CAPITULO 3

MODELO DE COMPORTAMIENTO PARA AMPLIFICADORES DE POTENCIA USANDO SERIES DE VOLTERRA MODIFICADAS

3.1. INTRODUCCION

Este proyecto se basa íntegramente en el artículo [1]. En él se propone la simplificación de los modelos de comportamiento basados en series de Volterra con el fin de reducir su complejidad para PAs. Se expone una nueva forma de representar las series de Volterra completamente adaptada para la formulación en el dominio discreto. Un nuevo modelo que puede ser fácilmente extraído tomando muestras en el dominio del tiempo, de la entrada y la salida del amplificador.

Como se comentó en el capítulo anterior, las no linealidades y los efectos de memoria serán muy importantes a la hora de modelar PAs en diseños de sistemas de banda ancha. Las series de Volterra, puesto que son válidas para dicho fin, han sido usadas por muchos investigadores. Sin embargo, como también se comentó, en casos de PAs con no linealidades fuertes y/o con efectos de memoria a largo plazo, la complejidad que alcanzan estos modelos los invalida. Complejidad, que se debe a que el número de coeficientes a estimar en el modelo se incrementa con el grado de no linealidad y con la longitud de la memoria del sistema.

Entonces, para vencer las limitaciones de las series de Volterra, en [1] se proponen las series de Volterra Modificadas. Y como además, los modelos propuestos para señales de onda continua no son simulables en programas como Matlab, Scilab,..., se trabaja con series de Volterra Modificadas en el **dominio discreto**.

3.2. SERIES DE VOLTERRA MODIFICADAS

Las series de Volterra Modificadas van a tener la propiedad de separar los efectos puramente estáticos de los efectos de memoria, íntimamente relacionados en las series clásicas. Suponiendo que la duración de la memoria es lo suficientemente corta respecto al periodo de la señal, las series de Volterra clásicas podrán ser truncadas a un sumatorio finito que va a permitir modelar no solo las no linealidades débiles sino también las más fuertes. Serán las series de Volterra Modificadas.

En un principio, un sistema no lineal cualquiera de **orden finito** con **memoria finita** puede ser representado por una serie de Volterra truncada en el dominio discreto como:

$$y(n) = \sum_{p=1}^P y_p(n) \quad (1)$$

donde:

$$y_p(n) = \sum_{i_1=0}^{N-1} \dots \sum_{i_p=0}^{N-1} h_p(i_1, \dots, i_p) \prod_{j=1}^p x(n-i_j) \quad (2)$$

$x(n)$ e $y(n)$ son la entrada y la salida respectivamente y $h_p(i_1, \dots, i_p)$ los kernel de Volterra. **P es el orden mas alto de no linealidad, y N representa la máxima longitud de memoria del sistema.**

Es muy difícil identificar kernels de orden superior a cinco, por lo que esta formulación solo serviría para sistemas **débilmente no lineales**. Para superar este obstáculo se propone introducir la **función de desviación dinámica** $e(n, i)$:

$$e(n, i) = x(n-i) - x(n) \quad (3)$$

Sustituyendo (3) en (2) vemos que (1) puede ser expresada a través de:

$$y(n) = y_s(n) + y_d(n) \quad (4)$$

donde $y_s(n)$ representa la característica **estática** del sistema, y puede ser expresada como la serie de potencia de $x(n)$:

$$y_s(n) = \sum_{p=1}^P a_p x^p(n) \quad (5)$$

mientras, $y_d(n)$ es una convolución puramente **dinámica** y multidimensional en lo que concierne a $e(n,i)$ y esta controlada por $x(n)$;

$$y_d(n) = \sum_{p=1}^P \sum_{r=1}^p x^{p-r}(n) \sum_{i_1=0}^{N-1} \dots \sum_{i_r=0}^{N-1} g_{p_r}(i_1, \dots, i_r) \prod_{j=1}^r e(n, i_j) \quad (6)$$

En muchas aplicaciones es posible **truncar la memoria** del sistema a un intervalo de tiempo relativamente corto y centrado en el instante en el que se evalúa la salida. Si dicho tiempo de memoria es “suficientemente corto”, entonces la $e(n,i)$ toma valores pequeños. Luego $y_d(n)$ puede ser linealizada con respecto a $e(n,i)$ conduciendo a:

$$y(n) = \sum_{p=1}^P a_p(n) x^p(n) + \sum_{p=1}^P x^{p-1}(n) \sum_{i=0}^{N-1} g_{p_1}(i) e(n, i) \quad (7)$$

donde solamente el kernel de primer orden de la parte dinámica tiene que ser identificado para caracterizar sistemas no lineales con memoria.

En este modelo, las no linealidades estáticas y desviación dinámica pueden ser estimadas por separado. La parte estática es generalmente extraída bajo condiciones de gran señal, mientras que la $g_{p_1}(i)$ bajo condiciones de pequeña señal.

Sin embargo, si resustituimos (3) en (7), obtenemos:

$$y(n) = \sum_{p=1}^P h_p(n) x^p(n) + \sum_{p=1}^P x^{p-1}(n) \sum_{i=0}^{N-1} h_{p_1}(i) x(n-i) \quad (8)$$

Comparando (7) con (8), la ventaja de este último es que la salida es lineal respecto a los coeficientes o kernels, propiedad que también poseía la serie clásica de Volterra en (1). **Esta propiedad de linealidad nos permitirá emplear algoritmos de identificación de sistemas lineales para extraer el modelo de un modo directo.**

Con un formato similar a (8) podríamos reescribir (1) como:

$$y(n) = \sum_{p=1}^P h_p(n) x^p(n) + \sum_{p=1}^P \left\{ \sum_{r=1}^p \left[x^{p-r}(n) \sum_{i_1=1}^{N-1} \dots \sum_{i_r=i_{r-1}}^{N-1} h_{p_r}(i_1, \dots, i_r) \prod_{j=1}^r x(n-i_j) \right] \right\} \quad (9)$$

Comparando con (2), podemos ver que la secuencia de productos de la entrada ha sido reorganizada en (9). Más adelante, emplearemos esta representación para llegar a un modelo más reducido.

3.2.1. Reducción de la desviación

En la nueva representación de las series de Volterra (9), r esta directamente relacionada con el posible orden de los productos de las entradas retrasadas. Por tanto, $r = 1$ significa que solamente una de las entradas retrasadas esta incluida en el producto, $x^{p-1}(n)x(n-i_j)$. Es fácil ver por tanto que (8) es igual a truncar el modelo general (9) tomando $r = 1$.

Truncar al primer orden la parte dinámica permite modelar con exactitud sistemas altamente no lineales, pero su efectividad esta limitada sólo a aquellos sistemas donde la duración de la memoria es pequeña comparada con el inverso del ancho de banda de la señal envolvente. Desafortunadamente en la práctica nos encontraremos muchas veces que la duración de la memoria es mayor comparada con el inverso del ancho de banda de la señal envolvente.

Pero afortunadamente, también en la práctica, los efectos de memoria en los amplificadores tienden a declinar con el tiempo, lo que significa que las entradas con grandes tiempos de retraso tienen menos efecto en la señal de salida.

Por tanto, es razonable limitar r a un valor a un valor pequeño, entre 1 y M , para reducir la complejidad del modelo pero manteniendo una buena exactitud del mismo. En [1], se llama a esto "Reducción de la desviación", y r representa el **orden de la desviación dinámica**.

Por ejemplo, tomando $1 \leq r \leq 2$, la parte dinámica en (9) queda truncada al segundo orden. Nuestro modelo de Volterra será:

$$\begin{aligned}
 y(n) = & \sum_{p=1}^P h_p(n) x^p(n) \\
 & + \sum_{p=1}^P \left[x^{p-1}(n) \sum_{i=1}^{N-1} h_{p_1}(i) x(n-i) \right] \\
 & + \sum_{p=2}^P \left[x^{p-2}(n) \sum_{i_1=1}^{N-1} \sum_{i_2=i_1}^{N-1} h_{p_2}(i_1, i_2) x(n-i_1) x(n-i_2) \right]
 \end{aligned} \tag{10}$$

La decisión de a que orden queremos truncar depende en la práctica de las características del amplificador y de la fidelidad que se requiera.

3.2.2. Modelo de Volterra paso baja equivalente

En el análisis y el diseño de los sistemas de comunicación, la mayoría de los simuladores usan envolventes complejas en banda base para evaluar el funcionamiento del sistema. Por consiguiente es necesario desarrollar la banda base compleja o la representación paso baja equivalente para los modelos de Volterra, para así poder manejar portadoras moduladas.

Consideramos $x(t) = \text{Re} \left[\tilde{x}(t)e^{j\omega_0 t} \right]$ e $y(t) = \text{Re} \left[\tilde{y}(t)e^{j\omega_0 t} \right]$ como señales de entrada y salida de un amplificador de potencia, donde ω_0 es la frecuencia de portadora y $\tilde{x}(t)$ e $\tilde{y}(t)$ representan las envolventes complejas de entrada y salida respectivamente. Siguiendo el mismo procedimiento usado para obtener (9), el nuevo modelo de Volterra paso de baja equivalente puede ser escrito en el dominio discreto como:

$$\tilde{y}(n) = \sum_{i=0}^r \tilde{y}_i(n) \quad (11)$$

donde

$$\tilde{y}_0(n) = \sum_{k=0}^{\frac{P-1}{2}} \tilde{h}_0(k) |\tilde{x}(n)|^{2k} \tilde{x}(n) \quad (12)$$

$$\begin{aligned} \tilde{y}_1(n) = & \sum_{i=1}^{N-1} \tilde{h}_1(i) \tilde{x}(n-i) \\ & + \sum_{k=1}^{\frac{P-1}{2}} \left[|\tilde{x}(n)|^{2k} \sum_{i=1}^{N-1} \tilde{h}_{k_{11}}(i) \tilde{x}(n-i) \right] \end{aligned} \quad (13)$$

$$+ \sum_{k=1}^{\frac{P-1}{2}} \left[|\tilde{x}(n)|^{2(k-1)} \tilde{x}^2(n) \sum_{i=1}^{N-1} \tilde{h}_{k_{12}}(i) \tilde{x}^*(n-i) \right]$$

$$\tilde{y}_2 = \sum_{k=0}^{\frac{P-3}{2}} \left[|\tilde{x}(n)|^{2k} \tilde{x}^*(n) \sum_{i_1=1}^{N-1} \sum_{i_2=i_1}^{N-1} \tilde{h}_{k_{21}}(i_1, i_2) \tilde{x}(n-i_1) \tilde{x}(n-i_2) \right]$$

$$\begin{aligned}
 & + \sum_{k=0}^{\frac{P-3}{2}} \left[|\tilde{x}(n)|^{2k} \tilde{x}(n) \sum_{i_1=1}^{N-1} \sum_{i_2=1}^{N-1} \tilde{h}_{k_{22}}(i_1, i_2) \tilde{x}(n-i_1) \tilde{x}^*(n-i_2) \right] \\
 & + \sum_{k=1}^{\frac{P-3}{2}} \left[|\tilde{x}(n)|^{2(k-1)} \tilde{x}^3(n) \sum_{i_1=1}^{N-1} \sum_{i_2=i_1}^{N-1} \tilde{h}_{k_{23}}(i_1, i_2) \tilde{x}^*(n-i_1) \tilde{x}^*(n-i_2) \right]
 \end{aligned} \tag{14}$$

...

Aquí, $(\bullet)^*$ representa la conjugada, mientras que $|x|$ la magnitud de x . $\tilde{y}_0(n)$ es la parte estática, mientras que $\tilde{y}_1(n)$ e $\tilde{y}_2(n)$ representan los términos dinámicos de primer y segundo orden respectivamente. Los términos de orden mayor se obtendrán de la misma forma.

En aplicaciones reales, con el fin de reducir la complejidad del modelo, solo nos quedaremos con algunos términos de (11), es decir, usando “Reducción de la desviación” limitaremos r a un valor pequeño.

3.3. EXTRACCION DEL MODELO

Una exigencia clave para la extracción modelo de comportamiento del PA es que el conjunto de ondas a la entrada contenga la mayor parte de frecuencias y amplitudes de interés, como por ejemplo, el ancho de banda y el rango dinámico entero del amplificador de potencia bajo estudio. En otras palabras, el sistema debe ser probado exhaustivamente con el fin de observar todas las no linealidades.

Como ya mencionamos antes, la salida del modelo de Volterra es lineal en lo que concierne a sus coeficientes. Podemos decir que los coeficientes que aparecen en (9) son una generalización de los coeficientes de respuesta al impulso $h(\dots)$ que definen un modelo lineal. Por tanto, una posible solución a la estimación de los parámetros del modelo de Volterra es tratarlo como un problema de regresión grande pero estándar.

En particular, nosotros podríamos formar un vector θ que contenga todos los coeficientes conocidos $h(\dots)$, y definir una matriz X que incluya todos los productos $x(n-i_1)\dots x(n-i_N)$ que aparecen en el modelo. Se considera $x(n)$ desde $n = N$ hasta L . Esto es así porque para obtener las señales retrasadas que nos hacen falta en las expresiones de los productos de entrada tenemos que considerar que el principio de la señal de entrada está en la muestra N , y así, contando con las muestras anteriores podemos componer las señales retrasadas. **L es la cantidad total de datos disponibles.**

Si además asumimos la presencia de un error e , $e = [e(N), \dots, e(L)]^T$, entonces, el modelo de Volterra puede ser escrito como:

$$y = X\theta + e \quad (15)$$

donde $y = [y(N), \dots, y(L)]^T$.

Hemos dicho que los simuladores trabajan con envolventes complejas en banda base, por lo que en este proyecto tendremos θ y $\tilde{X}$, que se formarán de igual manera a partir de (11). Es decir, **nuestro modelo** es:

$$\tilde{y} = \tilde{X}\theta + e \quad (16)$$

CAPÍTULO. 3. Modelo de comportamiento para PAs usando Series de Volterra M.

Por otro lado, a la hora de estimar los parámetros se usará el ajuste de mínimos cuadrados:

$$\hat{\theta} = (\tilde{X}^H \tilde{X})^{-1} \tilde{X}^H \tilde{y} \quad (17)$$

comúnmente usado, y que tiene la ventaja de la simplicidad en la notación y de su aplicabilidad general. (Además de ésta, habrá más técnicas para estimar los parámetros del modelo).

3.4. MODELO EN SCILAB

Como ya se ha dicho, se parte de la expresión (11) que repetimos aquí:

$$\tilde{y}(n) = \sum_{i=0}^r \tilde{y}_i(n) \quad (11)$$

donde, como también vimos:

$$\tilde{y}_0(n) = \sum_{k=0}^{\frac{P-1}{2}} \tilde{h}_0(k) |\tilde{x}(n)|^{2k} \tilde{x}(n) \quad (12)$$

$$\begin{aligned} \tilde{y}_1(n) = & \sum_{i=1}^{N-1} \tilde{h}_1(i) \tilde{x}(n-i) \\ & + \sum_{k=1}^{\frac{P-1}{2}} \left[|\tilde{x}(n)|^{2k} \sum_{i=1}^{N-1} \tilde{h}_{k1}(i) \tilde{x}(n-i) \right] \end{aligned} \quad (13)$$

$$\begin{aligned} & + \sum_{k=1}^{\frac{P-1}{2}} \left[|\tilde{x}(n)|^{2(k-1)} \tilde{x}^2(n) \sum_{i=1}^{N-1} \tilde{h}_{k12}(i) \tilde{x}^*(n-i) \right] \\ \tilde{y}_2 = & \sum_{k=0}^{\frac{P-3}{2}} \left[|\tilde{x}(n)|^{2k} \tilde{x}^*(n) \sum_{i_1=1}^{N-1} \sum_{i_2=i_1}^{N-1} \tilde{h}_{k21}(i_1, i_2) \tilde{x}(n-i_1) \tilde{x}(n-i_2) \right] \\ & + \sum_{k=0}^{\frac{P-3}{2}} \left[|\tilde{x}(n)|^{2k} \tilde{x}(n) \sum_{i_1=1}^{N-1} \sum_{i_2=1}^{N-1} \tilde{h}_{k22}(i_1, i_2) \tilde{x}(n-i_1) \tilde{x}^*(n-i_2) \right] \end{aligned} \quad (14)$$

$$+ \sum_{k=1}^{\frac{P-3}{2}} \left[|\tilde{x}(n)|^{2(k-1)} \tilde{x}^3(n) \sum_{i_1=1}^{N-1} \sum_{i_2=i_1}^{N-1} \tilde{h}_{k23}(i_1, i_2) \tilde{x}^*(n-i_1) \tilde{x}^*(n-i_2) \right]$$

En el punto **3.2.2** vimos que es razonable limitar r a un valor pequeño para reducir la complejidad del modelo, y aún así se mantiene una buena exactitud del mismo. Nosotros tomaremos $r=2$, por lo que no habrá que seguir desarrollando (11) más allá de (12),(13) y (14).

Se trata ahora de identificar nuestro modelo en estas expresiones y darle forma en Scilab. Es decir, identificar θ y $\tilde{X}$ de:

$$\tilde{y} = \tilde{X}\theta + e \quad (16)$$

en (11),(12),(13) y (14).

Hemos dicho que r toma un valor fijo igual a dos, pero N y P serán valores que podemos variar.

Lo primero fue construir $\tilde{X}$. Desarrollando y ordenando (11),(12),(13) y (14) para distintos valores de N y P se puede apreciar el método de formación que siguen las columnas de la matriz. Trasladando esto a Scilab se programó la función **“matriz5.sci”** (Capítulo 7 Anexos: apartado 7.1.1). Esta función se encarga de la creación de $\tilde{X}$, y sólo necesita como parámetros de entrada $\tilde{x}(n)$ y el valor de los parámetros N y P .

Para probar su correcto funcionamiento, como primera comprobación, se usó la función **“NMSE2.sci”** (Capítulo 7 Anexos: apartado 7.1.2). Esta función toma como parámetros de entrada $\tilde{x}_{med}(n)$, N y P . $\tilde{x}_{med}(n)$ es un vector cualquiera, que inventamos nosotros para probar que **“matriz5.sci”** funciona correctamente. Pues bien, con estos parámetros de entrada, lo primero que hace la función **“NMSE2.sci”** es llamar a **“matriz5.sci”**. Una vez creada $\tilde{X}$ calcula $\tilde{y}_{med}(n)$, $\tilde{y}_{med} = \tilde{X}\theta$, donde θ es un vector columna también inventado formado por unos, con tantos unos como columnas tiene $\tilde{X}$. Seguidamente, la función estima el modelo, $\hat{\theta} = (\tilde{X}^H \tilde{X})^{-1} \tilde{X}^H \tilde{y}$, y comprueba lo que cabe esperar, que $\theta \equiv \hat{\theta}$. Lo que realmente devuelve la función es ese error que hace que θ y $\hat{\theta}$ no sean exactamente iguales.

El error que aquí se ha estimado es el error *NMSE*:

$$NMSE = 10 \log_{10} nmse \quad (18)$$

donde:

$$nmse = \frac{\sum_{n=1}^{L-1} |\tilde{y}_{med}(n) - \tilde{y}_{pred}(n)|^2}{\sum_{n=1}^{L-1} |\tilde{y}_{med}(n)|^2} \quad (19)$$

y lo que llamamos $\tilde{y}_{pred}(n)$ es, $\tilde{y}_{pred} = \tilde{X}\hat{\theta}$.

En todas las comprobaciones realizadas, para distintas $\tilde{x}_{med}(n)$ y distintas N y P , se devolvieron valores de $NMSE$ siempre aceptables, entorno a -300dB, con lo que podemos concluir que nuestra función **“matriz5.sci”** funciona correctamente.

En el capítulo siguiente se probará nuestro modelo con medidas reales tomadas en el laboratorio, se trabajará con entradas y salidas reales de un amplificador de potencia común (MAX2430 de Maxim).