Implementaciones en Matlab de los Algoritmos Adaptativos para los Sistemas de Antenas Inteligentes

J. Alvarez[1], M. Chuez[2], P. Vargas[3]
Facultad de Ingeniería Eléctrica y Computación (FIEC)
Escuela Superior Politécnica del Litoral
Campus Gustavo Galindo, Km 30.5 vía Perimetral
Apartado 09-01-5863. Guayaquil, Ecuador
juanfranciscoalvarezalvarado@yahoo.com[1], chugo1900@hotmail.com[2], pvargas@espol.edu.ec[3]

Resumen

Actualmente existen implementaciones en matlab de algunos algoritmos adaptativos como son el algoritmo mínimo cuadrado (LMS) y algoritmo mínimo cuadrado recursivo (RLS), pero no hay disponibles implementaciones para los algoritmos de módulo constante (CMA) y el algoritmo de matriz inversa (DMI), sabiendo esto nos planteamos realizar aplicaciones en CMA y DMI, tanto para señales gaussianas como para señales senosoidales para antenas, para esto nos planteamos algunas interrogantes ¿Qué tanto variara la velocidad de convergencia y la potencia de estas antenas aplicando los mismos valores de entrada en estos algoritmos?, ¿Qué sucederá si variamos el número de antenas?, ¿Qué resultados obtendríamos en el comportamiento de los gráficos de curvas de error, curvas de pesos estimados, curvas de la señal a la salida del filtro vs la señal de referencia del sistema?, ¿Qué problemas podrían ocurrir al implementar los algoritmos CMA y DMI?. (Algunos programadores ingresan valores de ruido a la entrada del sistema para simular señales reales) y otros lo colocan a la salida del filtro en los algoritmos adaptativos, ¿podríamos comparar los resultados obtenidos de ingresar el ruido a la entrada del sistema con el ruido ingresado a al salida del filtro?, ¿ que sucederá si cambiamos la señal de referencia con una señal parecida a la original?, ¿Qué tanto aumenta el error con este cambio?. En este trabajo respondemos estas interrogantes y analizamos los resultados obtenidos con estos cuatro algoritmos adaptativos, LMS, RLS, CMA y DMI.

Palabras Claves: *Implementaciones en Matlab de los Algoritmos Adaptativos para los Sistemas de Antenas Inteligentes, filtros, wiener, kalman, Algoritmo, Implementación, Simulación, LMS, RLS, CMA, DMI.*

Abstract

There are currently matlab implementations of some adaptative algorithms such as least square algorithm (LMS) and recursive least square algorithm (RLS), but there is no available implementations for the constant modulus algorithm (CMA) and the inverse matrix algorithm (DMI) knowing this we plan to carry CMA and DMI applications, both for Gaussian signals as sinusoids signals for antennas, for this we ask some questions , how much will vary the speed of convergence and the power of the antennas using the same input values these algorithms?. What happens if we vary the number of antennas?. What would obtain results of the behavior of graphics error curves, estimated weight curves, curves of the signal at the output of the filter vs the reference signal system?. What problems might occur when implementing the algorithms CMA and DMI?. (some programmers entering noise values to the input of the system to simulate real signals) and others placed at the output of adaptative filter algorithms, can we compare the results of the noise entering the system input with the noise entered not the output of the filter?. What will happen if we change the reference signal with a signal similar to the original?. How much the error increases with this change?. In this paper we answer these questions and analyze the results obtained with these four adaptative algorithms, LMS, RLS, CMA and DMI.

Keywords: *Matlab implementations of algorithns for smart antenna systems, filters, wiener, kalman, algorithm, implementation, simulation, LMS, RLS, CMA, DMI.*

1. Introducción

Realizamos la implementación en matlab de algunos algoritmos adaptativos como son el algoritmo LMS (algoritmo mínimo cuadrado), el algoritmo RLS (algoritmo mínimo cuadrado recursivo), el algoritmo CMA (algoritmo de modulo constante y el algoritmo DMI (algoritmo de matriz inversa directa), hicimos el

análisis de estos algoritmos basados en su señal de referencia y en el caso del CMA que es un algoritmo adaptativo ciego se lo realizo en un entorno RLS para ayudar a su convergencia, se determino la capacidad de red, la cobertura y el desempeño de las antenas inteligentes en estos diferentes algoritmos adaptativos, se determino la potencia de transmisión en su relación con la capacidad y cobertura. Se hizo un seguimiento de la señal deseada, para tener simulaciones mas exactas se utilizo valores de antenas recomendados por la central matlab y en otros casos se uso valores aproximados a la realidad.

2. El Algoritmo Adaptativo Mínimo Cuadrado LMS

Este algoritmo utiliza la aproximación estocástica para el cálculo de la gradiente de la función mínima cuadrática MMSE (Mínimum Mean Square Error). A la minimización de la función de cálculo de gradiente se la conoce como descenso por gradiente (steepest descent), el error cuadrático medio mínimo siempre sigue la dirección tangente a la superficie, el algoritmo LMS requiere un mayor número de iteraciones para llegar a la convergencia, también requiere un mayor número de coeficientes para funcionar, su cálculo de error es inferior al de otros algoritmos adaptativos, pero tiene la ventaja de ser mas fácil de entender matemáticamente, tiene características lineales lo que garantiza que el sistema sea más robusto (es decir que no se caiga el sistema), por eso es ampliamente usado, solo y en híbridos con otros algoritmos.

El algoritmo LMS utiliza el conformador de haces adaptativos el cual es una técnica que permite una máxima radiación hacia un usuario deseado y nulos en la dirección de las señales interferentes, figura 3 (sistema de un arreglo adaptativo)

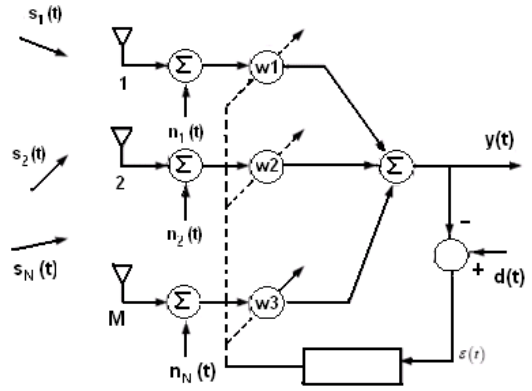


Figura 3. Sistema de arreglo adaptativo

Las $s_N(t)$ representan las señales incidentes en los elementos de las antenas. A estas señales se les suma el ruido $n_N(t)$, para después ser ambos ponderados por el procesador de señales digitales controlado por un algoritmo adaptativo, De esta manera se obtiene en la salida, la suma de las ponderaciones variables de cada elemento de antena, designada como y(t). Los pesos Wm son calculados iterativamente basándose en la salida del arreglo y(t), la señal de referencia d(t), que es una aproximación de la señal deseada, y las ponderaciones pasadas. La salida del arreglo esta dado por la ecuación 1.

$$y(t) = W^H x(t) \quad (1)$$

Donde W^H es la transpuesta conjugada compleja del vector de ponderación W.

El vector respuesta a los datos muestreados, en la salida del arreglo, esta dado por:

$$x(t) = s(t)\, a(\theta_0) + \sum_{i=1}^{N_n} ui(t) a(\theta_i) + n(t) \quad (2)$$

El error cuadrático medio a la salida del conformador del haz y de la señal de referencia se lo expresa con la ecuación 3.

$$\varepsilon^2(t) = [\, d^*(t) - W^H x(t)\,]^2 \quad (3)$$

Con estos valores obtenemos la expresión iterativa para la actualización de pesos (ecuación 4)

$$w(k+1) = w(k) - \frac{1}{2}\mu\nabla\,(E\{e^2(k)\}) \quad (4)$$

.x(t) es el vector respuesta a los datos muestreados a la salida de un arreglo en función del tiempo

Con estos valores obtenemos la gradiente de la función de costo (ecuación 5)

$$\nabla(E\{e^2(k)\}) = -2x(k)e^*(k) \quad (5)$$

Cuando reemplazamos esta gradiente ecuación 5 en 4 se obtiene la forma generalizada del algoritmo LMS. Ecuación 6, esta forma generalizada del algoritmo LMS también se la llama función de control o ecuación general de los algoritmos LMS.

$$w(k+1) = w(k) + \mu_{LMS}\, x(k)e^*(k) \quad (6)$$

μ_{LMS} **es un parametro constante**

Al implementar en matlab el algoritmo LMS para señales senosoidales se presentaron algunos problemas, como por ejemplo como ingresar la señal senosoidal en lugar de una señal gaussiana, esto se resolvió ingresando la señal senosoidal de entrada en un lazo for con una cantidad de valores de muestra.

```
for k=1:400
entrada(k)=sin((2*pi*k)/M);
End
```

se suele ingresar la misma señal de entrada como señal de referencia para recuperación de la señal en los algoritmos adaptativos, también es importante recordar que la señal de referencia se coloca a la salida del filtro, y se la suele restar del valor de y(t) para obtener el valor del error..

```
N=60
%begin of algorithm
```

```
w = zeros ( orden  , 1 )
for n = orden : N
 u = entrada(n:-1:n-orden+1);
y(n)= w' * u;  % y(n) señal a la salida
e(n)=senal_referencia(n)-y(n);
 j=j+1
 if e(n)<error
 fprintf('el numero de iteraciones hasta que se completo el error ');
  j
 end
if n < 20
mu=cota/2
else
mu=cota/4
end
w = w + mu * u * e(n);
end
for n =  N+1 : totallength
u = entrada(n:-1:n-orden+1) ;
y(n) = w' * u ;   % y(n) señal a la salida
e(n)=senal_referencia(n)-y(n);
end
```

Al realizar esta implementación tenemos que señalar 2 cosas importantes que sucedieron, uno al ingresar los valores de entrada de una señal senosoidal tuvimos que utilizar la transpuesta de la señal de entrada, porque se realizaba una multiplicación de matrices en el algoritmo.

.entrada=entrada'

Y segundo para el cálculo de la potencia de la señal utilizamos la formula:

Potencia=mean(senal_referencia.^2)

Con este valor de potencia obtenemos los valores de cota que servían para obtener los valores de mu de la constante de ajuste.

```
% calculo de la cota de la contante de ajuste
cota=1/((orden+1)*potencia);
```

Para utilizar la presente implementación se requiere un matlab7 o superior.

3. El Algoritmo Adaptativo Mínimo Cuadrado Recursivo RLS

El algoritmo adaptativo mínimo cuadrado recursivo es una extensión de el algoritmo mínimo cuadrado, presenta varias ventajas como el disminuir el número de iteraciones para llegar a su convergencia, además permite hacer operaciones matemáticas mas complejas que el mínimo cuadrado, el algoritmo RLS utiliza el filtro de kalman para hallar su solución, el filtro de kalman esta basado en la matriz de auto correlación de datos y del vector P. Las formulas más importantes en el algoritmo RLS son:

$$y(n) = w^T(n)x(n) = \sum_{k=1}^{M} w_k(n)x_k(n) \quad n=1,2,.....,N \qquad (7)$$

Donde W(n) son los valores de los pesos que se generan por el algoritmo RLS, con estos valores de pesos obtenemos el valor de y(n) a la salida del filtro, el valor del error esta dado por la ecuación 8.

$$e(n) = d(n) - y(n) = d(n) - w^T(n)x(n) \qquad (8)$$

Las principales formulas utilizadas por el algoritmo RLS son (ecuaciones 9,10 y 11)

$$\hat{w}(n) = \hat{w}(n-1) + q(n)\left[d^*(n) - \hat{w}^H(n-1)x(n)\right] \qquad (9)$$

$$q(n) = \frac{\gamma^{-1}R_{xx}^{-1}(n-1)x(n)}{1+\gamma^{-1}x^H(n)R_{xx}^{-1}(n-1)x(n)} \qquad (10)$$

$$R_{xx}^{-1}(n) = \gamma^{-1}[R_{xx}^{-1}(n-1) - q(n)x(n)R_{xx}^{-1}(n-1)] \qquad (11)$$

Estas ecuaciones provienen del filtro de kalman (figura 2)

Estas ecuaciones RLS las podemos dividir en dos grupos principalmente, en un grupo de ecuaciones para predecir la señal (ecuación 12 y 13).

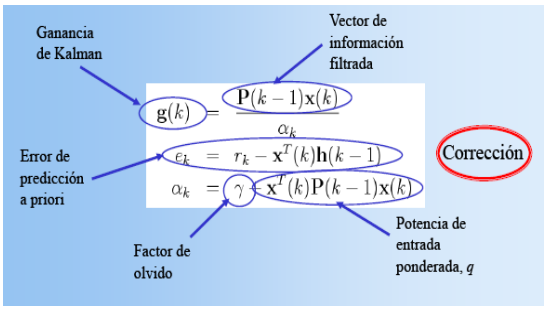


Figura 2. Filtro de Kalman.

$$h(k) = h(k-1) + g(k) \qquad (12)$$

Predicción

$$P(k) = \frac{1}{\gamma}[P(k-1) - g(k)x^T(k)P(k-1)] \qquad (13)$$

Y un grupo de ecuaciones de función correctiva (ecuaciones 14, 15 y 16)

$$g(k) = \frac{P(k-1)x(k)}{\alpha_k} \qquad (14)$$

$$\epsilon_k = r_k - x^T(k)h(k-1) \qquad \text{Corrección} \qquad (15)$$

$$\alpha_k = \gamma + x^T(k)P(k-1)x(k) \qquad (16)$$

Es de señalar que el algoritmo RLS tuvo un desempeño más exacto que el algoritmo LMS para señales gaussianas, y en señales senosoidales un resultado casi idéntico.

4. El algoritmo Adaptativo de Módulo Constante (CMA)

El algoritmo adaptativo de módulo constante (CMA) es un algoritmo de igualación ciega, es decir que no requiere señal de referencia, ya que estos generan su propia señal de referencia desde las señales receptadas, esta es una ventaja que poseen sobre los algoritmos lms y rls, aunque en ocasiones no se utilice esta propiedad del algoritmo cma, ya que muchas veces se prefiere hacer un hibrido lms-cma o rls-cma para aprovechar las características lineales que poseen los algoritmos lms y rls.

Las principales formulas utilizadas por el algoritmo CMA son: (ecuaciones 17,18,19,20,21 y 22)

Ecuación de pesos cma (ecuación 17)

$$J(w(n)) = E\left[(R-|y(n)|^2)^2\right] \quad (17)$$

Donde R es una constante positiva que depende de las propiedades del sistema, valor de u, numero de elementos, etc. Esto lo vemos en la ecuación 18.

$$w(n+1) = w(n) + \mu(R-y(n)y^*(n))y(n)u^*(n) \quad (18)$$

Donde el asterico denota el valor conjugado de un escalar. El algoritmo CMA tiene su máximo rendimiento cuando el orden del filtro es 2. Esto lo vemos en la ecuación general de los algoritmo cma (ecuación 19)

$$J(\mathbf{w}) = E[(|y(n)|^p - R_p)^2] \quad (19)$$

El algoritmo CMA proviene del algoritmo de SATO, desarrollado en 1974, siendo la única diferencia que el algoritmo de SATO es de orden 1, y el algoritmo CMA es de orden 2. En el orden 2 es cuando el algoritmo CMA tiene su máximo rendimiento.

El valor de Rp lo obtenemos de los valores estimados de la señal de entrada evaluados para un numero de canal p. ecuación 20

$$R_p = \frac{E[|s(n)|^{2p}]}{E[|s(n)|^p]} \quad (20)$$

En la ecuación 21 vemos la ecuación de pesos para cualquier valor de p

$$\mathbf{w}_{n+1} = \mathbf{w}_n + \mu \mathbf{x}_n (R_p - |y(n)|^p)|y(n)|^{p-2} y(n)\mathbf{x}_n \quad (21)$$

p=1 ➡ Sato

p=2 ➡ CMA

Para valores complejos se utiliza Xn conjugada ecuación 22

Con constelaciones complejas

$$W_{n+1} = W_n + \mu X_n (R_p - |y(n)|^p)\,|y(n)|^{p-2} y(n)\, X_n^* \quad (22)$$

El algoritmo CMA suele emplearse de dos maneras distintas una es para igualación ciega y la otra es para desconvolucion.

el lazo de actualización de pesos del algoritmo cma es el siguiente:

```
N=80;   % N=80
%begin of the algorithm
lamda = 0.999994115 ;
delta=100000000000000% delta = 1e10 ;
P = delta * eye (orden);
if x==1
    w=[0.15]
    w=w'
end
.
.
end
if x==5
    w=[0.15 0.10 0.20 0.30 -0.20]
    w=w'
end
Rp =1;
for n = orden : N
    u = entrada(n:-1:n-orden+1)
    phi = u' * P ;
    k = phi'/(lamda + phi * u );
    y(n)= w'*u ;
    y(n)
    T(n)=[abs(y(n))].^2
    T2(n)=Rp-T(n);
    T3=T2(n)*u
    e=T3*y(n)
    T4=e'
    T5=T4*u
    mu=5*10^-7
    w=(w)-(mu*T5)
    P = ( P - k * phi ) / lamda;
    Recordedw(1:orden,n)=w
end
for n =  N+1 : totallength
    u = entrada(n:-1:n-orden+1) ;
    y(n) = w' * u ;
    T(n)=[abs(y(n))].^2;
    T2(n)=1-T(n);
    e(n)=T2(n)*y(n);
end
```

Es de tener presente que el algoritmo CMA para señales gaussianas tiende a no converger, para señales senosoidales si converge algunas veces pero converge menos veces que el lms, rls y dmi. Algunos autores proponen ingresar diferentes valores de Rp para ayudar a la convergencia, también proponen ingresar valores en un vector pivote que ayude a la convergencia y evite la perdida de señal, pudiendo este vector pivote tomar diferentes valores. El algoritmo cma tiende a girar sobre los valores ingresados en W en muchas ocasiones. También se suele cambiar el valor de landa para ayudar a la convergencia, debe de manipularse cuidadosamente este valor de lamda, ya que se puede perder fácilmente la señal y no hallar la convergencia. Funciona mejor cuando se ingresan el mismo número de antenas con el mismo número de orden del canal, si se ingresan diferentes numero suele no funcionar el algoritmo

5. El algoritmo Adaptativo de Inversión de Matriz Directa (DMI)

El algoritmo adaptativo de inversión de matriz directa (DMI) es un algoritmo de igualación no ciega de canal, también llamado algoritmo de matriz inversa, suele utilizarse como un algoritmo de búsqueda, en medicina es utilizado es utilizado en redes neuronales al igual que el algoritmo cma.

El algoritmo dmi tiene la característica de que utiliza el filtro de wiener dentro de su programación, por lo que es utilizado en algunos casos para funciones lineales su estructura es parecida a la del algoritmo lms, converge más rápidamente que el algoritmo lms y requiere señal de referencia.

El algoritmo DMI es un algoritmo que utiliza matrices en un lazo de control, puede usar multicanales, en función de frecuencia, fase, etc. Sus formulas principales son:

$$\hat{R}_{xx} = \sum_{i=N_1}^{N_2} x(i)x^H(i) \quad (23)$$

$$\hat{r}_{xd} = \sum_{i=N_1}^{N_2} d^*(i)x(i) \quad (24)$$

$$w = \hat{R}_{xx}^{-1}\hat{r}_{xd} \quad (25)$$

El algoritmo dmi utiliza las formulas de Rxx de la correlación cruzada, pero evaluada entre dos valores N1 y N2, donde N1 y N2 son los limites de iteraciones en donde se evaluara la sumatoria de Rxx que será uno de los valores que nos permitirá obtener los pesos en el algoritmo dmi, este valor de Rxx a su inversa se la multiplica por el valor de rxx.

Donde rxx será igual a d(n)*u, d(n) es la señal de referencia, y u es la señal de entrada al filtro evaluada en el intervalo N1, N2.

El lazo de evaluación de pesos para el algoritmo dmi es:

```
%begin of algorithm
Rxx=0;
rxx=0;
w = zeros ( orden  , 1 )
N=80
for n = orden : N
    u = entrada(n:-1:n-orden+1);
u(find(u>1))=(u(1)+u(2)+u(3)+u(4)+u(5
))/5;
     u(find(u<-
1))=(u(1)+u(2)+u(3)+u(4)+u(5))/5;
    u
    n
    y(n)= w' * u;
    e(n)=senal_referencia(n)-y(n);
    j=j+1
     if e(n)<error
        fprintf('el     numero     de
iteraciones hasta que se completo el
error ');
        j
     end
    if n < 20
        mu=cota/2;
    else
        mu=cota/4;
    end
    T=u*u';
    if j==1
        Rxx=T*0;
    end
    Rxx=Rxx+T;
    T2=senal_referencia(n)*u;
    if j==1
        rxx=T2*0;
    end
    rxx=rxx+T2;
    T3=inv(Rxx);
    w = (inv(Rxx))*rxx;
w(find(w>1))=(w(1)+w(2)+w(3)+w(4)+w(5
))/5;
w(find(w<-
1))=(w(1)+w(2)+w(3)+w(4)+w(5))/5;
w
end
```

Para evitar las pérdidas de señal se suelen utilizar diferentes métodos de búsqueda.

6. Implementacion en Matlab

Para la realización de esta simulación hemos utilizado el programa Matlab y Simulink v.R2008a, el programa Simulink lo utilizamos para hacer un menú desde el cual llamar a todas las implementaciones lms, rls, cma y dmi tanto para señales gaussianas como para señales senosoidales. Al correr los programas se presentaran múltiples pantallas que guiaran al usuario en la correcta utilización del programa y presentara los resultados de la opción escogida de una manera ordenada. Figura 3

Figura 3. Menú de las implementaciones en Matlab

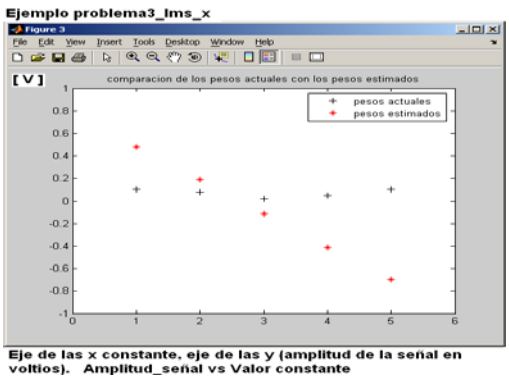
Comparación de resultados para los algoritmo lms, rls , cma y dmi utilizando una señal senosoidal.

Tabla 1 Datos de entrada

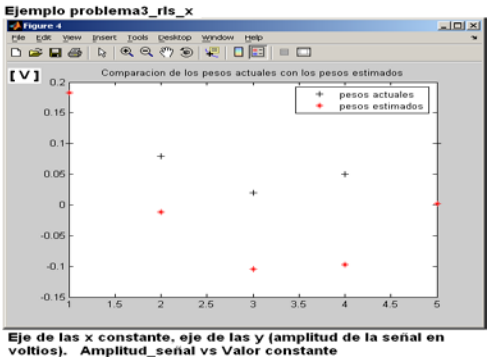
Comparación de los resultados de los algoritmos lms, rls, cma y dmi

Tipo de señal utilizada:	senosoidal
Aplicaciones en matlab utilizados:	problema3 lms, problema3 rls, problema3 cma y Problema3 dmi
Nombre de los Ejemplos utilizados en la comparación:	problema3_lms_x problema3_rls_x problema3_cma_x y problema3_dmi_x.
Orden de la señal:	5
Numero de antenas:	5
Valores iniciales de entrada para las antenas:	V1= 0.10 V V2= 0.08 V V3= 0.02V V4=0.05 V V5=0.10 V
Valores de ruido ingresados para las antenas	ruido1= 0.01 V, ruido2=0.01 V, ruido3= 0.01 V, ruido4=0.02 V, ruido5=0.02 V
opción_ruido=1	opción_señal=2
numero de muestras	400
N	80
rango de seguridad	200
Error del sistema	0.0000001
Constantes de ajuste lms:	mu=0.32 y mu=0.15
Constantes de ajuste rls:	lamda = 0.999994115 delta = 100000000000000
Constante de ajuste cma:	(u)cma=5*10^-7
Constante de ajuste dmi	El algoritmo dmi no usa ninguna constante de ajuste

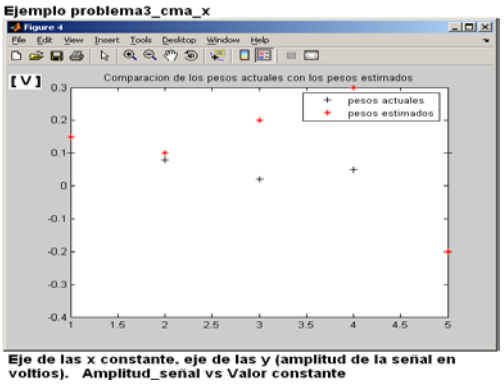
Nota1.- la señal obtenida por el algoritmo cma en algunos casos fue aproximadamente de la mitad de la señal de referencia, conservaba la misma forma de la señal pero disminuía su amplitud



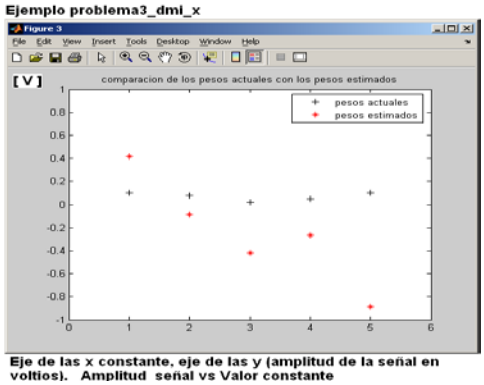
Pantalla1 Comparación de los pesos estimados de una señal senosoidal con los pesos reales o actuales de una señal senosoidal utilizando un algoritmo lms, orden=5, numero de pesos=5.



Pantalla 2. Comparación de los pesos estimados de una señal senosoidal con los pesos reales o actuales de una señal senosoidal utilizando un algoritmo rls. Orden=5, numero pesos=5.

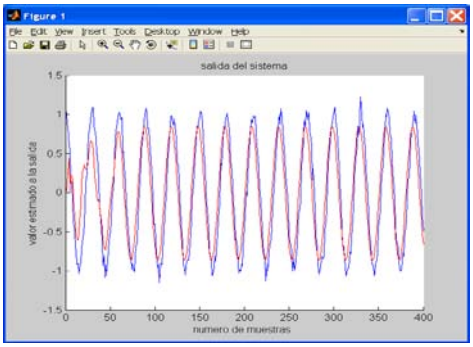


Pantalla 3. Comparación de los pesos estimados de una señal senosoidal con los pesos reales o actuales de una señal senosoidal utilizando un algoritmo cma, orden=5, numero de pesos=5.

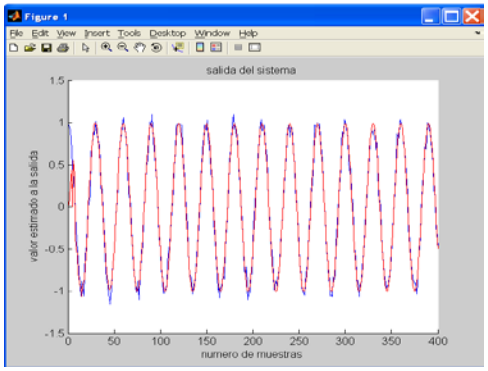


Pantalla 4. Comparación de los pesos estimados de una señal senosoidal con los pesos reales o actuales de una señal senosoidal utilizando un algoritmo dmi, orden=5, numero de pesos=5

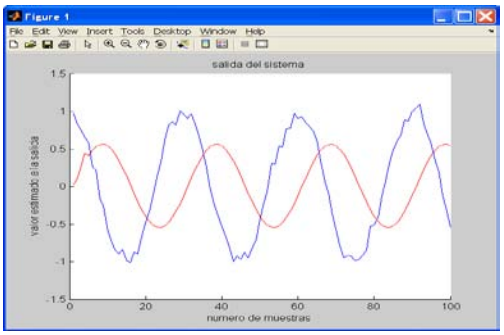
A continuacion colocaremos las pantallas de la comparacion de la señal de referencia con la señal obtenida en cada uno de estos algoritmos.
Nota.- la señal de referencia estara representada con color azul y la señal obtenida por los algoritmos adaptativos de color rojo.



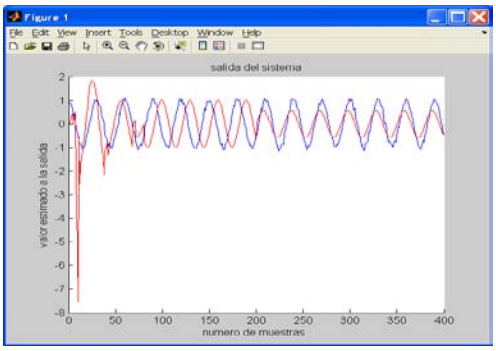
Pantalla 5. Comparación de una señal senosoidal con la señal de referencia del sistema. Algoritmo lms



Pantalla 6. Comparación de una señal senosoidal con la señal de referencia del sistema. Algoritmo rls.



Pantalla 7. Comparación de una señal senosoidal con la señal de referencia del sistema. Algoritmo cma



Pantalla 8. Comparación de una señal senosoidal con la señal de referencia del sistema. Algoritmo dmi

No nos es posible colocar en el presente extracto todas las graficas generadas por las diferentes aplicaciones y las potencias encontradas en las comparaciones, las que hemos puesto en el presente extracto es para dar una idea del comportamiento de las curvas en un ejemplo dado.

7. Conclusiones y resultados

1.- El algoritmo lms casi siempre converge, en un porcentaje muy superior al de los algoritmos rls, cma y dmi.
2.- Al aplicar señales senosoidales la convergencia de los valores ingresados con los valores estimados fue inferior al de los valores obtenidos para señales gaussianas
3.- No se puede utilizar el mismo valor de N en el lazo de los pesos y en el lazo de seguridad del algoritmo dmi.
4.- El valor de N en el lazo de cálculo de pesos del algoritmo dmi debe ser un numero bajo entre N=40 y N=80, la curva se pierde para valores más bajos como N=10 y tiene muchos saltos en la parte de la curva en donde N esta activa. Si se coloca un número menor de N, por ejemplo N=4 el programa dice error, valor de N debe ser mayor o igual al número de orden del filtro. Esto es causado por los valores del lazo for que depende del valor del orden del filtro.
5.- En el algoritmo dmi, se obtiene la convergencia a pesar del valor de N tan bajo, porque la convergencia es..mas..rápida.
6.- Los algoritmos cma y dmi algunas veces no convergen.
7.- En el algoritmo dmi fue necesario utilizar un método de búsqueda de la señal, porque la señal se perdía (para esto utilizamos la función find).
8.- Se podría cambiar el método de búsqueda utilizado en el programa dmi por otro más optimo.

9.- En el algoritmo cma colocando buenos valores de Rp se podría mejorar el desempeño del algoritmo.
10.- Al ingresar los ruidos adicionales en la entrada del filtro la curva y la convergencia fueron más exactos que cuando se aplico el ruido a la salida.
11.- Al utilizar la misma señal de entrada (opcion señal =1) como señal de referencia la convergencia fue mejor que cuando se usaba otra señal que la reemplace (opción=2).
12.- Una solución para disminuir el error en el algoritmo CMA fue hacer un hibrido CMA-RLS, que con solo unas pocas líneas de código se obtenía una disminución..del..error..bastante..buena.
13.- En el algoritmo cma su convergencia fue superior que la del algoritmo rls por lo menos 4 o 5 veces más rápida.
14.- En el algoritmo cma en el lazo de seguridad este garantiza la convergencia de la curva y que de una buena señal en este intervalo, pero causa que el grafico del error no se pueda evaluar en el intervalo de control, esto debe de ser causado por alguna característica del algoritmo_cma.
15.- Al utilizar el algoritmo cma se presento algo interesante, solo permitía al programa funcionar si el numero de orden del filtro era igual al número de antenas, entonces optamos por escribir un mensaje de advertencia en el programa señalando esta limitación.
16.- se tiene que utilizar un valor bajo como constante de ajuste en el algoritmo cma. (u)cma=5*10^-7.
17.- En algunos casos fue necesario variar los valores de los filtros para ayudar a que converga la señal
18.- En el algoritmo cma con señales senosoidales en algunos casos la señal obtenida fue de casi de la mitad de la señal de referencia, es decir que tendría pérdida de potencia, aunque la señal si converge..

8. Recomendaciones

1.- Como futuras líneas de investigación planteamos que se construya algún circuito eléctrico o antena que simule un sistema de antenas real, también debería investigarse una mejora en el direccionamiento de la antena DOA. Buscar la fórmula adecuada para determinar la distancia a la antena utilizando la potencia, también se podría plantear hacer un algoritmo music y un algoritmo DKP (Back Propagation) .
También podría hacerse un algoritmo NCMA (algoritmo de modulo contante normalizado)
2.- Podrían hacerse algoritmos hibridos lms-dmi, lms-music, u otros algoritmos híbridos que permitan obtener más rápidas convergencias, menor error, y utilizar las mejores características de los algoritmos adaptativos.

9. Referencias

[1] Ahmed El Zooghby, *"Smart Antenna Engineering"*, Artech House, London 2005
[2] The Mathworks Inc., *"SIMULINK Dynamic System Simulation for MATLAB®"*, www.mathworks.com, 1999.
[3] Tapan K Sarkas Michaele Wicks, Magdalena Salazar, Palma and Robert J. *"SMART ANTENNAS"*, 2007
[4] Alexander D. Poularikas and Zayed M. Ramadan , *"Adaptative Filtering Primer With Matlab"*, Taylor y Francis Group, A CRC PRESS BOOK, 2006
[5] SMART ANTENNAS FOR WIRELESS COMUNICATIONS, Liberty, Joseph C, and Rappaport, Theodore S. Prentice Hall PTR, Estados Unidos de America 1999.
[6] ZENG.H.H, L. Relationships Between the Constant Modulus and Wiener receivers. IEEE transactions on Information Theory, (IT-44), New Jersey, p. 1523-1537, 1988.
[7] Jarabo Amores María Pilar, Técnicas de optimización de ingeniería, el algoritmo LMS, p 2-8 Mayo del 2007
[8] Tamer Abdelazim Mellik, Ph.D. demo Algorithm lms, demo Algorithm rls. Central matlab octubre 2003
[9] Dr. Luis M. San José Revuelta, Procesado Adaptativo de señales Universidad de Carabobo p 4-9, Mayo 2008
[10] Departamento de Ingeniería de comunicaciones DICOM, Universidad de Cantabria, Igualación e identificación ciega, Tratamiento avanzado de señal en telecomunicaciones. Grupo de tratamiento avanzado de señal GTAS, p 4-11, p 13-16. 2007
[11] Blight, J. D., Dailey, R.L. and Gangass, D , *"Practical control law design for aircraft using multivariable techniques"*, Internacional Journal of Control, Vol. 59, No 1, 93-137. (1994)

Juan F. Alvarez

Maribel R. Chuez

Ing. Pedro Vargas
Director de Proyecto de graduación