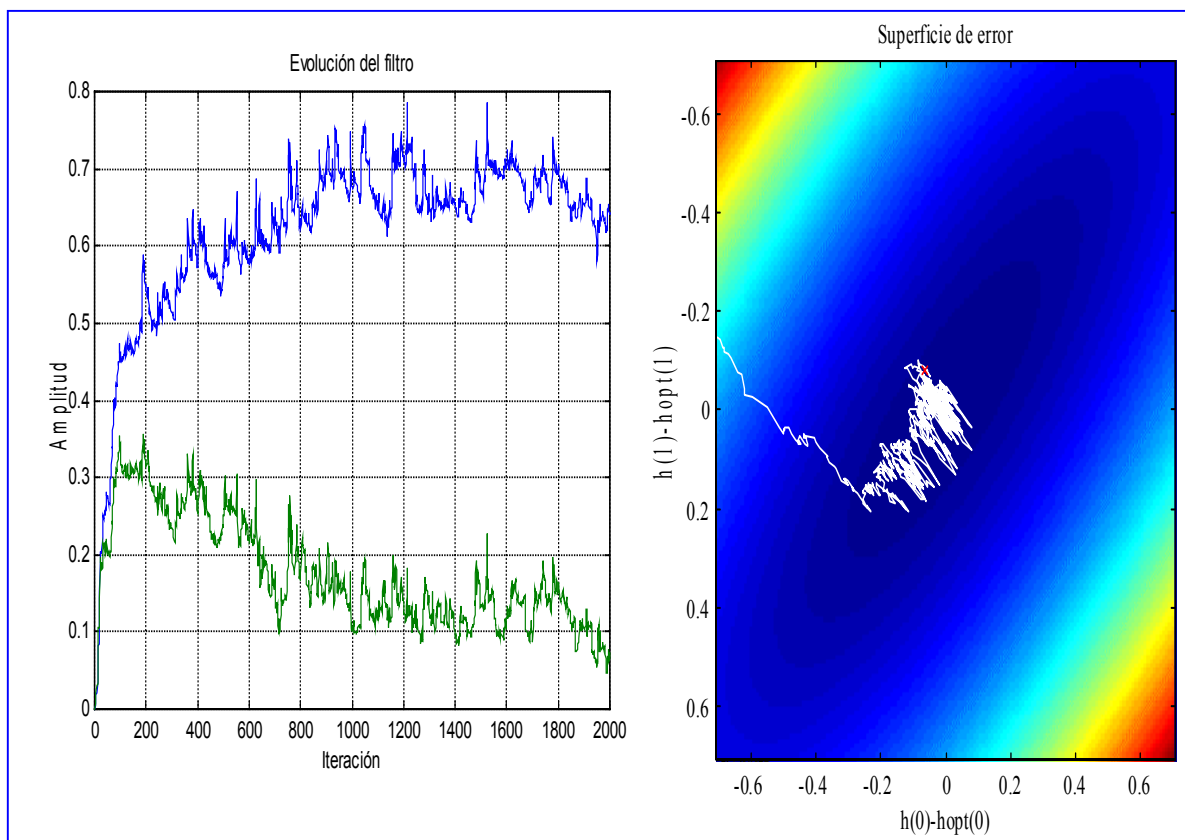


Capítulo V

SISTEMAS ADAPTATIVOS



- V.1. INTRODUCCION
- V.2. MSE Y METODOS DE GRADIENTE
- V.3. DISEÑO Y CONVERGENCIA DE
ALGORITMOS DE GRADIENTE.
- V.4. EL ALGORITMO LMS
- V.5. DSD Y METODOS DE BUSQUEDA
ALEATORIA
- V.6. ALGORITMO RLS
- V.7. FILTRO DE KALMAN
- V.8. CONCLUSIONES
- V.9. EJERCICIOS
- V.10. BIBLIOGRAFIA

V.1 INTRODUCCION.

En la mayor parte de escenarios en los que se requiere tratamiento de la señal los diseños óptimos dan lugar a una degradación de las prestaciones cuando las condiciones del escenario cambian con el tiempo. El avance en mejores diseños y su extremada dependencia con el espectro o la correlación de las señales a tratar, obliga a prevenirse de la degradación de la calidad si las señales a procesar cambian sus propiedades. De este modo, cuando un sistema de tratamiento de señal es capaz de adaptarse automáticamente al entorno de señal, se dice que este es adaptativo. La arquitectura genérica de un sistema adaptativo será una etapa de proceso (habitualmente un sistema lineal) al que se le superpone una estructura de aprendizaje. La estructura de aprendizaje observa las condiciones e introduce las modificaciones pertinentes en el sistema de proceso. Lo que se vera no es mas que la versión digital de los sistemas realimentados en electronica analogica que, al igual de ser los que inspiraron estos sistemas, recientemente dieron lugar a sistemas de codificacion y decodificacion que actuan proximos al limite de Shanon y son denominados como turbo-codigos.

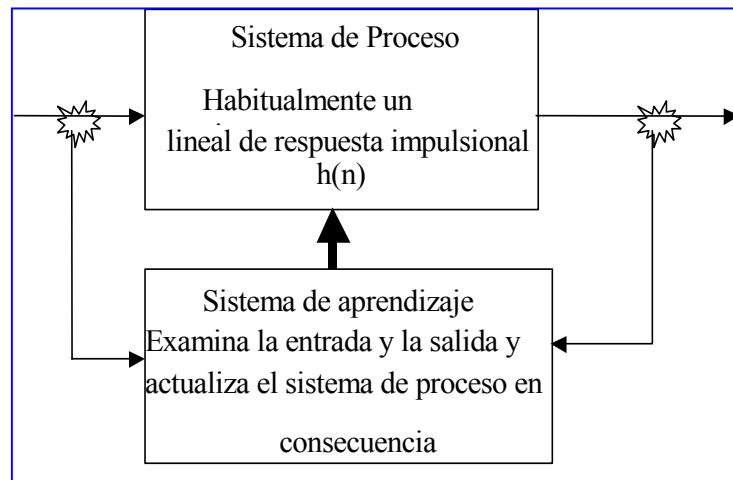


Figura V.1. Arquitectura general de un sistema adaptativo. Se muestra el sistema de proceso y el correspondiente sistema de aprendizaje que, observando entrada y salida, actualiza el sistema que realiza el procesamiento de señal.

El concepto de sistema adaptativo es de uso extendido y de más antigüedad en tratamiento de señal de lo que pudiera parecer. Tal vez, uno de los sistemas adaptativos más antiguos sea el denominado control automático de ganancia de un amplificador. En este caso sencillo, el sistema de proceso es básicamente el amplificador que multiplica la entrada por una determinada ganancia $g(n)$:

$$d(n) = g(n).x(n) \quad (V.1)$$

Como puede observarse, el carácter adaptativo se traduce, habitualmente, en el carácter variante del sistema lineal. En este caso, la ganancia del amplificador obviamente va a cambiar con el tiempo.

En el sistema de aprendizaje se dispone de la señal de referencia $r(n)$, o simplemente de su potencia P_r , basta pues calcular la potencia de la señal de salida del amplificador, según:

$$P_d(n) = \beta.P_d(n-1) + (1-\beta).d^2(n) \quad (V.2)$$

Esta expresión permite seguir como evoluciona la potencia o nivel de señal que el amplificador entrega a lo largo del tiempo. Por lo tanto, esta estimación de la potencia, comparada con la potencia de la referencia, permite generar una señal para corregir la ganancia del amplificador:

$$\varepsilon(n) = P_r - P_d(n) \quad (V.3)$$

Finalmente la regla de aprendizaje queda:

$$g(n+1) = g(n) + \mu \varepsilon(n) \quad (V.4)$$

Siendo μ un parámetro relacionado con el procedimiento o manera de enseñar al amplificador a modificar su ganancia como se podrá ver mas adelante.

El presente tema se dedica a explorar aquellos procedimientos de aprendizaje más usados para sistemas lineales. En concreto, son los métodos de aprendizaje para el diseño de un filtro MSE o de Wiener que mantengan a éste cerca de su carácter óptimo en cualquier condición de la señal de entrada.

Antes de comenzar, es de destacar que el diseño de sistema adaptativo entraña el mismo problema que el diseño de un sistema cualquiera aunque este sea invariante. En un diseño se parte de un sistema inicial y en sucesivos pasos de diseño este se acaba perfeccionando al problema que ha de resolver. Es decir, si se desea diseñar un sistema $h(n)$ a partir de un sistema donde la respuesta impulsional inicial es cero (punto de partida del diseño), el problema se puede enfocar como el denominado diseño bloque a partir de N datos para el filtro MSE, tal y como se ha visto en el capítulo anterior; o bien, y esta es su relación con sistemas adaptativos, en como encontrar la regla de aprendizaje que haga evolucionar o enseñe al sistema inicial no óptimo a hacer su trabajo bien, es decir, a converger a $h(n)$. En definitiva, el diseño de reglas de aprendizaje entraña no sólo que el sistema aprenda a cambiar cuando la señal de entrada cambia sino que además, se autodiseña cuando comienza. Es importante destacar que un sistema adaptativo o dotado de aprendizaje, puede aprender que el mejor diseño que puede conseguir es un sistema invariante por lo que no es del todo correcto asociar a los sistemas lineales adaptativos la idea de que siempre se trata de sistemas variantes. A nivel de ejemplo, lo que se ha tratado de explicar es que si, en el caso del control automático de ganancia, la potencia de entrada de la señal al amplificador es constante, pero diferente de la de referencia, la regla (V.4) hace que el sistema sea variante hasta que la potencia que entrega también lo sea e igual a la de referencia. Una de las cuestiones importantes es saber si este sistema variante, al cabo de un tiempo, alcanzara una ganancia constante óptima. Dicho de otro modo, si se puede trabajar sobre un sistema variante para que este encuentre automáticamente el sistema invariante óptimo.

Lo comentado sobre diseño y autoaprendizaje es crucial para comprender los apartados que siguen. La razón es que, la exposición de los métodos de aprendizaje es más comprensible si se plantea el diseño de un sistema invariante por autoaprendizaje, que si se plantea el diseño de un sistema variante. Todas las reglas o algoritmos válidos para el autodiseño del sistema invariante son válidas, con leves matices, para mantener luego al sistema vigilante de los cambios que se produzcan en las condiciones de diseño y éste realice automáticamente las correcciones necesarias.

Así pues, se va a plantear el problema de diseño adaptativo como la regla que hace que un sistema inicial erróneo converja a la solución óptima MSE. Es decir, cuál es la regla que hace que la respuesta del sistema converja, secuencialmente a partir de los datos, a la solución de Wiener (inversa de la autocorrelación de los datos por el vector de correlación cruzada referencia datos).

El primer grupo o familia esta fuertemente ligado al carácter cuadrático del objetivo MSE y a la existencia de un solo extremo para este tipo de funciones.

V.2. MSE Y METODOS DE GRADIENTE.

Como se ha mencionado en el apartado anterior, la explicación se centrará desde el principio para el diseño de un filtro de Wiener. En primer lugar, dadas la señal referencia $d(n)$, la respuesta impulsional del filtro FIR $\underline{a}_n$, nótese que depende del índice n ya que esta sujeta a un proceso de aprendizaje y por ello cambiara con el tiempo, y el vector de datos $\underline{X}_n$ de la señal de entrada, el MSE en función de los coeficientes del filtro viene dado por:

$$\xi(\underline{a}_n) = E \left\{ \left| d(n) - \underline{a}_n^H \underline{X}_n \right|^2 \right\} \quad (V.5)$$

Esta expresión, puede escribirse en términos de la matriz de correlación de la entrada y del vector de correlación cruzada, ya definidos en el capítulo anterior.

$$\xi(\underline{a}_n) = P_d + \underline{a}_n^H \underline{\underline{R}} \underline{a}_n - \underline{a}_n^H \underline{P} - \underline{P}^H \underline{a}_n \quad (V.6)$$

Los coeficientes óptimos del filtro se obtienen tomando el gradiente e identificándolo al vector cero

$$\frac{\partial \xi(\underline{a}_n)}{\partial \underline{a}_n^H} = \nabla \xi = \underline{\underline{R}} \underline{a}_{opt} - \underline{P} = \underline{0} \quad \Rightarrow \quad \underline{a}_{opt} = \underline{\underline{R}}^{-1} \underline{P} \quad (V.7)$$

lo que permite formular el MSE para cualquier respuesta impulsional en función de la óptima

$$\xi(\underline{a}_n) = \xi_{min} + (\underline{a}_n - \underline{a}_{opt})^H \underline{\underline{R}} (\underline{a}_n - \underline{a}_{opt}) \quad (V.8)$$

donde ξ_{min} es el error mínimo y que se obtiene cuando se utilizan los pesos óptimos.

$$\xi_{min} = P_d - \underline{P}^H \underline{\underline{R}}^{-1} \underline{P} \quad (V.9)$$

La importancia de (V.8) radica en que revela la dependencia cuadrática del criterio de calidad con los coeficientes y que, en consecuencia, existe un solo mínimo. Las curvas de nivel de la superficie definida por la ecuación (V.8) son elipses. Tal y como se puede ver en la Figura V.2, el mecanismo de aprendizaje es el siguiente: para pasar desde un punto $\underline{a}_n$ a otro mejor $\underline{a}_{n+1}$, basta tomar la dirección contraria al gradiente del MSE y moverse una determinada cantidad en la mencionada dirección. Nótese que, de manera deliberada, no se ha indicado si el subíndice n denota iteraciones o muestras, digamos que por el momento más bien indica iteraciones en el proceso de aprendizaje.

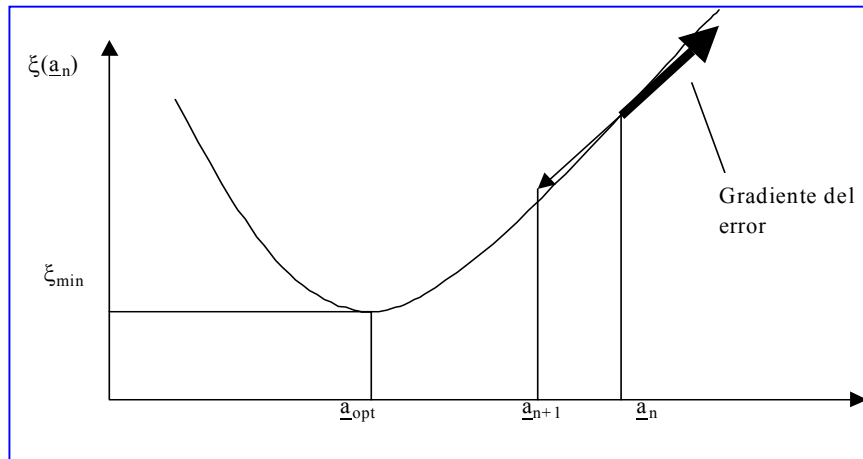


Figura V.2. Curva del MSE en función de los pesos. Su comportamiento cuadrático conlleva que el gradiente cambiado de signo, en cualquier posición, marca la dirección a seguir para alcanzar el mínimo.

Es decir, si en un momento dado el algoritmo se sitúa en el valor $\underline{a}_n$, basta moverse en la dirección contraria al gradiente para tener un filtro de menor MSE o error.

De lo anterior, se deduce fácilmente la regla de aprendizaje

$$\underline{a}_{n+1} = \underline{a}_n - \mu \nabla \xi(\underline{a}_n) \quad (V.10.a)$$

la cantidad μ denominado 'step-size' o paso de adaptación, determina la velocidad de aprendizaje que se desea imprimir al sistema.

El dibujo anterior usa el eje de abscisas para situar vectores. Aunque esta representación es adecuada para la explicación, una representación en dos dimensiones revela que el método del gradiente tan solo denota una dirección de menor error pero, dependiendo de su posición, no necesariamente la

dirección del mínimo (la dirección no apunta al mínimo, véase en la figura V.4). Las curvas de nivel representan las líneas de igual error cuadrático medio. Nótese sólo en el caso de que estas líneas de igual error sean circunferencias, el gradiente en cualquier punto apuntara siempre al mínimo.

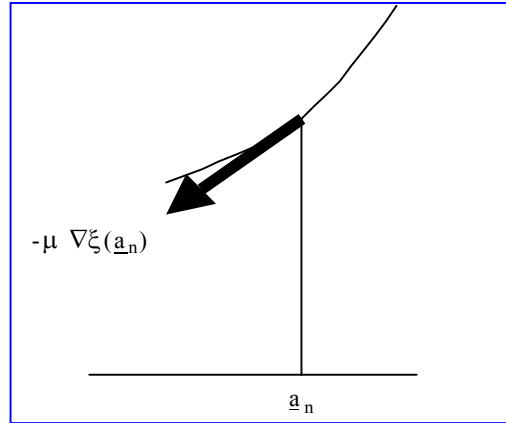


Figura V.3. Dirección del gradiente negativo.

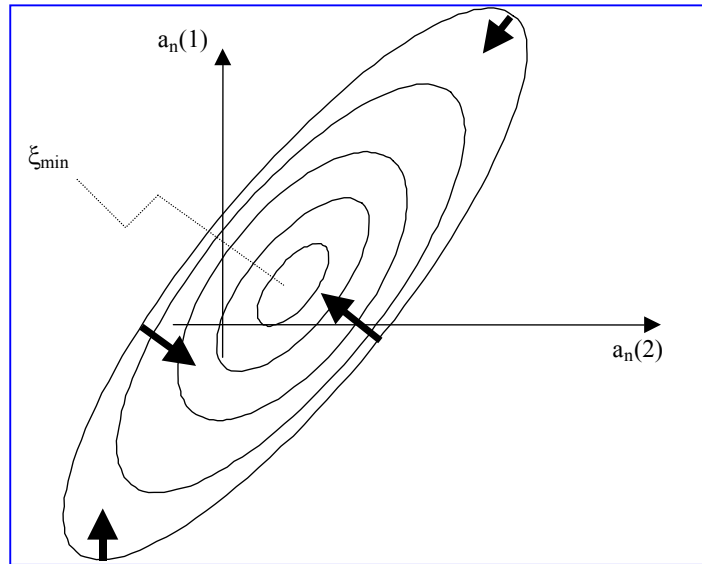


Figura V.4. Curvas de error para el caso de un filtro de dos coeficientes junto con diversos vectores de gradiente donde puede apreciarse la diferente dirección que toman dependiendo del punto (y magnitud que es inversamente proporcional a la separación entre curvas de nivel) si bien todos indican dirección de menor error.

Se comprobó cuál es la relación entre la forma de las curvas de nivel y la matriz de correlación, para lo cual se realiza un cambio de variable sobre (V.8):

$$\xi(\tilde{\underline{a}}_n) = \xi_{min} + \tilde{\underline{a}}_n^H \underline{\underline{R}} \tilde{\underline{a}}_n \quad (\text{V.10.b})$$

Este cambio centrará las curvas de nivel en el valor óptimo $\underline{a}_{opt}$. De este modo, el gradiente de (V.8), en cualquier punto de la superficie, puede expresarse en términos de la nueva variable como:

$$\nabla \xi(\tilde{\underline{a}}_n) = \underline{\underline{R}} \tilde{\underline{a}}_n \quad (\text{V.10.c})$$

En particular, el gradiente en los puntos extremos de los ejes de las curvas de nivel es un vector que pasa por el origen de coordenadas y por lo tanto ha de ser de la forma $k\tilde{\underline{a}}_n$, de lo que se deduce que las direcciones de los ejes principales vienen dadas por los autovectores de $\underline{R}$.

Para relacionar los autovalores de $\underline{R}$ con la superficie de error, se calculara la curvatura de la superficie en las direcciones de los ejes. Si se tiene en cuenta la descomposición en autovalores y autovectores de $\underline{R}$ (véase la ecuación (I.32) del primer capítulo) y se efectúa, de nuevo, un cambio de variable sobre la ecuación de la superficie de error se obtiene, finalmente, (V.10.d).

$$\xi = \xi_{min} + \tilde{\underline{a}}_n^H (\underline{E} \underline{\Lambda} \underline{E}^H) \tilde{\underline{a}}_n = \xi_{min} + \underline{z}_n^H \underline{\Lambda} \underline{z}_n = \xi_{min} + \sum_{i=1}^Q \lambda_i |z_n(i)|^2 \quad (V.10.d)$$

Los ejes de las curvas de error en las nuevas coordenadas $\underline{z}_n$ quedan alineados con los ejes de representación (figura V.5). La curvatura buscada se obtiene de calcular la segunda derivada de la función de error respecto a $z_n(i)$, siendo igual a $2\lambda_i$. Así pues, la dirección del eje corto (véase la Figura V.5) estará asociada al autovector de mayor autovalor, y viceversa. En definitiva, la excentricidad de las curvas de nivel de la función de error dependerá de cuan distintos sean los autovalores de la matriz de correlación. El impacto de esta observación en la velocidad de convergencia se analizará con más precisión en el próximo apartado.

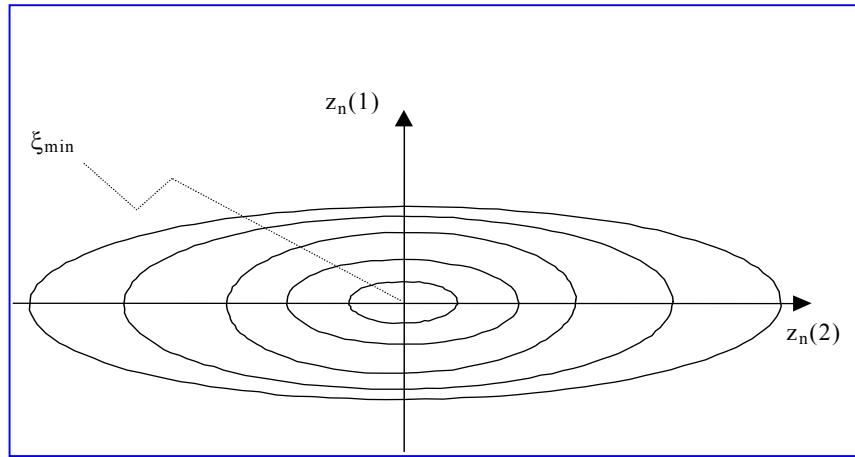


Figura V.5. Curvas de error de la superficie de la figura V.4, representadas sobre las nuevas variables $\underline{z}_n = \underline{E}^H (\underline{a}_n - \underline{a}_{opt})$. Los ejes principales, autovectores de la matriz de autocorrelación de los datos, de las elipses quedan alineados con los ejes de representación.

V.3. DISEÑO Y CONVERGENCIA DE ALGORITMOS DE GRADIENTE.

Los algoritmos denominados de gradiente emplean el gradiente del MSE para proporcionar la actualización de los coeficientes. Dicho gradiente es fácil de obtener de la expresión del MSE y resulta ser:

$$\underline{a}_{n+1} = \underline{a}_n - \mu \nabla \xi_n = \underline{a}_n - \mu (\underline{R} \underline{a}_n - \underline{P}) \quad (V.11)$$

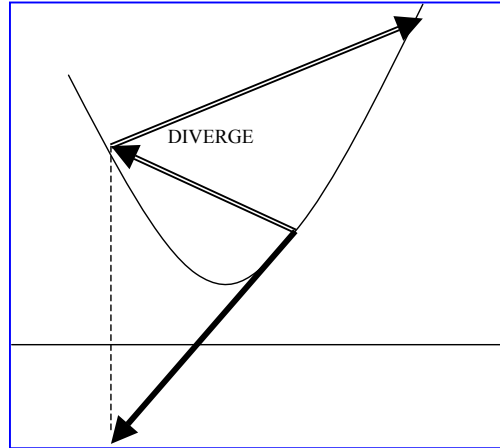


Figura V.6. Cuando el paso de adaptación es muy grande el algoritmo diverge.

Al tratar de diseñar cual es la selección adecuada del parámetro μ (que controlará la velocidad de aprendizaje o convergencia), y antes de pasar a su análisis formal, es interesante notar que existe una velocidad máxima de convergencia o valor máximo del parámetro para que el algoritmo no diverja. Como puede verse en la Figura V.6, si se avanza demasiado en la dirección del gradiente negativo, al margen de pasar al otro ‘lado’ de la curva, se podría llegar a un punto de mayor error (o ignorancia de los parámetros óptimos) que en el que se estaba, con lo que, en lugar de aprender, el sistema habrá perdido conocimiento y sucesivamente se llegará a valores de coeficientes más alejados de los óptimos y también de mayor error. En definitiva, el algoritmo diverge.

Para analizar esta divergencia con detalle, es necesario resolver la ecuación en diferencias que expresa la regla de adaptación al escribirla como

$$\Delta \underline{a}_n = \underline{a}_{n+1} - \underline{a}_n = -\mu \underline{R} \underline{a}_n + \mu \underline{P} \quad (\text{V.12})$$

Esta ecuación tiene, al igual que en circuitos RLC, un régimen transitorio, proporcionado por la ecuación homogénea, y un permanente, proporcionado por la solución particular. Esta última, se calcula bajo la condición de diferencia finita igual a cero. Al igualar a cero el termino de la derecha de la ecuación (V.12), se obtiene el permanente que prueba que el sistema tiene como tal la solución óptima.

$$-\mu \underline{R} \underline{a}_p + \mu \underline{P} = 0 \quad \Rightarrow \quad \underline{a}_p = \underline{R}^{-1} \underline{P} = \underline{a}_{opt} \quad (\text{V.13})$$

Respecto a la solución transitoria, ésta ha de calcularse de la ecuación homogénea. Dicha ecuación se obtiene de anular la excitación o termino que no depende del vector de coeficientes,

$$\Delta \underline{a}_n = -\mu \underline{R} \underline{a}_n \quad (\text{V.14})$$

Las frecuencias propias, o modos propios, de la ecuación se obtienen de probar una función exponencial (entendida como el producto componente a componente del vector $\underline{z}_i$ consigo mismo, de este modo, $\underline{z}_i^m$ ha de entenderse como el producto del escalar z_i^m por el vector $\underline{z}_i$):

$$\underline{z}_i^{m+1} - \underline{z}_i^m = -\mu \underline{R} \underline{z}_i^m \quad \Rightarrow \quad -\frac{(z_i - 1)}{\mu} \underline{z}_i = \underline{R} \underline{z}_i \quad (\text{V.15})$$

De esta última ecuación se desprende que $(z_i - 1)/\mu$ es un autovalor de la matriz de autocorrelación y el vector $\underline{z}_i$ es el autovector correspondiente. De este modo, las frecuencias propias son iguales a los Q autovalores λ_i ($i=1, \dots, Q$) de la matriz de autocorrelación de los datos, para una longitud del filtro igual a Q. En resumen, la solución a la ecuación de aprendizaje viene dada por (V.16), que es la suma de la solución transitoria más la solución particular, encontrada anteriormente. Las Q constantes α_i dependen de las condiciones iniciales o respuesta del filtro en el origen o comienzo del proceso de aprendizaje $\underline{a}_0$:

$$\underline{a}_n = \sum_{i=1}^Q (1 - \mu \lambda_i)^n \underline{z}_i \alpha_i + \underline{a}_{opt} \quad (V.16)$$

Forzando la solución inicial, la ecuación definitiva para la evolución de los coeficientes, que la regla del gradiente produce en los coeficientes del filtro, viene dada por (V.17). Para obtener esta expresión se ha usado la dependencia de una matriz en sus autovalores y autovectores (véase Capítulo I).

$$\underline{a}_n = (\underline{I} - \mu \underline{R})^n (\underline{a}_0 - \underline{a}_{opt}) + \underline{a}_{opt} \quad (V.17)$$

De esta expresión, o de la expresión de las frecuencias propias puede deducirse bajo qué condición la evolución es estable, es decir, el transitorio tiende a cero y la solución adaptativa converge a la óptima. Dicha condición viene dada por (V.18). El lector puede ver que en este caso se fija que las raíces estén dentro del círculo unidad, al igual que en un sistema analógico se fijaban al semiplano izquierdo.

$$|1 - \mu \lambda_i| < 1 \quad \forall i = 1, \dots, Q \quad (V.18)$$

Esta condición habría de verificarse para cada uno de los autovalores pero, dado que la matriz de autocorrelación es definida positiva, sus autovalores son positivos y ordenados, en consecuencia si la condición anterior se verifica para el autovalor máximo se verificara para todos los demás. Además, como el paso de adaptación ha de ser positivo, para no cambiar la dirección del gradiente, se llega fácilmente a la condición de convergencia:

$$\mu < \frac{2}{\lambda_{max}} \quad (V.19)$$

Es decir, el paso de adaptación viene limitado por el doble de la inversa del autovalor máximo de la matriz de autocorrelación.

Una cota más interesante y de más fácil cálculo para el paso de adaptación se obtiene del hecho de que (ver Capítulo I) la traza de una matriz es igual a la suma de sus autovalores. Como en el caso de matriz de autocorrelación todos los autovalores son positivos, como ya se ha comentado, puede concluirse que la traza de la matriz (suma de los autovalores) es mayor que su autovalor máximo. Por lo anterior, una cota segura del parámetro μ que siempre garantiza convergencia viene dada por:

$$\mu < \frac{2}{\text{Traza}(\underline{R})} < \frac{2}{\lambda_{max}} \quad (V.20)$$

Como quiera que la traza es la suma de los valores de la diagonal principal, la traza de la matriz de autocorrelación de $Q \times Q$ es Q veces la autocorrelación de cero, es decir, Q veces la potencia de la señal de entrada. Para garantizar que se está en convergencia, el valor seleccionado para el paso de adaptación vendrá dado por (V.21) donde el parámetro α estará siempre comprendido entre cero y uno. Su valor preciso e impacto correspondiente se determinará más adelante. Esta expresión es del todo viable ya que depende de la potencia de la señal entrada, igual que en el sistema de control automático de ganancia, expuesto al comienzo del presente tema.

$$\mu = \frac{2\alpha}{P_x(n)} \quad (V.21)$$

Como puede verse la potencia del denominador se ha previsto variante con el tiempo. De hecho su estimación (con una memoria aproximada de $1/(1-\beta)$ muestras) vendría dada por:

$$P_x(n+1) = \beta \cdot P_x(n) + (1 - \beta) \cdot \underline{X}_n^H \cdot \underline{X}_n \quad (V.22)$$

El lector puede comprobar que el estimador anterior tiene como valor esperado la traza de la matriz de autocorrelación. La elección de la memoria con que se estima la potencia viene dictada por la situación practica y el tiempo que se requiere para que el algoritmo adapte su paso de adaptación al escenario. Es de señalar que, en la practica, se acostumbra a poner un umbral mínimo P_{xo} para el estimador de potencia, pues si, por accidente, el sistema se queda sin señal de entrada o esta se hiciera en algún momento muy pequeña, el paso de adaptación crecería mucho y los pesos también produciendo eventualmente saturación en la dinámica proporcionada por el procesador de los coeficientes. Por todo lo anterior, en la practica, se prefiere (V.23).

$$P_x(n+1) = \begin{cases} \beta \cdot P_x(n) + (1-\beta) \cdot \underline{X}_n^H \cdot \underline{X}_n & \text{si } P_x(n+1) > P_{xo} \\ P_{xo} & \text{en caso contrario} \end{cases} \quad (V.23)$$

En ocasiones y con la intención de ahorrar el máximo de operaciones, la expresión (V.21) se realiza mediante tabla, con entrada igual a la envolvente de la señal de entrada y salida la μ más adecuada. Así mismo, se puede redondear la potencia a potencias de dos y así permitir el cálculo de la μ por desplazamientos en el registro que contiene el parámetro α .

Una vez determinado el cálculo preciso del paso de adaptación para garantizar la convergencia del algoritmo, se hace necesario determinar cuantas iteraciones o actualizaciones serán necesarias para alcanzar la convergencia. Como en cualquier sistema físico modelable con ecuaciones diferenciales lineales, la constante de tiempo efectiva del sistema viene determinada por la constante de tiempo más larga, es decir, la asociada a la respuesta transitoria que más tiempo tarda en atenuarse. A la vista de (V.16), y teniendo presente que todos los autovalores son positivos, la constante de tiempo vendrá determinada por el numero de iteraciones necesarias para que el término correspondiente al mínimo autovalor haya, prácticamente, desaparecido. El criterio habitual es considerar que un termino influye poco en la evolución cuando se ha reducido a un décimo de su valor inicial. Con este criterio, el numero de iteraciones N_c necesarias para converger será aquel que verifique:

$$(1 - \mu \lambda_{min})^{N_c} = 0,1 \quad (V.24)$$

Que, despejando, pasa a ser (V.25).

$$N_c = \frac{2,30}{-Ln(1 - \mu \lambda_{min})} \quad (V.25)$$

Sustituyendo la expresión del paso de adaptación en función del autovalor máximo y teniendo en cuenta que, en este análisis, el denominado ‘eigenvalue spread’ (dispersión de autovalores) es mucho mayor que la unidad, se puede usar el primer termino del desarrollo de Taylor para obtener la expresión:

$$N_c = \frac{2,30}{-Ln\left(1 - 2\alpha \frac{\lambda_{min}}{\lambda_{max}}\right)} \approx \frac{2,30}{2\alpha} \frac{\lambda_{max}}{\lambda_{min}} \quad (V.26)$$

Antes de proseguir con una expresión más próxima a los datos o señal de entrada del tiempo de convergencia, la expresión anterior proporciona detalles muy interesantes relativos al funcionamiento del aprendizaje mediante el gradiente. Al observar que el autovalor mínimo limita la convergencia, es decir, los modos débiles asociados a autovalores pequeños tardan más en converger que los fuertes, se puede concluir que el filtro aprende más rápido a realizar su trabajo con las zonas frecuenciales o modos de alta energía, dejando para el final aquellas de más baja energía. Recuerde que los autovalores de una matriz, cuando su dimensión tiende a infinito, tienden a ser las muestras del espectro de potencia de la señal de entrada. Dicho de otro modo, si se diseña un ecualizador, el sistema empezara a ecualizar correctamente las zonas donde la señal de entrada trae mucha energía y dejara para el final las zonas de poca o muy pequeña energía. Este comportamiento de los algoritmos de gradiente se comprende gráficamente si se recuerda que, en el caso de dos dimensiones, el eje corto es el que corresponde al mayor autovalor y el eje

largo al del autovalor más pequeño. Reproduciendo en parte la Figura V.4, la Figura V.7 muestra como en el eje corto (autovalor máximo), con una iteración prácticamente ya se llega al mínimo; mientras que, en el eje largo, aun quedan más iteraciones para converger.

Volviendo a la interpretación práctica de la expresión del tiempo de convergencia, como se ha comentado el autovalor máximo se puede estimar como Q veces la potencia de la señal de entrada. Esta potencia para señal y ruido incorrelados será la suma de ambos, es decir, P_x será la suma de la potencia de señal útil P_s más la de ruido P_w . Con respecto al autovalor mínimo, éste puede estimarse interpretando los autovalores como muestras de la densidad espectral, luego el mínimo autovalor puede estimarse como el mínimo de la densidad espectral de la entrada. Si el ruido que contiene la señal de entrada es blanco, el autovalor mínimo estará muy próximo a la densidad espectral de este P_w/B_t , siendo B_t el ancho de banda de la señal de entrada, en un sistema muestreado la inversa del periodo de muestreo. En definitiva, una estimación practica del tiempo de convergencia vendría dada por:

$$TN_c = \text{Tiempo de convergencia} = \frac{2,30}{2\alpha} Q \frac{P_s}{P_w} = \frac{2,30}{2\alpha} Q \text{ SNR} \quad (\text{V.27})$$

En consecuencia, el orden del filtro y una densidad espectral con una gran distancia entre su máximo y su mínimo prolongaran los tiempos de convergencia o aprendizaje de los métodos de gradiente.

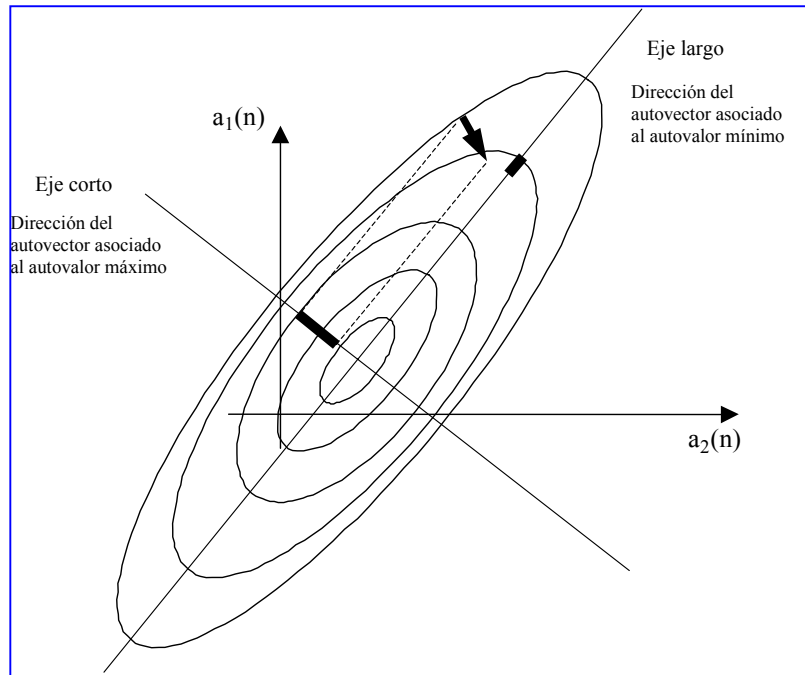


Figura V.7. Una iteración de gradiente muestra como la distancia al mínimo se alcanza según el eje del autovector máximo rápidamente mientras que en el otro eje, autovector mínimo, aun esta lejos de aproximarse al mínimo.

V.4 EL ALGORITMO LMS.

Al tratarse de una implementación del método de gradiente, en primer lugar, se dejara de hablar de iteraciones para el índice n . Ahora el índice n recobra su papel habitual, como índice temporal de la entrada $x(n)$, y se actualiza el vector de datos $\underline{X}_n$ también de la manera habitual.

Como todo procedimiento practico de gradiente, el LMS (Least Mean Square algorithm, debido a B. Widrow) toma una estimación del gradiente teórico, que se reproduce a continuación para el instante n .

$$\underline{\nabla} \xi(n) = \underline{R} \underline{a}_n - \underline{P} = E \{ \underline{X}_n \underline{X}_n^H \} \underline{a}_n - E \{ d^*(n) \underline{X}_n \} \quad (\text{V.28})$$

Esta expresión revela el carácter teórico de esta regla de aprendizaje. El LMS estima el gradiente de un modo aparentemente grosero pero de una sencillez y calidad espectaculares. De hecho el LMS aproxima los valores esperados de la formula anterior por sus valores instantáneos. Al tomar esta estimación y teniendo en cuenta que la salida del filtro es:

$$y(n) = \underline{a}^H \underline{X}_n \quad (V.29)$$

Se obtiene la siguiente regla de adaptación:

$$\underline{a}_{n+1} = \underline{a}_n + \mu \underline{X}_n (d^*(n) - y^*(n)) = \underline{a}_n + \mu \underline{X}_n \varepsilon^*(n) \quad (V.30)$$

Donde $\varepsilon(n)$ es el error o señal diferencia entre la referencia y la salida del filtro. Un esquema del LMS se presenta en la Figura V.8 donde puede apreciarse su simplicidad.

Todo lo expresado para el método de gradiente es válido, en términos de valores esperados o medios, para el LMS. De hecho, es fácil mostrar la validez de esta afirmación comprobando, de nuevo cuál sería el valor del μ más adecuado. Nótese que si se denomina $\varepsilon(n)$ al error que producen los pesos $\underline{a}_n$ con el vector de datos $\underline{X}_n$, se puede calcular cual es el error $\varepsilon_o(n)$ que producen, con los mismos datos, los nuevos coeficientes. La condición de convergencia es que dicho error, en potencia, ha de ser menor que el anterior. Pasando a expresar lo anterior, se obtiene (V.31).

$$\varepsilon_o(n) = d(n) - \underline{a}_{n+1}^H \underline{X}_n = d(n) - (\underline{a}_n^H - \mu \underline{X}_n^H \varepsilon(n)) \underline{X}_n = \varepsilon(n) (1 - \mu \underline{X}_n^H \underline{X}_n) \quad (V.31)$$

Con lo que la condición resulta ser:

$$\mu < \frac{2}{\underline{X}_n^H \underline{X}_n} \quad (V.32)$$

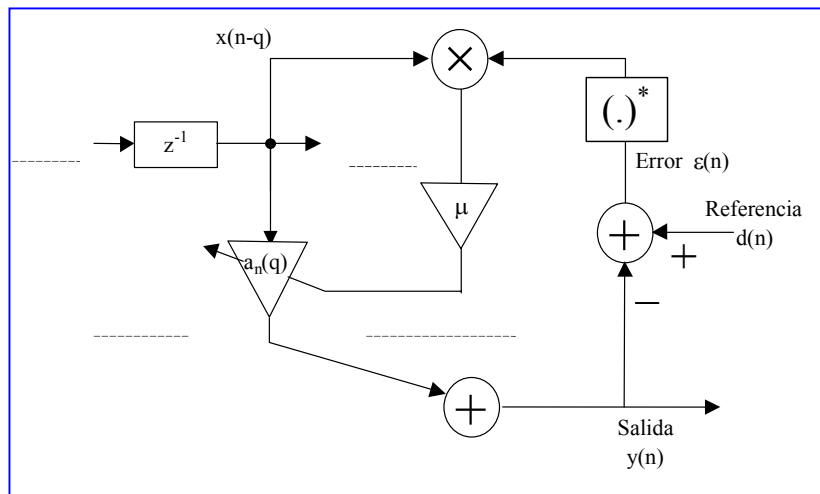


Figura V.8. Detalle de la implementación del LMS para el peso q del filtro. Cada peso del FIR se actualiza con una fracción del producto (o mezcla) de la muestra de entrada por el conjugado del error.

En esta expresión puede reconocerse el denominador que no es más que el valor instantáneo de Q veces la potencia de la señal de entrada. Por supuesto, el valor instantáneo haría fluctuar excesivamente el paso de adaptación y se toma precisamente el promedio que se propuso en el apartado anterior.

Pasando al adecuado diseño del parámetro α del paso de adaptación, éste ha de abordarse bajo el principio de que el LMS usa una variable aleatoria, y no el gradiente determinístico, en la actualización de los pesos. Este carácter aleatorio impide que el aprendizaje finalice. Incluso cuando el algoritmo

alcanzase el mínimo, se puede decir que permanecería nervioso, despierto o deseoso de aprender más, moviéndose alrededor del óptimo por si acaso éste cambiase. El efecto es que los coeficientes del filtro pasan a ser una variable aleatoria cuya media es el filtro óptimo y cuya matriz de covarianza es necesario determinar:

$$\begin{aligned} E\{\underline{a}_n\} &= \underline{a}_{opt} \\ \underline{\Sigma}_a &= E\{\tilde{\underline{a}}_n \tilde{\underline{a}}_n^H\} \\ \text{siendo } \tilde{\underline{a}}_n &= \underline{a}_n - \underline{a}_{opt} \text{ el denominado error de coeficientes} \end{aligned} \quad (V.33)$$

Para calcular la varianza del error de coeficientes, de la ecuación de actualización de coeficientes en el LMS se resta en ambos lados el vector óptimo, con lo que se obtiene (V.34).

$$\underline{a}_{n+1} = \underline{a}_n - \mu \varepsilon^*(n) \underline{X}_n \quad (V.34)$$

Antes de proseguir, se descompondrá el error entre la salida y la referencia en dos términos, terminos que aparecen de considerar la referencia compuesta de dos contribuciones diferentes: La primera es aquella parte de la referencia que no se puede cancelar con la entrada y viceversa que denominaremos $w(n)$; y la segunda, que es aquella que con los pesos óptimos se suprimiría. Luego:

$$d(n) = w(n) + \underline{a}_{opt}^H \underline{X}_n \quad (V.35)$$

Así pues, el error será (destacándose que la potencia de $w(n)$ es precisamente el MSE mínimo ξ_{min}):

$$\varepsilon(n) = w(n) - \underline{a}_n^H \underline{X}_n \quad (V.36)$$

Al usar esta última expresión en (V.34) y multiplicando a ambos lados por su transpuesto conjugado se obtiene la expresión siguiente:

$$\begin{aligned} \underline{a}_{n+1} &= \underline{a}_n + \mu \underline{X}_n (w^*(n) - \underline{X}_n^H \underline{a}_n) \\ \underline{a}_{n+1} \underline{a}_{n+1}^H &= [\underline{a}_n + \mu \underline{X}_n (w^*(n) - \underline{X}_n^H \underline{a}_n)] [\underline{a}_n^H + \mu (w(n) - \underline{a}_n^H \underline{X}_n) \underline{X}_n^H] \end{aligned}$$

Además, al tomar el valor esperado, asumiendo que $w(n)$ es independiente del resto de variables aleatorias y que los coeficientes del filtro son independientes de las muestras de la señal de entrada. Notese que al estar proximos a convergencia el error sera octogonal a los datos. En definitiva, se obtiene:

$$\underline{\Sigma}_a^{n+1} = [\underline{I} - \mu \underline{R}] \underline{\Sigma}_a^n [\underline{I} - \mu \underline{R}] + \mu^2 \xi_{min} \underline{R} \quad (V.37)$$

Llegados a este punto, si se considera el régimen permanente del algoritmo se puede suponer que la covarianza de los coeficientes se ha estabilizado y no depende de n , con lo cual se obtiene:

$$0 = \mu \xi_{min} \underline{I} - \underline{R} \underline{\Sigma}_a \underline{R}^{-1} - \underline{\Sigma}_a + \mu \underline{R} \underline{\Sigma}_a \quad (V.38)$$

Finalmente, teniendo en cuenta que el paso de adaptación se escogerá habitualmente de forma que $\mu \ll 1/\lambda_{max}$, el último término de la ecuación (V.38) desaparece. Es fácil comprobar que en estas condiciones la expresión para la covarianza de los coeficientes:

$$\underline{\Sigma}_a \cong \frac{\mu}{2} \xi_{min} \underline{I} \quad (V.39)$$

Esta es precisamente la solución de la ecuación V.38. Antes de proseguir, nótese que esta expresión revela que, para valores del paso de adaptación lejos de la cota superior (para que no diverja), los coeficientes evolucionan independientemente unos de otros ya que la matriz de covarianza es

diagonal. Obviamente, esto no ocurre al comienzo del aprendizaje por no darse las condiciones en que esta expresión se ha derivado. En términos de inteligencia artificial y considerando a los coeficientes como neuronas, al comienzo estas colaboran y cuando el aprendizaje ha terminado y están en proceso de espera de cambios para retomar el proceso estas evolucionan, alrededor de su valor, de manera independiente, eso sí, pendientes de los cambios que pudieran producirse en el escenario para comenzar de nuevo su periodo de aprendizaje o colaborativo. Se puede decir, que en el proceso de mejora del conocimiento, la búsqueda de nueva información es independiente; mientras que, la asimilación de este es un proceso colaborativo.

Con lo anterior queda demostrado que el sistema adaptativo, bajo LMS, va a mostrar una fluctuación a su salida, que el usuario de ésta percibirá como ruido y que es debida a que el filtro no presenta un ajuste perfecto a los coeficientes óptimos, dado el carácter aleatorio del instrumento de aprendizaje que es el gradiente instantáneo. La cuestión es que este ruido de desajuste, en el caso de ecualización para sistemas de comunicaciones, hará disminuir la SNR a la salida del ecualizador y podría afectar seriamente a la tasa de error del sistema. De hecho, este ruido de desajuste provocara que la potencia del error sea siempre, en media, superior al mínimo. Este efecto se hace evidente escribiendo la ecuación del MSE en función del error de coeficientes:

$$\xi(n) = \xi_{min} + \tilde{\underline{a}}_n^H \underline{R} \tilde{\underline{a}}_n \quad (V.40)$$

Como puede verse la potencia del error se ha convertido en una variable aleatoria. Al tomar su valor esperado y teniendo en cuenta que (según V.39):

$$E\{\tilde{\underline{a}}_n^H \underline{R} \tilde{\underline{a}}_n\} = \text{Traza}[\underline{\Sigma}_a \underline{R}] = \frac{\mu}{2} \xi_{min} \text{Traza}(\underline{R}) \quad (V.41)$$

Se puede pues concluir que el exceso de error, llamado también desajuste, y definido como se indica en (V.42),

$$\aleph \equiv \frac{E\{\xi(n)\} - \xi_{min}}{\xi_{min}} 100 \% \quad (V.42)$$

pasa a ser

$$\aleph = \frac{\mu}{2} \text{Traza}(\underline{R}) \quad (V.43)$$

En definitiva, si se toma (V.44),

$$\mu = \frac{2\alpha}{P_x(n)} \quad (V.44)$$

el ruido de desajuste será de $\alpha\%$. En forma numérica, un alfa de 0.1 equivale a tener un exceso de error o ruido de desajuste del 10%. Existe por tanto un compromiso entre velocidad de adaptación y nivel de desajuste. Esto completa el diseño del LMS.

El contenido de este apartado demuestra la manera correcta de proceder ante cualquier otra regla de aprendizaje, esté basada en el gradiente o no. Vale pues la pena recordar brevemente los pasos seguidos. En primer lugar se ha de probar la convergencia al óptimo o diseño deseado y comenzar la asignación pertinente de parámetros para que esto sea así. En segundo lugar se ha de analizar la velocidad de convergencia, tanto sobre una colección limitada de datos, iteraciones, como en caso contrario sobre vectores de datos ilimitado. Por último, en fase de seguimiento se ha de determinar el ruido de desajuste, o lo que es lo mismo su velocidad de seguimiento o capacidad de reacción a cambios. Es también destacable la orientación, en términos de sistema de aprendizaje, en todas las fases mencionadas. Nótese que altas velocidades de convergencia, o aprendizaje rápido, llevan asociados desajustes elevados. También se puede interpretar que con pasos de adaptación altos se aprende mucho pero mal pues el sistema ‘medita’ poco sobre lo aprendido y esa es la razón de su nerviosismo en seguimiento. En

resumen, esta más ávido por aprender que por sedimentar lo aprendido. El compromiso entre desajuste y velocidad de convergencia puede apreciarse en la Figura V.9.

El apartado que sigue analiza otros métodos de gradiente que muestran que en la base del LMS está una manera burda pero eficiente de estimar el gradiente.

V.5 EL DSD Y METODOS DE BUSQUEDA ALEATORIA.

Como ya se ha mencionado el LMS estima el gradiente del MSE mediante su valor instantáneo, en este apartado se presentaran otros métodos alternativos a la estimación instantánea de la derivada del MSE con respecto a los pesos.

El DSD (Differential Steepest Descent) es un algoritmo de la familia de gradiente que calcula el gradiente de forma independiente para cada coeficiente del filtro. Para ello, e imaginando que se esta en el instante n en una valor del MSE igual a ξ_n , se toma uno de los pesos $a_n(q)$ ($q=0, \dots, Q-1$) y se le añade una perturbación δ , a continuación se mide, usando M vectores de datos cuanto vale el MSE $\xi_n(\delta)$. Procediendo del mismo modo con una perturbación $-\delta$ se obtiene el nuevo error $\xi_n(-\delta)$. Llegado este punto, y después de $2M$ vectores de datos, se dispone de lo necesario para estimar una de las componentes del vector gradiente como sigue:

$$\nabla_n(q) = \frac{\xi_n(q, \delta) - \xi_n(q, -\delta)}{2\delta} \quad (V.45)$$

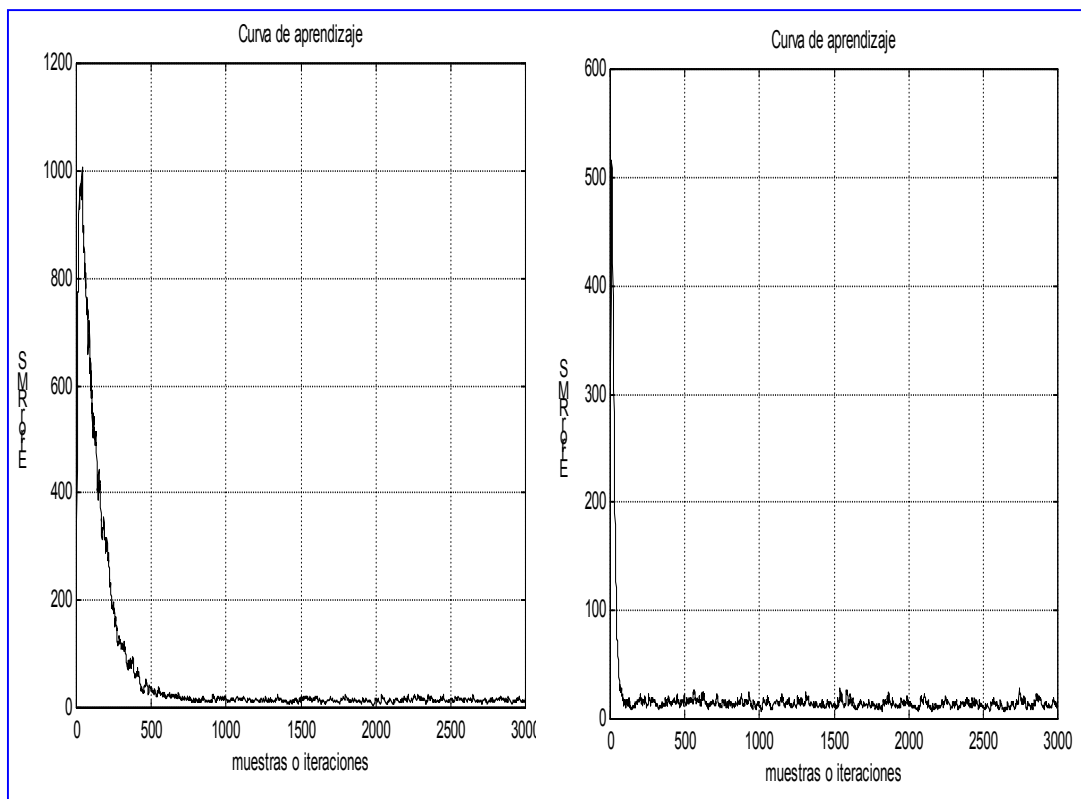


Figura V.9. Curva de aprendizaje o potencia del error para el algoritmo LMS con 11 coeficientes, para dos valores diferentes del parámetro μ . A la izquierda el parámetro es diez veces mayor que a la derecha. Nótese la mayor velocidad de convergencia a la par que un mayor desajuste a la derecha que a la izquierda.

Nótese que con pesos complejos (componentes i-q en comunicaciones banda trasladada) el numero de variables a perturbar es igual a doble del numero de coeficientes. Este proceso, repetido para las Q componentes del gradiente en parte real e imaginaria, permite obtener una estimación de gradiente para ser usada en la regla de adaptación:

$$\underline{a}(n+1) = \underline{a}(n) - \frac{\mu}{2} \nabla_n \quad (\text{V.46})$$

En indudable que el DSD utiliza una estimación del gradiente mucho mejor que el instantáneo del LMS, el problema radica en que se requieren $2MQ$ vectores de datos para una sola iteración. Los criterios de convergencia, velocidad y dependencia de ésta con la dispersión de autovalores de la matriz de correlación (eigenvalue spread) se mantienen, siempre teniendo en cuenta que cada iteración o adaptación requiere muchos más datos que en el LMS. Así pues, el DSD será más lento que el LMS y, salvo por el desajuste, la ventaja que se puede apreciar es que no requiere del vector de datos en la estimación del gradiente ni para realizar la iteración. La Figura V.10 muestra, para un filtro de un solo coeficiente real, cómo se lleva a cabo la estimación del gradiente.

Esta forma de estimar el gradiente origina en si un exceso de error, indicado en la Figura V.10. Cuando el algoritmo pasa a implementar los pesos $a_n(q)$, en realidad nunca se encuentra en ellos, es decir, al mismo tiempo que se filtran vectores de datos, el algoritmo está en todo momento implementando las perturbaciones para medir el siguiente gradiente. Esto quiere decir que, aunque nominalmente los pesos son el vector $\underline{a}_n$ el filtro nunca los utiliza, sino que utiliza los valores perturbados a fin de evaluar el siguiente gradiente. Este proceso constante de perturbación hace que realmente el sistema se encuentre, a nivel de error o calidad, entre los dos valores perturbados del MSE y en media en el valor indicado en la gráfica.

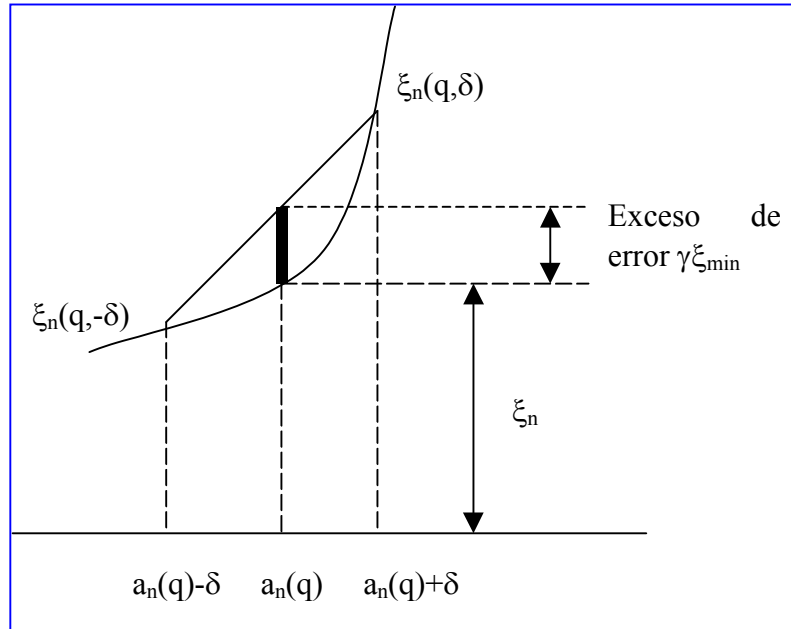


Figura V.10. Estimación del gradiente en el algoritmo DSD. Se muestra el exceso de error que la propia forma de estimar el gradiente original.

Para su cálculo, se desarrollara en serie de Taylor, hasta su tercer termino, los dos valores del MSE con perturbación:

$$\begin{aligned} \xi_n(q, -\delta) &= \xi_n - \xi'_n \delta + \xi''_n \frac{\delta^2}{2} \\ \xi_n(q, \delta) &= \xi_n + \xi'_n \delta + \xi''_n \frac{\delta^2}{2} \end{aligned} \quad (\text{V.47})$$

En base a esta expresión, la media sería directamente el doble del último termino. Como el exceso de error, de cara al cálculo del desajuste final, se normaliza con respecto al error mínimo, se tendrá (V.48). En esta expresión se ha utilizado que la derivada segunda del MSE con respecto al coeficiente es el valor de la autocorrelación en el origen:

$$\gamma = \frac{\xi_n'' \delta^2}{\xi_{min}} = \frac{\delta^2 r_x(0)}{\xi_{min}} \quad (V.48)$$

Por último, (V.48) es el exceso de error debido a la estimación del gradiente con respecto al coeficiente q , como el algoritmo realiza secuencialmente el mismo proceso para todos los coeficientes, el valor a tomar será el promedio de los Q excesos de error. De este modo, el exceso de error motivado por el procedimiento de estimación es:

$$\gamma = \frac{\delta^2 \text{Traza}(\underline{\underline{R}})}{Q \xi_{min}} \quad (V.49)$$

Además del exceso de error provocado por la permanente perturbación de los pesos, esta el tradicional, ya mostrado para el LMS, exceso de error debido a que la estimación del gradiente es una variable aleatoria. De hecho esta variable aleatoria proviene de que cada MSE es estimado con M vectores de datos.

$$\xi_n(q, \delta) = \frac{1}{M} \sum_n^{n+M-1} |\varepsilon(n)|^2 \quad (V.50)$$

Por lo tanto, asumiendo que el algoritmo ya ha convergido y se encuentra en seguimiento con error próximo al error mínimo, la media del estimador de gradiente será cero y su varianza (V.51).

$$E\left\{|\nabla_n|^2\right\} = \frac{E\left\{\xi_n^2(\delta)\right\} + E\left\{\xi_n^2(-\delta)\right\}}{4\delta^2} \quad (V.51)$$

Donde se ha omitido el índice q por comodidad en la presentación, y se ha asumido que los errores son independientes en cada instante n .

Como los sucesivos valores del error de la salida con la referencia en todo momento se consideran independientes (recuérdese que se analiza el exceso de error en la convergencia) y se supone gaussianidad, el valor esperado del cuadrado del estimador (V.50) es igual a:

$$E\left\{\xi_n^2(\delta)\right\} = \frac{1}{M^2} ME\left\{|\varepsilon(n)|^4\right\} = \frac{3\xi_{min}^2}{M} \quad (V.52)$$

Que, a su vez, sustituida en (V.51) proporciona la varianza del gradiente.

$$E\left\{|\nabla_n|^2\right\} = \frac{3}{2} \frac{\xi_{min}^2}{M\delta^2} \quad (V.53)$$

De la expresión anterior, que es el exceso de MSE que se produciría debido al gradiente en un coeficiente con paso de adaptación igual a la unidad, puede derivarse el exceso de error global al tener presente que son Q coeficientes y que el paso de adaptación es $\mu/2$.

$$\aleph = \frac{3}{4} \frac{\mu Q \xi_{min}}{M\delta^2} \quad (V.54)$$

El ruido de desajuste total será igual al anterior más el debido al exceso motivado por la estimación del gradiente. En definitiva el ruido de desajuste para el algoritmo DSD es:

$$\mathfrak{N}_{total} = \frac{3}{4} \frac{\mu Q \xi_{min}}{M \delta^2} + \gamma = \frac{3}{4} \frac{\mu Q \xi_{min}}{M \delta^2} + \frac{\delta^2 \text{Traza}(\underline{\underline{R}})}{Q \xi_{min}} \quad (\text{V.55})$$

Con respecto a convergencia, al ser éste un método de gradiente el paso de adaptación tiene la misma cota de convergencia, que viene limitada por la traza de la matriz de autocorrelación de los datos,

$$\mu = \frac{2\alpha}{\text{Traza}(\underline{\underline{R}})} \quad (\text{V.56})$$

Por lo anterior, lo único que se ha de determinar es el tamaño de la perturbación δ que más interesa. En este sentido, es interesante ver que existe un compromiso y, mientras que, para el exceso de estimación, interesa la perturbación más pequeña posible, sin embargo, para el ruido de la estimación interesa lo más grande posible (se deja al lector el razonar el porqué de este compromiso). En lo que se refiere al óptimo, basta derivar e igualar a cero la expresión del desajuste total. Procediendo de este modo, se obtiene el tamaño de perturbación óptima:

$$\delta_{opt}^2 = \sqrt{\frac{3}{2} \frac{Q^2 \alpha \xi_{min}^2}{M \text{Traza}^2(\underline{\underline{R}})}} \quad (\text{V.57})$$

Y el desajuste total mínimo como (V.58).

$$\mathfrak{N}_{total \text{ óptimo}} = \sqrt{\frac{6\alpha}{M}} \quad (\text{V.58})$$

Esta última expresión revela la ventaja del DSD sobre el LMS pues el desajuste disminuye con la raíz cuadrada del numero de términos empleados en la estimación del MSE. Recuérdese que, en cualquier caso, su desventaja radica en que se requieren $2MQ$ vectores por cada iteración. También ha de destacarse que en la adaptación, el vector de datos no es necesario lo cual en ciertas aplicaciones puede resultar una gran ventaja en términos de complejidad de implementación y tecnología disponible.

El comportamiento del algoritmo en términos de velocidad de convergencia y nivel de desajuste queda ilustrado en la figura V.11.

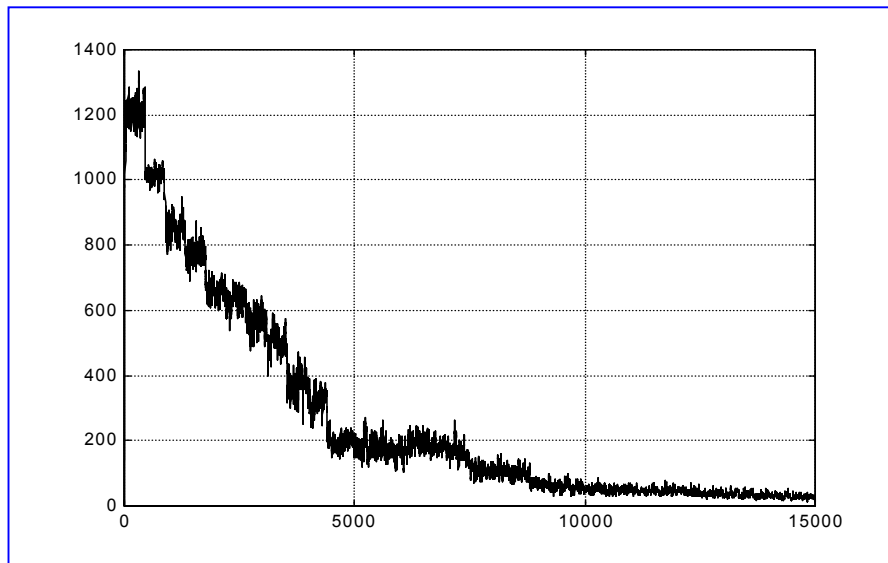


Figura V.11. Evolución del error o curva de aprendizaje para el DSD. Los parámetros son: $\delta=0.1$, el numero del muestras para promediar por cada peso de 20, $\alpha = 0.1$. El FIR diseñado es el mismo que en la figura V.9. Nótese el incremento producido en el numero de muestras para la convergencia aunque el desajuste es notablemente bajo.

Como ha podido verse el DSD es un método de estimar el gradiente del MSE. Básicamente puede denominarse, también como el primer método de los llamados de perturbación, en el sentido de que el aprendizaje es derivado directamente de una perturbación de los coeficientes del filtro que se desea adaptar. Este concepto de perturbación podría generalizarse y, a su vez, simplificarse con un mecanismo más natural de aprendizaje que sería de prueba y error. Dicho de otro modo, se podría provocar, a la vez, un vector de perturbaciones cualquiera $\underline{\delta}_n$ para los coeficientes. Si con dicha perturbación, el error aumenta esta perturbación no se tiene en cuenta; caso contrario se adoptan los coeficientes resultantes. Evidentemente, el algoritmo tiene una lógica aplastante e incluso podría solucionar el problema de la adaptación de filtros no-lineales ya que garantiza que ningún aprendizaje no positivo (o que haga decrecer el criterio de error) es incorporado al filtro. Así mismo, al ser de prueba y error, el algoritmo funcionaría sobre cualquier otro objetivo, aunque no fuese cuadrático, siempre y cuando no tenga mínimos locales.

Este algoritmo, que se resume a continuación, se denomina LRS (Linear Random Search) o de búsqueda lineal aleatoria.

Instante n. Pesos disponibles $\underline{a}_n$

- 1.- Generar un vector aleatorio con distribución uniforme en sus componentes $\underline{\delta}_n$.
- 2.- Actualizar los pesos como $\underline{a}_{n+1} \Rightarrow \underline{a}_n + \frac{\mu}{2} \underline{\delta}_n$
- 3.- Calcular el nuevo error usando M vectores de datos ξ_{n+1}
- 4.- $\text{Si } \xi_{n+1} < \xi_n \quad \underline{a}_n \Rightarrow \underline{a}_{n+1}$
 $\text{En caso contrario } \underline{a}_n \Rightarrow \underline{a}_n$
- 5.- $n \Rightarrow n + 1$.

Es fácil comprobar que el desajuste del LRS viene dado por:

$$\aleph = \frac{\mu Q \sigma_{\delta}^2}{2M} \quad (\text{V.59})$$

Donde σ_{δ} es la varianza de cada componente del vector de perturbación. Sus prestaciones quedan ilustradas en la Figura V.12.

El LRS en si sugiere múltiples variantes como controlar la distribución del vector de perturbación, usar diferentes distribuciones para cada uno de los pesos del filtro, tomar el modulo en lugar del cuadrado del error para ahorrar operaciones, etc. Todo lo contenido en este capítulo se desarrollo en los años 70 en el contexto de procesamiento de señal para reconocimiento de formas y ha servido de inspiración a lo que hoy se denomina de manera más precisa algoritmos genéticos o aprendizaje en redes neuronales. Se recomienda al lector que, antes de utilizar algoritmos genéticos o neuronales, comprenda y se familiarice con los algoritmos de gradiente que todavía hoy son materia de investigación y publicaciones prestigiosas. De otro modo, se podría caer en el defecto de tratar de solucionar problemas con herramientas mucho más complicadas e incomprensibles que un LMS, DSD o LRS, que en el 90% de las situaciones son más que suficiente para una aplicación concreta. Antes de cerrar el apartado y con el fin de apuntar las posibilidades de los métodos de gradiente se presentara una mejora sobre el LRS.

Básicamente el algoritmo, denominado GRS (Guided Random Search), consiste en aprovechar en el LRS el hecho de que el error haya disminuido (y que implica que la dirección es correcta) manteniendo e incluso aumentando el desplazamiento en la dirección encontrada. La idea es permanecer en la dirección correcta hasta que se es tangente a una curva de error. Llegado ese momento o cercano, la dirección correcta será ortogonal (dependiendo de lo próximo que se este al punto de tangencia con una superficie de error constante) comenzándose de nuevo el algoritmo. De hecho el GRS usa, en parte, una propiedad del algoritmo AG (Gradiente Acelerado) existente en la literatura. Esta propiedad establece que para criterios de error cuadrático, la recta que une el punto de comienzo con el segundo punto de tangencial es la dirección que apunta directamente al mínimo. Esta propiedad del AG se esquematiza en la Figura V.13. El GRS usa de hecho esta propiedad ya que cuando se agota la dirección correcta,

encontrada por búsqueda aleatoria, procede a realizar una búsqueda en la dirección ortogonal, aleatoria aun, a la usada.

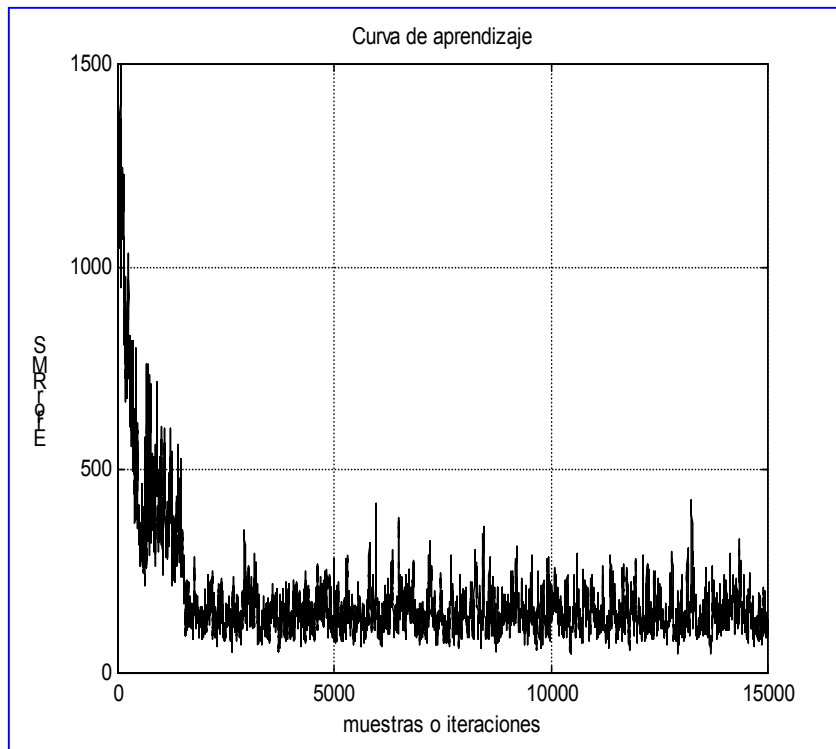
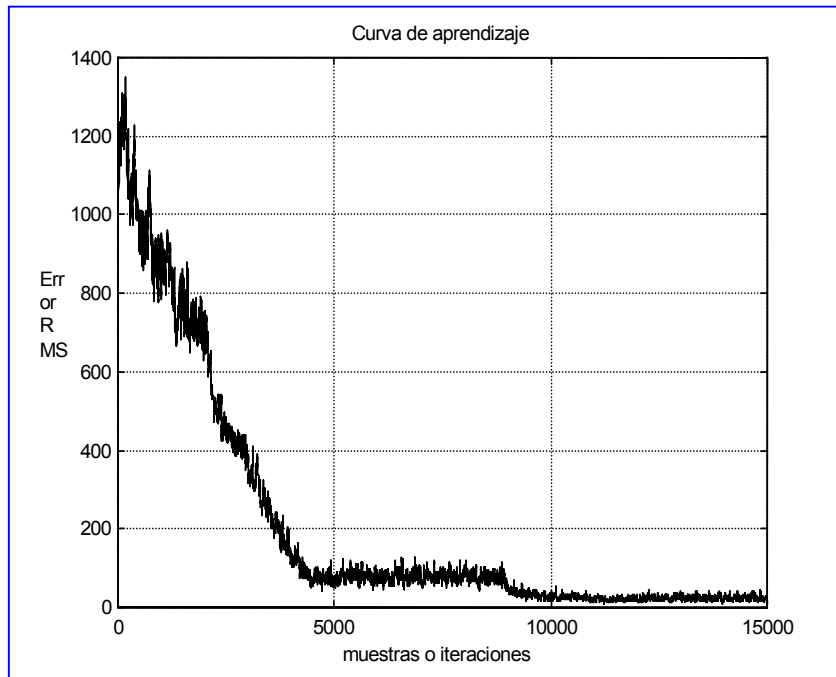


Figura V.12. Curva de aprendizaje para el algoritmo LRS. La distribución de la perturbación es uniforme de varianza 0.3 con un paso de adaptación igual a 0.1 (en la parte superior) y 0.5 (en la inferior). El sistema diseñado es el mismo que en las figuras V.9 y V.11.

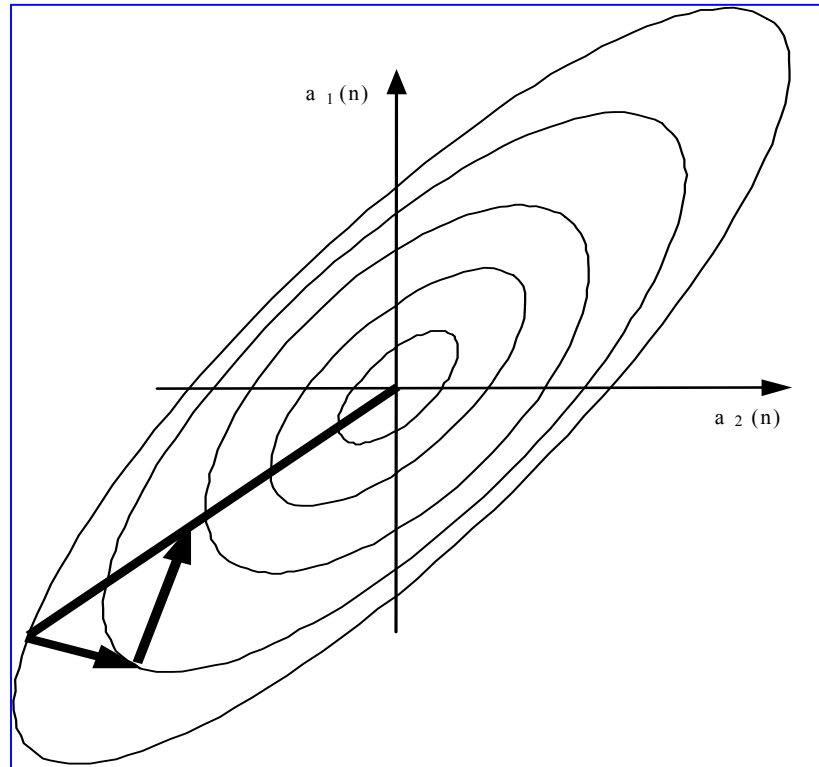


Figura V.13. Propiedad del AG para superficies de error cuadrático. Dos direcciones ortogonales y tangentes a curvas de error, al unir las, dan la dirección directa al mínimo.

Un resumen del GRS sería el siguiente:

Instante n. Pesos disponibles $\underline{a}_n$

- 1.- Generar un vector aleatorio con distribución uniforme en sus componentes $\underline{\delta}_n$.
- 2.- Actualizar los pesos como $\underline{a}_{n+1} \Rightarrow \underline{a}_n + \frac{\mu_o}{2} \underline{\delta}_n$.
- 3.- Calcular el nuevo error usando M vectores de datos

Si $\xi_{n+1} > \xi_n$ $\underline{a}_n \Rightarrow \underline{a}_n$ $\mu_o \Rightarrow \mu_{basico}$ $n \rightarrow n+1$ pasar al paso 1

En caso contrario $i = 2$

4. (#) $\underline{a}_n \Rightarrow \underline{a}_{n+1}$ $\mu_o \Rightarrow \beta \mu_o$ con $\beta \geq 1$

$$\underline{a}_{n+i} \Rightarrow \underline{a}_{n+1} - \frac{\mu_o}{2} \underline{\delta}_n$$

$$\begin{cases} \text{si } \xi_{n+1} > \xi_n & \mu_o \Rightarrow \mu_{basico} & n \rightarrow n+1 & \text{pasar al paso 1 generando} \\ \text{un vector de perturbación ortogonal al anterior:} \\ \text{si } \xi_{n+1} < \xi_n & i \rightarrow i+1 & \text{volver a (\#)} \end{cases}$$

Finalmente, en la figura V.14 se muestran algunas trayectorias del GRS en su búsqueda del mínimo global.

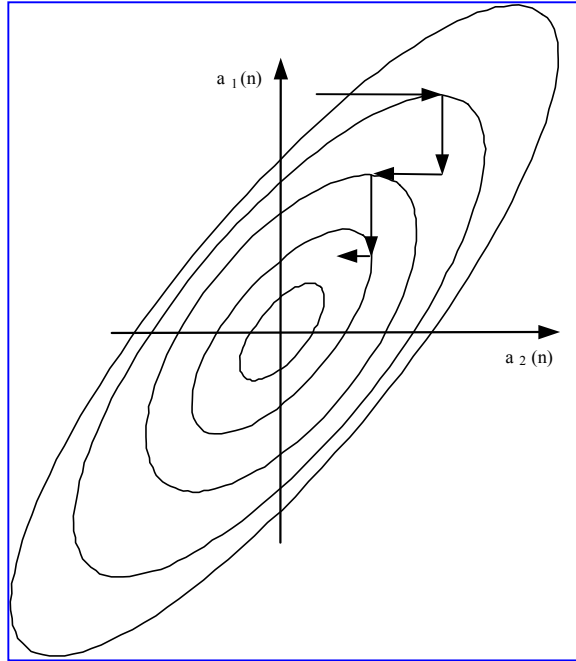


Figura V.14. Evolución del GRS. Aproximada con trazos ortogonales y direcciones tangentes a curvas de igual error.

El lector puede apreciar que el numero de variantes sobre el concepto de gradiente y los diferentes modos de estimación de éste, incluyendo métodos de perturbación aleatoria o aleatoria guiada sería muy amplio. No obstante, el objetivo era mostrar su interés y evidenciar que se trata de una familia de métodos útiles en aquellas situaciones en que se tiene claro la función a minimizar y el conjunto de parámetros con los que se pretende llevar a cabo la minimización. La extensión a sistemas no lineales no es difícil aunque se escapa de los objetivos del presente capítulo.

V.6. EL ALGORITMO RLS (Recursive Least Squares).

Una posibilidad que se aparta, aparentemente, de los métodos de gradiente es recurrir a la estimación, muestra a muestra, de los dos componentes de la solución de Wiener, es decir, de la matriz de autocorrelación de los datos y del vector $\underline{P}$. Dichas estimaciones se realizan vía un promedio de un numero M de muestras más recientes del vector de datos $\underline{X}_n$ y de la referencia $d(n)$. Habitualmente y a fin de simplificar su coste computacional, el promedio se realiza con un IIR de un coeficiente, que se denominara β , y que realiza un promediado exponencial, de base el parámetro mencionado, de las actualizaciones con una longitud efectiva de $1/(1-\beta)$ (análogamente a como se hacía en la ecuación V.2). En resumen, tanto matriz como vector se actualizan según se indica a continuación

$$\begin{aligned}\underline{R}_{n+1} &= \beta \cdot \underline{R}_n + (1-\beta) \cdot \underline{X}_n \cdot \underline{X}_n^H \\ \underline{P}_{n+1} &= \beta \cdot \underline{P}_n + (1-\beta) \cdot \underline{X}_n \cdot d^*(n)\end{aligned}\tag{V.60}$$

Estas expresiones ya fueron analizadas en un capítulo previo, así como sus propiedades como estimador insesgado de las funciones a estimar.

Una vez realizada la actualización (V.60), el filtro que se realiza para procesar el siguiente vector de datos viene dado por:

$$\underline{a}_{n+1} = \underline{R}_{n+1}^{-1} \underline{P}_{n+1}\tag{V.61}$$

Al margen de otros comentarios, nótese que esta manera de proceder implementa la solución óptima para un sistema que dispusiese tan solo de M (igual a $1/(1-\beta)$) datos de la señal de entrada. Es

decir, el filtro tendrá una buena capacidad de reacción a los cambios y se adaptará a ellos llegando en M muestras al óptimo local. Nótese que en términos de desajuste, el óptimo local coincidirá con el global, de infinitas muestras, en tanto en cuanto el parámetro β este próximo a la unidad. Como siempre, al hacer el desajuste pequeño se incrementa el tiempo de convergencia (el que tarda el sistema en reaccionar frente a un cambio o no estacionaridad), ya sea en la referencia como en los datos o entrada.

De una manera más formal, es trivial comprobar que (V.60) y (V.61) producen la solución de mínimo MSE(n), definido este como sigue:

$$MSE(n) = (1 - \beta) \cdot \sum_{m=-\infty}^n \beta^{n-m} \cdot |\varepsilon(m)|^2 = (1 - \beta) \cdot \sum_{m=-\infty}^n \beta^{n-m} \cdot \left| d(m) - \underline{a}_{n+1}^H \underline{X}_m \right|^2 \quad (V.62)$$

O bien

$$MSE(n) = \beta MSE(n-1) + (1 - \beta) |\varepsilon(n)|^2 \quad (V.63)$$

Estas expresiones son las que dan nombre al procedimiento o algoritmo pues es un procedimiento de minimización recursivo del error cuadrático (RLS).

El procedimiento expuesto para actualizar los pesos del filtro adolece de dos problemas. El primero es que no se dispone, por el momento, de una dependencia explícita entre los coeficientes en un instante y el anterior. La segunda, es que el procedimiento, sin duda de mayor complejidad que el LMS, requiere, también aparentemente, de la inversión de la matriz de autocorrelación. La solución a ambos problemas radica en la solución del segundo. De hecho, en el Capítulo I se expuso el denominado lema de la inversa que permite escribir la ecuación de recursión para la matriz de autocorrelación en términos de la matriz inversa. Enunciando de nuevo el lema de la inversa, este establece que si una matriz admite la descomposición:

$$\underline{\underline{A}} = \underline{\underline{B}} + \underline{\underline{C}} \underline{\underline{D}} \underline{\underline{C}}^H \quad (V.64)$$

Entonces, su inversa se puede escribir como:

$$\underline{\underline{A}}^{-1} = \underline{\underline{B}}^{-1} - \underline{\underline{B}}^{-1} \underline{\underline{C}} \left[\underline{\underline{D}} + \underline{\underline{C}}^H \underline{\underline{B}}^{-1} \underline{\underline{C}} \right]^{-1} \underline{\underline{C}}^H \underline{\underline{B}}^{-1} \quad (V.65)$$

Al aplicar el lema de la inversa a la matriz de autocorrelación de datos se obtiene:

$$\underline{\underline{R}}_{n+1}^{-1} = \frac{1}{\beta} \underline{\underline{R}}_n^{-1} - \frac{1}{\beta} \underline{\underline{R}}_n^{-1} \underline{\underline{X}}_n \left[\underline{\underline{I}} + \underline{\underline{X}}_n^H \underline{\underline{R}}_n^{-1} \underline{\underline{X}}_n \frac{1-\beta}{\beta} \right]^{-1} \frac{1-\beta}{\beta} \underline{\underline{X}}_n^H \underline{\underline{R}}_n^{-1} \quad (V.66)$$

Además,

$$\underline{\underline{a}}_{n+1} = \underline{\underline{R}}_{n+1}^{-1} \left(\beta \underline{\underline{P}}_n + (1 - \beta) \underline{\underline{X}}_n d^*(n) \right) \quad (V.67)$$

Este desarrollo permite que su uso, conjunto con la obtenida del lema de la inversa, permite encontrar la forma de actualizar los pesos anteriores ($\underline{\underline{a}}_n = \underline{\underline{R}}_n^{-1} \underline{\underline{P}}_n$) para obtener los nuevos. Después de agrupar términos, se obtiene que los nuevos pesos sean iguales a los anteriores más un vector que multiplica al error instantáneo.

$$\underline{\underline{a}}_{n+1} = \underline{\underline{a}}_n + \underline{\underline{K}}_n \varepsilon^*(n) \quad (V.68)$$

Nótese pues que la primera diferencia con respecto al LMS es que el vector $\mu \cdot \underline{\underline{X}}_n$ se ha reemplazado por un vector de ganancia. Este cambio es consecuencia del cambio entre minimizar un error

instantáneo a un promedio de los errores cometidos en las últimas muestras de salida. La expresión de este nuevo vector es:

$$\underline{K}_n = \left(\frac{\alpha}{1 + \phi} \right) \underline{R}_n^{-1} \underline{X}_n \quad (\text{V.69})$$

En esta expresión se aprecia que el RLS mantiene el vector de datos $\underline{X}_n$ pero difiere en que el paso de adaptación que ahora pasa a ser una matriz de ganancia que es, salvo una constante, la inversa de la matriz de autocorrelación. Más interesante es la interpretación cuando se describen las constantes que figuran en la fórmula anterior:

$$\begin{aligned} \alpha &= \frac{1 - \beta}{\beta} \\ \phi &= \alpha \cdot \left(\underline{X}_n^H \underline{R}_n^{-1} \underline{X}_n \right) \\ \underline{K}_n &= \frac{(1 - \beta) \underline{R}_n^{-1} \underline{X}_n}{\beta + (1 - \beta) \cdot \left(\underline{X}_n^H \underline{R}_n^{-1} \underline{X}_n \right)} \end{aligned} \quad (\text{V.70})$$

La última fórmula en (V.70), revela que el RLS es similar al LMS cuando la matriz de autocorrelación de los datos es diagonal, es decir, cuando estos son ruido blanco. El lector puede comprobar que en este caso las actualizaciones siguiendo el RLS o el LMS son prácticamente las mismas. Esto no debe sorprender ya que se recordara que para ruido blanco la dispersión de autovalores es mínima y el gradiente apunta en cualquier punto al mínimo, es decir, el LMS bajo ruido blanco resulta un algoritmo perfecto. Esta es la razón por la que muchos autores recomiendan que si se desea emplear el LMS es muy recomendable blanquear los datos con un predictor lineal. El hecho es que el predictor más el LMS juntos, y ambos adaptativos, acostumbran a tener mayor complejidad que el RLS directamente por lo que el procedimiento mencionado está prácticamente en desuso.

Las ecuaciones anteriores completan el algoritmo RLS para una iteración. A continuación se resumen los pasos a seguir en cada vector de datos.

Iteración en el instante n

Datos $\underline{a}_n, \underline{R}_n^{-1}, d(n)$ y $\underline{X}_n$

- 1.- Calcular la salida del filtro $y(n) = \underline{a}_n^H \underline{X}_n$
 - 2.- Calcular el error entre la salida y la señal de referencia $\varepsilon(n) = d(n) - y(n)$
 - 3.- Calcular $\phi = \alpha \left(\underline{X}_n^H \underline{R}_n^{-1} \underline{X}_n \right)$ siendo $\alpha = \frac{1 - \beta}{\beta}$
 - 4.- Evaluar el vector ganancia $\underline{K}_n = \frac{\alpha \underline{R}_n^{-1} \underline{X}_n}{1 + \phi}$
 - 5.- Actualizar los coeficientes $\underline{a}_{n+1} = \underline{a}_n + \underline{K}_n \varepsilon^*(n)$
 - 6.- Actualizar la matriz inversa $\underline{R}_{n+1}^H = \frac{1}{\beta} \underline{R}_n^{-1} - \underline{K}_n \underline{K}_n^H \frac{1 + \phi}{1 - \beta}$
- $n \Rightarrow n + 1$

El algoritmo RLS es sin duda el mejor algoritmo adaptativo para la minimización del MSE. Sus prestaciones no dependen de la dispersión de autovalores. La convergencia es del orden de la longitud del filtro, es decir, para un filtro de Q coeficientes se tarda Q iteraciones o vectores de datos en converger y su desajuste se minimiza con valores de β próximos a la unidad. Por ello, y siempre que su mayor complejidad lo permita, no tiene rival en la minimización del MSE o en el diseño de filtro de Wiener. De todos modos, su éxito relativo es debido a que, dentro del área de teoría de control, Kalman proporcionó

una forma más elegante y formal de presentarlo. No solo esto, sino que, además el filtro de Kalman proporciona más versatilidad a su empleo y por tanto el ámbito de su aplicación es mayor. Por todo ello, y dado que el próximo apartado presenta el filtro de Kalman, las afirmaciones sobre convergencia y desajuste se posponen al próximo apartado, mucho más interesante que el actual, como podrá ver el lector.

V.7 EL FILTRO DE KALMAN.

La deducción del filtro de Kalman puede, o debe, hacerse en términos del estimador óptimo de media condicional basado en el modelo de Gauss-Markov de un proceso. Aunque esta presentación sería la formal, no se usará en el presente apartado por tres razones: La primera es que la formalidad mencionada se paga en términos de dificultad conceptual de conocimientos en estimación y modelado. La segunda es que esta presentación es la que el lector puede encontrar en la literatura siempre que esta aborda el tema de filtrado óptimo o de Kalman y no tiene interés repetirla aquí de nuevo. La tercera es que esa presentación formal se separa de los conceptos y herramientas que hasta ahora se han expuesto y que son familiares en un curso de procesamiento de señal. La presentación que sigue es original y se basa en conceptos perfectamente descritos anteriormente. En cualquier caso, el lector interesado en esta presentación formal encontrará útiles los ejercicios resueltos correspondientes al Capítulo II ya que en uno de ellos se describe formalmente el estimador de media condicional.

El primer cambio que tiene lugar es que a diferencia de los anteriores algoritmos expuestos en el presente capítulo, se establece un modelo de señal sobre el que después se derivará el algoritmo y que se pasa a explicar a continuación. Este modelo, denominado de Gauss-Markov es válido para cualquier proceso gaussiano.

El problema se plantea suponiendo que existe un fenómeno físico que está controlado por un conjunto de Q variables que se denominará vector de estado y denotaremos por $\underline{A}_n$. A nivel de ejemplo, supóngase que el vector de estado son las temperaturas internas de un horno de combustión. Es claro, que el vector de estado no está disponible para su medida, si se renuncia a introducir sensores que, al margen de su costo, perturbarían el funcionamiento del horno. Este vector de estado tiene una evolución con el tiempo que se describe con dos sumandos. Un sumando es una componente determinista de su evolución que fuerza que los valores de las variables dependen de los anteriores vía una matriz denominada de transición $\underline{F}$. El segundo sumando, es la parte impredecible de la evolución de los parámetros del vector de estado y viene dado por un vector de variables aleatorias que se denominará $\underline{v}_n$. Este vector es independiente del vector de estado $\underline{A}_n$, su media es el vector cero, y su matriz de covarianza es $\underline{V}$.

En definitiva la ecuación de estado viene dada por (V.71), insistiéndose en sus dos componentes, la predecible o prevista y la imprevisible,

$$\underline{A}_{n+1} = \underline{F}_n \underline{A}_n + \underline{v}_n \quad (\text{V.71})$$

Donde:

$$E\{\underline{A}_n \underline{v}_n^H\} = \underline{0} \quad ; \quad E\{\underline{v}_n\} = \underline{0} \quad ; \quad E\{\underline{v}_n \underline{v}_n^H\} = \underline{V}_n \quad (\text{V.72})$$

Antes de proseguir veamos algunos ejemplos que muestran hasta qué punto la ecuación de estado y sus componentes son una parte esencial en aplicaciones. En primer lugar, podíamos imaginar que se trata de conocer la posición x y la velocidad v_x de un móvil, ambos valores serán las componentes del vector de estado. Si se supone que el móvil funciona básicamente en velocidad constante maniobrando poco en aceleración o deceleración, la primera componente de la ecuación de estado sería:

$$\begin{bmatrix} x(n+1) \\ v_x(n+1) \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \cdot \begin{bmatrix} x(n) \\ v_x(n) \end{bmatrix} \quad (\text{V.73})$$

Siendo T es el intervalo de muestreo o intervalo entre dos medidas de posición y velocidad. La primera ecuación es la del espacio para un móvil de velocidad constante y la segunda establece que es un movimiento de velocidad constante. En resumen, esta primera parte de la ecuación de estado viene regulada por las hipótesis realizadas sobre el fenómeno físico en observación. En cualquier caso, la

hipótesis de velocidad constante puede no ser estrictamente cierta y el móvil puede tener una cierta maniobrabilidad que, sin poderla escribir como algo determinista en el primer termino, si que altera posición y velocidad. Esta es la razón del segundo termino, es decir, todo aquello que no se puede prever en la evolución del estado se plantea como aleatorio o imprecisión así hablaremos de una imprecisión en la posición $v_1(n)$ y una imprecisión en la velocidad que será $v_2(n)$.

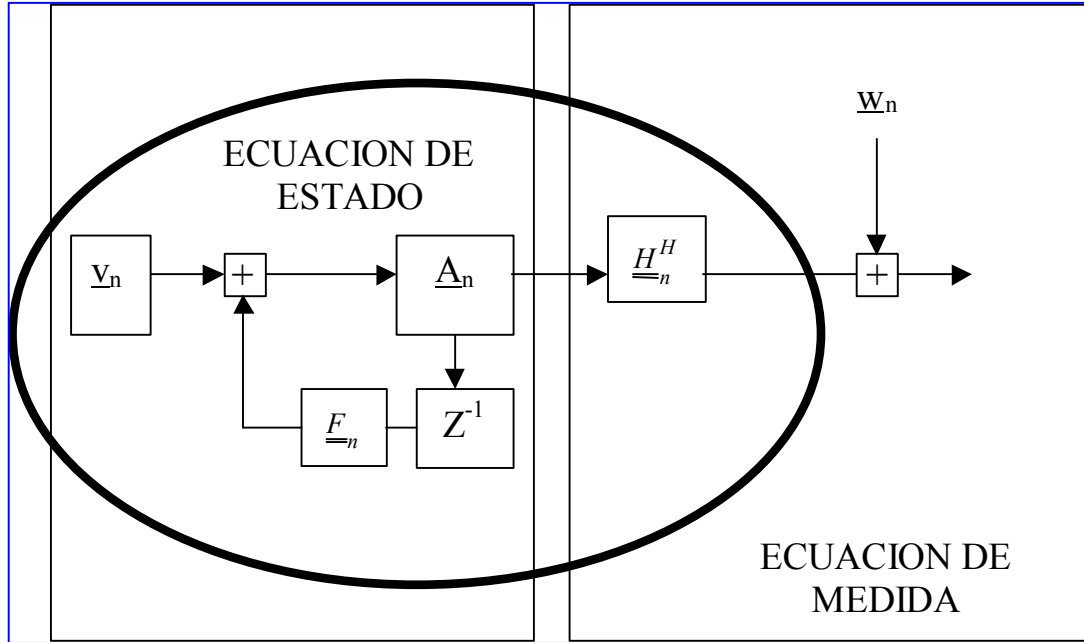


Figura V.15. El modelo de señal donde se muestra la ecuación de estado como descriptor del fenómeno físico correspondiente y la ecuación de medida como única muestra de la evolución del vector de estado.

De este modo, la ecuación de estado completa sería

$$\begin{bmatrix} x(n+1) \\ v_x(n+1) \end{bmatrix} = \begin{bmatrix} 1 & T \\ 0 & 1 \end{bmatrix} \begin{bmatrix} x(n) \\ v_x(n) \end{bmatrix} + \begin{bmatrix} v_1(n) \\ v_2(n) \end{bmatrix} \quad (V.73)$$

Recuerde pues, que la matriz de covarianza de la imprecisión ha de establecerse de acuerdo con las previsiones sobre la maniobrabilidad del vector de estado y el desconocimiento del diseñador sobre ésta. Por último, se considera que tanto la matriz de transición como la matriz de covarianza de la imprecisión son conocidas.

La segunda ecuación del modelo tiene su origen en que, en general y como en el caso del horno de combustión, la medida o observación del vector de estado no es posible directamente. Así las temperaturas interiores se miden en el exterior. Esto implica una matriz de transición $\underline{H}$ que establece la relación entre las variables del vector de estado y la medida de estas $\underline{z}_n$, además ha de considerarse el ruido de medida que los transductores introducen $\underline{w}_n$. El ruido de medida es habitualmente de media cero y covarianza $\underline{W}_n$ siendo incorrelado con el resto de variables. De este modo, la denominada ecuación de medida viene dada por (V.74).

$$\underline{z}_n = \underline{H}_n^H \underline{A}_n + \underline{w}_n \quad (V.74)$$

Donde

$$E\{\underline{w}_n \underline{z}_n^H\} = \underline{0} \quad ; \quad E\{\underline{w}_n \underline{A}_n^H\} = \underline{0} \quad ; \quad E\{\underline{w}_n\} = \underline{0} \quad ; \quad E\{\underline{w}_n \underline{w}_n^H\} = \underline{W}_n \quad (V.75)$$

En definitiva, el modelo de señal establece el vector $\underline{z}_n$ como señal disponible y a utilizar para conseguir la estimación más precisa posible del vector de estado. Se denomina filtro de Kalman al proceso de la medida para la estimación del estado. A continuación se describen las ecuaciones del modelo

$$\begin{aligned}\underline{A}_{n+1} &= \underline{F}_n \underline{A}_n + \underline{v}_n \\ \underline{z}_n &= \underline{H}_n^H \underline{A}_n + \underline{w}_n\end{aligned}\quad (\text{V.76})$$

Para comenzar el diseño del filtro o estimador, se analizara, en primer lugar, como se puede copiar el modelo, dentro de lo posible. De la ecuación de medida se dispone únicamente de un vector de estado estimado, probablemente incorrecto y de un ruido de medida desconocido. Con lo anterior, la ecuación de filtrado o copia de la segunda del modelo será (V.77), pues recuerde que solo la matriz de transición y la covarianza del ruido de medida son conocidas:

$$\hat{\underline{z}}_n = \underline{H}_n^H \hat{\underline{A}}_n \quad (\text{V.77})$$

Evidentemente esta estimación de la medida será errónea y la determinación del error será

$$\underline{\varepsilon}_n = \underline{z}_n - \hat{\underline{z}}_n \quad (\text{V.78})$$

Este error es el único capaz de ayudar a mejorar la estimación que antes se ha usado del vector de estado. Antes de indicar como se utiliza el error para mejorar el estado, es interesante expresar el error de medida en función del error de estado. Restando la ecuación de medida de la ecuación de filtrado se obtiene (V.79) que revela la dependencia entre el error observado y el oculto o de las componentes del vector de estado:

$$\underline{\varepsilon}_n = \underline{z}_n - \hat{\underline{z}}_n = \underline{H}_n^H (\underline{A}_n - \hat{\underline{A}}_n) + \underline{w}_n = \underline{H}_n^H \tilde{\underline{A}}_n + \underline{w}_n \quad (\text{V.79})$$

Es mas, teniendo en cuenta que el ruido de medida está incorrelado con el vector de estado, se puede calcular la potencia del error de medida en función de la covarianza del error de estado que se denominara $\underline{\Sigma}_n$

$$\underline{\xi}_n \equiv E\{\underline{\varepsilon}_n \underline{\varepsilon}_n^H\} = \underline{H}_n^H \underline{\Sigma}_n \underline{H}_n + \underline{W}_n \quad (\text{V.80})$$

donde

$$\underline{\Sigma}_n \equiv E\{\tilde{\underline{A}}_n \tilde{\underline{A}}_n^H\} \quad (\text{V.81})$$

Esta ecuación revela, como no podía ser de otro modo, que el error de medida viene dado por dos términos: la potencia del ruido de medida y el error de coeficientes a través de la matriz de medida.

Una vez caracterizada la ecuación de filtrado y la dependencia del error de medida con el error de estado, se pasará a ver como se puede usar el error de medida para mejorar la estimación del estado. Lo más intuitivo y lógico es pensar que la nueva estimación del vector de estado se modificará de acuerdo al error de medida. Por ello, también parece lógico establecer como ecuación de adaptación la siguiente:

$$\hat{\underline{A}}_{n+1} = \underline{F}_n \hat{\underline{A}}_n + \underline{K}_n \underline{\varepsilon}_n \quad (\text{V.82})$$

Para abordar el diseño de la matriz de ganancia, está claro que su diseño debería ser tal que el nuevo error de vector de estado fuese mínimo. El nuevo vector de error se obtiene de restar a la ecuación de estado del modelo la ecuación anterior. Al hacerlo se obtiene:

$$\tilde{\underline{A}}_{n+1} = \underline{F}_n \tilde{\underline{A}}_n + \underline{v}_n - \underline{K}_n \underline{\varepsilon}_n \quad (\text{V.83})$$

Nótese que se trata de mejorar la estimación, es decir, de minimizar la covarianza del vector de la izquierda de la ecuación anterior, disponiendo el error de medida como dato. Recuperando del tema de filtro de Wiener el principio de ortogonalidad, este establecía que el óptimo se obtenía de forzar la ortogonalidad del error con los datos. Este principio aplicado a la ecuación anterior sería:

$$E\{\tilde{\underline{A}}_{n+1} \underline{\varepsilon}_n^H\} = \underline{0} \quad (\text{V.84})$$

Al sustituir (V.83) se obtiene que la ecuación de diseño de la matriz de ganancia es:

$$\underline{0} = \underline{F}_n E\{\tilde{\underline{A}}_n \underline{\varepsilon}_n^H\} - \underline{K}_n \underline{\xi}_{\underline{\varepsilon}_n} \quad (\text{V.85})$$

Donde se ha tenido presente la independencia de la imprecisión con el error de medida y que la covarianza del error de medida ya se definió y calculó previamente (véase V.80).

De los términos necesarios para calcular la matriz de ganancia, tan solo el valor esperado no está ya disponible. Este término puede calcularse con ayuda de la expresión del error de medida y resulta ser:

$$E\{\tilde{\underline{A}}_n \underline{\varepsilon}_n^H\} = E\{\tilde{\underline{A}}_n \tilde{\underline{A}}_n^H\} \underline{H}_n = \underline{\Sigma}_n \underline{H}_n \quad (\text{V.86})$$

Y, al ser usada en la anterior, proporciona la expresión de la matriz de ganancia.

$$\underline{K}_n = \underline{\xi}_{\underline{\varepsilon}_n}^{-1} \underline{F}_n \underline{\Sigma}_n \underline{H}_n \quad (\text{V.87})$$

En este momento, disponiendo de una estimación del vector de estado y de su matriz de covarianza, al disponer del error de medida, se calcula la matriz de ganancia y se actualiza la estimación del vector de estado. Para completar la iteración es necesario dejar disponible la estimación de la matriz de covarianza del nuevo estado. Esta ecuación, que permite actualizar también la matriz de covarianza del error de estado, se obtiene de la ecuación (V.83), teniendo presente la expresión ya obtenida para la matriz de ganancia. Después de sencillas manipulaciones se obtiene (V.88) que completa el algoritmo o filtro de Kalman.

$$\underline{\Sigma}_{n+1} = \underline{F}_n \underline{\Sigma}_n \underline{F}_n^H - \underline{K}_n \underline{\xi}_{\underline{\varepsilon}_n} \underline{K}_n^H + \underline{V}_n \quad (\text{V.88})$$

A modo de resumen los pasos a seguir en cada iteración son los siguientes:

Instante n

Datos $\underline{z}_n; \hat{\underline{A}}_n; \underline{\Sigma}_n$

- 1.- Filtrar el vector de estado $\hat{\underline{z}}_n = \underline{H}_n^H \hat{\underline{A}}_n$
 - 2.- Evaluar el error de medida $\underline{\varepsilon}_n = \underline{z}_n - \hat{\underline{z}}_n$
 - 3.- Estimar su covarianza $\underline{\xi}_{\underline{\varepsilon}_n} = \underline{H}_n^H \underline{\Sigma}_n \underline{H}_n + \underline{W}_n$
 - 4.- Calcular la matriz de ganancia $\underline{K}_n = \underline{\xi}_{\underline{\varepsilon}_n}^{-1} \underline{F}_n \underline{\Sigma}_n \underline{H}_n$
 - 5.- Actualizar la estimación del vector de estado $\hat{\underline{A}}_{n+1} = \underline{F}_n \hat{\underline{A}}_n + \underline{K}_n \underline{\varepsilon}_n$
 - 6.- Actualizar la estimación de la covarianza $\underline{\Sigma}_{n+1} = \underline{F}_n \underline{\Sigma}_n \underline{F}_n^H - \underline{K}_n \underline{\xi}_{\underline{\varepsilon}_n} \underline{K}_n^H + \underline{V}_n$
- $n \Rightarrow n+1$

Este algoritmo es denominado filtro de Kalman. Sus prestaciones quedan representadas en la Figura V.16. El lector puede comparar y ver su estricta similitud con el algoritmo RLS. La diferencia es que temas de control de plantas, seguimiento de movimiento, etc. no hubiesen saltado al panel de aplicaciones de no ser por el planteamiento tan elegante de un algoritmo para un modelo de señal tan general. Además, todos los parámetros que participan en el filtro de Kalman tienen un sentido físico y no meramente algebraico como se vera a continuación.

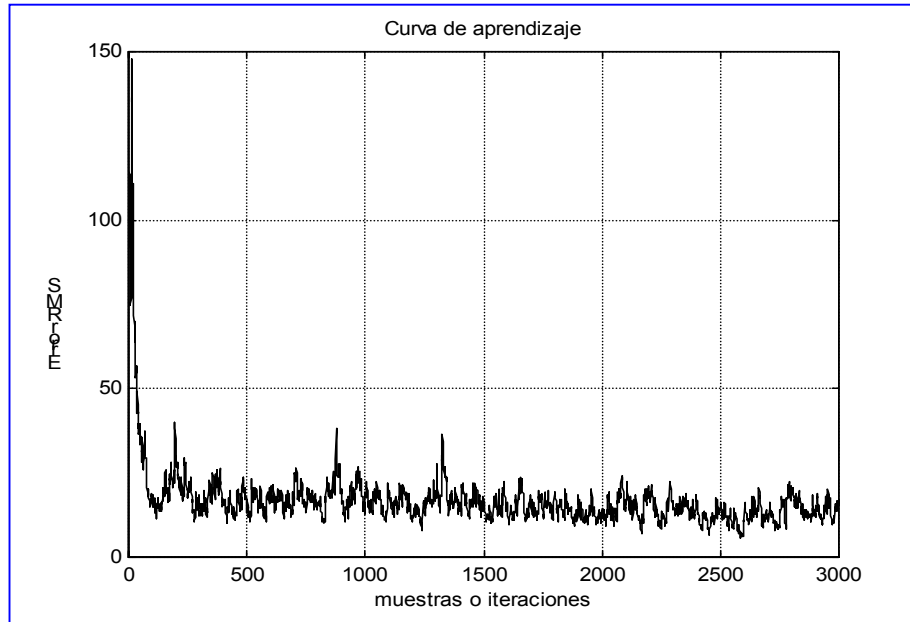


Figura V.16. Curva de aprendizaje del filtro de Kalman en la misma situación que el resto de algoritmos. Puede apreciarse su extraordinaria rapidez de convergencia y el escaso desajuste, ambos muy por encima en calidad que el resto de algoritmos.

Aunque la explicación que sigue podría realizarse sobre el caso de seguimiento de movimiento en una o en dos dimensiones, dado que el capítulo que nos ocupa es el diseño adaptativo de filtros, lo que sigue usará, tanto el modelo, como el algoritmo para diseño de un filtro de Wiener. La primera cuestión es como se encaja el modelo de señal con el problema de filtrado. Es fácil, se ha de pensar que, dada la referencia $d(n)$ del filtro de Wiener y el vector conteniendo las muestras de la entrada, existe un filtro ideal y desconocido de coeficientes $\underline{A}_n$ (dependerá de n si ha de ser variante) que proporciona un error de medida independiente de los datos, es decir, ortogonal a estos y por lo tanto óptimo:

$$w(n) = d(n) - \underline{A}_n^H \underline{X}_n \quad (\text{V.89})$$

Esta ecuación, tan natural en el problema de filtrado, no es ni más ni menos que la ecuación de medida en el modelo de Gauss-Markov.

$$d^*(n) = \underline{X}_n^H \underline{A}_n + w^*(n) \quad (\text{V.90})$$

Como puede verse, lo que era un vector $\underline{z}$ ahora pasa a ser un escalar que es igual al conjugado de la señal de referencia. Pero lo más destacable es que las muestras de la entrada juegan el papel de la matriz de medida que, en el escalar mencionado, nos permiten observar los coeficientes óptimos. También el ruido de medida pasa a ser el error mínimo. Es escalar y su covarianza será por tanto $\zeta_{\min}$.

La ecuación de estado es aun más interesante. Esta ha de reflejar un modelo de evolución de los coeficientes de ese filtro óptimo desconocido que se pretende estimar. Afortunadamente esta ecuación se simplifica mucho, pues lo normal es que de la evolución de los coeficientes se desconozca todo, o bien se trate de un sistema invariante. Un sistema invariante tendría como ecuación de estado:

$$\underline{A}_{n+1} = \underline{A}_n \quad (\text{V.91})$$

No obstante, parece más general y previsor el incluir una cierta ignorancia en el carácter invariante del filtro buscado, por esta razón se añade a la anterior un termino aleatorio de matriz de covarianza $\underline{V}$. Lo que sí suele ser cierto es que los coeficientes presentan una evolución incorrelada entre ellos por lo que, habitualmente se utilizara una matriz $\underline{V}$ diagonal con igual valor en cada elemento. Aunque no es estrictamente formal, lo plausible de la suposición anterior puede buscarse en la covarianza de coeficientes encontrada con motivo del cálculo del desajuste en el LMS (véase la ecuación (V.39)):

$$\underline{A}_{n+1} = \underline{A}_n + \underline{v}_n \quad (\text{V.92})$$

Es importante recordar dos cuestiones: en primer lugar la matriz de transición es la unidad, es decir la matriz $\underline{F}$ no aparecerá en el algoritmo correspondiente; en segundo lugar, la varianza que se sitúe en la matriz $\underline{V}$ es el síndrome de no estacionaridad del proceso o, de otro modo, cuan variante será el filtro óptimo.

Con este modelo para el problema de filtrado, la secuencia de ecuaciones del filtro de Kalman van apareciendo de manera fácil o sencilla. El filtro adaptativo que se implementa es:

$$y^*(n) = \underline{X}_n^H \hat{\underline{A}}_n \quad \text{igual a la tradicional} \quad y(n) = \hat{\underline{A}}_n^H \underline{X}_n \quad (\text{V.93})$$

El error de medida, denominado simplemente error en Wiener, es:

$$\varepsilon(n) = d^*(n) - y^*(n) \quad (\text{V.94})$$

Que, dado que es un escalar, tiene como potencia

$$\xi_n = \varsigma_{min} + \underline{X}_n^H \underline{\Sigma}_n \underline{X}_n \quad (\text{V.95})$$

Y, la matriz de ganancia, que pasa a ser un vector igual que en el RLS, es

$$\underline{K}_n = \frac{\underline{\Sigma}_n \underline{X}_n}{\varsigma_{min} + \underline{X}_n^H \underline{\Sigma}_n \underline{X}_n} \quad (\text{V.96})$$

La ecuación de actualización de los coeficientes es

$$\hat{\underline{A}}_{n+1} = \hat{\underline{A}}_n + \frac{\underline{\Sigma}_n}{\varsigma_{min} + \underline{X}_n^H \underline{\Sigma}_n \underline{X}_n} \underline{X}_n \varepsilon(n) \quad (\text{V.97})$$

Finalmente, la de actualización de la matriz de covarianza cierra el algoritmo:

$$\underline{\Sigma}_{n+1} = \underline{\Sigma}_n - \underline{K}_n \xi_n \underline{K}_n^H + \underline{V}_n \quad (\text{V.98})$$

Queda ahora la asignación de los valores iniciales de los coeficientes $\underline{A}_0$ y de su matriz de covarianza $\underline{\Sigma}_0$. Respecto al primero, cualquier valor es válido pues la convergencia no depende del valor inicial. Respecto a la matriz de covarianza de los coeficientes, puede elegirse diagonal a fin de que el algoritmo reaccione rápidamente. Lo que es realmente importante es que, dado que la elección del vector inicial es arbitraria, es lógico pensar que éste tiene un error grande. Por esta razón hay de situar las diagonales de la matriz de covarianza al máximo valor permisible por la dinámica del procesador.

El hecho de que la covarianza $\underline{\Sigma}$ representa el error de coeficientes permite conocer detalles relevantes del algoritmo. En primer lugar nótese que si el error es muy grande al comienzo del algoritmo, se puede suponer que la matriz de covarianza es un factor grande por la matriz identidad:

$$\underline{\underline{\Sigma}}_n \approx \sigma^2 \underline{\underline{I}} \quad (\text{V.99})$$

Al sustituir esta matriz d covarianza de los coeficientes en la matriz de ganancia y considerar que el factor σ es muy grande, se tendrá (V.100).

$$\underline{\underline{K}}_n \approx \lim_{\sigma \rightarrow \infty} \frac{\sigma^2 \underline{\underline{X}}_n}{\zeta_{min} + \sigma^2 \underline{\underline{X}}_n^H \underline{\underline{X}}_n} = \frac{\underline{\underline{X}}_n}{\underline{\underline{X}}_n^H \underline{\underline{X}}_n} \quad (\text{V.100})$$

Es decir, el filtro de Kalman, en los instantes iniciales actúa como un LMS de paso de adaptación igual a la unidad. Después cuando la matriz de covarianza de coeficientes decrece la matriz de ganancia se hace muy pequeña. De hecho, la matriz de covarianza, para valores pequeños de ésta y siguiendo la recursión que aparece en la ecuación (V.98), tendera a la matriz $\underline{\underline{V}}$. Si a su vez, esta se ha fijado diagonal, puede decirse que en seguimiento el algoritmo tiene una ganancia igual a:

$$\underline{\underline{K}}_n \approx \frac{\sigma_v^2}{\zeta_{min}} \underline{\underline{X}}_n \quad \text{con} \quad \underline{\underline{V}} = \sigma_v^2 \underline{\underline{I}} \quad (\text{V.101})$$

Esta expresión revela que, efectivamente, el índice de estacionaridad que se decide para el filtro influencia directamente el valor final del paso de adaptación del algoritmo. De otro modo, si el sistema a identificar fuese invariante el paso de adaptación acabaría siendo cero y el desajuste estrictamente cero. También es de destacar que el denominador de la matriz de ganancia, no es más que el estimador de la traza que empleaba el LMS para normalizar el paso de adaptación. La potencia del error mínimo actúa de salvaguarda (impide que en un momento dado la ganancia se haga infinito) haciendo que si la señal de entrada es muy pequeña la ganancia no se haga excesivamente grande. Este valor puede situarse a la unidad en aquellas aplicaciones donde se desconozca su valor.

Como ha podido verse la convergencia es controlada por la covarianza de coeficientes. Para estimar de qué orden es la velocidad de convergencia del algoritmo se examinara como evoluciona el valor diagonal de dicha matriz. Esta manera de obtener la velocidad de convergencia no es rigurosa aunque su resultado muestra una concordancia extraordinaria con lo observado en la práctica. Al rescribir (V.98) para el caso de matriz de covarianza diagonal, para el valor q de la diagonal, se obtiene:

$$\sigma_{n+1}^2 = \sigma_n^2 \left(1 - \frac{\sigma_n^2 |x(n-q)|^2}{\zeta_{min} + \sigma_n^2 \underline{\underline{X}}_n^H \underline{\underline{X}}_n} \right) + \sigma_v^2 \quad (\text{V.102})$$

Dado que los valores iniciales son grandes y que la no estacionaridad toma valores pequeños (entre 10^{-3} y 10^{-9}), se puede escribir (V.103)

$$\sigma_n^2 \approx \sigma_n^2 \left(1 - \frac{\sigma_n^2 P_x}{\zeta_{min} + \sigma_n^2 Q P_x} \right) \approx \sigma_n^2 \left(1 - \frac{1}{Q} \right) \quad (\text{V.103})$$

De la que puede concluirse que el numero de iteraciones para que la covarianza se reduzca un orden de magnitud es igual a $Q \cdot \ln(10)$, es decir, es proporcional al orden del filtro y, lo que es más importante, es independiente de la dispersión de autovalores.

Con respecto al desajuste, este vendrá dado por la traza de la matriz de covarianza de coeficientes multiplicada por la matriz de autocorrelación de los datos. Como

$$\lim_{n \rightarrow \infty} \left(\text{Traza} \left[\underline{\underline{\Sigma}}_n \underline{\underline{R}} \right] \right) = \sigma_v^2 \text{Traza}(\underline{\underline{R}})$$

El desajuste será:

$$M = \frac{\sigma_v^2 \text{Traza}(\underline{\underline{R}})}{\zeta_{min}} \quad (\text{V.104})$$

Que, como era de prever, depende directamente del denominado índice de no-estacionaridad con que se denominó al valor en la diagonal de la matriz $\underline{\underline{V}}$.

Lo expuesto es una visión rápida y cuando menos efectiva del filtro de Kalman. Efectiva ya que reúne todo lo necesario para su implementación. Tal vez valga la pena remarcar que en aplicaciones diferentes de la de filtrado, explicada con detalle, la clave está en la calidad del modelo ya que, como el lector ha podido apreciar, éste es el que dicta el algoritmo oportuno. Es interesante señalar que la flexibilidad del modelo es alta y es fácil encontrar en su aplicación en seguimiento de trayectorias para radar de variables de estado de dimensión 9 y matrices de transición completas. En cualquier caso, dando por hecho que este filtro sustituye al RLS, puede decirse que es el mejor algoritmo adaptativo conocido hoy. Existen refinamientos para su estabilidad numérica o para la reducción en el número de operaciones que, para los objetivos del capítulo, no viene al caso exponer. También existen versiones para filtrado no lineal de uso extendido en sistemas de recuperación de portadora en comunicaciones que también se salen del ámbito que aquí se persigue. Como tal, filtrado de Kalman es por sí solo el único contenido de asignaturas de pregrado y postgrado en teoría de control y automática. Aquí se ha expuesto, como ya se ha reiterado, una breve exposición que permite su uso y que en su presentación sacrifica la formalidad por su sencillez.

V.7 CONCLUSIONES

En este tema se ha presentado los algoritmos más relevantes que permiten un diseño adaptado a los cambios del escenario de filtros de mínimo MSE con respecto a una referencia. Estos sistemas son de uso obligado en el modelado o ecualización de sistemas variantes como es la señal de voz, en el primer caso, y los canales de comunicaciones en el segundo.

La familia de algoritmos más usados y los primeros en usarse en procesamiento de señal son los de gradiente. Usados por primera vez en reconocimiento formas supervisado en los 60, pasaron a ser de uso extendido en procesamiento de señal, en arrays en primer lugar y después en reductores de ruido, más tarde a todos los campos de aplicación. Puede decirse que la FFT, la Recursión de Levinson y el LMS son los algoritmos más implementados.

Una vez presentada la utilidad del gradiente en el aprendizaje, los algoritmos denominados de gradiente se diferencian entre sí en la manera en que la estimación de este se lleva a cabo. El más popular denominado LMS o de gradiente instantáneo es el que se presenta en primer lugar en honor a lo extendido de su uso como ya se ha comentado. Al mismo tiempo, parámetros de calidad como velocidad de convergencia y desajuste se introdujeron en la presentación del LMS.

A continuación se expuso un modo más riguroso de estimar el gradiente denominado DSD. El valor del DSD es que sugiere inmediatamente algoritmos de búsqueda aleatoria. Estos algoritmos no solamente resuelven el problema lineal planteado por el MSE, sino que además permiten resolver problemas de minimización no-lineales. También desarrollados en reconocimiento de formas en el comienzo de los 60, hoy se redescubren en las redes neuronales o sistemas denominados de inteligencia artificial. Se han descrito los dos más básicos pasando del carácter estrictamente aleatorio de la búsqueda del LRS al guiado del GRS.

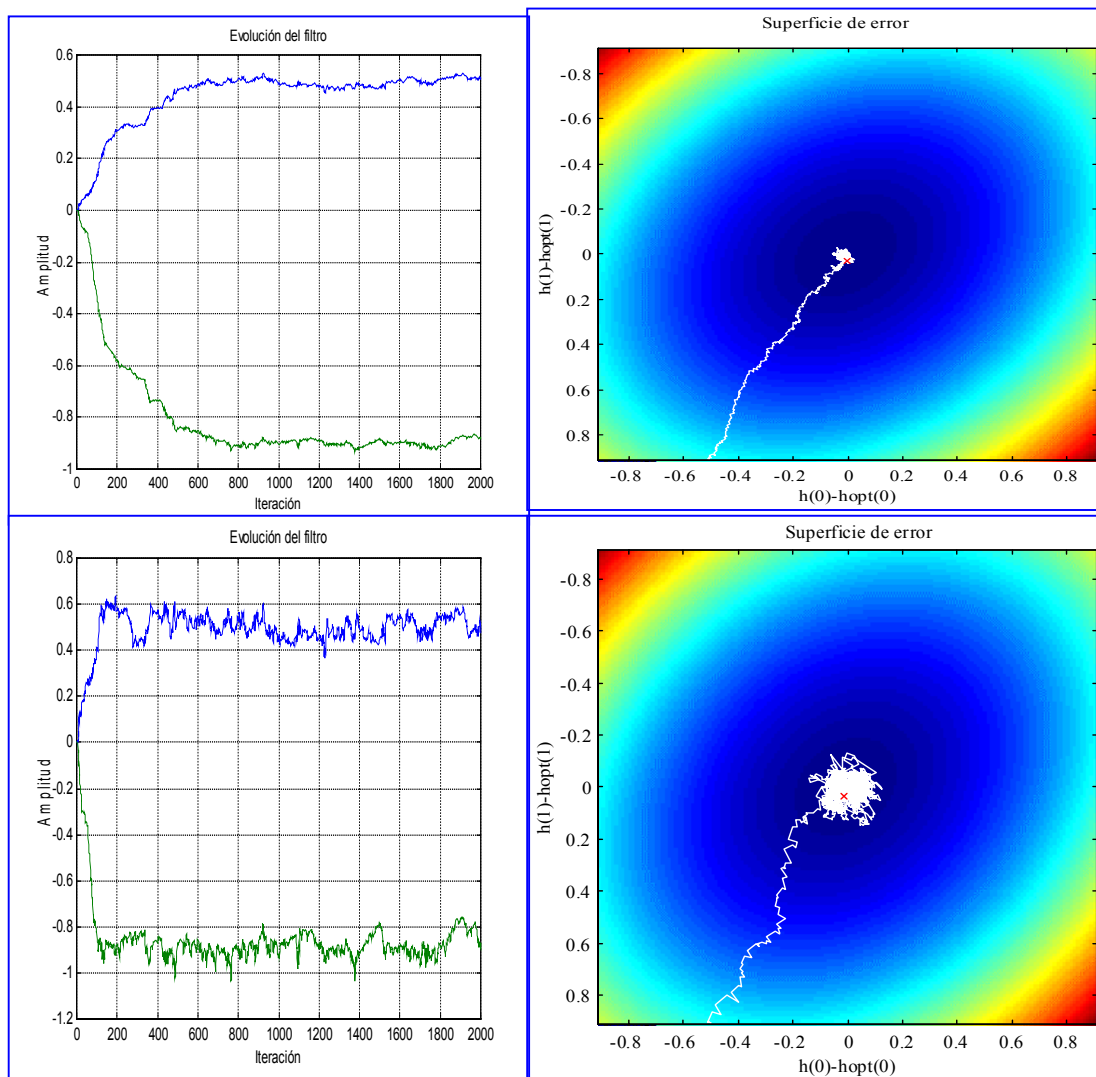
Como algoritmo que no usa el gradiente se ha presentado el RLS. Como se ha podido observar el RLS surge de una manipulación eficiente del álgebra que rodea al proceso de adaptación. Este carácter meramente algebraico promueve que sea superado, por amplitud, flexibilidad y sentido físico por el denominado filtro de Kalman, presentado en último lugar. Aunque de manera breve y de forma original, la presentación relaciona RLS con el LMS y prueba las características más destacables del filtro de Kalman. De hecho es de destacar como cambiar el criterio de mínimo error con la referencia a mínimo error con coeficientes desconocidos permite un diseño tan elaborado y efectivo como el filtro de Kalman.

V.8 EJERCICIOS.

1.- Si se utiliza el algoritmo LMS para la obtención de un filtro de Wiener en el que los datos pueden considerarse estacionarios en una longitud de M muestras, de potencia P_x en dicho intervalo. Además los datos contienen señal mas ruido blanco, este ultimo de potencia σ^2 , responda a las siguientes preguntas:

- Cual es la expresión del parámetro μ para que el desajuste sea de un 10%?
- ¿Que expresión utilizaría para la actualización de la potencia de los datos?
- ¿Cual es el valor de las iteraciones para obtener convergencia en función del desajuste y la relación señal a ruido de los datos?
- En cuanto se incrementa el desajuste si el filtro se implementa con b bits y una dinámica de $\pm A_{\max}$?

2.- Dado un proceso AR(2) generado por el siguiente polinomio $A(z)=1-0.5z^{-1}+0.9z^{-2}$ se emplea el algoritmo LMS para calcular los coeficientes del predictor. Los coeficientes obtenidos a cada muestra así como la evolución sobre las curvas de error se representan en la figura siguiente para dos elecciones diferentes del parámetro del algoritmo:



- Calcule la dispersión de autovalores que presenta el problema.
- Cual es el parámetro de paso de adaptación elegido en cada caso aproximadamente
- Cuánto es el desajuste que se obtiene en cada elección.

3.- Tomando la estructura lattice para la implementación de un predictor lineal, conteste a la siguiente pregunta:

- Dados los Parcours, cual es la relación que iterativamente permite calcular los coeficientes del predictor lineal.

Asumiendo que es cierto que los errores backward son ortogonales entre ellos, puede considerarse que cada sección de la lattice es un filtro de Wiener a diseñar, de izquierda a derecha, independiente de las anteriores. Este filtro de Wiener puede diseñarse también adaptativo empleando el algoritmo LMS (gradiente instantáneo). Considerando que la entrada es estacionaria, responda a las siguientes preguntas:

- Cual es la ecuación de adaptación del Parcor K_q en el instante n , en función de el error forward y el backward? (Note que, al ser la entrada estacionaria, cualquiera de los dos Parcors pueden usarse para contestar esta pregunta ya que ambos serán iguales)
- Cual es la expresión del parámetro μ para tener un desajuste del 10%?
- Demuestre que, para que el desajuste sea el mismo en todas las secciones, el parámetro μ crece a medida que se avanza de sección, siempre de izquierda a derecha. Para ello, deduzca la expresión del parámetro mencionado, el usado en la sección q , en función del utilizado en la anterior y K_{q-1} .
- Considere ahora que el proceso no es estacionario y que el gradiente instantáneo se obtiene, en la sección q , de minimizar

$$\xi_{q+1} = \gamma \cdot E\left(\left[e_{q+1}^f(n)\right]^2\right) + (1 - \gamma) \cdot E\left(\left[e_{q+1}^b(n)\right]^2\right)$$

¿Cuál es la regla de aprendizaje del LMS en esta situación?

4.- Definido un filtro FIR según la ecuación $y(n) = \underline{A}_n^T \cdot \underline{X}_n$ siendo:

$$\underline{A}_n^T = [a(0), a(1), \dots, a(Q-1)]$$

$$\underline{X}_n^T = [x(n), x(n-1), \dots, x(n-Q+1)]$$

donde el superíndice (T) indica transpuesto, se pretende analizar el comportamiento de este cuando se pretende obtener una salida $y(n)$ lo más próxima a una señal de referencia $d(n)$ en sentido de error cuadrático medio ξ .

El método de diseño es adaptativo usando el algoritmo LMS.

Responda a las siguientes preguntas:

a.- Considerando que en el instante n el filtro implementado es $\underline{A}_n$, demuestre que el error cuadrático medio (MSE) ξ vendría dado por la expresión:

$$\xi = \xi_{min} + (\underline{A}_n - \underline{A}_{opt})^T \cdot \underline{R} \cdot (\underline{A}_n - \underline{A}_{opt})$$

donde

$\underline{A}_{opt}$ es la solución de Wiener

$\underline{R}$ es la matriz de autocorrelación de la señal de entrada $x(n)$ de orden Q

ξ_{min} es el MSE de la solución de Wiener

Dado que $\underline{A}_n$, en un algoritmo estocástico como el LMS, pasa a ser un vector de variables aleatorias, entonces también ξ será una variable aleatoria.

b) Demuestre que el valor esperado de ξ viene dado por

$$E(\xi) = \xi_{min} + \text{traza}(\underline{\Sigma} \cdot \underline{R})$$

donde

$\text{traza}(\cdot)$ es la suma de los elementos de la diagonal principal

$\underline{\Sigma}$ es la matriz de covarianza de $\underline{A}_n$

(NOTA: El valor esperado de $\underline{A}_n$ en el algoritmo LMS, con la elección adecuada del parámetro μ es $\underline{A}_{opt}$)

 Considere a partir de ahora que la regla de adaptación de los coeficientes es $\underline{A}_{n+1} = \underline{A}_n + \mu \underline{X}_n (d(n) - y(n))$ y que se encuentra en la zona donde el algoritmo ha convergido, es decir,

$$E[\underline{a}_{n+1} \underline{a}_{n+1}^T] = E[\underline{a}_n \underline{a}_n^T] = \underline{\Sigma} \quad \text{siendo } \underline{a}_n = \underline{A}_n - \underline{A}_{opt}$$

Además, considere para resolver el próximo apartado que $E[\underline{X}_n \varepsilon(n) \underline{a}_n^T] = \underline{R} \underline{\Sigma}$ siendo $\varepsilon(n) = d(n) - y(n)$

 c) Demuestre que la matriz de covarianza es diagonal e igual a:

$$\underline{\Sigma} = \left(\frac{\mu}{2} \right) \xi_{min} \underline{I} \quad \text{siendo } \underline{I} \text{ la matriz identidad}$$

d) Demuestre que el desajuste del algoritmo LMS pasa a ser

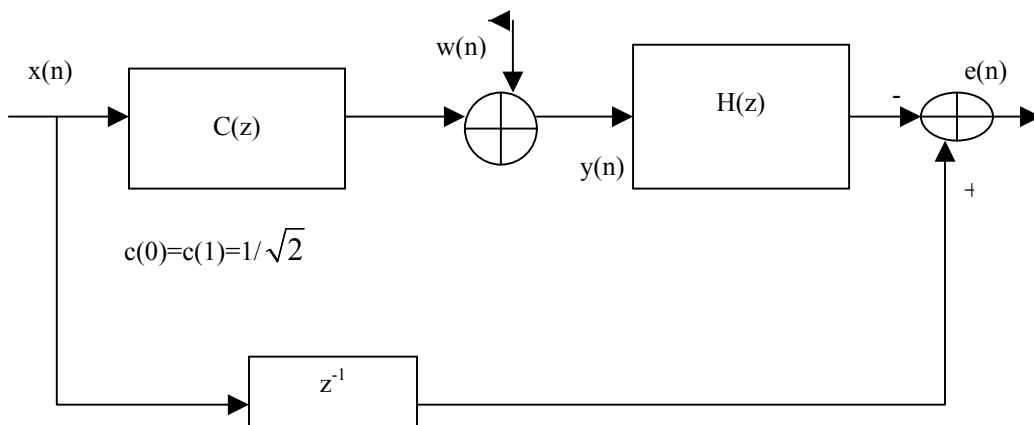
$$M = \frac{E(\xi) - \xi_{min}}{\xi_{min}} = \left(\frac{\mu}{2} \right) \text{traza}(\underline{R})$$

e) Calcule como se incrementa el desajuste del algoritmo si los coeficientes del filtro, comprendidos entre +1 y -1, se cuantifican con b bits y como ha de modificarse el μ para hacer que el desajuste se mantenga el mismo que sin cuantificación de coeficientes.

f) Demuestre que el desajuste permite conocer la relación señal a ruido SNR a la salida del filtro adaptativo en comparación a la optima según la siguiente relación:

$$M = \left(\frac{SNR_{opt}}{SNR} \right) - 1$$

5.- We like to equalize a communications channel with z-transform denoted by $C(z)$ modeled by means an FIR filter with two coefficients $c(0)=c(1)=1/\sqrt{2}$. The input signal $x(n)$ is a BPSK modulation with uncorrelated symbols known at receiver site. With this reference an equalizer $H(z)$ has to be designed using the MSE criterion (Wiener filtering). The noise $w(n)$ is white Gaussian noise with power N_0



a.- Provide the equations that determine the coefficients of the equalizer when its impulse response is of M taps.

b.- Which is the expression of the 2-tap Wiener equalizer as a function of N_0 ?

c.- Write down the expression of the MSE as a function also of N_0 .

Assume that the estimation of the equalizer coefficients is done by means the LMS algorithm.

d.- Provide the updating equations of the LMS algorithm

- c.- Find out the maximum steep size in order to guarantee the convergence of the algorithm
- f.- Describe the impact of the noise power N_0 in the convergence rate of the algorithm
- g.- Assuming tha the missadjustment noise has to be 0.1%, which is the proper set for the steep-size μ
- h.- With the selection done in the previous section, how many updates are necessary to achieve a 1% convergence of the algorithm?

V.9 REFERENCIAS

- [1] B.D.O. Anderson, J.B. Moore. "Optimal Filtering". Prentice Hall. Englewood Cliffs. N.J. 1979.
- [2] R. A. Bonzingo, T.W. Miller. "Introduction to adaptive arrays". John Wiley & Sons. N.Y. 1980.
- [3] B. Widrow. "Adaptive filters I: Fundamentals". Standford Electronic Laboratories, Standford CA, Rept. SEL-66-126 (Tech. Rept. 6746-6). December 1966.
- [4] B. Widrow, P.E. Mantey, L.J. Griffiths, B.B. Goode. "Adaptive antenna systems". Proc. IEEE, Vol. 55, No. 12, pp. 2143-2159, December 1967.
- [5] D.G. Luenberger. "Optimization by vector space methods". Wiley, New York, Chapter 10, 1969.
- [6] R.T. Compton. "An adaptive array in a spread spectrum communications systems". Proc. IEEE, Vol. 66, No. 3, pp. 289-298, March 1978.
- [7] B. Widrow, J.M. McCool. " A comparison of adaptive algorithms based on the method of steepest descent and random search". IEEE Trans. on antennas and propagation, Vol. AP 24, No. 5, pp. 615-638, September 1976.
- [8] D. Godard. "Channel equalization using Kalman filter for fast data transmission". IBM J. Res. Dev. , May 1974, pp. 267-273.
- [9] J.P. Lawrence and F.P. Emad. "An analytic comparison of random searching and gradient searching for the extremum of a known obejctive function". IEEE Trans. oAutom. Control, Vol. AC-18, No. 6, Dec. 1973, pp. 669-671.
- [10] A.N. Mucciardi. "A new class of search algorithms for adaptive computation". Proc. 1973 IEEE conference on Decision and Control, Dec. San Diego, Paper No. WA5-3, pp. 94-100.