Adaptool
Un Entorno Gráfico para Procesamiento Adaptivo

Mario Alcaraz, Juan E. Cousseau, Osvaldo E. Agamennoni

Departamento de Ing. Eléctrica
Universidad Nacional del Sur,
Av. Alem 1253, Bahía Blanca, 8000 Argentina.

Palabras Clave: Entorno gráfico, filtros adaptivos IIR, análisis.

Resumen: En el presente trabajo se presenta un entorno gráfico diseñado en base a un ambiente de cálculo numérico y matricial y una interface gráfica disponibles, que denominamos genericamente **Adaptool**. La premisa fue generar un ambiente para la simulación y evaluación de problemas especificos de filtrado adaptivo de señales, con énfasis en aplicaciones de ecualización, identificación de sistemas y cancelamiento de interferencias con filtros adaptivos de realización IIR. La premisa de diseño del entorno es su modularidad, de forma que el mismo pueda ser expandido tanto en sus aplicaciones como en sus prestaciones, de forma simple y consistente.

Adaptool
Un Entorno Gráfico para Procesamiento Adaptivo

Resumen

En el presente trabajo se presenta un entorno gráfico diseñado en base a un ambiente de cálculo numérico y matricial y una interface gráfica disponibles, que denominamos genericamente **Adaptool**. La premisa fue generar un ambiente para la simulación y evaluación de problemas especificos de filtrado adaptivo de señales, con énfasis en aplicaciones de ecualización, identificación de sistemas y cancelamiento de interferencias con filtros adaptivos de realización IIR. Con ese objetivo se definieron una serie especificaciones respecto al diseño de los distintos elementos del entorno, así como su uso, manejo, y formatos de datos con los cuales trabajar. La premisa de diseño del entorno es su modularidad, de forma que el mismo pueda ser expandido tanto en sus aplicaciones como en sus prestaciones, de forma simple y consistente.

Entre los elementos implementados se cuenta con una serie de controles gráficos y sus funciones auxiliares asociadas, los cuales permiten disponer de algunas de las necesidades básicas del procesamiento adaptivo, como ser generación de señales, algoritmos para filtrado adaptivo y visualización de resultados. A partir de estos elementos es posible configurar aplicaciones dentro del entorno, como por ejemplo identificación de sistemas y cancelamiento de interferencias, quedando abierta la posibilidad de complementarlo con nuevos desarrollos de aplicaciones.

1 Introducción

El procesamiento adaptivo de señales en forma genérica es aquel en el cual el sistema que lo realiza, se adecua a una respuesta característica deseada por el usuario [1][2]. La Figura 1 muestra el diagrama de bloques básico de un sistema adaptivo genérico. En cada intervalo de tiempo, una muestra de la señal de entrada $x(n)$ es procesada por un filtro variante en el tiempo, generando una salida $y(n)$. Está señal es comparada con una referencia $d(n)$ para generar una señal de error $e(n) = d(n) - y(n)$. Finalmente, este error es usado por un algoritmo que ajusta los coeficientes del filtro adaptivo para minimizar un criterio de desempeño determinado.

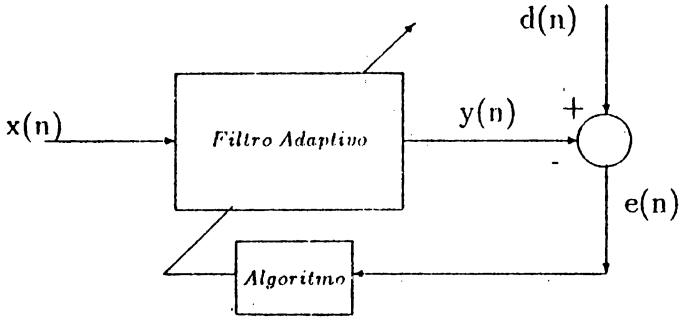

Figura 1: Esquema genérico de filtrado adaptivo.

La caracterización de un sistema adaptivo completo esta asociada a los siguientes aspectos: aplicación, realización del filtro adaptivo y algoritmo de adaptación.

Algunas de las aplicaciones de interés para este trabajo son las siguientes

Identificación de sistemas: En este caso a medida que se minimiza el error por medio del proceso adaptivo (Figura 2), la función transferancia del sistema adaptivo se acerca a una que reproduce en forma aproximada a la del sistema desconocido.

Ecualización: También llamada modelado inverso, pues el objetivo es obtener un modelo semejante a la inversa de la función transferencia de un sistema (Figura 3), para poder reducir en alguna medida los efectos que este hubiera producido sobre una señal de entrada.

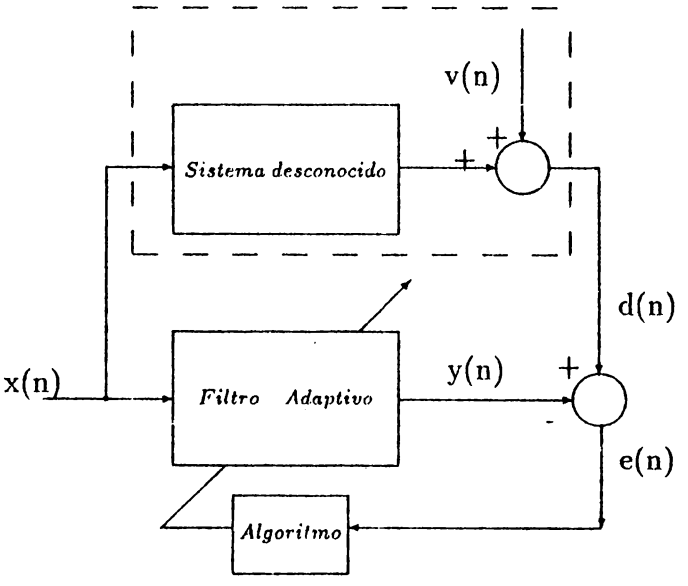

Figura 2: Configuración de identificación de sistemas.

En el aspecto relacionado con la realización del filtro adaptivo, la elección de la estructura está influenciada por la complejidad computacional del filtro y por el número de iteraciones necesarias para alcanzar un determinado nivel de desempeño. Basicamente, podemos considerar dos clases generales de filtros adaptivos

- **Realización adaptiva FIR (respuesta finita al impulso)**: Para esta realización, la salida $y(n)$ es una combinación lineal de los coeficientes del filtro, con una función de error cuadrático medio (MSE definida por $E[c^2(n)]$) con un único mínimo [1].

- **Realización adaptiva IIR (respuesta infinita al impulso)**: Una ventaja potencial frente a la realización FIR es que esta realización modela adecuadamente sistemas físicos debido a su estructura polo-cero. En este caso la salida $y(n)$ dependerá de $x(n)$ y de valores pasados de $y(n)$. Por lo que en general el MSE puede no tener un único mínimo. Tiene una menor complejidad computacional que un filtro FIR con similares características en su respuesta. Aunque es posible tener en cuenta la realización canónica en forma directa para la implementación de filtros IIR adaptivos, la dificultad para mantener la estabilidad de esta estructura lleva a considerar otras realizaciones como por ejemplo la paralela, cascada o lattice [3].

De forma general, los filtros adaptivos IIR todavía no han sustituido a los FIR, debido a problemas de complejidad computational (asociadas al algoritmo de adaptación) y velocidad de convergencia. Por la estructura recursiva de los mismos, se plantean problemas de estabilidad del filtro además de los del algoritmo de adaptación, convergencia a soluciones en mínimos locales y

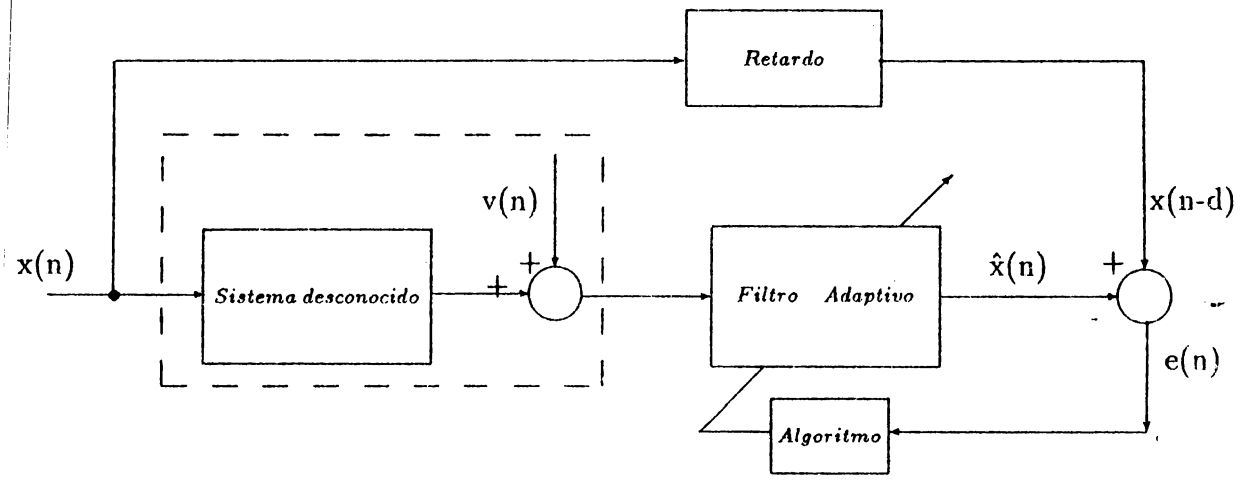

Figura 3: Configuración de ecualización.

una convergencia lenta. En consecuencia, distintos algoritmos para filtrado adaptivo IIR han sido propuestos en la literatura tratando de superar estos problemas.

Con el proposito de disponer de una herramienta de simulación adecuada para el análisis de algoritmos IIR, se construyó el entorno **Adaptool**. En la presentación del trabajo serán discutidos en la sección 2 los algoritmos básicos que forman parte del entorno. En la sección 3 se describen las características escenciales del entorno **Adaptool**. Se incluyen además en la sección 4 algunos ejemplos de aplicación y finalmente las concluciones en la sección 5.

2 Algoritmos

El algoritmo de adaptación es el procedimiento usado para ajustar los coeficientes del fitro adaptivo a fín de minimizar un criterio dado. El algoritmo está determinado por el método de busqueda (o algoritmo de minimización), la función objetivo y la naturaleza de la señal error. La elección del algoritmo determina diversos aspectos cruciales del filtrado adaptivo en general, tales como la existencia de soluciones sub-óptimas u óptimas polarizadas, la velocidad de convergencia y la complejidad computacional del mismo. Se analizará en esta sección una metodología general para los algoritmos IIR, de especial interés en este trabajo.

Una forma de análisis de algoritmos asociados a filtros adaptivos IIR, puede basarse en la aplicación de identificación de sistemas y la estructura adaptiva IIR en forma directa. La señal deseada entonces para esta aplicación en particular se define por

$$d(n) = \frac{B(q^{-1})}{A(q^{-1})}x(n) + v(n) \tag{1}$$

donde $A(q^{-1}) = 1 - \sum_{i=1}^{n_a} a_i q^{-i}$ y $B(q^{-1}) = \sum_{j=0}^{n_b} b_j q^{-j}$ son polinómios sin factores comunes en el operador retardo unitario q^{-1}, y $x(n)$ y $v(n)$ son la señal de entrada y el ruido de medición, respectivamente, considerado en general a este último de media cero y varianza acotada. El filtro adaptivo implementado con la estructura directa está definido por:

$$y(n) = \frac{\hat{B}_n(q^{-1})}{\hat{A}_n(q^{-1})}x(n) \tag{2}$$

donde $\hat{A}_n(q^{-1}) = 1 - \sum_{i=1}^{N} \hat{a}_i(n)q^{-i}$ y $\hat{B}_n(q^{-1}) = \sum_{j=0}^{M} \hat{b}_j(n)q^{-j}$.
Definiendo

$$\begin{aligned}
\theta(n) &= [a_1 ... a_{n_a}; b_0 ... b_{n_b}]^T \\
\phi(n) &= [\frac{B(q^{-1})}{A(q^{-1})}x(n-1) ... \frac{B(q^{-1})}{A(q^{-1})}x(n-n_a); x(n) ... x(n-n_b)]^T \\
\hat{\theta}(n) &= [\hat{a}_1(n) ... \hat{a}_N(n); \hat{b}_0(n) ... \hat{b}_M(n)]^T \\
\hat{\phi}(n) &= [y(n-1) ... y(n-N); x(n) ... x(n-M)]^T
\end{aligned}$$

las ecuaciones (1) y (2) pueden reescribirse como

$$\begin{aligned}
d(n) &= \theta(n)^T \phi(n) + v(n) \\
y(n) &= \hat{\theta}(n)^T \hat{\phi}(n)
\end{aligned}$$

La función objetivo es función directa de una señal de error genérica, que a su vez es función de las señales $x(n)$, $y(n)$ y $d(n)$, esto es, $J(n) = J[e(n)] = J[e(x(n), y(n), d(n))]$. Esta función debe minimizarse para que $y(n)$ se aproxime a $d(n)$. Usando este planteo, se puede considerar que un algoritmo de filtrado adaptivo posee tres aspectos relevantes en particular: definición del algoritmo de minimización, definición de la forma de la función objetivo y definición de la señal error. Estos aspectos determinan una manera simple y estructurada de interpretar, analizar y estudiar un algoritmo en particular. De hecho, la mayoria de los algoritmos adaptivos conocidos pueden ser visualizados así, o con pequeñas variaciones de esta forma [5].

La definición del algoritmo de minimización afecta esencialmente la velocidad de convergencia del proceso adaptivo. Algunos de los métodos mas comunmente usados en el campo del procesamiento adaptivo de señales son:

- Método Gauss-Newton: Esta clase de algoritmos minimiza la función objetivo $J[e(n)]$ usando una estimación recursiva de la inversa de la matriz Hessiana y del gradiente. El algoritmo puede expresarse como:

$$\hat{\theta}(n+1) = \hat{\theta}(n) - \mu P(n) \nabla_{\hat{\theta}} \{J[e(n)]\} \tag{3}$$

donde $\nabla_{\hat{\theta}}\{J[\cdot]\}$ es el operador gradiente, μ es el factor de convergencia y

$$P(n) = \frac{1}{1-\mu}[P(n-1) - \frac{P(n-1)\nabla_{\hat{\theta}}(n-1)\nabla_{\hat{\theta}}^T(n-1)P(n-1)^T}{\frac{1-\mu}{\mu} + \nabla_{\hat{\theta}}^T(-1n)P(n-1)\nabla_{\hat{\theta}}(n-1)}] \tag{4}$$

Además, el vector gradiente $\nabla_{\hat{\theta}}\{J[e(n)]\}$ es reemplazado por una estimación computacionalmente eficiente.

- Método de gradiente estocástico: Con este tipo de algoritmo se busca el mínimo de la función objetivo siguiendo la dirección opuesta al vector gradiente instantáneo de dicha función. En consecuencia, la ecuación de actualización asume la forma

$$\hat{\theta}(n+1) = \hat{\theta}(n) - \mu \nabla_{\hat{\theta}} \{J[e(n)]\} \tag{5}$$

En general, los métodos de gradiente son computacionalmente mas simples, pero el método Gauss-Newton usualmente requieren un menor número de iteraciones para acercarse al punto mínimo. Sin embargo, pueden aparecer problemas de inestabilidad debido a la formas recursivas de estimar la matriz Hessiana inversa. Para el análisis de los distintos algoritmos en este trabajo, se adoptará el método de gradiente estocástico, fundamentalmente por la simplicidad de las expresiones resultantes.

2.1 Algoritmo de Gradiente Recursivo (OE)

Este algoritmo tiene como base la minimización del error medio cuadrático de predicción. Por su relación con métodos generales del área de identificación de sistemas, este método se denomina también de *Máxima Verosimilitud Aproximada* (AML) [5]. Por la forma de obtener las componentes del regresor aqui lo denominamos de *Gradiente Recursivo* (OE). Teniendo en cuenta que como función objetivo. Este algoritmo puede definirse como:

$$\hat{\theta}(n+1) = \hat{\theta}(n) - \mu \nabla_{\hat{\theta}}[e^2_{OE}(n)] \tag{6}$$

donde $e_{OE}(n) = e(n) = d(n) - y(n)$ es el error de predicción o error de salida. Desarrollando el gradiente se obtiene la siguiente expresión

$$\frac{1}{2}\nabla_{\hat{\theta}}[e^2_{OE}(n)] = e_{OE}(n)\nabla_{\hat{\theta}}[e_{OE}(n)] = -e_{OE}(n)\nabla_{\hat{\theta}}[y(n)] \tag{7}$$

Asumiendo que los parámetros varían lentamente, puede derivarse con respecto a $\hat{\theta}(n)$, donde despues de algunas simplificaciones [5] se obtiene:

$$\begin{aligned} \nabla_{\theta}[y(n)] &\cong [\tfrac{1}{\hat{A}_{n-1}(q^{-1})}y(n-1) \cdots \tfrac{1}{\hat{A}_{n-N}(q^{-1})}y(n-N)\ \tfrac{1}{\hat{A}_n(q^{-1})}x(n) \cdots \tfrac{1}{\hat{A}_{n-M}(q^{-1})}x(n-M)]^T \\ &= [y^f(n-1) \cdots y^f(n-N)\, x^f(n) \cdots x^f(n-M)]^T \\ &= \hat{\phi}_{OE}(n) \end{aligned} \tag{8}$$

donde $y^f(n) = \frac{1}{\hat{A}_n(q^{-1})}y(n)$ y $x^f(n) = \frac{1}{\hat{A}_n(q^{-1})}x(n)$

El cálculo de las componentes del vector gradiente requiere un filtrado de las señales de entrada y salida, para obtener las componentes del regresor $\hat{\phi}_{OE}(n)$. Por lo tanto se puede expresar el algoritmo OE como

$$\hat{\theta}(n+1) = \hat{\theta}(n) + \mu e_{OE}(n)\hat{\phi}_{OE}(n) \tag{9}$$

Se observa entonces que existen tres elementos dentro de la actualización de los parámetros: el factor de velocidad de convergencia μ, el error $e_{OE}(n)$ y el regresor $\hat{\phi}_{OE}(n)$. La característica mas importante del algoritmo OE es que solo puede ser garantizada su convergencia en forma local, por lo que la posible existencia de mínimos locales pueden afectar la convergencia total del algoritmo.

2.2 Algoritmo de Error de Ecuación (EE)

Con el algoritmo *Least Mean Squares Equation Error* [2], o de Error de Ecuación (EE), se pretende superar el problema de la existencia de mínimos locales del algoritmo anterior, logrando una relación lineal entre la señal de referencia y los coeficientes del filtro adaptivo. Para esto, se elige la señal error como

$$e_{EE}(n) = \hat{A}_n(q^{-1})d(n) - \hat{B}_n(q^{-1})x(n) \tag{10}$$

El gradiente de la función objetivo queda en este caso

$$\frac{1}{2}\nabla_{\hat{\theta}}[e^2_{EE}(n)] = e_{EE}(n)\nabla_{\hat{\theta}}[e_{EE}(n)] \tag{11}$$

o en forma explícita

$$\nabla_{\hat{\theta}}[e_{EE}(n)] = \nabla_{\hat{\theta}}[\hat{A}_n(q^{-1})d(n) - \hat{B}_n(q^{-1})x(n)] = [d(n-1) \ldots d(n-N); x(n) \ldots x(n-M)]^T = \hat{\phi}_{EE}(n) \tag{12}$$

Se puede apreciar que es mas simple computacionalmente que el método de Gradiente Recursivo, debido a que solo se compone de muestras anteriores de la entrada y la salida deseada. Queda entonces la siguiente expresión para el algoritmo de adaptación:

$$\hat{\theta}(n+1) = \hat{\theta}(n) + \mu e_{EE}(n)\hat{\phi}_{EE}(n) \tag{13}$$

La desventaja que presenta este algoritmo es la polarización de la solución resultante ante la presencia de ruido de medición.

2.3 Algoritmo Steiglitz-McBride (SM)

En este algoritmo se combinan las buenas características de los algoritmos OE y EE, solución no polarizada y global única, respectivamente. Para alcanzar este objetivo, el algoritmo Steiglitz-McBride [4] está basado en una señal error $e_{SM}(n)$, que es una función lineal de los coeficientes del filtro adaptivo. La intepretación física de la señal error es similar a la del algoritmo OE, teniendo una solución global no polarizada. La señal error $e_{SM}(n)$ está dada por:

$$e_{SM}(n) = [\frac{\hat{A}_n(q^{-1})}{\hat{A}_{n-1}(q^{-1})}]e_{EO}(n) = [\frac{1}{\hat{A}_{n-1}(q^{-1})}]e_{EE}(n) \tag{14}$$

El vector gradiente para está señal es

$$\frac{1}{2}\nabla_{\hat{\theta}}[e^2_{SM}(n)] = e_{SM}(n)\nabla_{\hat{\theta}}[e_{SM}(n)] = -e_{SM}(n)[\frac{1}{\hat{A}_{n-1}(q^{-1})}]\hat{\phi}_{EE}(n) \tag{15}$$

Considerando la aproximación de variación lenta de los parámetros es posible obtener el regresor para este algoritmo es el siguiente [6]

$$\begin{aligned}
\tfrac{1}{\hat{A}_{n-1}(q^{-1})}\hat{\phi}_{EE}(n) &= [\tfrac{1}{\hat{A}_{n-1}(q^{-1})}d(n-1)\cdots\tfrac{1}{\hat{A}_{n-1}(q^{-1})}d(n-N)\ \tfrac{1}{\hat{A}_{n-1}(q^{-1})}x(n)\cdots\tfrac{1}{\hat{A}_{n-1}(q^{-1})}x(n-M)] \\
&\cong [\tfrac{1}{\hat{A}_{n-1}(q^{-1})}d(n-1)\cdots\tfrac{1}{\hat{A}_{n-N}(q^{-1})}d(n-N)\ \tfrac{1}{\hat{A}_{n}(q^{-1})}x(n)\cdots\tfrac{1}{\hat{A}_{n-M}(q^{-1})}x(n-M)] \\
&= [d^f(n-1)\cdots d^f(n-N)\,x^f(n)\cdots x^f(n-M)]^T \\
&= \hat{\phi}_{SM}(n)
\end{aligned} \tag{16}$$

de forma que la ecuación de adaptación queda

$$\hat{\theta}(n+1) = \hat{\theta}(n) + \mu e_{SM}(n)\hat{\phi}_{SM}(n) \tag{17}$$

Existen otros algoritmos para filtros adaptivos IIR, con distintas características e implementaciones, pero todos ellos tienen los mismos elementos en común, que son el error y el regresor. Por otra parte la estructura del filtro adaptivo puede variar (sobre todo para simplificar el test de estabilidad), pudiendo ser alternativamente paralela, cascada, lattice, etc.. También puede variar el método de optimización (utilizando por ejemplo métodos de gradientes conjugados o descomposición QR de la inversa de la matriz Hessiano [1]). Sin embargo el regresor y el error siempre pueden discriminarse como estructuras en común.
En la sección siguiente se insertarán los algoritmos estudiados dentro de un contexto de filtrado adaptivo, para mostrar los requerimientos necesarios para su funcionamiento y evaluación.

3 Filtrado adaptivo IIR: Entorno Adaptool

En la sección anterior se planteo la forma genérica que adoptan algunos algoritmos para filtrado adaptivo: filtrado propiamente dicho, cálculo del error, cálculo del regresor y actualización de los

coeficientes. Estos pasos se deben repetir iterativamente sobre las señales que ingresan al procesador adaptivo, las cuales están dadas por la aplicación utilizada. Se aprecian entonces algunas de los elementos necesarios dentro de un entorno de procesamiento adaptivo, los cuales deben ser implementados.

La herramienta fundamental es el procesador adaptivo, el cual debe implementar diversos algoritmos de adaptación y minimización, sobre diversas estructuras de filtrado. Este debe ser insertado facilmente dentro de las aplicaciones particulares, que implementarán las diversas configuraciones de procesamiento adaptivo que se deseen evaluar. Por otra parte, se necesita disponer de señales sobre las cuales actuar. Estas señales pueden ser generadas en el entorno, o poder ser ingresadas desde el exterior, por ejemplo desde otros entornos de procesamiento de señales [9].

Finalmente se debe poder evaluar la aplicación realizada, a fín de analizar las ventajas y desventajas de los diversos algoritmos. Una herramienta para esto es la curva de aprendizaje, la cual consiste en un gráfico del error cuadrático en función del tiempo. Otras formas de visualización del proceso adaptivo se refieren a los coeficientes del filtro adaptivo, observandose su evolución en el tiempo. Por ejemplo, en el caso de los filtros IIR, es común graficar en el caso de un denominador de segundo orden, el coeficiente de orden uno versus el termino independiente. Superponiendo a este gráfico una curva de niveles de la función de error, se aprecia de que forma se acercan los coeficientes del filtro a una solución.

A partir de las necesidades planteadas anteriormente para el filtrado adaptivo, se construyó el conjunto de funciones Adaptool, basado en el programa *MATLAB for Windows* [7]. Este último es un ambiente computacional para procesamiento y visualización numérica de alto desempeño que integra análisis numérico, calculo matricial, procesamiento de señales y gráficos. Posee además facilidades para el uso y creación de una interface gráfica [8]. Esta interface consiste en una serie de objetos gráficos, tales como menues, botones, listas y campos, los cuales al ser seleccionados por el usuario producen una acción determinada en el entorno.

3.1 Controles Adaptool

En la implementación del entorno para el filtrado adaptivo **Adaptool**, se definió una serie de controles creados a partir de objetos graficos de MATLAB. Cada uno de estos controles está destinado a la selección de opciones e ingreso de valores que luego serán utilizados por funciones específicas del entorno de adaptación. Además, dentro del funcionamiento del mismo control está la posibilidad de seleccionar sobre que funciones se aplicarán estas selecciones.

Todo el conjunto de posibilidades que posee un control se halla programada dentro de una función MATLAB que controla su creación y funcionamiento. Estas funciones tienen una sintáxis semejante, ya que en sus parámetros se debe ingresar un manipulador que identifique al control correspondiente, una cadena de caracteres donde se indica la tarea a realizar sobre el control, y algunos parámetros opcionales. El manipulador es similar a los utilizados por los controles MATLAB y es un valor número asignado en el momento de su creación para individualizarlo. Las tareas que puede ejecutar un control son su creación, asignación de valores al control desde la función en lugar del usuario por medio de la interface gráfica, lectura de estos valores y de selecciones hechas en el mismo, etc. De esta forma, cualquier función o aplicación que acceda a estas funciones puede obtener la información que el usuario ingreso en el control.

Estos controles permiten alcanzar un alto grado de modularidad en la construcción de aplicaciones dentro del entorno, ya que no es necesario programar en cada una los distintos elementos de la interface gráfica. Por otra parte, si se realiza una modificación en uno de estos controles, como agregar nuevas opciones, la misma estará disponible en todas las aplicaciones que los use.

Los controles disponibles son los siguientes:

- *Sggencal*: Generador de señales determinísticas.

- *Noisecal*: Generador de señales estocásticas.
- *Ldveccal*: Cargar y salvar vectores de señales en variables o archivos.
- *Ldmatcal*: Cargar y salvar matrices en variables.
- *Stfircal*: Filtro adaptivo FIR (filtrado de la señal y cálculo de gradientes de los coeficientes).
- *Stiircal*: Filtro adaptivo IIR(filtrado de la senñal y cálculo de gradientes delos coeficientes).
- *Minimcal*: Minimización de función objetivo (actualización de coeficientes del filtro adaptivo).
- *Dialocal*: Ventana para mensajes de error.

A partir de ellos se pueden construir las aplicaciones propiamente dichas, siendo solo necesario utilizar la función correspondiente a cada control para su creación y posterior manejo (modificar párametros, obtener los ingresados por el usuario,etc.).

Existen una serie de funciones relacionadas con algunos de los controles (*Stfircal* , *Stiircal* , *Minimcal*). En estos controles existe la posibilidad de seleccionar diversos métodos de minimización o algoritmos da adaptación, los cuales están disponibles como funciones MATLAB, distintas a las del control. Cuando se hace una selección en un control, se obtiene el nombre de las funciones correspondientes a la selección, las que son invocadas luego por la aplicación que hace uso del control. Con esto se logra una mayor velocidad, y flexibilidad en la implementación, o sea, solo se debe modificar el control gráfico para ingresar la nueva opción. Las funciones implementan el cálculo del error y el regresor en los algoritmos de adaptación y la actualización de los coeficientes en los métodos de minimización. Por la existencia de los puntos en común vistos anteriormente, pueden independizarse los diversos métodos y algoritmos, siendo solo necesario respetar la convención establecida para el paso de los parámetros correspondientes. Esto es utilizado por las aplicaciones para obtener los diversos valores calculados por las funciones sin necesidad de tener en cuenta el algoritmo específico.

3.2 Aplicaciones

El siguiente nivel dentro del entorno Adaptool es el de las aplicaciones, herramientas gráficas para resolver una situación específica de procesamiento adaptivo, construidas a partir de los controles gráficos vistos anteriormente. Se observan varias categorias de aplicaciones: las destinadas a la generación y manipulación de señales, los procesadores adaptivos propiamente dichos, las diversas configuraciones adaptivas y aquellas para la evaluación de los resultados obtenidos. Las aplicaciones disponibles son las siguientes:

- *Generad*: generación de señales de diversa naturaleza.
- *ModPlant*: modelado de plantas AR, MA y ARMA con agregado de ruido de medición.
- *AdaptFIR*: procesamiento adaptivo FIR.
- *AdaptIIR*: procesamiento adaptivo IIR.
- *SigPlot*: graficado de señales.
- *TrayPlot*: graficación de trayectoria de coeficientes.
- *Cancel*: cancelamiento adaptivo de interferencias.

- *Equaliz*: ecualización adaptiva.
- *Ident*: identificación adaptiva de sistemas.
- *Ensemble*: promediado de curvas de aprendizaje.

El protocolo para el manejo de las aplicaciones es similar al existente en los controles gráficos. Existe una función MATLAB con la cual se crea cada una de las aplicaciones dentro del entorno, pudiendo existir multiples instancias de las mismas. Para identificar a cada una, al ser creadas, la función correspondiente devuelve un manipulador, de la misma manera que con los controles gráficos. Sin embargo fué necesario agregar un protocolo de identificación de las distintas instancias de las aplicaciones, teniendo en cuenta el momento de su creación. Esto es debido a que el programa MATLAB puede repetir un mismo identificador del tipo usado por estas dentro de una misma sesión de trabajo, pudiendo causar problemas para individualizarlas.

El formato gráfico de las aplicaciones es similar en todas ellas: una ventana con una serie de controles gráficos **Adaptool** y una barra de ménu. Con los primeros se puede seleccionar diversos parámetros dentro de la aplicación. En la barra de menú se disponen de diversas opciones, algunas de las cuales permiten abrir otras aplicaciones. Por ejemplo, desde una aplicación de identificación de sistemas, se puede abrir una instancia para la generación de señales, las cuales son utilizadas por la primera. De esta manera se logra una interconexión rápida y simple entre las mismas. Se simplifica además de esta forma la programación de las mismas al poder disponer en una aplicación de funciones ya implementadas por otras.

4 Utilización del entorno de procesamiento adaptivo

Para ejemplificar el uso del entorno **Adaptool**, se utilizará la configuración de identificación de sistemas. Como primer paso se inicia la aplicación de identificación Ident, invocandola desde la linea de comandos del MATLAB por medio del comando **ident**. Se tiene entonces disponible la ventana correspondiente, que solo posee una barra de menú, desde donde se abren las aplicaciones necesarias para la identificación: modelado de planta (*ModPlant*) y procesador adaptivo (*AdaptFIR* o *AdaptIIR*). Aunque estas aplicaciones son independientes de la de identificación, en este contexto aparecen asociadas a la misma. Esto se refleja en los nombres que las mismas adoptan. En este caso se utilizará un filtro adaptivo IIR.

Con la aplicación *ModPlant* se modela el sistema desconocido, pudiendose elegir el tipo de modelo a utilizar (AR, MA, ARMA) desde la barra de menú. Se desea modelar una planta con la siguiente función transferencia

$$y_0(n) = \frac{0.5 - 0.3q^{-1}}{1 + 0.3q^{-1} - 0.2q^{-2}} x(n) \tag{18}$$

Se selecciona entonces un filtro ARMA en el menú **Plant** y luego desde la linea de comandos de MATLAB se ingresa los coeficientes del filtro por medio de las ordenes

```
>>a=[1, -.3, .2]; b=[.5, -.3];
```

Finalmente, en los controles correspondientes se ingresan los nombres de las variables del entorno MATLAB que contienen los coeficientes necesarios para el modelado y se los lleva a la aplicación. También puede agregarse un ruido de medición, especificando la relación señal-ruido deseada. En el menú **Noise** se selecciona el tipo de ruido, apareciendo el control correspondiente. Como se observa en la Figura 4 se seleccionó ruido gaussiano de media cero y varianza igual a uno, con una frecuencia de muestreo $f_s = 1$ Hz y duración 1000 seg. La relación señal ruido se eligó de 60 dB.

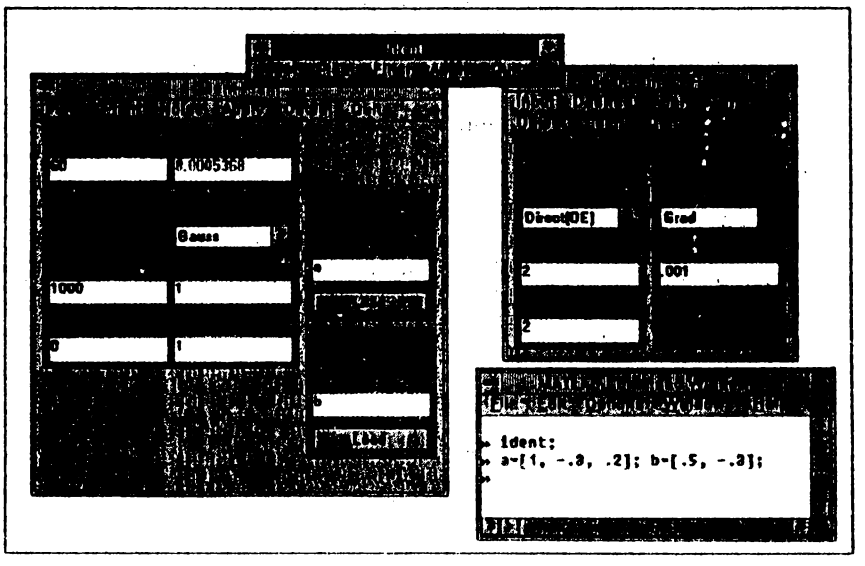

Figura 4: Identificación, modelado de planta y filtro adaptivo IIR.

La señal de entrada a la planta se obtiene de una aplicación **Generad**, asociada a la de modelado, ya que se inicia desde uno de los menues de esta. No solo se pueden generar señales de distinta naturaleza, sino también obtener aquellas almacenadas en archivos o variables del entorno MATLAB. Si se desea como entrada ruido Gaussiano, desde el menú se activa el control correspondiente, y luego se ingresan los parámetros para el mismo; en este caso media cero, varianza igual a uno y para ser consistente con las demás señales del ejemplo, la misma frecuencia de muestreo y duración ingresadas anteriormente (Figura 5).

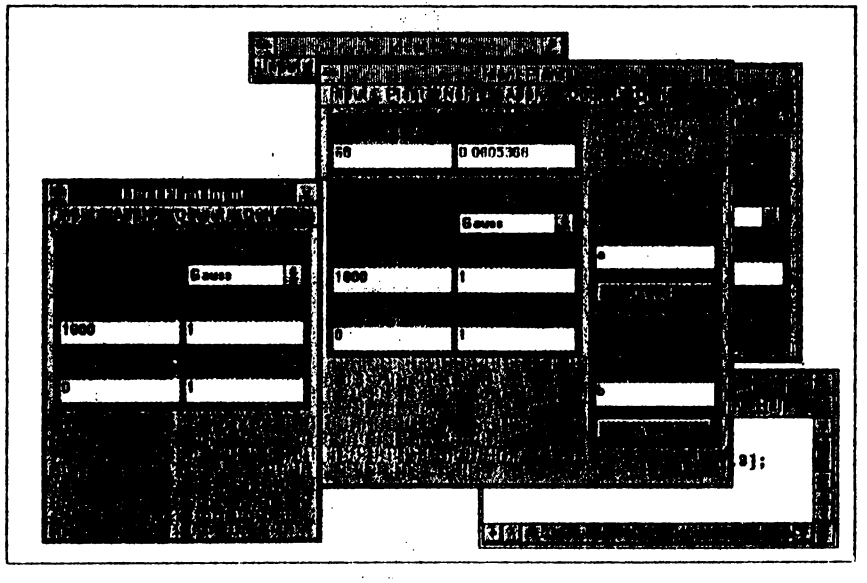

Figura 5: Generación de entrada de la planta

Si se desea, puede ejecutarse una aplicación, por ejemplo la de modelado, separada del resto de la configuración. Pulsando el menú **Apply** en ella, se activa **Generad** y luego la de modelado. Desde el menú **Output** de ambas se pueden obtener las señales correspondientes, para almacenar o graficar. La opción de graficado invoca una aplicación **Sigplot** (Figura 6). En la misma pueden variarse los parámetros del gráfico, por medio de un control propio.

Finalmente para ejecutar la aplicación de identificación de sistemas, se deben ingresar los

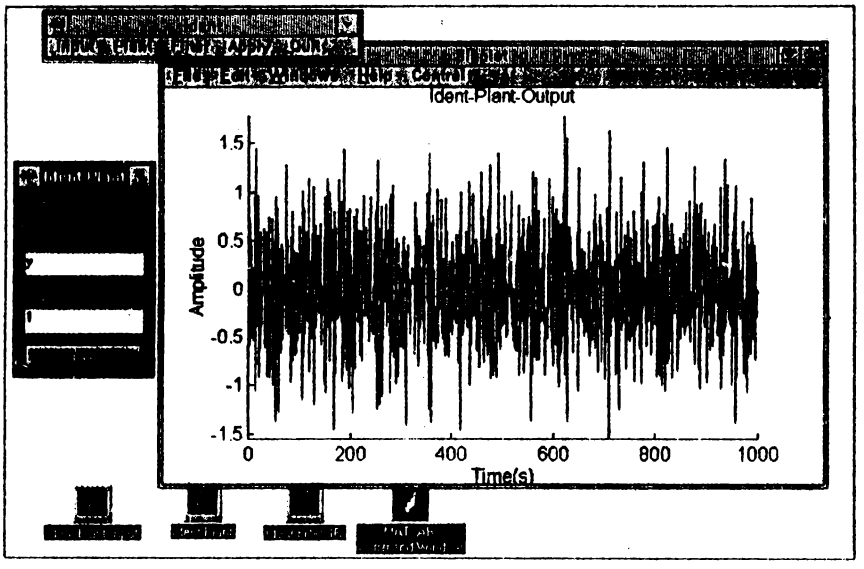

Figura 6: Generación de la salida de la planta

parámetros que configuren al procesador adaptivo, en cuanto a algoritmo y método de minimización a utilizar. A través de los controles gráficos correspondientes, se especifica tipo de estructura y algoritmo, cantidad de coeficientes del filtro y factor de convergencia (Figura 7).

En este ejemplo se utilizó un filtro IIR con realización directa con numerador y denominador de segundo orden, adaptado por medio de un algoritmo OE con minimización por el método de gradiente, con factor de convergencia $\mu = 0.01$. También puede elegirse como se visualizará la evolución de los coeficientes durante la adaptación, por ejemplo como trayectoria de los mismos. Con el menú **Apply** de la aplicación de identificación, se inicia el proceso de adaptación, poniendose en funcionamiento todas las demás. Una vez finalizado el mismo, en el procesador adaptivo pueden obtenerse los resultados (gráficos y vectores de coeficientes del filtro). Los coeficientes se deben salvar como variables en el entorno MATLAB, estando disponibles para otras aplicaciones o funciones.

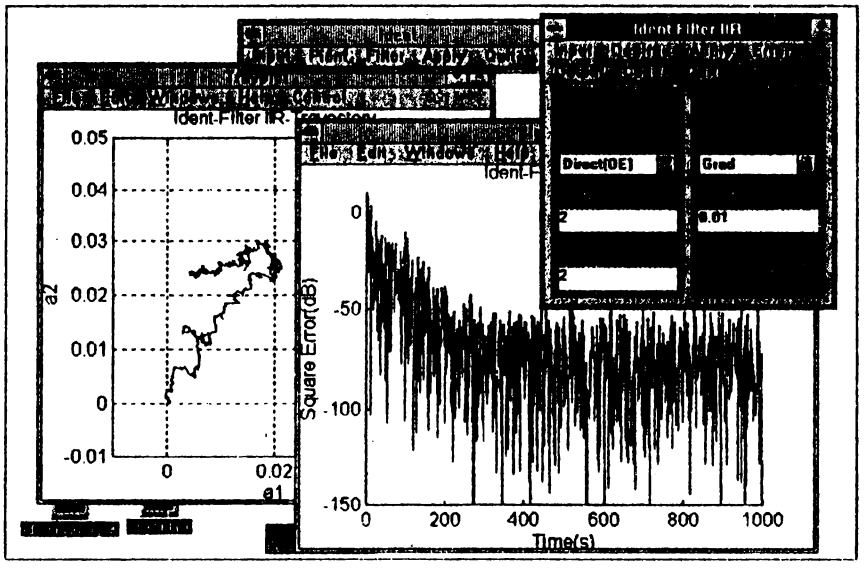

Figura 7: Aplicación de identificación de sistemas.

5 Conclusiones

Debido a que el análisis de los algoritmos de filtrado adaptivo IIR es complejo y se encuentra en constante avance, es necesario disponer de herramientas adecuadas, tanto para el estudio de los algoritmos existentes como para el análisis o el disen o de nuevos algoritmos. En este trabajo se presentó una herramienta para la simulación de filtros adaptivos IIR para satisfacer los requerimientos anteriores. Se buscaron los bloques básicos constitutivos del filtrado adaptivo IIR, poniendo en evidencia sus características, especificaciones y puntos en común. A partir de esto se construyó dentro de un ambiente gráfico destinado al cálculo numérico, una serie de herramientas para la implementación de los diversos algoritmos de adaptació IIR, así como aplicaciones que hagan uso de los mismos. De fundamental importancia en la definición de los mismos fue la modularidad, con el objeto de poder realizar distintas aplicaciones de una forma simple y consistente, así como la posibilidad de expansión de los mismos, para poder incorporar en el futuro nuevos algoritmos, configuraciones y prestaciones al entorno.

Referencias

[1] S. Haykin, *Adaptive Filter Theory*, Prentice-Hall, Englewood-Cliffs, 1991.

[2] B. Widrow y S. Stearns, *Adaptive Signal Procesing*, Prentice-Hall, Englewood-Cliffs, 1985.

[3] J. J. Shynk, "Adaptive IIR filtering", *IEEE Acoustics, Speech and Signal Processing Magazine*, pp. 4–21, Abril 1989.

[4] K. Steiglitz y L. F. McBride, "A technique for the identification of linear systems", *IEEE Trans. on Automatic Control*, vol. AC-10, pp. 461–464, Julio 1965.

[5] L.Ljung y T. Soderstrom, *Theory and Practice of Recursive Identification*, The MIT Press, Cambridge, 1983.

[6] H. Fan y W.K. Jenkins, "A new adaptive IIR filter", IEEE Trans. on Circuits and Systems, vol. CAS-33, pp.939-947, Octubre 1986.

[7] *MATLAB User's Guide*, The MathWorks,Inc., Agosto 1992.

[8] *Building a Graphical User Interface*, The MathWorks,Inc., Octubre 1993.

[9] D. W. Brown *SPC Toolbox User's Guide*, Naval Postgraduate School, Monterey, CA., Mayo 1994.