag{3.41}$$

La principal desventaja de esta generalización se encuentra en el problema de optimización cuadrático, donde se manejan $n\left(l-1\right)$ variables sujetas a $2n\left(l-1\right)+2l$ restricciones de desigualdad. Esta

desventaja es realmente visible en experimentos empíricos, donde se muestra que este tipo de solución es lenta, además no existe ninguna inclusión de técnicas que mejoren la robustez del sistema, ni ningún estudio teórico sobre la cota de error [3].

3.3.6.2. Máquinas biclasificadoras SV generalizadas

Este tipo de máquina construye una función clasificadora global a partir de un conjunto de funciones biclasificadoras. Existen técnicas de descomposición y reconstrucción que permiten a las SVM biclasificadoras manejar problemas de multiclasificación con mayor simplicidad y/o menor tiempo de respuesta que una SVM generalizada a multiclasificación.

3.3.6.2.1. Arquitecturas de descomposición estándares En el esquema de descomposición estándar se construyen m máquinas biclasificadoras en paralelo, que son entrenadas sobre modificaciones del conjunto de aprendizaje, creándose una matriz de descomposición. Los elementos de unas clases son asignados a salidas positivas, los de otras a salidas negativas, y si es el caso, los restantes no son tenidos en consideración en aquel clasificador en particular.

3.3.6.2.1.1. Uno contra el resto Conocido como 1-v-r (del ingles *one-versus-rest*), este esquema se basa en la idea de que si existe un grupo de n datos de entrenamiento donde existen l clases ($l > 2$), se pueden tener un grupo de m clasificadores binarios (donde $m = l$), cada uno entrenado para separar una clase del resto de clases existentes ($l - 1$). La descomposición se realiza de la siguiente manera: existe un grupo de datos (n_j) que pertenecen a la j-ésima clase ($j \in \{1, \ldots, l\}$), a los cuales se les dará una etiqueta positiva ($t_j = +1$) y al resto de datos ($n_r = n - n_j$) se les dará una etiqueta negativa ($t_r = -1$) para el entrenamiento de la i-ésima SVM. Así se crea una matriz de descomposición (D_{1-v-r}) de m filas y l columnas:

$$D_{i,j} = \begin{cases} +1 & \text{si } n_h \in n_j \\ -1 & \text{si } n_h \in n_r \end{cases} \tag{3.42}$$

Por ejemplo, una máquina de clasificación multiclase (1-v-r) con $l = 5$, se obtiene $m = 5$. Entonces la correspondiente matriz de descomposición es la siguiente:

$$D_{1-v-r} = \begin{pmatrix} +1 & -1 & -1 & -1 & -1 \\ -1 & +1 & -1 & -1 & -1 \\ -1 & -1 & +1 & -1 & -1 \\ -1 & -1 & -1 & +1 & -1 \\ -1 & -1 & -1 & -1 & +1 \end{pmatrix} \tag{3.43}$$

En esta arquitectura propuesta por [23, 10], el tiempo de entrenamiento es proporcional al número de clases, y debido a que el entrenamiento de cada biclasificador es con el conjunto de datos de entrenamiento completo su costo computacional es alto.

3.3.6.2.1.2. Uno contra uno Conocido como 1-v-1 (del ingles *one-versus-one*), se realiza implementando $m = \frac{l(l-1)}{2}$ clasificadores binarios. Luego, el entrenamiento de la i-ésima SVM se realiza con solo 2 de las l ($l > 2$) clases existentes en el grupo de n datos de entrenamiento, otorgándole etiqueta positiva ($t_j = +1$) a los datos (n_j) que pertenecen al subgrupo de datos de la clase j ($j \in \{1, \ldots, l\}$), y etiqueta negativa ($t_p = -1$) a los datos (n_p) que pertenecen al subgrupo de datos de la clase p ($p \in \{1, \ldots, l\}$ y $p \neq j$). Los demás datos ($n_r = n - n_j - n_p$) no se utilizan en el entrenamiento de la i-ésima SVM por lo tanto son etiquetados con cero ($t_r = 0$), creándose la matriz de descomposición (D_{1-v-1})

$$D_{i,j} = \begin{cases} +1 & \text{si } n_h \in n_j \\ -1 & \text{si } n_h \in n_p \\ 0 & \text{si } n_h \in n_r \end{cases} \tag{3.44}$$

Por ejemplo, para una máquina de clasificación multiclase (1-v-1) con $l = 5$, se obtiene $m = 10$. Entonces la correspondiente matriz de descomposición es la siguiente:

$$D_{1-v-1} = \begin{pmatrix} +1 & -1 & 0 & 0 & 0 \\ +1 & 0 & -1 & 0 & 0 \\ +1 & 0 & 0 & -1 & 0 \\ +1 & 0 & 0 & 0 & -1 \\ 0 & +1 & -1 & 0 & 0 \\ 0 & +1 & 0 & -1 & 0 \\ 0 & +1 & 0 & 0 & -1 \\ 0 & 0 & +1 & -1 & 0 \\ 0 & 0 & +1 & 0 & -1 \\ 0 & 0 & 0 & +1 & -1 \end{pmatrix} \tag{3.45}$$

3.3.6.2.2. Arquitectura de descomposición ECOC La técnica ECOC (del ingles *Error Correcting Output Codes*) [13], utiliza la codificación estándar para obtener robustez contra fallos en las máquinas biclasificadoras. Se denomina codificación estándar a cada una de las posibles particiones de todo el conjunto de clases $y_i \in \{1, \ldots, l\}$, en problemas de biclasificación, que asignan etiquetas positivas $t_p = +1$ a los patrones de entrenamiento n_j de un cierto subconjunto de clases Y_j, y etiquetas negativas $t_p = -1$ a los patrones de entrenamiento n_r representantes del resto de clases Y_r. La de descomposición generada por

$$D_{i,j} = \begin{cases} +1 & \text{si } n_h \in n_j \\ -1 & \text{si } n_h \in n_r \end{cases} \tag{3.46}$$

debe ser tan diferente como sea posible en términos de la distancia Hamming para añadir redundancia, en este caso $m = 2^{l-1} - 1$. Por ejemplo, para una máquina de clasificación multiclase (ECOC) con $l = 4$, se obtiene $m = 7$. Entonces la correspondiente matriz de descomposición es la siguiente:

$$D_{ECOC} = \begin{pmatrix} +1 & -1 & -1 & -1 \\ +1 & -1 & -1 & +1 \\ +1 & -1 & +1 & -1 \\ +1 & -1 & +1 & +1 \\ +1 & +1 & -1 & -1 \\ +1 & +1 & -1 & +1 \\ +1 & +1 & +1 & -1 \end{pmatrix} \tag{3.47}$$

3.3.6.2.3. Métodos de reconstrucción Cada máquina biclasificadora entrenada emite una respuesta en forma numérica $z^i = g_i(\vec{x})$ a una entrada $\vec{x}$. La información más importante en esta respuesta, en principio, se encuentra en el signo $s^i = f_i(\vec{x}) = \text{sign}\ (g_i(\vec{x}))$ que adopta la función de decisión. En la determinación de la respuesta final facilitada por el método de reconstrucción de la maquina de aprendizaje multiclase han de ser tomados en consideración los siguientes elementos:

- Las predicciones numéricas parciales de los nodos de dicotomía, $z^i = g_i(\vec{x})$.
- El signo de las predicciones numéricas, $s^i = f_i(\vec{x}) = \text{sign}\ (g_i(\vec{x}))$.
- Un elemento interprete de las predicciones numéricas y binarias, $\Theta\left(z^i, s^i\right)$, con el fin de asignar o no, una o varias clases como posible respuesta de clasificación a una entrada $\vec{x}$.
- Un elemento $\Psi\left(\Theta\left(z^1, s^1\right), \ldots, \Theta\left(z^m, s^m\right)\right)$ de combinación de las predicciones, que tenga o pueda tener en consideración las predicciones numéricas, sus signos y/o la clase o clases asignadas.

3.3.6.2.4. Esquemas de votación Son la forma de reconstrucción más habitual. Se tiene en consideración solo el signo de las predicciones de todos las máquinas biclasificadoras. Estos signos son interpretados en función de las clases implicadas en las máquinas biclasificadoras utilizado en el esquema de descomposición.

- i-ésimo 1-v-r máquina biclasificadora

$$\Theta\left(s^i\right) = \begin{cases} y_i & \text{si} \quad s^i = +1 \\ \emptyset & \text{si} \quad s^i = -1 \end{cases} \tag{3.48}$$

- i-ésimo 1-v-1 máquina biclasificador

$$\Theta\left(s^i\right) = \begin{cases} y_j & \text{si} \quad s^i = +1 \\ y_p & \text{si} \quad s^i = -1 \end{cases} \tag{3.49}$$

- i-ésimo ECOC máquina biclasificadora

$$\Theta\left(s^i\right) = \begin{cases} Y_j & \text{si} \quad s^i = +1 \\ Y_r & \text{si} \quad s^i = -1 \end{cases} \tag{3.50}$$

Tras la interpretación de las predicciones, el elemento de combinación Ψ realiza un recuento del número de clases votadas, acción de la que toma el nombre de esquema de reconstrucción, que posee diferentes variantes. Se define a continuación algunas de estas posibilidades para las arquitecturas de descomposición

- Votación por unanimidad: se determina como respuesta aquella única clase que haya obtenido todos los votos posibles en las predicciones.
- Votación por mayoría absoluta: se determina como respuesta final aquella única clase que haya obtenido más de a mitad de los votos posibles.
- Votación por mayoría simple: se determina como respuesta final aquella única clase que haya obtenido más votos que el resto de clases.

3.4. Caracterización del sistema en falla.

La metodología propuesta utiliza los valores eficaces de tensión por fases, línea y secuencia cero[4] durante la falla. Manejando en total 7 valores eficaces de tensión[5] y obteniendo dos descriptores de cada señal. Así una falla queda caracterizada con 14 descriptores.

[4] Teniendo las señales de tensión de las fases en el tiempo, se pueden calcular las tensiones de línea realizando las correspondientes restas de señales en el tiempo y la tensión de secuencia cero realizando la suma. Luego obtener para cada una de estas señales su valor eficaz.

[5] 3 tensiones de fase, 3 tensiones de línea y secuencia cero.

3.4.1. Detección de la falla

La propuesta presentada en este documento no contempla en ningún momento la detección de la falla, por ello esta implícitamente diseñado para sistemas de distribución que cuenten con equipos de detección de fallas. Para los intereses de este estudio se toma como criterio de detección de fallas la existencia o no de huecos de tensión en las fases medidas en la subestación.

Definición de hueco de tensión

Según el estándar (IEEE Std 1159, 95) y la norma colombiana (NTC 5000, 02), un hueco de tensión o caída de tensión (sag) es la reducción del valor eficaz de la tensión entre el 0,9 y 0,1 p.u. de su valor nominal, durante 8,33ms a 1 minuto, caracterizada a frecuencia industrial (60Hz en Colombia).

3.4.2. Caracterización de los huecos de tensión

Cuando ocurre una falla en el sistema de distribución, la tensión medida en la subestación presenta, generalmente, un hueco de tensión como el mostrado en la Figura 3.4. Por ello, caracterizando el hueco de tensión medido en la subestación es posible determinar la ubicación más probable de la falla.

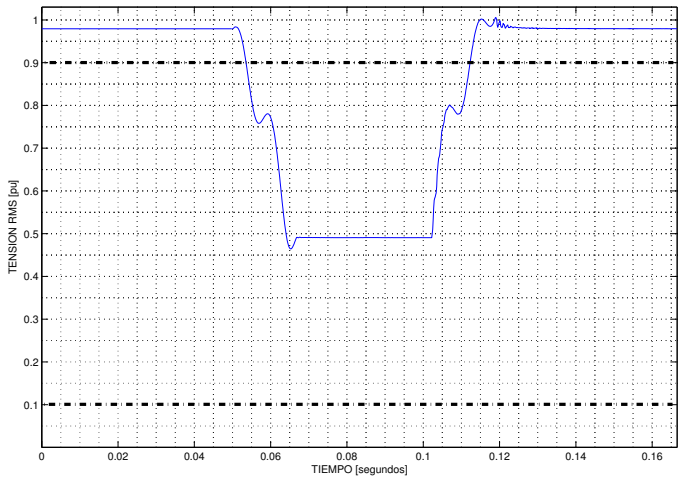


Figura 3.4: Hueco de tensión medido en un subestación en presencia de una falla.

3.4.2.1. Descriptores

La base o punto de partida para la caracterización del hueco es lo descrito en [19]. Los descriptores monofásicos utilizados en este estudio son (ver Figura 3.5):

- Profundidad del hueco de tensión (h).

- Pendiente de caída (m).

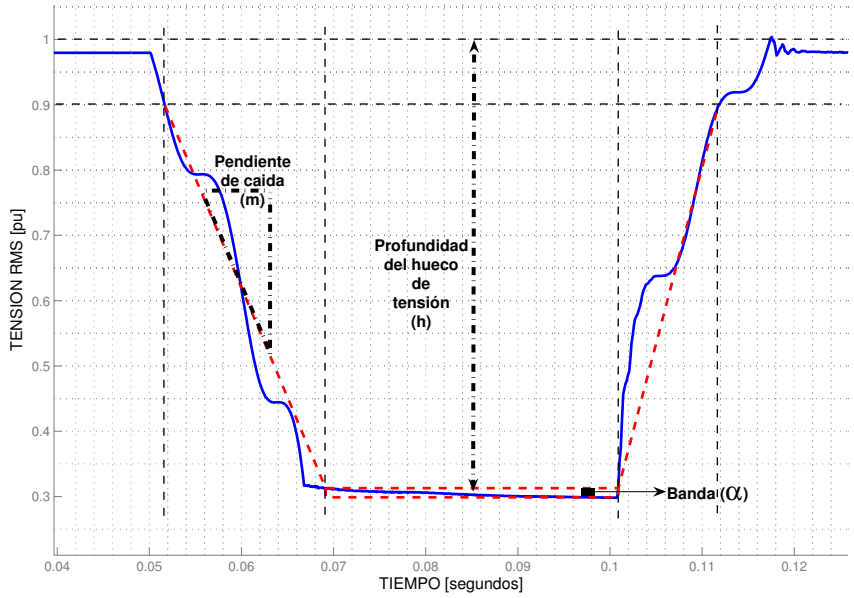


Figura 3.5: Descriptores base.

Debido a que dos fallas ubicadas en dos lugares diferentes dentro del sistema de distribución, vistas desde la subestación podrían producir huecos similares y localizarse erróneamente, se concluye que esta información no es suficiente para determinar la ubicación del evento, pero al observar el comportamiento de las fases no falladas, las tensiones de línea y de secuencia cero se obtiene una marcada diferencia entre estas fallas. En las Figuras 3.6 y 3.7 se muestra el comportamiento de las tensiones eficaces ante una falla monofásica en la fase A, ubicada en dos lugares[6] diferentes.

[6]Que distan 34,6 kilómetros uno del otro.

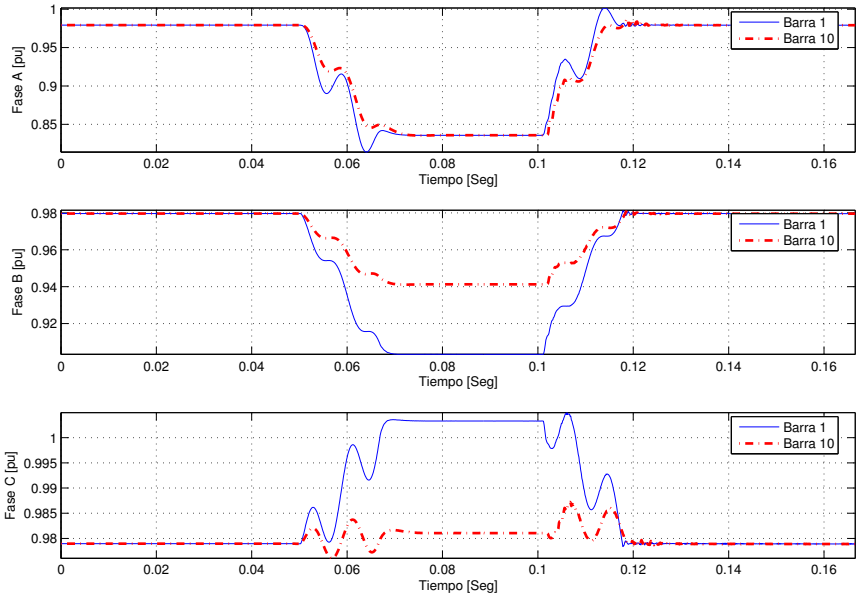


Figura 3.6: Comportamiento de las tensiones de fase ante una falla monofásica en la fase A en dos lugares diferentes.

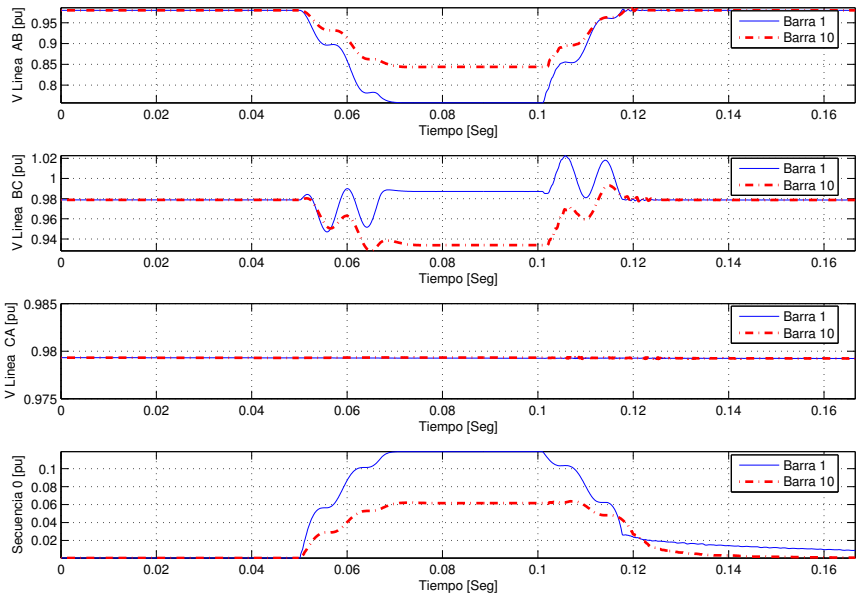


Figura 3.7: Comportamiento de las tensiones de línea y secuencia cero ante una falla monofásica en la fase A en dos lugares diferentes.

Por tal motivo se obtienen descriptores de las siguientes señales[7]:

- Fase o fases con hueco de tensión.
- Fase o fases sin hueco de tensión.
- Linea.
- Secuencia cero.

La definición de los descriptores empleados en este trabajo son:

- Magnitud (h): Está dada en p.u. y es la máxima caída o aumento[8] en cada una de las siete señales de tensión.
- Pendiente inicial (m): Es la rata de variación de tensión que presenta cada una de las siete señales al ocurrir la falla.

Para la obtención de estos descriptores es importante tener en cuenta los dos siguientes aspectos:

1. Ubicar el estado estable del hueco de tensión. Esto es necesario debido a que la parte transitoria del hueco de tensión varía con respecto al ángulo de incidencia de la falla como se muestra en la Figura 3.8, lo cual, por obvias razones, produciría diferentes descriptores para el mismo tipo de falla, en el mismo lugar y con la misma resistencia de falla imposibilitando una caracterización de la falla. Por ello los descriptores deben ser obtenidos con base en parte estable del hueco de tensión.

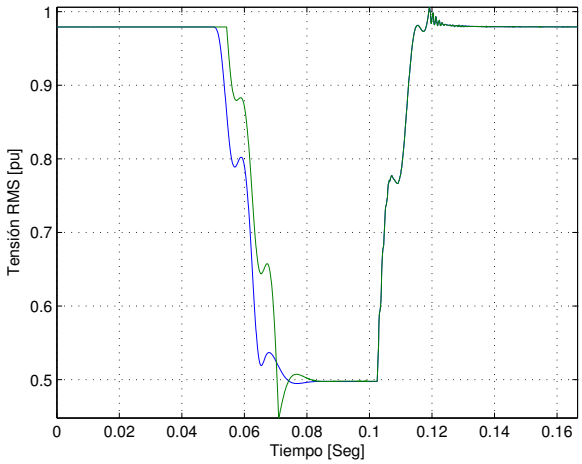


Figura 3.8: Falla monofásica en la misma fase a diferente ángulo.

[7] Todos son valores eficaces de tensión.

[8] Si las tensiones de fase o línea son menores a uno en p.u. representa una caída $(1-V_{min})$. Si son mayores a uno en p.u. representa un aumento $(1-V_{max})$. En la tensión de secuencia cero representa siempre un aumento $(-(0-V_{max}))$.

2. Tiempo de inicio del hueco[9] y duración de la parte estable del hueco[10]. Además de la razón del anterior numeral, son necesarios porque proveen los intervalos de tiempo para obtener los descriptores de las fases no falladas, las tensiones de línea y de secuencia cero. En caso de una falla en varias fases, se obtienen los descriptores propios de cada hueco y son utilizados los intervalos de tiempo del hueco más profundo[11] para observar las demás señales de tensión[12] y obtener sus descriptores.

Esto es ilustrado en la Figura 3.9 donde se observan los descriptores de las tensiones de fase. Nótese que la fase C está en falla y provee los intervalos de tiempo para obtener los descriptores de las fases A y B.

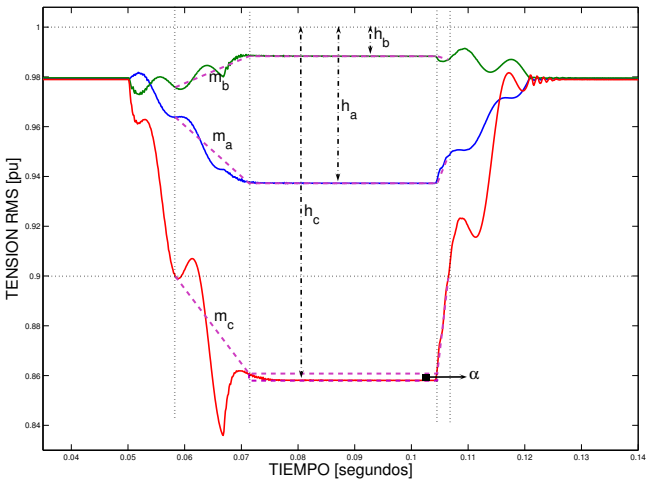


Figura 3.9: Descriptores basado en los intervalos de la fase fallada.

Aunque es obvia la pérdida de información en la parte transitoria de la falla, ésta no es relevante para la metodología de propuesta en este trabajo y por ello es aceptable su aproximación[13].

3.5. Metodología

3.5.1. Zonificación de la red.

Se utiliza para ubicar la zona más probable de localización de la falla. El criterio de zonificación queda a cargo del operador que implemente esta metodología, basado en el conocimiento de la

[9] Instante en el cual el valor eficaz de la tensión de fase es por primera vez menor a 0.9 p.u.

[10] Tiempo durante el cuál el valor eficaz de la tensión permanece en una banda de amplitud α sobre el valor mínimo de la parte estable del hueco de tensión . En este caso se selecciona $\alpha = 0{,}2 * h$.

[11] El de mayor magnitud (h)

[12] Fase no fallada (si aplica), tensiones de línea y de secuencia cero

[13] La pendiente inicial (m) es trazada con dos puntos.

topología, protecciones, longitudes de las redes, usuarios, estrato, etc., de su sistema. Se recomienda utilizar una zona por cada circuito ramal y se pueden agrupar varios circuitos en una zona o dividir un circuito en varias zonas.

3.5.2. Adquisición de datos de falla

Se refiere a la creación de una base de datos que contenga las señales[14] vistas en la subestación cuando ha ocurrido una falla en el sistema de distribución. Esta puede ser obtenida ya sea basada en el historial de fallas existente en la subestación o por la simulación del sistema fallado con algún software especializado.

3.5.3. Preprocesamiento de la señal

Obtener el valor eficaz de las tensiones de fase, linea y secuencia cero[15]. Seguidamente se obtienen los descriptores para cada falla y se guardan como vector. Estos a su vez son agrupados y etiqueteados dependiendo de la zona a la cual pertenece la falla.

Se realiza un escalamiento simple de los vectores (entre [-1,1] [16]). Este paso es muy importante para evitar dificultades numéricas durante los cálculos. Se puede utilizar el escalamiento que se desee y se recomienda escalar únicamente los descriptores de pendiente.

3.5.4. Entrenamiento de la SVM

Hace referencia a la técnica de inteligencia artificial. Como se vió anteriormente, las máquinas de soporte vectorial necesitan la definición a priori tanto del parámetro de penalización C, como de la función kernel y sus respectivos parámetros. Por ello el modelo queda definido de la siguiente manera:

- Función kernel: RBF (ver capitulo 4 y anexo B).
- Parámetro de la función kernel: Se determina mediante validación cruzada y búsqueda en malla (ver anexo B).
- Parámetro de penalización: Se determina mediante validación cruzada y búsqueda en malla (ver anexo B).

[14] 3 tensiones de fase, 3 tensiones de línea y tensión secuencia cero

[15] El presente estudio se realizó en base a simulaciones que no contenían componentes armónicas. Por lo tanto se aconseja realizar un filtrado a las señales de tensión, para trabajar solo con la componente fundamental y evitar posibles efectos negativos sobre la localización de fallas.

3.5.5. Prueba

Se realiza una prueba final con datos desconocidos[16] con el fin de obtener la precisión final del modelo. Estos son generalmente el 20 % de la base de datos y son extraídos justo antes de la etapa de entrenamiento de la SVM. En caso de no estar preprocesados estos datos, deben recibir exactamente el mismo tratamiento dado a los datos de entrenamiento en la etapa de preprocesamiento[17].

[16] Estos son datos que la SVM nunca vió en la etapa de entrenamiento.
[17] Esto también aplica en caso de ser un dato real de falla que se desea localizar.

Capítulo 4

Resultados

4.1. Introducción

La determinación del modelo que permita localizar la zona más probable de falla se realiza en este capítulo. Esto se logra comparando los resultados obtenidos al utilizar las funciones kernel de base radial (RBF), sigmoide y polinomial, sobre los mismos datos de entrenamiento y de prueba. El número de descriptores utilizados en cada clasificador puede cambiar, esto se debe a que en pruebas preliminares[1], se obtuvo el mínimo número de descriptores (mostrados en la sección 3.4) que se pueden usar en cada caso sin alterar el resultado. Los datos de entrenamiento son seleccionados en pruebas preliminares que permitieron ver datos innecesarios para que la SVM realice una buena clasificación.

4.2. Sistema de prueba

Se trabaja el sistema de distribución especificado en el anexo A.

4.3. Pruebas realizadas

La simulación del circuito se realiza en EMTP/ATP para los diferentes tipos de falla (monofásicas, bifásicas, bifásicas a tierra, trifásicas y trifásicas a tierra) en todas las barras mostradas en la Figura A.1 y con resistencias de falla de 0,05, 2, 5, 10, 15, 20, 25, 30, 35 y 50 Ω, obteniéndose una base de datos de 1 410 señales de tensión (fase A, B y C) medidas desde la cabecera del circuito.

Para la detección de falla, se toma como referencia el valor de 0,9 pu del valor eficaz de la señal de tensión de fase, es decir para valores menores (Vrms <0,9 pu) se considera que existe una falla. Entonces de los 1 410 datos, 1 391 corresponden a datos de fallas y 19 a datos de no fallas (el valor eficaz de la tensión no estuvo debajo del 0,9 pu).

[1] No incluidas en este libro.

El circuito se divide en partes, para ubicar la zona más probable de localización de falla. Básicamente se busca discriminar el ramal donde se encuentra la falla, bajo este criterio las zonas son:

- Zona 1: barras 1, 2, 3, 4, 5, 6, 7, 12.
- Zona 2: barras 8, 9, 10, 11.
- Zona 3: barras 13, 14, 15, 16, 17.
- Zona 4: barras 18, 19, 20, 21.

Se define el porcentaje de acierto como:

$$\frac{\#\text{ de datos clasificados correctamente}}{\#\text{ total de datos clasificados}} \cdot 100 \tag{4.1}$$

El desempeño de la SVM, se mide a partir de sus aciertos definidos como se muestra en la ecuación (4.1) sobre los datos de prueba. Cabe resaltar que estos datos de prueba no se utilizan en el entrenamiento y por tanto este porcentaje es una medida del comportamiento de la máquina clasificadora ante datos desconocidos.

4.4. Resultados

Se desarrolló una toolbox en MATLAB para manejar las SVMs (ver anexo D) para realizar las pruebas pertinentes. El entrenamiento de la máquina de soporte vectorial se realiza según lo descrito en el anexo B, para las funciones kernel polinomial, sigmoide y RBF.

Para resolver el problema de localización de fallas, se proponen dos esquemas. Las figuras de resultados muestran tanto los porcentajes de acierto de entrenamiento y de validación, como el porcentaje de datos de entrenamiento que son vectores de soporte.

4.4.1. Localizador independiente del tipo de falla - LIF

Para este esquema propuesto se trabaja con dos clasificadores independientes entre sí, cuya conexión se muestra en la Figura 4.1.

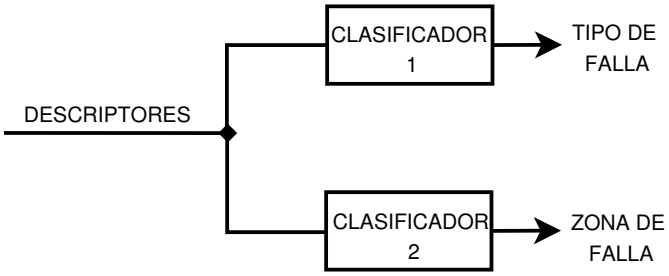


Figura 4.1: Esquema LIF.

4.4.1.1. Clasificador 1

Esta máquina de aprendizaje es la encargada de clasificar las fallas según su clase:

- Clase 1: Falla monofásica.
- Clase 2: Falla bifásica.
- Clase 3: Falla bifásica a tierra.
- Clase 4: Falla trifásica y falla trifásica a tierra[2].

Los descriptores utilizados por este clasificador son los huecos de las tensiones de fase y la tensión de la secuencia cero. Así se tiene un total de cuatro descriptores. La distribución de los datos de entrenamiento y prueba se muestran en la Tabla 4.1.

Tabla 4.1: Datos de entrenamiento y prueba del clasificador 1

	Datos de entrenamiento			Datos de prueba		
Clase	Barras	Resistencias Ω	Total	Barras	Resistencias Ω	Total
1	1-2; 6-14; 16-21	0,05; 10; 20; 35 y 50	164	3-5; 15	0,05; 10; 20; 35 y 50	267
				1-21	2; 5; 15; 25 y 30	
2	1-2; 6-12	0,05; 10; 20; 35 y 50	135	3-5	0,05; 10; 20; 35 y 50	225
				1-12	2; 5; 15; 25 y 30	
3	1-2; 6-12	0,05; 10; 20; 35 y 50	135	3-5	0,05; 10; 20; 35 y 50	225
				1-12	2; 5; 15; 25 y 30	
4	1-2; 6-12	0,05; 10; 20; 35 y 50	90	3-5	0,05; 10; 20; 35 y 50	150
				1-12	2; 5; 15; 25 y 30	
	TOTAL		524	TOTAL		867

En la fase de entrenamiento se obtienen los mejores porcentajes de aciertos de validación con los parámetros mostrados en la Tabla 4.2. Los resultados se muestran en la Figura 4.2.

Tabla 4.2: Parámetros escogidos en el entrenamiento para el clasificador 1

Polinómico			Sigmoide			RBF	
C	d	c	C	κ	Θ	C	σ
2^{16}	2^{2}	2^{1}	2^{20}	2^{8}	2^{1}	2^{100}	$2^{1,75}$

[2]Las fallas trifásicas y trifásicas a tierra son fallas simétricas y dado los descriptores planteados en esta tesis no existe una clara diferenciación de estas dos fallas, por ello se consideran como una sola clase.

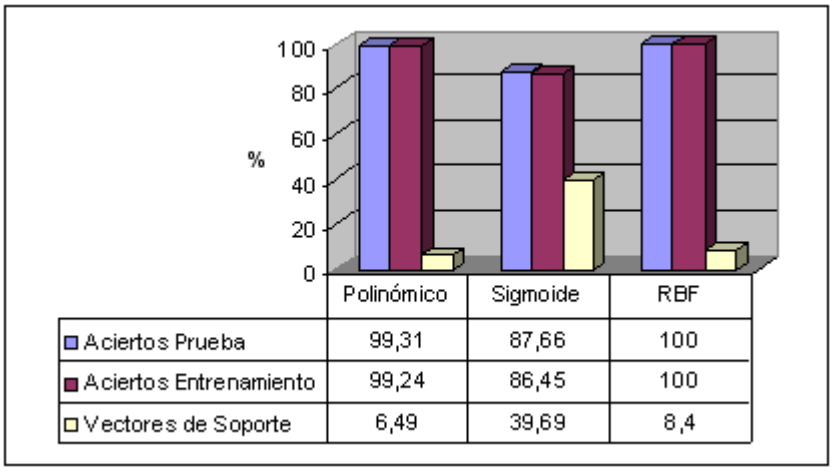


Figura 4.2: Resultados para el Clasificador 1.

4.4.1.2. Clasificador 2

Esta máquina de aprendizaje se encarga de clasificar las fallas según su zona como se indicó en la sección 4.3. Los descriptores utilizados por este clasificador son los 14 descriptores mencionados en la sección 3.4. La distribución de los datos de entrenamiento y prueba se muestran en la Tabla 4.3.

Tabla 4.3: Datos de entrenamiento y prueba del clasificador 2

	Datos de entrenamiento			Datos de prueba		
Clase	Barras	Resistencias Ω	Total	Barras	Resistencias Ω	Total
1	1-2; 6-7; 12	0,05; 10; 20; 35 y 50	275	3-5	0,05; 10; 20; 35 y 50	605
				1-7; 12	2; 5; 15; 25 y 30	
2	8-11	0,05; 10; 20; 35 y 50	220	8-11	0,05; 10; 20; 35 y 50	220
				8-11	2; 5; 15; 25 y 30	
3	13-14; 16-17	0,05; 10; 20; 35 y 50	15	15	0,05; 10; 20; 35 y 50	25
				13-17	2; 5; 15; 25 y 30	
4	18-21	0,05; 10; 20; 35 y 50	14	18-21	0,05; 10; 20; 35 y 50	17
				18-21	2; 5; 15; 25 y 30	
	TOTAL		524	TOTAL		867

En la fase de entrenamiento se obtienen los mejores porcentajes de aciertos de validación con los parámetros mostrados en la Tabla 4.4. Los resultados se muestran en la Figura 4.3.

Tabla 4.4: Parámetros escogidos en el entrenamiento para el clasificador 2

Polinómico			Sigmoide			RBF	
C	d	c	C	κ	Θ	C	σ
2^{-1}	2^2	2^1	2^7	$2^{-2,5}$	2^1	2^{22}	$2^{0,5}$

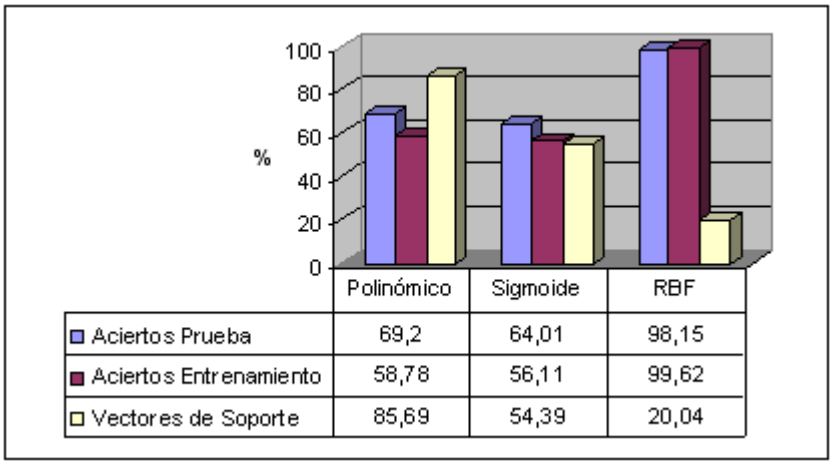


Figura 4.3: Resultados para el Clasificador 2.

4.4.2. Localizador dependiente del tipo de falla - LDF

En esta configuración, se trabaja con dos etapas conectadas como se muestra en la Figura 4.4.

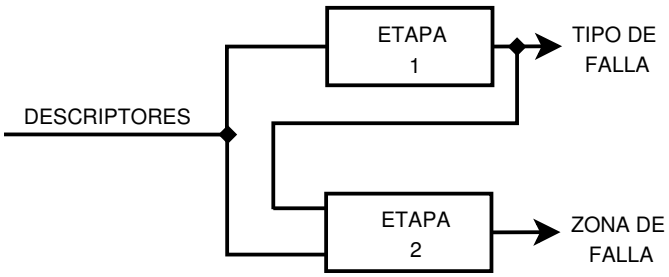


Figura 4.4: Esquema LDF.

4.4.2.1. Etapa 1

Etapa compuesta por una SVM, que tiene como función determinar el tipo de falla. Es el mismo Clasificador 1 del esquema LIF (ver sección 4.4.1.1).

4.4.2.2. Etapa 2

Formada por cuatro SVMs, las cuales no se conectan entre sí. Una de estas SVMs se activa dependiendo de la salida de la etapa 1. Cada SVM puede determinar la zona más probable de localización de la falla para una única clase de falla (clase 1, clase 2, clase 3 o clase 4).

4.4.2.2.1. SVM tipo 1 Localizador de fallas monofásicas. Los descriptores utilizados por este clasificador son los 14 descriptores mencionados en la sección 3.4. Los datos de entrenamiento y prueba se muestran en la Tabla 4.5.

Tabla 4.5: Datos de entrenamiento y prueba de la SVM tipo 1

	Datos de entrenamiento			Datos de prueba		
Clase	Barras	Resistencias Ω	Total	Barras	Resistencias Ω	Total
1	1-2; 6-7; 12	0,05; 10; 20; 35 y 50	75	3-5	0,05; 10; 20; 35 y 50	165
				1-7; 12	2; 5; 15; 25 y 30	
2	8-11	0,05; 10; 20; 35 y 50	60	8-11	0,05; 10; 20; 35 y 50	60
				8-11	2; 5; 15; 25 y 30	
3	13-14; 16-17	0,05; 10; 20; 35 y 50	15	15	0,05; 10; 20; 35 y 50	25
				13-17	2; 5; 15; 25 y 30	
4	18-21	0,05; 10; 20; 35 y 50	14	18-21	0,05; 10; 20; 35 y 50	17
				18-21	2; 5; 15; 25 y 30	
	TOTAL		164	TOTAL		267

En la fase de entrenamiento se obtienen los mejores porcentajes de aciertos de validación con los parámetros mostrados en la Tabla 4.6. Los resultados se muestran en la Figura 4.5.

Tabla 4.6: Parámetros escogidos en el entrenamiento para SVM tipo 1

Polinómico			Sigmoide			RBF	
C	d	c	C	κ	Θ	C	σ
2^8	$2^{1,5}$	2^1	2^{16}	$2^{-5,5}$	2^1	2^{100}	$2^{0,5}$

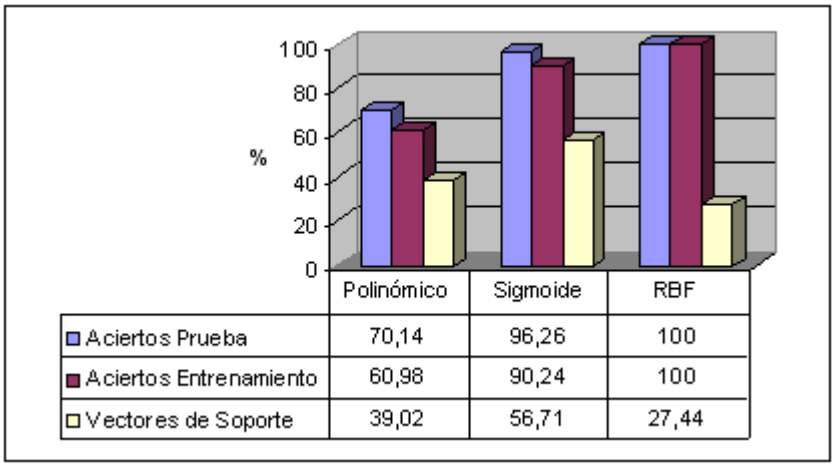


Figura 4.5: Resultados para SVM tipo 1.

4.4.2.2.2. SVM tipo 2 Localizador de fallas bifásicas. Los descriptores utilizados en esta SVM son las pendientes y los huecos de tensión tanto de fase como de línea, para un total de doce descriptores. Los datos de entrenamiento y prueba se muestran en la Tabla 4.7.

Tabla 4.7: Datos de entrenamiento y prueba de la SVM tipo 2

	Datos de entrenamiento			Datos de prueba		
Clase	Barras	Resistencias Ω	Total	Barras	Resistencias Ω	Total
1	1-2; 6-7; 12	0,05; 10; 20; 35 y 50	75	3-5	0,05; 10; 20; 35 y 50	165
				1-7; 12	2; 5; 15; 25 y 30	
2	8-11	0,05; 10; 20; 35 y 50	60	8-11	0,05; 10; 20; 35 y 50	60
				8-11	2; 5; 15; 25 y 30	
	TOTAL		135	TOTAL		225

En la fase de entrenamiento se obtienen los mejores porcentajes de aciertos de validación con los parámetros mostrados en la Tabla 4.8. Los resultados se muestran en la Figura 4.6.

Tabla 4.8: Parámetros escogidos en el entrenamiento para SVM tipo 2

Polinómico			Sigmoide			RBF	
C	d	c	C	κ	Θ	C	σ
2^8	2^2	2^1	2^{16}	2^{-8}	2^1	2^{50}	2^0

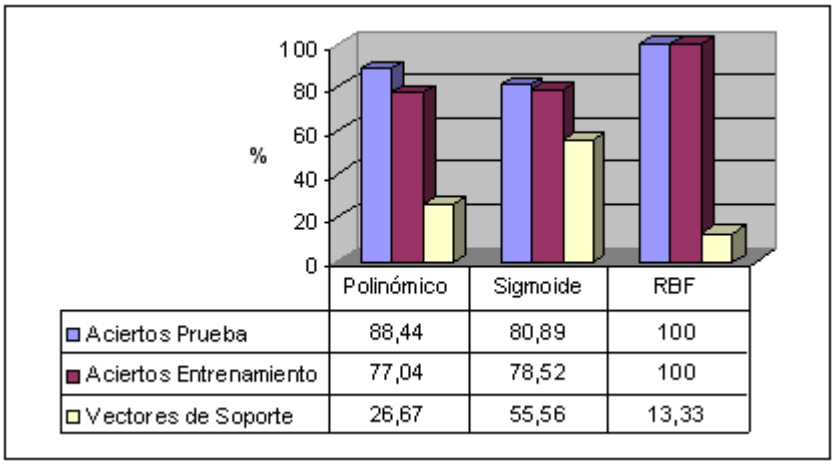


Figura 4.6: Resultados para SVM tipo 2

4.4.2.2.3. SVM tipo 3 Localizador de fallas bifásicas a tierra. Los descriptores utilizados por este clasificador son todos los 14 descriptores mencionados en la sección 3.4. Los datos de entrenamiento y prueba se muestran en la Tabla 4.9.

Tabla 4.9: Datos de entrenamiento y prueba de la SVM tipo 3

	Datos de entrenamiento			Datos de prueba		
Clase	Barras	Resistencias Ω	Total	Barras	Resistencias Ω	Total
1	1-2; 6-7; 12	0,05; 10; 20; 35 y 50	75	3-5	0,05; 10; 20; 35 y 50	165
				1-7; 12	2; 5; 15; 25 y 30	
2	8-11	0,05; 10; 20; 35 y 50	60	8-11	0,05; 10; 20; 35 y 50	60
				8-11	2; 5; 15; 25 y 30	
	TOTAL		135	TOTAL		225

En la fase de entrenamiento se obtienen los mejores porcentajes de aciertos de validación con los parámetros mostrados en la Tabla 4.10. Los resultados se muestran en la Figura 4.7.

Tabla 4.10: Parámetros escogidos en el entrenamiento para SVM tipo 3

Polinómico			Sigmoide			RBF	
C	d	c	C	κ	Θ	C	σ
2^6	$2^{1,5}$	2^1	2^{18}	$2^{-6,5}$	2^1	2^{50}	2^4

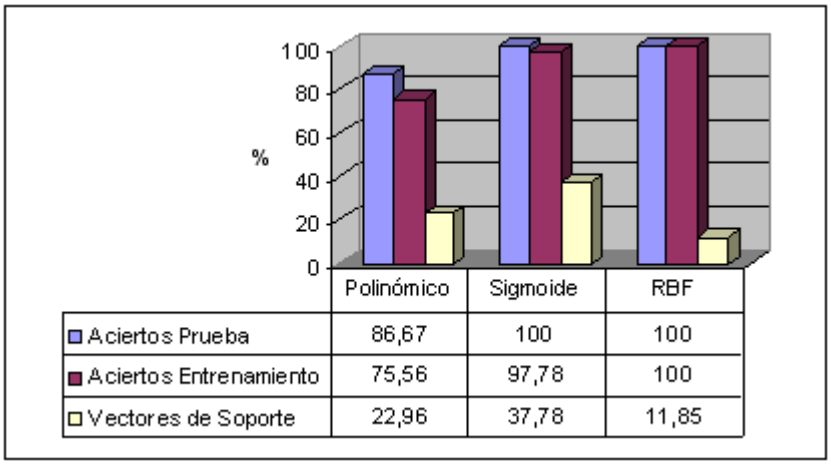


Figura 4.7: Resultados para SVM tipo 3

4.4.2.2.4. SVM tipo 4 Localizador de fallas trifásicas y trifásicas a tierra. Los descriptores utilizados en esta SVM son las pendientes y los huecos de tensión tanto de fase como de línea, para un total de doce descriptores. Los datos de entrenamiento y prueba se muestran en la Tabla 4.11.

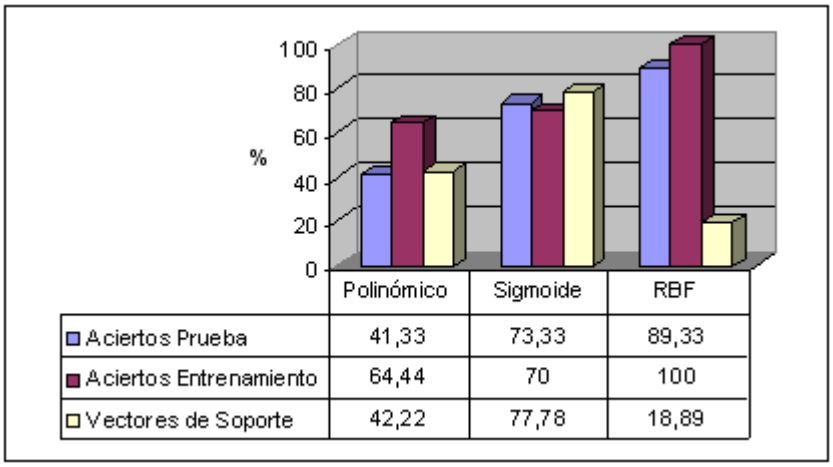
Tabla 4.11: Datos de entrenamiento y prueba de la SVM tipo 4

	Datos de entrenamiento			Datos de prueba		
Clase	Barras	Resistencias Ω	Total	Barras	Resistencias Ω	Total
1	1-2; 6-7; 12	0,05; 10; 20; 35 y 50	50	3-5	0,05; 10; 20; 35 y 50	110
				1-7; 12	2; 5; 15; 25 y 30	
2	8-11	0,05; 10; 20; 35 y 50	40	8-11	0,05; 10; 20; 35 y 50	40
				8-11	2; 5; 15; 25 y 30	
	TOTAL		90	TOTAL		150

En la fase de entrenamiento se obtienen los mejores porcentajes de aciertos de validación con los parámetros mostrados en la Tabla 4.12. Los resultados se muestran en la Figura 4.8.

Tabla 4.12: Parámetros escogidos en el entrenamiento para SVM tipo 4

Polinómico			Sigmoide			RBF	
C	d	c	C	κ	Θ	C	σ
2^{10}	2^1	2^1	2^{14}	2^{-7}	2^1	2^{100}	$2^{-1,75}$



	Polinómico	Sigmoide	RBF
Aciertos Prueba	41,33	73,33	89,33
Aciertos Entrenamiento	64,44	70	100
Vectores de Soporte	42,22	77,78	18,89

Figura 4.8: Resultados para SVM tipo 4

4.4.3. Modelo seleccionado

Se selecciona el esquema LDF y función kernel RBF (se discute con más detalle en el anexo C).

Capítulo 5

Conclusiones, aportes y trabajos futuros

5.1. Conclusiones

- Se logra demostrar que es posible una buena localización de fallas utilizando las SVMs, lo cual es el interés principal de este estudio .
- Debido a que las fallas monofásicas se presentan con más frecuencia que los otros tipos de falla, se propone una forma de localizar fallas con máquinas especializadas (LDF). Este método es más eficiente que el método que trabaja con todos los tipos de falla (LIF), pero se tiene la desventaja que la respuesta depende del tipo de falla (si hay error en el tipo de falla, no habrán buenos resultados en la localización de la falla).
- En un principio se entrenaron las SVMs sólo con los descriptores de las tensiones de fase, y no se lograron resultados superiores al 95 %. Ante el reto de mejorar estos resultados sin aumentar dificultad e inspirados en la transformación de Karrenbauer, se plantearon los descriptores de las tensiones de línea y de la secuencia cero logrando así los resultados mostrados en este documento.
- Cuando se obtiene un 100 % en la eficiencia de la SVM, se tiene una excelente clasificación con los datos de prueba, pero no significa que no se puedan tener clasificaciones erróneas. Para estar seguros de que la herramienta nunca se fuera a equivocar, sería necesario probar la máquina con todas las posibles fallas y valores de resistencias que se puedan presentar (lo cual es imposible), sin embargo vale la pena resaltar que los datos de prueba siempre fueron mayores que los datos de entrenamiento. Un resultado del 100 % proporciona alta confianza ante la clasificación.
- Las fallas trifásicas son las que presentan mayores dificultades para su localización. En LIF se presenta una localización errónea de 16 fallas (98,15 % de acierto) donde 12 de estas son trifásicas. De manera más clara se observa este comportamiento en LDF para fallas trifásicas, donde el porcentaje de acierto es del 89,33 %. Este fenómeno no es preocupante para esta

investigación debido a que en pocas ocasiones se presenta este tipo de falla en sistemas de distribución.

- Esta metodología a diferencia de algunos métodos algorítmicos, no ve comprometida su precisión con fallas de alta impedancia (alrededor de 40 Ω) [24].
- Para entrenar el localizador de fallas no se necesitan todos los datos de falla disponibles, con solo los datos de las barras ubicadas en los limites se puede caracterizar cada zona.
- Se logra una localización satisfactoria de las fallas, únicamente con descriptores de tensión lo cual es muy importante por dos razones. La primera es que en sistemas de distribución reales generalmente se tienen registros de tensiones y no de corrientes. La segunda es que la tensión (a diferencia de la corriente) varía poco para diferentes condiciones de carga, lo cual le da robustez a la metodología propuesta en este trabajo de grado.
- Para el desarrollo de este trabajo de grado la aplicación del conocimiento adquirido a través de la formación académica en la universidad jugó un papel importante, pero el rol más importante fue realizado por el espíritu investigativo nacido en este proceso, el cual no solo se convirtió en fuente de fortaleza para afrontar el reto de adquirir un nuevo conocimiento que en algunas ocasiones se mostró esquivo, sino que además nos guío para encontrar el camino y dar una solución practica que ofrece muy buenos resultados. Por ello este trabajo de grado significa la voluntad, entrega y dedicación de los autores para aplicar ingeniería en todo el sentido de la palabra a un problema real, presente no solo en Colombia sino en el mundo entero.

5.2. Aportes

- Se realizó una investigación y recopilación bibliográfica sobre el estado del arte de las SVMs, que puede ser utilizada como guia para entender fácilmente sus conceptos, con el fin de emprender nuevas investigaciones relacionadas con el tema.
- Se desarrolló una toolbox en MATLAB de fácil manejo (libre para la comunidad universitaria UIS) para la aplicación de la SVM a problemas de clasificación.
- Se demuestra que es posible localizar fallas en sistemas de distribución utilizando las *máquinas de soporte vectorial.*

5.3. Trabajos futuros

- El buen desempeño mostrado por la SVM para identificar el tipo de falla y localizar la zona más probable de ocurrencia de ésta, muestra un camino alentador para continuar adelante en

la investigación. Con el objeto de obtener la mayor información posible sobre la falla se plantea realizar un pre-procesamiento a la señal diferente al valor eficaz [1].

- Buscar alternativas que permitan hacer distinción entre la ocurrencia de una falla y un cambio de carga, para detectar aquellas fallas que podrían pasar desapercibidas y no confundir las sobrecargas con fallas[2]. Estudiando las condiciones normales o permisibles de funcionamiento del sistema y las condiciones de falla del mismo se puede estimar su estado de operación (condición normal o falla) utilizando SVMs.

- Hallar con un método determinístico el tipo de falla y aumentar el nivel de seguridad del clasificador LDF (cambiando el clasificador SVM para el tipo de falla por el método determinístico).

- Aunque en este trabajo se realizan pruebas con varios valores de resistencia de falla, se puede cambiar este rango de valores por los estadísticamente más frecuentes en sistemas de distribución.

- Adaptar las *máquinas de soporte vectorial* para tener una ubicación detallada de la falla (distancia y zona).

[1] Utilizado en este trabajo por sencillez y facilidad de implementación, pero no se puede desconocer la pérdida de información que implica el usarlo.

[2] En el presente estudio, debido a la lógica utilizada para la detección del evento, existen fallas que no son detectadas.

Bibliografía

[1] *http://www.kyb.tuebingen.mpg.de/bu/people/bs/svm.html.*

[2] *http://www.support-vector.net.*

[3] Cecilo Angulo. *Aprendizaje con máquinas núcleo en entornos de multiclasificación.* Tesis Doctoral, Universidad Politécnica de Catalunya, 2001.

[4] K. P. Bennett and O. L. Mangasarian. Robust linear programming discrimination of two linearly inseparable set. *Optimization Methods and Software*, 1:23–34, 1992.

[5] V. Blanz, V. Vapnik, and C. Burges. *Multiclass Discrimination with an Extended Support Vector Machine.* Talk given at AT&T Bell Labs, 1995.

[6] B. Boser, I. Guyon, and V. Vapnik. A training algorithm for optimal margin classifier. *Proc. 5^{th} ACM Workshop on Computational Learning Theory*, pages:144–152, 1992.

[7] Christopher Burges. A tutorial on support vector machines for pattern recognition. *Data Mining And Knowledge Discovery*, 2:121–167, 1998.

[8] Gilberto Carrillo Caicedo. *Fundamentos de protecciones.* Bucaramanga, 1990.

[9] Westinghouse Electric Corporation. *Electric Utility Engineering Reference Book: distribution systems, vol 3.* Pittsburgh, Pennsylvania, 1964.

[10] C. Cortes and V. Vapnik. Support vector networks. *Machine Learning*, 20:273–297, 1995.

[11] Novosel D., Hart D., Hu Y., and Myllymaki J. *System for locating faults and estimating fault resistence in distribution networks with tapped loads.* US Patent number 5,839,093, 1998.

[12] Ratan Das. *Determining the Locations of Faults in Distribution Systems.* Doctoral Thesis, University of Saskatchewan Saskatoon, Canada, 1998.

[13] T. G. Dietterich and G. Bakiri. Solving multiclass learning problems via error-correcting output codes. *Journal of Artificial Intelligence Research*, 2:263–286, 1995.

[14] Ricardo Henao, Fabian Ojeda, Mauricio Orozco, and Germán Castellanos. *Identificación de estados funcionales en bioseñales empleando SVM.* Universidad Nacional de Colombia, 2004.

[15] H. W. Kuhn and A. W. Tucker. Nonlinear programming. *Proc. 2nd Berkeley Symposium on Mathematical Statistics and Probabilistics*, pages:481–492, 1951. University of California Press.

[16] Chih-Jen Lin, Chih-Chung Chang, and Chih-Wei Hsu. *A Practical Guide to Support Vector Classification.* National Taiwan University, 2004.

[17] Chih-Jen Lin and Chih-Wei Hsu. *A comparison of methods for multi-class Support Vector Machines.* National Taiwan University, 2003.

[18] Chih-Jen Lin and S. Keerthi. Asymptotic behaviors of support vector machines with gaussian kernel. *Neural Computation*, 15:1667–1689, 2003.

[19] J. Mora and J. Colomer. *Voltage Sag Characterization and Classification for Diagnosis in Electric Power Quality Domain.* Researh Report, Department of Electronics, Computer Science and Automatic Control, Group eXiT, University of Girona, España, 2003.

[20] M. Saha. *Provoost F. Fault location in Médium Voltage Networks.* CIRED, Nice, Francia, 1999.

[21] Bernhard Schölkopf and Alex Smola. *Learning with Kernels Support Vector Machines, Regularization, Optimization and Beyond.* The MIT Press, Cambridge, 2002.

[22] T. Takagi, Y. Yamakoshi, R. Kondow, and T. Matsushima. Development of a new type fault locator using the one-terminal voltage and current data. *IEEE Transactions on Power Apparatus and Systems*, PAS-101, No 8:2892–2898, 1982.

[23] V. Vapnik. *The nature of statistical learning theory.* Springer Verlag, New York, 1995.

[24] Libardo Villamizar and Carlos Quiñones. *Implementacioón del método de Ratan Das para la localización de fallas en sistemas de distribución de energía eléctrica.* Tesis de grado, Escuela de Ingenierías Eléctrica, Electrónica y Telecomunicaciones, Universisdad Industrial de Santander, Colombia, 2005.

[25] J. Weston and C. Watkins. *Multi-Class Support Vector Machines.* In M. Verleysen, editor, Proceedings ESANN, Brussels, 1999.

Anexo A

Especificaciones del sistema de prueba.

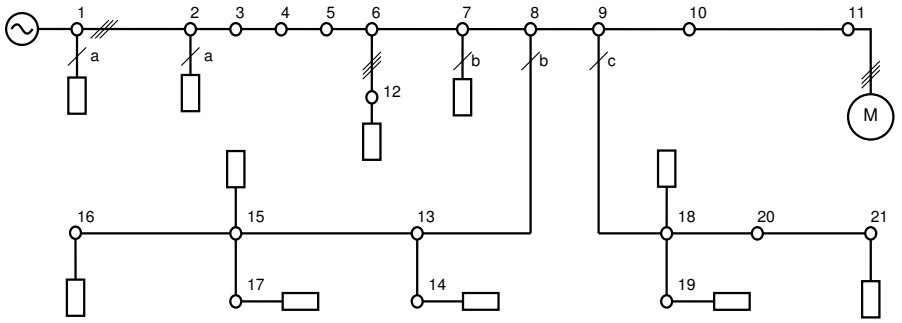


Figura A.1: Diagrama unifilar del modelo de sistema de distribución.

El sistema de distribución seleccionado para las pruebas de la metodología propuesta en el capitulo 3, es el modelo de sistema utilizado en [12] para la validación del método de localización de fallas de Ratan das. El diagrama unifilar de este sistema se muestra en la Figura A.1. En la Tabla A.1 se muestra los parámetros de la fuente del sistema. Los datos de las líneas se presentan en la Tabla A.4. La información de las cargas se especifica en la Tabla A.2 y el factor de potencia de cada tipo de carga se muestra en la Tabla A.3.

Tabla A.1: Parámetros del generador.

Tensión Base (kV)	Potencia Base (MVA)	Impedancia de Secuencia Positiva y Negativa (p.u.)	Impedancia de Secuencia Cero (p.u.)
25	1000	0,68283 + j2,98139	0,09496 + j1,39289

Tabla A.2: Datos de cargas del sistema.

Nodo	Fase	Carga (kVA)	Composición (%)		
			Calentador	Iluminación	Motor
1	A	15	99,8	0,1	0,1
2	A	15	99,8	0,1	0,1
7	B	15	99,8	0,1	0,1
11	A, B, C.	1000	0,1	0,1	99,8
12	A, B, C.	67,5	99,8	0,1	0,1
14	B	15	99,8	0,1	0,1
15	B	15	99,8	0,1	0,1
16	B	7,5	99,8	0,1	0,1
17	B	15	99,8	0,1	0,1
18	C	25	99,8	0,1	0,1
19	C	15	99,8	0,1	0,1
21	C	15	99,8	0,1	0,1

Tabla A.3: Factor de potencia.

Tipo de carga	Factor de potencia
Calentador	1
Iluminación	0,85 atraso
Motor	0,8 atraso

Tabla A.4: Parámetros de las lineas.

Sección entre los nodos	Longitud de la sección (km)	Impedancia Serie (Ohms / km)		Admitancia Paralelo (Ohms / km)	
		Secuencia Positiva y Negativa	Secuencia Cero	Secuencia Positiva y Negativa	Secuencia Cero
1 - 2	2,414	0,3480 + j0,5166	0,5254 + j1,704	j3,74 E-6	j2,49 E-6
2 - 6	16,092	0,3480 + j0,5166	0,5254 + j1,704	j3,74 E-6	j2,49 E-6
6 - 7	4,023	0,3480 + j0,5166	0,5254 + j1,704	j3,74 E-6	j2,49 E-6
7 - 8	5,150	0,5519 + j0,5390	0,7290 + j1,727	j3,59 E-6	j2,39 E-6
8 - 9	2,414	0,5519 + j0,5390	0,7290 + j1,727	j3,59 E-6	j2,39 E-6
9 - 10	4,506	0,5519 + j0,5390	0,7290 + j1,727	j3,59 E-6	j2,39 E-6
10 - 11	2,414	0,3480 + j0,5166	0,5254 + j1,704	j3,74 E-6	j2,49 E-6
11 - 12	2,414	0,3480 + j0,5166	0,5254 + j1,704	j3,74 E-6	j2,49 E-6
8 - 13	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
13 - 14	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
13 - 15	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
15 - 16	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
15 - 17	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
9 - 18	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
18 - 19	2,414	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
18 - 20	3,219	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6
20 - 21	3,219	7,3977 + j0,8998	7,3977 + j0,8998	j2,51 E-6	j2,51 E-6

Anexo B

Entrenamiento de la máquina de soporte vectorial

El objetivo del entrenamiento es encontrar el mejor valor de los hiperparámetros que definen la SVM para un tipo de kernel específico. A continuación se especifican elementos importantes de este proceso que influyen en la precisión de las SVMs, con el fin de localizar fallas en sistemas de distribución.

B.1. Método de descomposición

Para abordar el problema de multiclasificación se utiliza el método de descomposición uno contra uno (1-v-1), por sus excelentes resultados, robustez y menor exigencia computacional como se muestra en [17].

B.2. Función kernel

Dado los resultados obtenidos en este trabajo de grado (ver capítulo 4) se selecciona el kernel RBF. Además según [16] este kernel posee mayor simplicidad[1], menores complicaciones numéricas[2] y en [18] se muestra que puede comportarse como otras funciones kernel dependiendo de los parámetros.

B.3. Selección de parámetros

Los parámetros de la función kernel y el de penalización varían dependiendo del problema que se esté atacando, por ello no es posible encontrar un valor que se ajuste a la localización de fallas en general. Pero es posible determinar hiperparámetros que se ajusten a la localización de fallas en cada sistema de distribución al que se aplique esta metodología mediante el método de búsqueda en malla y validación cruzada. Estos métodos son empleados por sus grandes ventajas como fácil

[1] Solo necesita el ajuste de un parámetro kernel.

[2] Sus valores están entre 0 y 1, a diferencia del kernel polinómico que puede tender al infinito a medida que el grado del polinomio aumenta y el kernel sigmoide no es valido para algunos parámetros.

implementación, evita el sobre-entrenamiento (ver Figura B.1, obsérvese que el sobre-entrenamiento conlleva a una mala generalización de la SVM) y su alta competitividad con respecto a otros métodos de selección de hiperparámetros, según [16]. Durante este proceso, se obtienen unos errores de entrenamiento y de validación. Los parámetros seleccionados son los que presenten el menor error de validación.

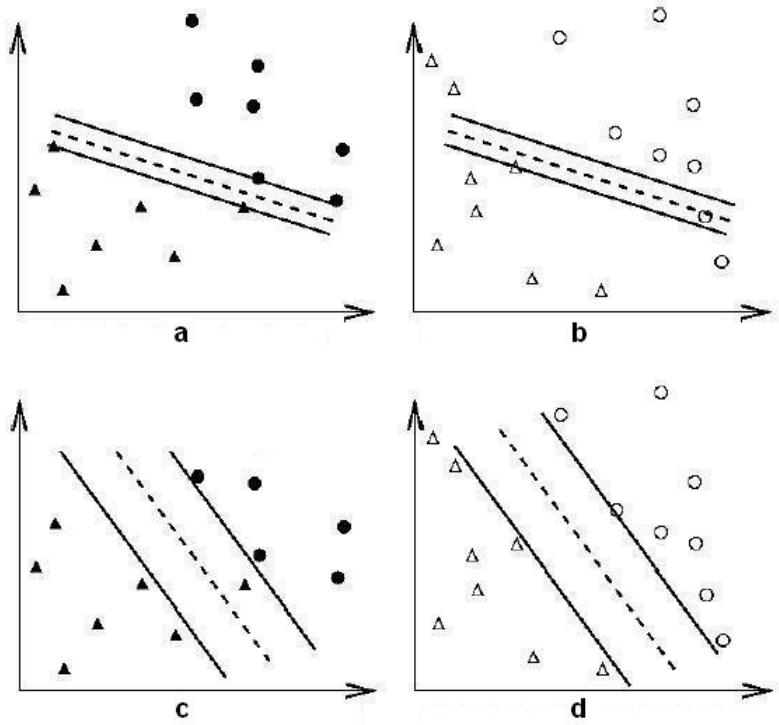


Figura B.1: *a.* SVM Sobre-entrenada y datos de entrenamiento. *b.* SVM Sobre-entrenada y datos de prueba. *c.* SVM sin Sobre-entrenamiento y datos de entrenamiento. *d.* SVM sin Sobre-entrenamiento y datos de prueba.

B.3.1. Validación cruzada [16].

El método consiste en dividir el bloque de datos de entrenamiento en n partes iguales. A continuación, para un cierto valor de hiperparámetros, se utilizan $n-1$ de las n partes para entrenar la máquina y la parte restante para hallar el error[3] de validación. Este proceso se realiza por tanto n veces, lo que permite usar todas las muestras para hallar un error de validación. Por último, se deben promediar los n valores de error de validación encontrados para obtener un solo error de validación asociado a los parámetros en uso.

[3]De igual forma se puede trabajar con el porcentaje de acierto.

B.3.2. Búsqueda en malla [16].

La búsqueda en malla aunque no ha sido presentada formalmente, en los últimos años los investigadores de las máquinas de aprendizaje la han empleado ampliamente por su simplicidad. Es básicamente el entrenamiento de la SVM con diferentes valores[4] para los parámetros del kernel y de penalización. En el caso del kernel RBF, la malla es de dos dimensiones (C, σ) y se recomienda hacerla con un paso en potencias de dos[5], para lograr una búsqueda rápida, luego se hace una rejilla más pequeña al rededor de los parámetros que mayor precisión hallan arrojado.

[4] Desde un punto inicial, siguiendo un paso de avance hasta un punto final.
[5] $C = \left[2^{-5}, 2^{-3}, ..., 2^{15}\right]$, $\sigma = \left[2^{-5}, 2^{-3}, ..., 2^{7}, ...\right]$

Anexo C

Selección del modelo

Los resultados obtenidos con la función kernel RBF son satisfactorios, superando el desempeño de las otras dos funciones kernel utilizadas en este estudio. Nótese también que en general el kernel RBF presenta un reducido porcentaje de vectores de soporte, lo que en términos computacionales se traduce en menores requerimientos de memoria para realizar la tarea impuesta. El comportamiento de los modelos en cada uno de los dos esquema (LIF y LDF) son mostrados en las siguientes figuras. Modelos basados en la función kernel polinomial Figuras C.1 y C.2. Modelos basados en la función kernel sigmoide Figuras C.3 y C.4. Modelo basado en la función kernel RBF Figuras C.5 y C.6.

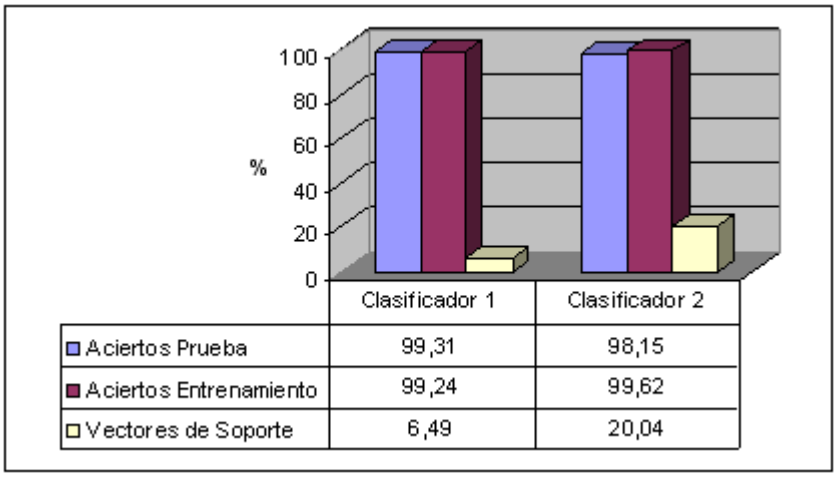
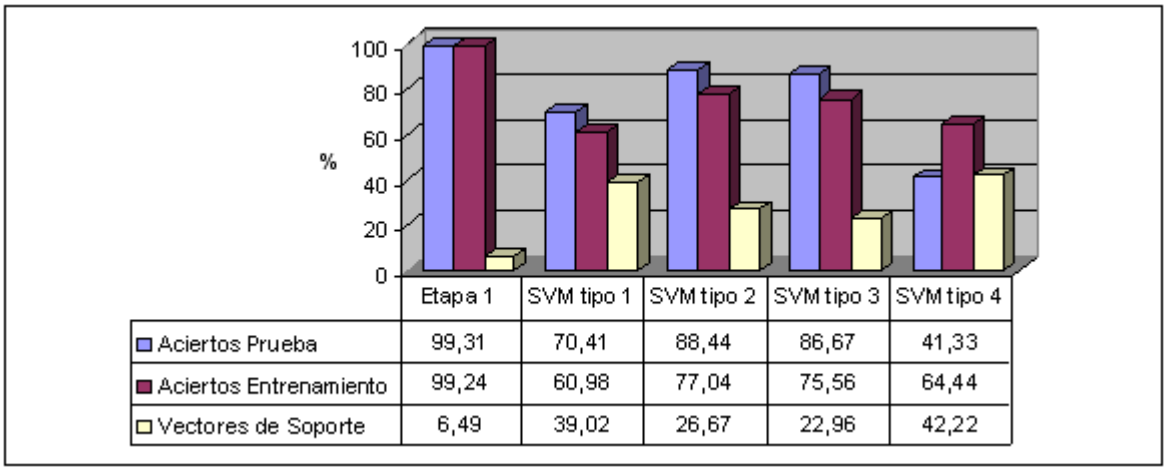
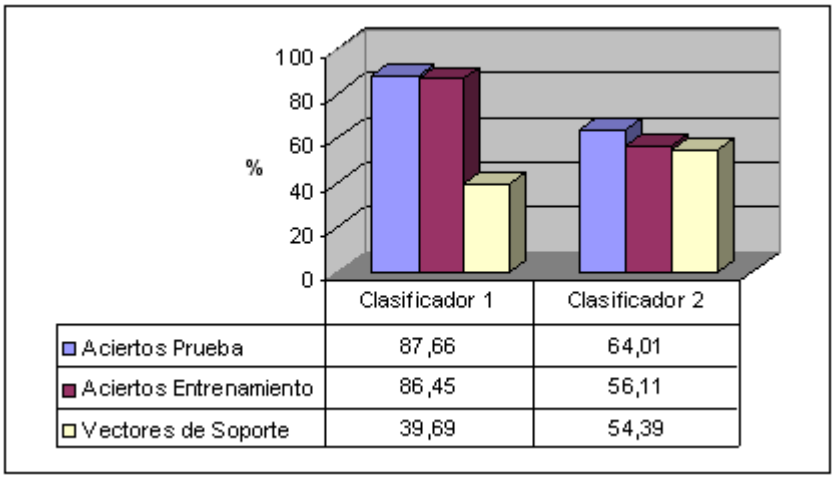


Figura C.1: Modelo LIF con kernel polinomial



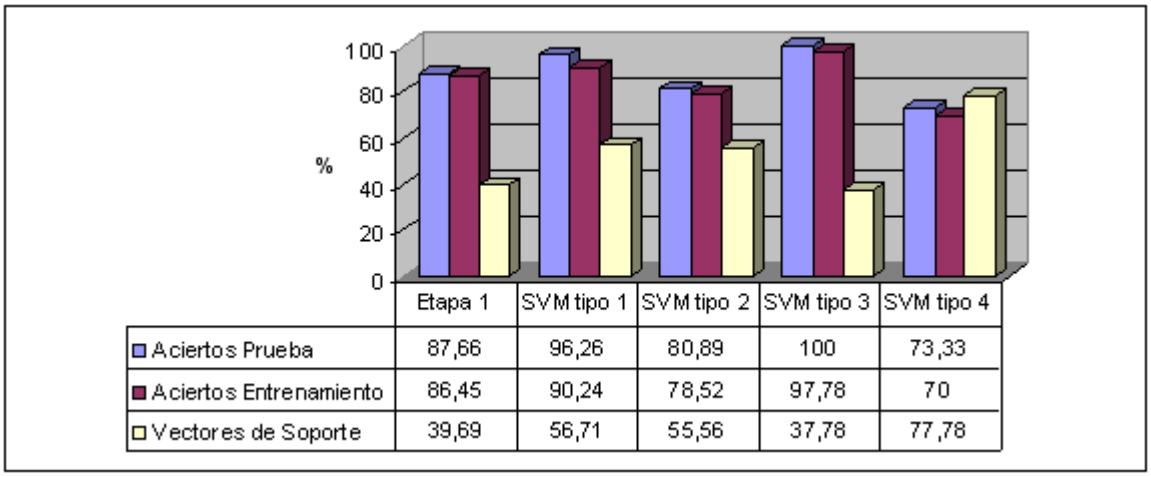
	Etapa 1	SVM tipo 1	SVM tipo 2	SVM tipo 3	SVM tipo 4
Aciertos Prueba	99,31	70,41	88,44	86,67	41,33
Aciertos Entrenamiento	99,24	60,98	77,04	75,56	64,44
Vectores de Soporte	6,49	39,02	26,67	22,96	42,22

Figura C.2: Modelo LDF con kernel polinomial



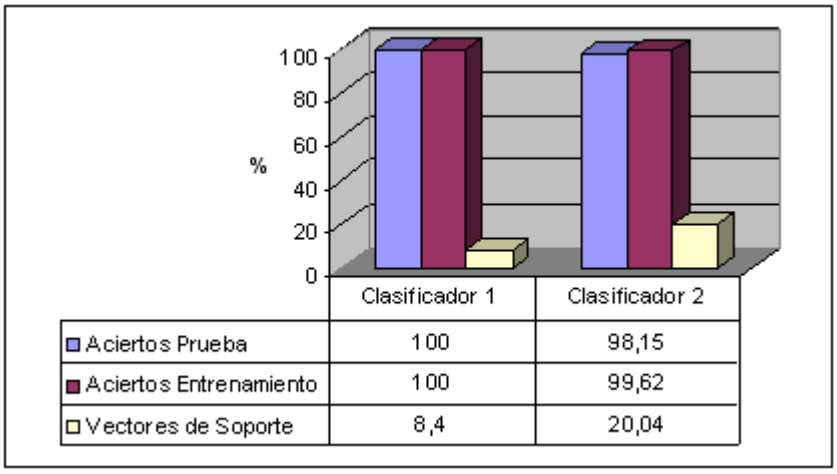
	Clasificador 1	Clasificador 2
Aciertos Prueba	87,66	64,01
Aciertos Entrenamiento	86,45	56,11
Vectores de Soporte	39,69	54,39

Figura C.3: Modelo LIF con kernel sigmoide



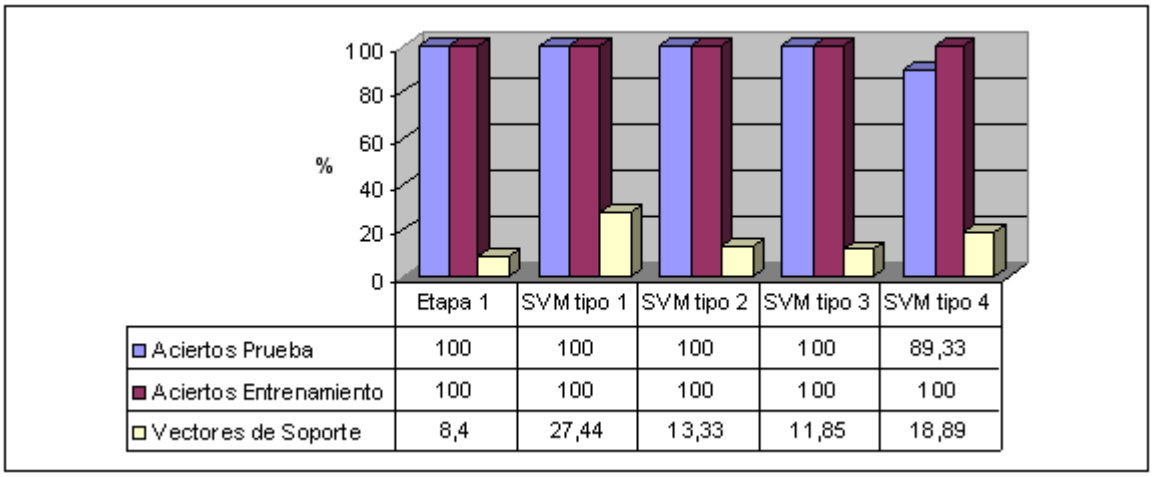
	Etapa 1	SVM tipo 1	SVM tipo 2	SVM tipo 3	SVM tipo 4
Aciertos Prueba	87,66	96,26	80,89	100	73,33
Aciertos Entrenamiento	86,45	90,24	78,52	97,78	70
Vectores de Soporte	39,69	56,71	55,56	37,78	77,78

Figura C.4: Modelo LDF con kernel sigmoide



	Clasificador 1	Clasificador 2
Aciertos Prueba	100	98,15
Aciertos Entrenamiento	100	99,62
Vectores de Soporte	8,4	20,04

Figura C.5: Modelo LIF con kernel RBF



	Etapa 1	SVM tipo 1	SVM tipo 2	SVM tipo 3	SVM tipo 4
Aciertos Prueba	100	100	100	100	89,33
Aciertos Entrenamiento	100	100	100	100	100
Vectores de Soporte	8,4	27,44	13,33	11,85	18,89

Figura C.6: Modelo LDF con kernel RBF

En los modelos con la función kernel RBF, cabe resaltar que el 98,15 % de aciertos de prueba en el esquema LIF representan 16 erróneas localizaciones de fallas de las cuales 12 son trifásicas, 1 bifásica, 2 bifásicas a tierra y 1 monofásica. En el caso del esquema LDF se presentan erróneas localizaciones solo en las fallas trifásicas (SVM tipo 4), siendo estas 16 en total.

Se escoge el esquema LDF basado en la función RBF dada su tendencia a errar la localización de las fallas menos comunes (trifásicas) y el gran desempeño que muestra al determinar acertadamente la zona en la que se ubican las fallas más comunes (monofásicas y bifásicas) en los sistemas de distribución de energía eléctrica.

Anexo D

Toolbox SVM

D.1. Introducción

Debido a que MATLAB carece de programas relacionados con SVM, y el software encontrado sobre esta técnica de inteligencia artificial no satisfizo los intereses de los autores, surgió la necesidad de implementar una paquete de funciones (toolbox) especializadas en SVMs. Esta toolbox fue realizada por Germán Morales y Alvaro Gómez durante la realización de este trabajo de grado.

Este apéndice es una guía rápida para el manejo de la toolbox *máquinas de soporte vectorial* desarrollada en MATLAB, dando una breve explicación sobre las principales funciones que componen esta toolbox (una explicación más detallada se encuentra en las ayudas de los programas[1]). Se muestra un ejemplo para el manejo de este software. Aunque existe interfaz gráfica para la toolbox, las funciones son independientes, facilitando su empleo desde cualquier otro programa

D.2. Instalación

Para instalar *svm_ tool*, es necesario tener instalado MATLAB 6.5 o superior. Siga los siguientes pasos para la instalación:

1. Ejecutar MATLAB 6.5 (o superior).
2. Copiar la carpeta *svm_ tool* a un directorio determinado.
3. Adicionar al *path* de MATLAB la carpeta *svm_ tool* y subcarpetas.

Después de los anteriores pasos se puede emplear cualquiera de las funciones que ofrece *svm_ tool*.

D.3. Formato de datos

Se darán las pautas a seguir en el manejo del formato de los datos y como se configura la toolbox para obtener los resultados deseados.

[1] Para obtener la toolbox escribir a german_yacko@yahoo.com.mx o agrchoco@yahoo.com

Tabla D.1: Formato de descriptores

Nombre [tipo de dato]	Descripción
`datos [struct]`	
`.X [NxP]`	Vectores de entrenamiento, donde se tienen N vectores de dimensión [1xP]
`.Y [Nx1]`	Etiquetas de los datos de entrenamiento (1,2,...,K)

Los vectores de entrenamiento se almacenan en una estructura como se muestra en la Tabla D.1. Estos datos son útiles en la etapa de entrenamiento de la SVM. Todo vector que se desee clasificar para conocer su etiqueta, se debe expresar en este formato.

Para configurar la SVM para la etapa de entrenamiento con las especificaciones deseadas, se debe seguir el formato mostrado en la Tabla D.2. .

Después de haber entrenado la SVM se obtienen resultados como se muestra en la Tabla D.3. El usuario no debe manipular esta estructura ya que esta posee toda la información necesaria para clasificar los nuevos datos.

D.4. Funciones

Para conocer las funciones disponibles en la toolbox, teclear:

```
>> help svm_tool

MAQUINAS DE SOPORTE VECTORIAL

Biclasificacion
  svm_2class  - SVM biclasificadora
  svm_2decis  - Clasificador SVM
  svm_kernel  - Evalua la funcion Kernel

Multiclasificacion
  k_clases    - SVM para multiclasificacion
  k_decis     - Clasificador de K clases
  gen_cod     - Genera la matriz de descomposicion
  exact_decis - Evalua exactitudes

Busqueda de parametros
  valcros     - Validacion cruzada
  aleatorio   - Arma grupos aleatorios

 Autores: German Morales UIS
          Alvaro Gomez   UIS
```

Tabla D.2: Estructura que configura la SVM

Nombre [tipo de dato]	Descripción
`opciones [struct]`	
`.C [1x1]` `.C [1xDIMC]`	Parámetro de penalización Parámetros de penalización que se utilizarán en la validación cruzada
`.Kernel [string]`	Tipo de kernel
`'linear'`	Lineal
`'rbf'`	RBF, gausiano
`'poly'`	Polinomial
`'sigmoid'`	Sigmoide
`.Kernel_par [2x1]` `.Kernel_par [2xDIMpar]`	Parámetro del kernel Parámetros del kernel que se utilizarán en la validación cruzada
* `.descom [string]`	Tipo de descomposición para multiclasificación
`'ovo'`	Uno contra uno (por defecto)
`'ovr'`	Uno contra el resto
`'ecoc_2c'`	Ecoc doble capa
* `.metodo [string]`	Algoritmo de optimización cuadrática
`'qpsolver'`	Algoritmo de optimización toolbox de MATLAB
`'loqo'`	Algoritmo de punto interior por Alex Smola
`'irwls'`	Algoritmo propuesto por Fernando Perez Cruz
`'smo'`	Optimización mínima secuencial (por defecto)
* `.eps`	Error que tolera el algoritmo de optimización (0,001 por defecto)
* `.tol`	Tolerancia de las condiciones KKT (0,001 por defecto)

*Campos opcionales

Tabla D.3: Estructura de la SVM entrenada

Nombre [tipo de dato]	Descripción
`modelo [struct]`	
`.Kernel [string]`	Tipo de kernel
`.Kernel_par [2x1]`	Parámetro del kernel
`.C [1x1]`	Parámetro de penalización
`.clases [1x1]`	Numero de clases K
`.exact_entr [1x1]`	Exactitud de entrenamiento
`.exact_test [1x1]`	Exactitud de prueba, este campo aparece por la validación cruzada
`.detalles [struct]`	
`.metodo [string]`	Metodo que se utilizo en problema de optimización
`.descom [string]`	Tipo de descomposición
`.eps [1x1]`	Error de parada
`.tol [1x1]`	Tolerancia de las condiciones KKT
`.num_SV [1x1]`	Numero de vectores de soporte
`.SV_y [1xK]`	Numero de vectores de soporte por clase
`.SVM [struct][1xL]`	Donde L es el numero de maquinas biclasificadoras que se ejecutan para la multiclasificar K clases L = size(D,1). D matriz de descomposición L = K*(K+1)/2 si se usa 'ovo' L = K si se usa 'ovr' L = K*(K+1) si se usa 'ECOC_2C'
`.SV [NixP]`	Vectores de soporte. Ni es el número de SV.
`.Alpha [Nix1]`	Multiplicadores de lagrange
`.b [1x1]`	Constante de la función de decisión
`.exact_entr [1x1]`	Exactitud de entrenamiento de cada maquina biclasificadora
`.clase1 [1xa1]`	Clases con etiqueta 1 implicadas en cada maquina biclasificadora
`.clase2 [1xa2]`	Clases con etiqueta 2 implicadas en cada maquina biclasificadora

```
Ultima modificacion: octubre/2005
```

Para obtener una ayuda detallada sobre el empleo de cualquiera de estas funciones, escribir *help* seguido del nombre de la función. Cada función de multiclasificación y de búsqueda de parámetros funciona para dos clases o más, y las funciones de biclasificación solo operan para dos clases.

D.4.1. svm_2class

Entrena la SVM de dos clases.

Sinopsis: `modelo = svm_2class(datos,opciones)`

Esta función es tal vez el corazón de la toolbox, debido a que contiene los algoritmos de optimización.

D.4.2. svm_2decis

Esta función clasifica los datos (clases = 2).

Sinopsis: `[Yi,fx] = svm_2decis(X,modelo)`

`Yi` corresponde a la clase de la entrada `X` dependiendo del modelo de SVM que se halla entrenado; y `fx` corresponde a la evaluación numérica de la función de decisión.

D.4.3. svm_kernel

Evalua la matriz kernel.

Sinopsis:
```
H = svm_kernel(X1,X2,tipo,par)
H = svm_kernel(X1,X2,opciones)
```

Halla la matriz kernel, donde:$H\left(i,j\right) = k\left(X1\left(:,i\right),X2(:,j)\right)$. Función necesaria para el entrenamiento y la clasificación. Donde los tipos de kernel disponibles son:

`'linear'`	: lineal	$H\left(a,b\right) = a' \times b$
`'rbf'`	: RBF (gausiano)	$H\left(a,b\right) = \exp\left(\frac{-,5\times\|a-b\|^2}{par(1)^2}\right)$
`'poly'`	: polinomial	$H\left(a,b\right) = \left(a' \times b + par\left(2\right)\right)^{par(1)}$
`'sigmoid'`	: sigmoide	$H\left(a,b\right) = \tanh\left(par(1) \times \left(a' \times b\right) + par\left(2\right)\right)$

D.4.4. k_clases

Entrena la SVM para múltiples clases (≥ 2).

Sinopsis:
```
modelo = k_clases(datos,opciones)
modelo = k_clases(datos,opciones,D)
```

Entrena la SVM para multiclasificación, donde el problema de K clases es descompuesto en problemas de clasificación binaria. Si se especifica `D` se trabaja con esta matriz de descomposición.

Si esta función se trabaja con dos clases únicamente, proporciona los mismos resultados que *svm_ 2class*.

D.4.5. k_decis

Esta función clasifica los datos (clases ≥ 2).

Sinopsis:
```
Yi = k_decis(X,modelo)
```

`Yi` corresponde a la clase de la entrada `X` dependiendo del modelo de SVM que se halla entrenado y el código de descomposición.

Si esta función se trabaja con dos clases únicamente, proporciona los mismos resultados que *svm_ 2decis*.

D.4.6. gen_cod

Genera una matriz de descomposición.

Sinopsis:
```
D = gen_cod(K,tipo)
```

Se genera la matriz de descomposicion D dependiendo del numero de clases y el tipo de código. Esta matriz de descomposicion es la que indica de que manera se utilizarán las máquinas biclasificadoras para la multiclasificación.

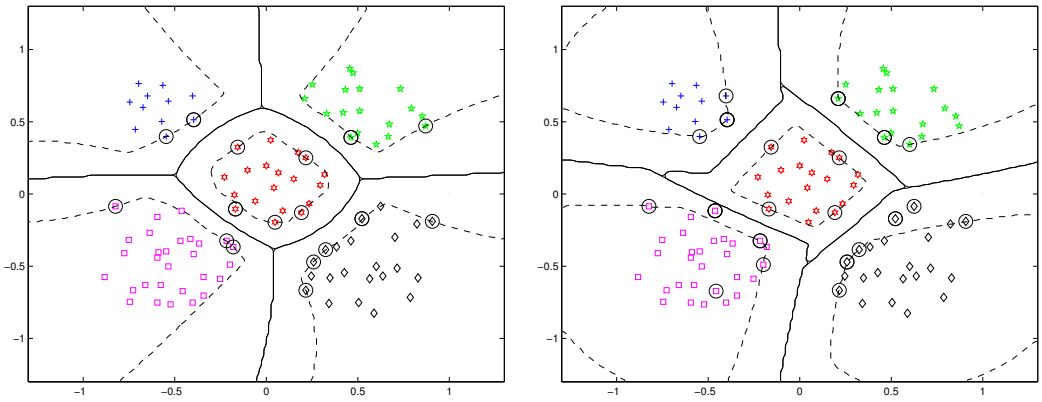
D.4.7. exact_decis

Evalua exactitudes.

Sinopsis:
```
exact = exact_decis(X,modelo,Y)
```

Etiqueta la entrada X con su correspondiente clase dependiendo del modelo de SVM que se halla entrenado y lo compara con la verdadera etiqueta Y sacando así el porcentaje de acierto.

Figura D.1: Derecha: multiclasificación uno contra el resto. Izquierda: multiclasificación uno contra uno.



D.4.8. valcros

Validación cruzada.

Sinopsis:
```
modelo = valcros(datos,opciones)
modelo = valcros(datos,opciones,datos_tst)
modelo = valcros(datos,opciones,datos_tst,partes)
modelo = valcros(datos,opciones,datos_tst,partes,D)
```

Esta función entrena las SVM con el método de validación cruzada para escoger parámetros adecuados. Si se ingresan datos de test, al final del entrenamiento se clasifican estos datos y se saca el porcentaje de acierto. Se puede especificar el numero de particiones que se van a tomar para la validación cruzada.

D.4.9. aleatorio

Esta función es utilizada por la validación cruzada para realizar las particiones del entrenamiento. Por ejemplo si se trabaja con cuatro particiones, y se tienen 100 datos, 20 de la clase uno y 80 de la clase 2, cada partición queda con 5 datos de la clase uno y 20 datos de la clase dos.

D.5. Ejemplos

A continuación se presentan ejemplos para mostrar el fácil manejo de las funciones de la toolbox *svm_ tool*.

D.5.1. Ejemplo de multiclasificación

Se cargan unos datos de ejemplo (contenidos en la toolbox), luego se procede a configurar la SVM, y se entrena:

```
>> datos = load('ejemplo1')
>> opciones=struct('Kernel','rbf','metodo','smo','descom','ovr','C',1e4,'Kernel_par',1);
>> modelo = k_clases(datos,opciones);
>> graficar(1,datos);
>> graficar(2,modelo);
```

Los resultados del entrenamiento y zonas de decisión se muestran en la Figura D.1. Después se entrena de nuevo cambiando el código de descomposición:

```
>> opciones.descom = 'ovo';
>> modelo = k_clases(datos,opciones);
>> figure;
>> graficar(1,datos);
>> graficar(2,modelo);
```

Los resultados se muestran en la Figura D.1.

D.5.2. Ejemplo de validación cruzada

Se cargan unos datos de ejemplo (contenidos en la toolbox):

```
>> datos = load('datos4')
>> datos_tst = load('datos4_tst')
>> graficar(1,datos);
```

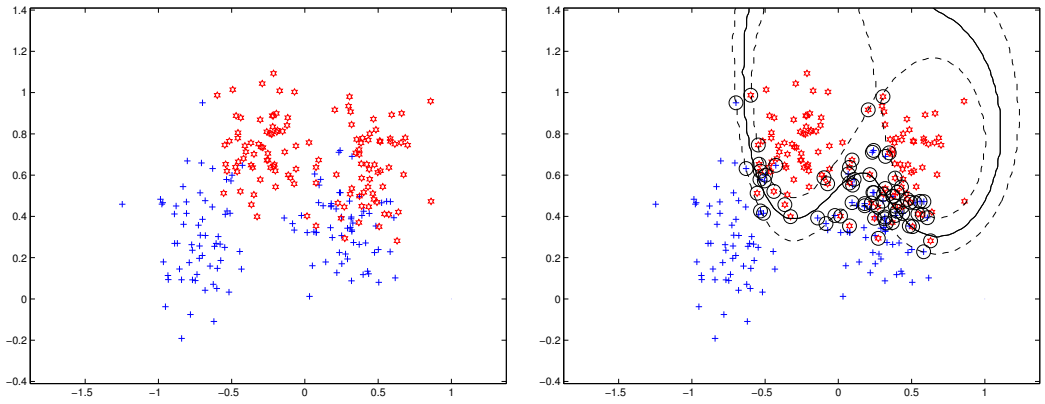
Aparece la gráfica mostrada en la Figura D.2, y en el área de trabajo se imprime:

```
datos =
   X: [250x2 double]
   Y: [250x1 double]
datos_tst =
   X: [250x2 double]
   Y: [250x1 double]
```

A continuación se procede a configurar la búsqueda de parámetros para la validación cruzada y después entrenar:

```
>> C = 2.^[5,6];
```

Figura D.2: Ejemplo validación cruzada.



```
>> par = 2.^[-2:0];
>> opciones = struct('Kernel','rbf','metodo','smo','C',C,'Kernel_par',par);
>> modelo = valcros(datos,opciones,datos_tst)
>> graficar(2,modelo);
```

Aparece la gráfica mostrada en la Figura D.2, y se imprime la siguiente área de trabajo:

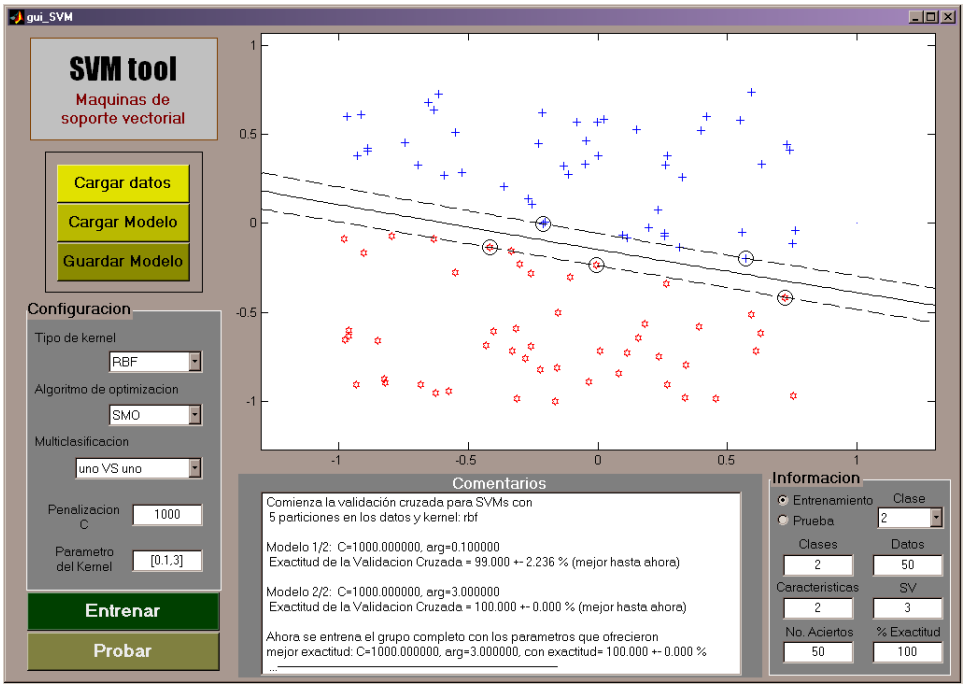
```
Comienza la validación cruzada para SVMs con
5 particiones en los datos y kernel: rbf

Modelo 1/6: C=32.000000, arg=0.250000
parte 1/5: #trn/#tst = 200/50, exact trn = 89.00%, exact tst = 90.00%
parte 2/5: #trn/#tst = 200/50, exact trn = 90.50%, exact tst = 76.00%
parte 3/5: #trn/#tst = 200/50, exact trn = 89.00%, exact tst = 88.00%
parte 4/5: #trn/#tst = 200/50, exact trn = 91.50%, exact tst = 76.00%
parte 5/5: #trn/#tst = 200/50, exact trn = 90.50%, exact tst = 94.00%
Exactitud de la Validacion Cruzada = 84.800 +- 8.319% (mejor hasta ahora)

Modelo 2/6: C=32.000000, arg=0.500000
parte 1/5: #trn/#tst = 200/50, exact trn = 88.00%, exact tst = 90.00%
parte 2/5: #trn/#tst = 200/50, exact trn = 91.00%, exact tst = 84.00%
parte 3/5: #trn/#tst = 200/50, exact trn = 89.50%, exact tst = 86.00%
parte 4/5: #trn/#tst = 200/50, exact trn = 91.00%, exact tst = 80.00%
parte 5/5: #trn/#tst = 200/50, exact trn = 88.50%, exact tst = 92.00%
Exactitud de la Validacion Cruzada = 86.400 +- 4.775% (mejor hasta ahora)

Modelo 3/6: C=32.000000, arg=1.000000
parte 1/5: #trn/#tst = 200/50, exact trn = 87.50%, exact tst = 86.00%
parte 2/5: #trn/#tst = 200/50, exact trn = 87.00%, exact tst = 82.00%
parte 3/5: #trn/#tst = 200/50, exact trn = 88.00%, exact tst = 84.00%
parte 4/5: #trn/#tst = 200/50, exact trn = 87.50%, exact tst = 84.00%
parte 5/5: #trn/#tst = 200/50, exact trn = 86.00%, exact tst = 92.00%
```

```
Exactitud de la Validacion Cruzada = 85.600 +- 3.847 %

Modelo 4/6: C=64.000000, arg=0.250000
parte 1/5: #trn/#tst = 200/50, exact trn = 88.00%, exact tst = 88.00%
parte 2/5: #trn/#tst = 200/50, exact trn = 91.50%, exact tst = 80.00%
parte 3/5: #trn/#tst = 200/50, exact trn = 91.00%, exact tst = 84.00%
parte 4/5: #trn/#tst = 200/50, exact trn = 91.00%, exact tst = 80.00%
Exactitud de la Validacion Cruzada = 83.000 +- 3.830 %

Modelo 5/6: C=64.000000, arg=0.500000
parte 1/5: #trn/#tst = 200/50, exact trn = 88.50%, exact tst = 90.00%
parte 2/5: #trn/#tst = 200/50, exact trn = 91.00%, exact tst = 84.00%
parte 3/5: #trn/#tst = 200/50, exact trn = 90.00%, exact tst = 88.00%
parte 4/5: #trn/#tst = 200/50, exact trn = 91.00%, exact tst = 80.00%
parte 5/5: #trn/#tst = 200/50, exact trn = 88.50%, exact tst = 94.00%
Exactitud de la Validacion Cruzada = 87.200 +- 5.404 % (mejor hasta ahora)

Modelo 6/6: C=64.000000, arg=1.000000
parte 1/5: #trn/#tst = 200/50, exact trn = 87.50%, exact tst = 86.00%
parte 2/5: #trn/#tst = 200/50, exact trn = 87.50%, exact tst = 82.00%
parte 3/5: #trn/#tst = 200/50, exact trn = 87.50%, exact tst = 82.00%
parte 4/5: #trn/#tst = 200/50, exact trn = 89.00%, exact tst = 84.00%
Exactitud de la Validacion Cruzada = 83.500 +- 1.915 %

Ahora se entrena el grupo completo con los parametros que ofrecieron
mejor exactitud: C=64.000000, arg=0.500000, y exactitud= 87.200 +- 5.404 %

La SVM ya entrenada con Validacion Cruzada se prueba con
250 datos y se obtiene una exactitud del 90.000 %
--------------------------------------------------------------------------
modelo =
      Kernel: 'rbf'
           C: 64
  Kernel_par: 0.5
      clases: 2
  exact_entr: 0.9000
  exact_test: 0.9000
    detalles: [1x1 struct]
         SVM: [1x1 struct]
```

Se muestra el resto de información contenida en *modelo:*

```
>>modelo.detalles
ans =
    metodo: 'smo'
       eps: 0.0010
       tol: 0.0010
```

Figura D.3: Interfaz gráfica




```
    num_SV: 74
      SV_y: [36 38]

>>modelo.SVM
ans =
            SV: [74x2 double]
         Alpha: [74x1 double]
             b: 2.2410
    exact_entr: 0.9000
```

Este ejemplo muestra en forma general como aprovechar la toolbox *svm_tool* con el método de validación cruzada para la búsqueda de parámetros.

D.6. Interfaz gráfica

Diseñada con el fin de proporcionar un ambiente amigable al usuario para el manejo de las funciones descritas anteriormente, ver Figura D.3. Para utilizarla se teclea *gui_SVM* en el prompt de MATLAB.