

Análisis espectral mediante el uso de la FFT

Mario Estévez Báez¹

Andrés Machado García²

José M. Estévez Carrera³

Material publicado originalmente en formato html en:

librosabiertos: analisis_espectral_mediante_el_uso_de_la_fft. InfoWiki. January 8, 2008, 09:08 CST.

Available at:

http://infomed20.sld.cu/wiki/doku.php?id=librosabiertos: analisis_espectral_mediante_el_uso_de_la_fft&rev=1199801296.

Accessed January 8, 2008

Introducción

El análisis espectral constituye un método de análisis poderoso para poner en evidencia periodicidades ocultas en una serie temporal, tal como lo es, por ejemplo, el electroencefalograma. El espectro de potencia refleja la energía de cada uno de los componentes de frecuencia del proceso estudiado. Permite distinguir los componentes espectrales y cuantificarlos. Este método de análisis en Neurofisiología Clínica, cuenta ya con más de 60 años de aplicación, existiendo equipos comerciales que permiten su empleo para el estudio de la actividad bioeléctrica del cerebro. Éste método de análisis es conocido y se maneja de forma diaria por el especialista promedio.

En el caso del electroencefalograma, el espectro de interés se encuentra entre 0.5 y 50 hertzios, mientras que para aquellas aplicaciones que muestren las fluctuaciones de variables como el neumograma, la presión arterial y el ritmo cardíaco, por poner solo algunos ejemplos, se requiere analizar frecuencias menores de 0.5 hertzios. Teniendo ello en cuenta, resulta conveniente revisar algunos aspectos básicos del método en cuestión, y en donde sea necesario se realizarán referencias específicas, vinculadas con el estudio de tales frecuencias, que por su carácter han sido denominadas extra-lentas y que están vinculadas con manifestaciones de fenómenos vitales del organismo del hombre y los animales (Estévez Báez M., 1982).

¹ Doctor en Medicina, Especialista de Fisiología de Segundo Grado, Investigador Titular, Profesor Consultante, Doctor en Ciencias Médicas, Académico Titular AIA, Instituto de Endocrinología y Enfermedades Metabólicas MINSAP.

² Licenciado en Cibernética-Matemática, Profesor Auxiliar, Maestro en Ciencias de la Computación Facultad de Biología, Universidad de La Habana, MES.

³ Licenciado en Informática, Instituto Superior de Medicina Militar “Dr. Luis Díaz Soto”

Series discretas de Fourier

La secuencia temporal $x(n)$, que es periódica, con período "T", puede ser representada, como hemos visto en otro acápite anterior, por la siguiente expresión:

$$x(n) = x(n + rT),$$

donde "r" es un valor entero cualquiera. Esta secuencia puede ser representada por una serie de Fourier, que corresponda a una suma de secuencias exponenciales complejas relacionadas, con frecuencias que sean múltiplos de la frecuencia fundamental ($2\pi/T$), asociada con la secuencia periódica $x(n)$. Las exponenciales periódicas complejas son de la siguiente forma:

$$e^{i(2\pi/T)kn}$$

Donde "i" es la unidad imaginaria y "k" es un número entero. La representación de la serie de Fourier de la secuencia temporal $x(n)$ toma la siguiente forma:

$$(a) \quad x(n) = \frac{1}{T} \sum_{k=0}^{T-1} x(k) e^{i(2\pi/T)kn}.$$

En esta expresión, queda bien definido que se trata de una secuencia de una señal de valores discretos y que por lo tanto, requiere solamente de "T" exponenciales complejas relacionadas.

Los coeficientes de la serie discreta de Fourier, pueden ser obtenidos de la secuencia temporal $x(n)$, por la relación que se muestra en la siguiente expresión:

$$(b) \quad x(k) = \sum_{n=0}^{T-1} x(n) e^{-i(2\pi/T)kn};$$

En esta expresión se puede generalizar que

$$x(k) = x(T + k), \quad \text{para cualquier entero "k".}$$

Las expresiones "a" y "b" constituyen una pareja (par) de análisis-síntesis, conociéndose como las expresiones de las Series Discretas de Fourier, para la representación de una secuencia periódica. Si se sustituye, para facilitar la notación, al término exponencial complejo de la siguiente manera:

$$W_T = e^{-i(2\pi/T)},$$

entonces, el par de ecuaciones de la Transformada de Fourier, para una secuencia temporal discreta, se puede escribir así:

$$\text{síntesis:} \quad x(n) = \frac{1}{T} \sum_{k=0}^{T-1} x(k) W_T^{-kn} ;$$

$$\text{análisis:} \quad x(k) = \sum_{n=0}^{T-1} x(n) W_T^{kn} ;$$

En la anterior exposición, se ha empleado la representación de las Series de Fourier, para secuencias temporales de tiempo discreto, en forma de exponenciales complejas. Esta forma de representación resulta útil, pero implica recordar correctamente, algunas de las propiedades de los números complejos y de las operaciones que pueden ser realizados con los mismos. A modo de recordatorio, en los próximos acápites marcar se expondrán en forma resumida, algunos de estos elementos.

Transformada rápida de Fourier

Retomando la expresión que presenta el acápite “Series discretas de Fourier”, para el cálculo de una serie temporal periódica de tiempo finito:

$$X(k) = \sum_{n=0}^{T-1} X(n) e^{-i(2\pi/T)kn}$$

que se puede simplificar como fue descrito anteriormente de la siguiente manera:

$$X(k) = \sum_{n=0}^{T-1} X(n) W_T^{kn} ;$$

y tomando en cuenta que la serie temporal tiene que ser compleja, entonces para calcular los diferentes valores, será necesario realizar operaciones con números complejos, que implicarán multiplicaciones, sustracciones y adiciones.

Si descomponemos en partes real (real) e imaginaria (im) la expresión anterior, tendríamos:

$$X(k) = \sum_{n=0}^{T-1} [(real(X(n), im X(n))(real W_T^{kn}, im W_T^{kn})]$$

para $k = 0, 1, \dots, T-1$;

Desarrollando de modo genérico el producto de números complejos que se muestra, se puede ver que su cálculo sería:

$$[(\text{real } X(n) * \text{real } W_T^{kn}) - (\text{im } X(n) * \text{im } W_T^{kn})] + \\ [(\text{real } X(n) * \text{im } W_T^{kn}) + (\text{im } X(n) * \text{real } W_T^{kn})];$$

Un cálculo inicial aproximado, permite decir que serían necesarias $4T^2$ multiplicaciones de números reales y de $T(4T - 2)$ sumas de reales. La cantidad de cálculos, y por ende del tiempo de cálculo, es aproximadamente proporcional al cuadrado de T , lo que representa un obstáculo serio, aún para las computadoras actuales. Está documentado, que ya en 1805, C.F. Gauss se había ocupado en buscar la manera de reducir el número de cálculos a realizar, aprovechando diferentes propiedades de la secuencia W_T^{kn} . C. Runge (1905) y posteriormente Danielson y Lanczos (1942) describieron algoritmos que lograban que el tiempo de cálculo fuese proporcional a $T \log T$ (Oppenheim A.V. y Shafer R.W., 1989).

A partir de los trabajos de J.W. Cooley y J.W. Tukey (1965), se hizo posible el desarrollo de algoritmos cada vez más eficientes para estos cálculos y que desde entonces han sido denominados como “algoritmos de la transformada rápida de Fourier” (FFT en lengua inglesa).

Hemos empleado numerosos de estos algoritmos, y actualmente utilizamos un “fabuloso” método descrito por Don Cross (2000), que más adelante será analizado.

Resolución y “aliasing”

La resolución del procedimiento del análisis espectral de una muestra finita (N) de valores de una secuencia temporal dada, se define por la siguiente expresión:

$$Rs = \frac{1}{Pm * N}$$

donde Rs _resolución y Pm _período de muestreo.

El período de muestreo de una señal va a determinar un valor que fue objeto de estudio por H. Nyquist (1928) y casi de modo simultáneo por el Académico Kotelnikov en la antigua URSS, que ha sido por ello llamado como frecuencia de Nyquist o frecuencia de Kotelnikov y que constituye la máxima frecuencia que puede ser detectable en el proceso de análisis. Si se denomina este valor como F_c , entonces el mismo queda definido mediante la expresión:

$$F_c = \frac{1}{2 * P_m}$$

El teorema del muestreo, enunciado por ambos investigadores plantea que una función temporal $h(t)$, muestreada con un período P_m , está limitada a un ancho de banda para las frecuencias mayores que F_c , o sea, si $h(f) = 0$ para todas las frecuencias mayores que F_c , entonces la función $h(t)$ estará determinada completamente por sus muestras. De forma explícita, la función $h(t)$ se representa por la expresión:

$$h(t) = P_m \sum_{n=-\infty}^{\infty} h(n) \frac{\{ \text{sen}[2\pi F_c(t - nP_m)] \}}{\pi(t - nP_m)};$$

Si la función $h(t)$ no está limitada en ancho de banda para los valores menores que la frecuencia de Nyquist (F_c), la densidad espectral de potencia espectral fuera del rango que va de $-\infty$ a $-F_c$ y de F_c a ∞ , se desplaza al intervalo entre $-F_c$ y F_c , distorsionando los resultados. A este fenómeno se le denomina en lengua inglesa “aliasing”.

El problema del *aliasing* es una consecuencia del uso de circuitos electrónicos, para el caso de que se trate de señales continuas que muestran el comportamiento de una señal que se produce de manera continua, y de la introducción del proceso de digitalización de las señales, o sea, *discretización* de valores de una señal continua en el tiempo. El primero de los casos no es objeto de análisis en esta parte del trabajo, pero sí lo será el segundo caso, por tratarse de la circunstancia con la cual se enfrenta el especialista que estudia la VRC en su trabajo cotidiano.

Una secuencia de valores discretos de una serie temporal, es de hecho, una serie sometida a *discretización*, y queremos en la mayoría de los casos, tratar de inferir las propiedades de la señal original (continua), a partir de la secuencia de valores digitalizados (discretizados). Si no se toma en cuenta este aspecto del teorema de muestreo, se pueden cometer errores importantes por la distorsión que el *aliasing* produce en los resultados.

En el caso concreto del análisis de la señal electrocardiográfica (ECG), este fenómeno se puede producir si el especialista no presta atención al modo en que fue realizado el registro de la señal original del ECG. Habitualmente, el registro original del ECG (continuo) es sometido a discretización para su análisis posterior. Si los sistemas de discretización son aplicados a señales continuas (analógicas), que han sido almacenadas en cinta magnética, las características de la velocidad de la cinta influirán en el proceso de análisis. Algo similar ocurre si lo almacenado en la cinta magnética es el valor discretizado de la señal del ECG. En este último caso, o cuando se utilizan dispositivos de memoria volátil, hay que prestar

atención no solamente a la frecuencia del proceso de discretización (P_m), sino que además debe controlarse que el dispositivo de traslación de la cinta, o el reloj interno que regula el proceso de discretización estén funcionando de modo estable. Un modo comúnmente utilizado es la grabación (en cinta o en memoria) de una señal de tiempo real, con lo cual se puede comprobar y corregir, si resulta necesario, cualquier desviación instrumental.

Generalmente, se considera que la frecuencia más elevada (rápida) que contiene información útil del ECG es de 50 – 100 Hz. De acuerdo con el teorema del muestreo, el P_m de tal proceso debe ser entonces de 0.01 o 0.005 Hz, ya que

$$F_c = \frac{1}{2 * P_m} \quad y \quad P_m = \frac{1}{F_c * 2} ;$$

En otras palabras, el periodo de muestreo debe ser de 2 a 5 milisegundos, para evitar la distorsión del *aliasing* en los indicadores espectrales calculados a partir de las mediciones discretas del ECG.

Existen diversos algoritmos para calcular la duración del ciclo cardíaco, considerado éste como el período de tiempo transcurrido entre el vértice de una onda “R” del complejo “QRS” del electrocardiograma y la inmediata ulterior. Para otras mediciones, tales como: el intervalo P-R, intervalos P-P y otros, igualmente resulta importante tener en cuenta el “ P_m ” empleado para la discretización. El propio algoritmo utilizado, puede afectar aún más el efecto del aliasing. El hecho de que muchos equipos comerciales utilicen una frecuencia de muestreo de 128 Hz ($P_m = 7.81$ ms), como base para el análisis posterior de las series de cardiointervalos R-R, ha sido estudiado para determinar si resultan válidos esos trabajos y sus limitaciones (Merri M., et al., 1990). En relación con el empleo de diferentes algoritmos para la detección de los vértices de las ondas “R” del ECG, también se han realizado investigaciones (Friesen G.M., et al., 1990).

Es recomendable, siempre que ello sea posible, emplear como período de muestreo de la señal electrocardiográfica un valor no mayor de 5 ms, aunque como antes ya señalamos, los estudios realizados con período de muestreo de 7.81 ms hayan tenido que aceptarse por razones de orden práctico. Cuando se emplean los algoritmos de la FFT, para realizar el análisis de frecuencia de series de cardiointervalos R-R, cuyos valores son reales, el componente $N/2$ de la serie de valores de salida de la FFT será el que corresponde a la frecuencia de Nyquist. Esto podrá ser comprobado en ejemplos que posteriormente serán presentados.

Veamos algunos ejemplos de interés para el lector, que permitirán comprender mejor lo expuesto con anterioridad.

Si la duración de una serie periódica temporal que se desea someter al análisis espectral tiene una de las duraciones que se muestran en la primera columna de la Tabla 1, los valores de la resolución del procedimiento serán los que se muestran en la columna correspondiente.

Tabla 1. Resolución del proceso espectral en dependencia de la duración de la muestra

Duración (s)	Resolución (Hz)
64	0.015625
128	0.0078125
256	0.00390625
1024	0.000976562
200	0.0050
400	0.0025

Pasemos ahora a una situación concreta: Se han tomado 500 muestras de una señal sinusoidal de frecuencia 0.08 Hz, empleando una frecuencia de discretización de 4 Hz. Se desea conocer la resolución que tendría el proceso de análisis espectral de esta serie temporal periódica y la frecuencia de Nyquist correspondiente.

De acuerdo a lo que ha sido expuesto anteriormente tendremos:

$$R_s = \frac{1}{(P_m * N)} = \frac{1}{(0.25 * 500)} = 0.008 \text{ Hz} ;$$

$$F_c = \frac{1}{(2 * P_m)} = \frac{1}{(2 * 0.25)} = 2 \text{ Hz} ;$$

Puede comprobarse además, que el componente de frecuencia N/2 (si se aplica la FFT) corresponderá al valor de la Fc. Veamos:

$$0.008 \text{ Hz} * 250 = 2 \text{ Hz} ;$$

De manera general, se puede decir que existen dos maneras principales de enfrentar la influencia negativa del *aliasing*. Son ellas:

- Conociendo el límite natural de ancho de banda de la señal a estudiar, muestrearla con una frecuencia al menos del doble de la frecuencia más alta que la misma posee.
- Proceder a un filtraje de la señal original, ya sea usando métodos de filtraje analógicos o digitales, para eliminar las frecuencias indeseadas y proceder luego a realizar un muestreo de la señal filtrada, con una frecuencia al

menos del doble de la frecuencia más alta que pueda quedar después del proceso de filtraje.

Algoritmos de la FFT. Algoritmo de Don Cross

En un acápite anterior fue señalada la aparición de los algoritmos de la FFT, para su utilización en aplicaciones con el uso de ordenadores. Considerando que en general, los diferentes algoritmos solo varían en cuanto a su implementación y eficiencia, vamos a limitarnos en este acápite a la descripción de los rasgos principales del algoritmo desarrollado por Don Cross (2000). Al finalizar el acápite se incluye el citado algoritmo en su implementación para Pascal.

La FFT es un algoritmo que convierte una función compleja, muestreada en tiempo, en una función muestreada en dominio de la frecuencia y que también es compleja. Como la mayor parte de las veces se opera con valores reales y el algoritmo de la FFT requiere que la señal *discretizada* esté expresada en valores complejos, resulta imprescindible entonces utilizar la propiedad de que un número real puede ser expresado como un número complejo, si como valor imaginario le colocamos el valor cero. Ejemplo: el valor real 750 puede ser expresado como el número complejo (750, 0), donde 750 es la parte real y 0 la parte imaginaria de ese número complejo.

Los valores de la secuencia temporal que se introducen como entrada al algoritmo de la FFT, deben estar ordenados en dos arreglos⁴ unidimensionales, que podrían llamarse `RealesEnt` e `ImaginEnt`, que tienen que tener el mismo número de elementos (N). El arreglo `RealesEnt` será el contenedor de los valores reales de la serie temporal que se somete al análisis y el arreglo `ImaginEnt`, tendrá el mismo número de elementos, siendo todos ellos de valor cero ("0"). El valor de " N ", que representa al número de elementos de la serie compleja temporal, tiene que ser una potencia entera de 2. El algoritmo no permite la entrada de valores que no cumplan esta condición, por lo que una serie de $N = 13,000$ será inadecuada, en tanto que una serie de $N = 16,384$ será adecuada. Esta limitación está asociada con particularidades del proceso de cálculo que facilita la realización de la FFT y es común a todos sus algoritmos. El menor número de valores sería "2", mientras que el número máximo estará en dependencia de las limitaciones de memoria disponible y del sistema operativo del ordenador. El tiempo de ejecución de este algoritmo es proporcional a la magnitud del número de estas operaciones que es $O(n \log(n))$, donde O _orden del algoritmo y n _número de elementos en la serie de entrada. Los resultados del procesamiento son entregados al usuario por el algoritmo mediante " n " valores complejos divididos en dos arreglos, que

⁴ En términos de lenguajes de programación, los arreglos son estructuras de datos que permiten almacenar una serie de valores, todos del mismo tipo.

podemos denominar **RealesSal** e **ImaginSal**. El número de valores de salida será similar al número de valores de entrada.

En la Tabla 2 se muestran, a modo de ejemplo concreto, los resultados de la aplicación del algoritmo de la FFT (Don Cross) a una serie temporal de 32 valores, correspondientes a la interpolación de una secuencia de cardiointervalos R-R, con periodo de muestreo de 1 segundo. A los efectos del ejemplo, puede observarse que se han introducido 32 muestras con valores complejos, como son mostrados en la Tabla, cuyos componentes reales aparecen en la columna con denominación **RealesEnt**, mientras que los componentes imaginarios correspondientes, aparecen bajo la columna denominada **ImaginEnt**, los cuales son todos de valor cero ("0").

Como la duración total de la serie es de 32 segundos, lo que resulta del hecho de que son 32 valores discretos ($N = 32$), muestreadas con periodo de 1 segundo ("Pm"), entonces:

$$Rs = \frac{1}{(Pm * N)} = \frac{1}{(1 * 32)} = 0.03125 \text{ Hz}$$

y

$$Fc = \frac{1}{(2 * Pm)} = \frac{1}{(2 * 1)} = 0.5 \text{ Hz}.$$

En las columnas siguientes a la denominada **ImaginEnt**, aparecen mostrados los resultados de la aplicación del algoritmo de la FFT, así como el cálculo de las densidades espectrales que han sido obtenidos a partir de los datos de salida del algoritmo de la FFT. La secuencia de valores reales de los 32 elementos de la serie temporal se muestra en un histograma secuencial en la Figura 1.

En la cuarta columna de la tabla se muestra el número de orden de las muestras frecuenciales, que corresponden a la salida de la FFT. Puede observarse que son 32 muestras, al igual que las 32 muestras de entrada a la FFT, pero la primera frecuencia en aparecer ha recibido el número de orden "0" en lugar de "1". Por ello, el último valor de la columna es el "31"

En la quinta columna se muestran los valores de las frecuencias discretas, para las cuales se han calculado los correspondientes resultados de la FFT. Podemos ver que la primera frecuencia discreta correspondería al valor "0", le sigue la que corresponde al valor de la resolución del proceso ("Rs"), que en este caso, como fue antes calculado es de 0.03125 Hz y en las subsiguientes filas de la columna, aparecen las frecuencias discretas con valores consecutivos múltiplos enteros de la resolución (2Rs, 3Rs, ... , nRs). En la sexta columna, se han precisado los valores del periodo ($1 / Rs$), que corresponden a cada valor de frecuencia discreta. Puede advertirse que el periodo correspondiente a cada frecuencia

discreta se va reduciendo, a medida que se va incrementando el valor correspondiente de la frecuencia, lo que es una natural consecuencia de la relación inversa entre la frecuencia y el periodo.

Tabla 2. Ejemplo de aplicación de la FFT a una muestra de 32 valores de una secuencia temporal (Ver explicación detallada en el texto).

N/O (1)	Real Ent (2)	Imag Ent (3)	N/O (4)	F[i] (5)	T (6)	Real Sal (7)	Imag Sal (8)	Potencia Espectro (9)	Amplitud Espectro (10)
1	970	0	0	-	-	30360	0	57608100	948.75
2	980	0	1	0.03125	32	42.74557	195.48389	2502.57	8.84
3	970	0	2	0.06250	16.0	250.53107	135.14452	5064.37	12.58
4	945	0	3	0.09375	10.67	53.60643	11.42716	187.76	2.42
5	965	0	4	0.12500	8.00	36.81981	-61.81981	323.59	3.18
6	980	0	5	0.18750	5.33	49.94675	-32.57616	22.240	2.63
7	970	0	6	0.21875	4.57	58.92056	-74.43283	563.24	4.19
8	930	0	7	0.25000	4.00	146.71811	69.08236	1643.66	7.17
9	935	0	8	0.28125	3.56	-105.00000	-5.00000	690.62	4.64
10	950	0	9	0.31250	3.20	-81.99338	-14.54823	13.230	0.64
11	960	0	10	0.34375	2.91	-52.05764	-6.85924	172.31	2.32
12	935	0	11	0.37500	2.67	-25.49541	11.19549	48.460	1.23
13	935	0	12	0.40625	2.46	-26.81981	-1.81981	45.160	1.19
14	955	0	13	0.43750	2.28	7.800090	16.20385	20.210	0.79
15	970	0	14	0.46875	2.13	-17.39399	-17.28189	37.570	1.08
16	970	0	15	0.50000	2.00	6.671820	-47.14164	141.68	2.10
17	945	0	16	0.53125	1.88	-10.00000	0.000000	6.2500	0.44
18	950	0	17	0.56250	1.78	6.671820	47.14164		
19	960	0	18	0.59375	1.68	-17.39399	17.28189		
20	960	0	19	0.62500	1.60	7.800090	-16.20385		
21	945	0	20	0.65625	1.52	-26.81981	1.819810		
22	935	0	21	0.68750	1.45	-25.4954	-11.19549		
23	935	0	22	0.71875	1.39	-52.05764	6.859240		
24	940	0	23	0.75000	1.33	-81.99338	14.54823		
25	920	0	24	0.78125	1.28	-105.00000	5.000000		
26	915	0	25	0.81250	1.23	146.71811	-69.08236		
27	915	0	26	0.84375	1.18	58.92056	74.432830		
28	935	0	27	0.87500	1.14	49.94675	52.576160		
29	920	0	28	0.90625	1.10	36.81981	61.819810		
30	925	0	29	0.93750	1.07	53.60643	-11.42716		
31	960	0	30	0.96875	1.03	250.53107	-135.1445		
32	980	0	31	1.00000	1.00	42.745570	-195.4834		

En las columnas séptima y octava, aparecen los valores de salida del algoritmo de la FFT, correspondientes a la serie de valores complejos de la serie temporal de entrada, cuyos valores reales e imaginarios se muestran en las columnas segunda y tercera. Resulta evidente que para la primera frecuencia, que ocupa el orden “0”, solamente se obtiene un valor real; el imaginario es cero. Esta frecuencia “0” del resultado, está teóricamente asociada al valor de la media aritmética de los

valores de la serie de entrada. Su magnitud (amplitud) es, en todo caso, proporcional a la media aritmética de los valores de entrada como ya se dijo, y por ello, podemos reducir su valor para una serie temporal de entrada dada, si a dicha serie se le calcula la media aritmética y luego se resta la misma de cada valor de la serie original. Más adelante volveremos sobre esta cuestión.

Una inspección de los valores calculados, tanto reales como imaginarios, muestra que las cifras obtenidas a partir de la frecuencia discreta " $N/2$ ", se repiten en la segunda mitad de la serie de valores en la salida de la FFT, constituyendo una imagen en espejo de los correspondientes a la primera mitad. También es de notar, que el valor del signo correspondiente a los valores imaginarios en la segunda mitad de la serie, resulta opuesto al que tuvo en la primera mitad. Debe también ser objeto de atención, el hecho de que el valor de salida imaginario, correspondiente a la frecuencia " $N/2$ ", es cero. Ello es consecuencia de que los valores de entrada a la FFT, fueron reales.

Por tanto, puede irse resumiendo que los valores a la salida del algoritmo de la FFT muestran una imagen especular de valores, cuando los valores de entrada han sido reales y que solo se diferencian en el signo, resultando que los signos de los valores imaginarios son opuestos en la segunda mitad, respecto a los signos en la segunda mitad.

Como en la inmensa mayoría de los casos emplearemos valores reales a la entrada de la FFT, resulta entonces que las salidas para las frecuencias positivas y negativas son similares y podemos decir que redundantes. Resulta, que estos valores son números conjugados complejos, lo que significa que sus partes reales son similares y que sus partes imaginarias tienen signo contrario. Vale recordar que en la parte que fue dedicada a la actualización acerca de la teoría de los números complejos, fue expuesto que dos números complejos son conjugados si se da la condición

$$(a, b) \text{ y } (a, -b) .$$

El algoritmo de Don Cross, tiene algunas facilidades que permiten que los cálculos se limiten a un solo componente de frecuencia discreta que se desee evaluar (ya sean valores de amplitud de componentes real e imaginario, o valor de la fase correspondiente). También es posible, usando un procedimiento especial del algoritmo, calcular la FFT de secuencias de números reales enteros, cuya longitud sea diferente de $2N$.

Nos parece conveniente ahora, que recordemos algunas expresiones de los números complejos que nos serán de utilidad inmediata. Podemos considerar el valor real e imaginario de salida de la FFT para cada frecuencia discreta en la forma "modulo-argumental" que se expuso en acápites anteriores:

$$(a, b) = \rho \varphi ;$$



Figura 1 Representación mediante un histograma secuencial de los valores de la secuencia utilizada en el texto como un ejemplo.

En este caso, $a_{\text{parte real}}$, correspondería al valor real obtenido y $b_{\text{parte imaginaria}}$, correspondería al valor imaginario. Recordando que

$$\rho = \sqrt[2]{a^2 + b^2} ;$$

se desprende que el módulo ρ estará dado por el valor real obtenido para la frecuencia discreta correspondiente, elevado al cuadrado, sumado con el cuadrado del valor imaginario y después de ser sumados ambos, calculando la raíz cuadrada se obtendría ρ . Consideremos como ejemplo los valores real e imaginario correspondientes a la salida de la FFT para la frecuencia discreta 0.3750 en la Tabla 2. Para obtener el módulo del resultado basta con operar:

$$\begin{aligned} \rho &= \pm \sqrt[2]{(-25.49541)^2 + (11.19549)^2} = \\ \rho &= \pm \sqrt[2]{650.0159311 + 125.3389963} = \\ \rho &= \pm \sqrt[2]{775.3549274} = \pm 27.84519577 ; \end{aligned}$$

Para encontrar el argumento ϕ , bastaría con calcularlo a partir de cualquiera de las expresiones:

$$\begin{aligned}\tan \varphi &= \frac{b}{a} = -0.43912 ; \\ \rho \sin \varphi &= b ; \\ \rho \cos \varphi &= a ; \\ \varphi &= \arctan \frac{b}{a} = 156^{\circ} 17' 34''.31 ;\end{aligned}$$

El punto $\rho\varphi$ en un plano de coordenadas, es el afijo que corresponde en este caso a la frecuencia discreta en cuestión. Como hemos visto en otro acápite anterior, este valor $\rho\varphi$ son las coordenadas polares del número complejo correspondiente. En este caso, el afijo se encontrará en el segundo cuadrante, con valor en el eje de las abscisas de -25.49541 y en el de las ordenadas de 11.19549 .

En el acápite “Métodos de Análisis de Procesos”, al señalar un ejemplo del primer tipo de los mismos, fue mencionado el caso de la transformada de Fourier. Se indicó que los valores de la serie temporal, representados en las coordenadas tiempo-intensidad, resultan absolutamente equivalentes a la representación en coordenadas de frecuencia-fase-intensidad. En el ejemplo que venimos analizando en este acápite, las coordenadas tiempo-intensidad de la serie de entrada están representados por los que se muestran en las columnas segunda y tercera de la Tabla 2, en tanto que las de frecuencia-fase-intensidad, lo están en las columnas séptima y octava. Sin embargo, no queda claro de la simple inspección de la tabla, cuál es la información de fase. La fase asociada a cada valor de frecuencia discreta está dada por el valor del ángulo φ , expresado generalmente en radianes y obtenido mediante la expresión ya antes mostrada:

$$\varphi = \arctan \frac{b}{a} ;$$

Los valores reales e imaginarios del valor complejo de salida de la FFT para cada frecuencia discreta son portadores, por lo tanto, de la información referente a la intensidad y a la fase. Teniendo estos elementos, se puede reconstituir unívocamente la serie original de valores de entrada a la FFT, aplicando el proceso de la FFT inversa, posibilidad que también brinda el algoritmo de Don Cross que venimos analizando, y en general, todos aquellos algoritmos que usan el concepto de la FFT.

Debe advertirse, sin embargo, que a los efectos del cálculo de la fase, se toman solo los denominados valores principales de la función “*arco tangente*”. En el caso de la frecuencia discreta, para la cual se efectuaron algunos cálculos anteriormente, ese valor principal de la fase correspondería a -0.414 radianes en la frecuencia discreta 11. Los valores principales de la función “*arc tan*” tienen que cumplir la condición:

$$-\frac{\pi}{2} < \arctan x < \frac{\pi}{2} ;$$

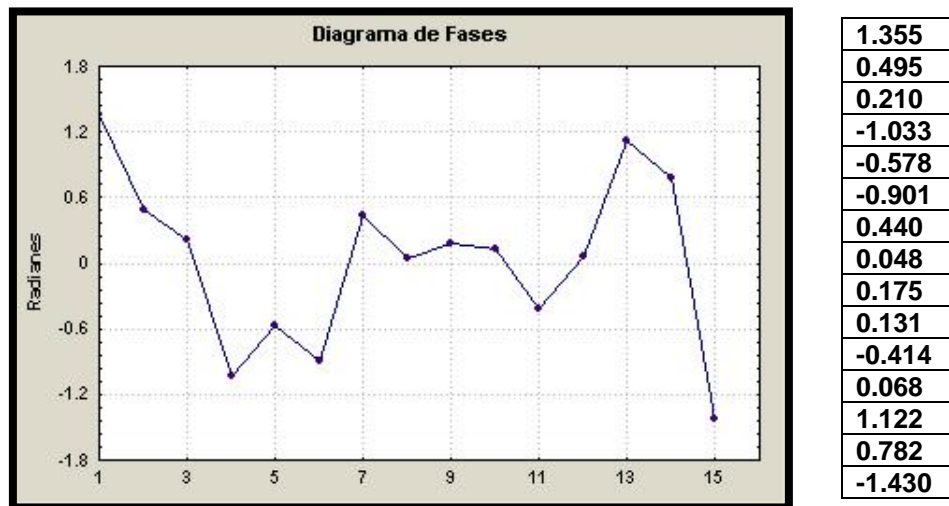


Figura 2 Diagrama de fases del espectro de la secuencia utilizada en el ejemplo.

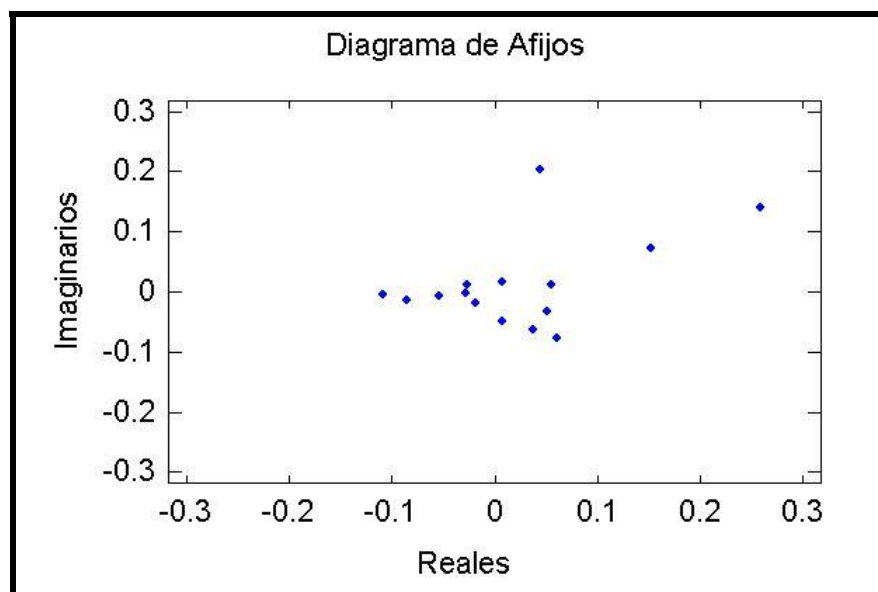


Figura 3 Diagrama de afijos de la serie de salida de la FFT del ejemplo en el texto.

Por ello, calculando la fase para las 15 primeras frecuencias discretas del ejemplo, se obtienen los valores que se muestran tabulados a la derecha de la Figura 2. En la parte izquierda, se muestra una curva que describe el perfil de la variación de la fase de la secuencia de entrada. En las abscisas se han situado los valores de las frecuencias discretas (Hz), en tanto en las ordenadas, aparecen los valores de la fase expresados en radianes. Este diagrama se denomina diagrama de fase y

junto al espectro de potencia, constituyen los resultados principales del análisis espectral. Más adelante, veremos la forma de calcular el espectro de potencia.

En la Figura 3 se muestra un diagrama de afijos que se corresponde con los valores calculados en el ejemplo.

Algoritmo de Don Cross (Código)

fourier.pas - Don Cross <dcross@intersrv.com>

```
=====*)

{$N+,E+} (* Allows code to use type 'double' and run on any iX86 machine *)
{$R-}(* Turn off range checking...we violate array bounds rules *)

unit Fourier;

interface

(*-----
procedure fft

Calculates the Fast Fourier Transform of the array of complex numbers
represented by 'Realln' and 'ImagIn' to produce the output complex
numbers in 'RealOut' and 'ImagOut'.
-----*)
procedure fft (
  NumSamples: word; { must be a positive integer power of 2 }
  var Realln: array of double;
  var ImagIn: array of double;
  var RealOut: array of double;
  var ImagOut: array of double );
(*-----
procedure ifft

Calculates the Inverse Fast Fourier Transform of the array of complex numbers represented by
'Realln' and 'ImagIn' to produce the output complex numbers in 'RealOut' and 'ImagOut'.
-----*)
procedure ifft (
  NumSamples: word; { must be a positive integer power of 2 }
  var Realln: array of double;
  var ImagIn: array of double;
  var RealOut: array of double;
  var ImagOut: array of double );
(*-----
procedure fft_integer

Same as procedure fft, but uses integer input arrays instead of
double. Make sure you call fft_integer_cleanup after the last
time you call fft_integer to free up memory it allocates.
-----*)
procedure fft_integer (
  NumSamples: word;
  var Realln: array of integer;
  var ImagIn: array of integer;
  var RealOut: array of double;
  var ImagOut: array of double );
```

```
(*-----
procedure fft_integer_cleanup

    If you call the procedure 'fft_integer', you must call
    'fft_integer_cleanup' after the last time you call 'fft_integer'
    in order to free up dynamic memory.
-----*)
```

```
procedure fft_integer_cleanup;
```

```
(*-----
procedure CalcFrequency
```

This procedure calculates the complex frequency sample at a given index directly. Use this instead of 'fft' when you only need one or two frequency samples, not the whole spectrum.

It is also useful for calculating the Discrete Fourier Transform (DFT) of a number of data which is not an integer power of 2. For example, you could calculate the DFT of 100 points instead of rounding up to 128 and padding the extra 28 array slots with zeroes.

```
-----*)
procedure CalcFrequency (
    NumSamples: word; { can be any positive integer }
    FrequencyIndex: word; { must be in the range 0 .. NumSamples-1}
    var Realln: array of double;
    var ImagIn: array of double;
    var RealOut: double;
    var ImagOut: double );
```

implementation

```
function IsPowerOfTwo ( x: word ): boolean;
```

```
var i, y: word;
```

```
begin
```

```
    y := 2;
```

```
    for i := 1 to 15 do begin
```

```
        if x = y then begin
```

```
            IsPowerOfTwo := TRUE;
```

```
            exit;
```

```
        end;
```

```
        y := y SHL 1;
```

```
    end;
```

```
    IsPowerOfTwo := FALSE;
```

```
end;
```

```
function NumberOfBitsNeeded ( PowerOfTwo: word ): word;
```

```
var i: word;
```

```
begin
```

```
    for i := 0 to 16 do begin
```

```
        if (PowerOfTwo AND (1 SHL i)) <> 0 then begin
```

```
            NumberOfBitsNeeded := i;
```

```
            exit;
```

```
        end;
```

```
    end;
```

```
end;
```

```
function ReverseBits ( index, NumBits: word ): word;
```

```
var i, rev: word;
```

```
begin
```

```
    rev := 0;
```

```
    for i := 0 to NumBits-1 do begin
```

```
        rev := (rev SHL 1) OR (index AND 1);
```

```
        index := index SHR 1;
```

```
    end;
```

```
    ReverseBits := rev;
```



```

end;

procedure FourierTransform (
  AngleNumerator: double;
  NumSamples: word;
  var Realln: array of double;
  var Imagln: array of double;
  var RealOut: array of double;
  var ImagOut: array of double );
var
  NumBits, i, j, k, n, BlockSize, BlockEnd: word;
  delta_angle, delta_ar: double;
  alpha, beta: double;
  tr, ti, ar, ai: double;
begin
  if not IsPowerOfTwo(NumSamples) or (NumSamples<2) then begin
    write ( 'Error in procedure Fourier: NumSamples=',    NumSamples );
    writeln ( ' is not a positive integer power of 2.' );
    halt;
  end;
  NumBits := NumberOfBitsNeeded (NumSamples);
  for i := 0 to NumSamples-1 do begin
    j := ReverseBits ( i, NumBits );
    RealOut[j] := Realln[i];
    ImagOut[j] := Imagln[i];
  end;
  BlockEnd := 1;
  BlockSize := 2;
  while BlockSize <= NumSamples do begin
    delta_angle := AngleNumerator / BlockSize;
    alpha := sin ( 0.5 * delta_angle );
    alpha := 2.0 * alpha * alpha;
    beta := sin ( delta_angle );
    i := 0;
    while i < NumSamples do begin
      ar := 1.0;  (* cos(0) *)
      ai := 0.0;  (* sin(0) *)
      j := i;
      for n := 0 to BlockEnd-1 do begin
        k := j + BlockEnd;
        tr := ar*RealOut[k] - ai*ImagOut[k];
        ti := ar*ImagOut[k] + ai*RealOut[k];
        RealOut[k] := RealOut[j] - tr;
        ImagOut[k] := ImagOut[j] - ti;
        RealOut[j] := RealOut[j] + tr;
        ImagOut[j] := ImagOut[j] + ti;
        delta_ar := alpha*ar + beta*ai;
        ai := ai - (alpha*ai - beta*ar);
        ar := ar - delta_ar;
        INC(j);
      end;
      i := i + BlockSize;
    end;
    BlockEnd := BlockSize;
    BlockSize := BlockSize SHL 1;
  end;
end;
end;

```

```

procedure fft (
  NumSamples: word;
  var Realln: array of double;
  var Imagln: array of double;

```

```

    var RealOut: array of double;
    var ImagOut: array of double );
begin
    FourierTransform ( 2*PI, NumSamples, Realln, Imagln, RealOut, ImagOut );
end;

procedure ifft (
    NumSamples: word;
    var Realln: array of double;
    var Imagln: array of double;
    var RealOut: array of double;
    var ImagOut: array of double );
var
    i: word;
begin
    FourierTransform ( -2*PI, NumSamples, Realln, Imagln, RealOut, ImagOut );

    (* Normalize the resulting time samples... *)
    for i := 0 to NumSamples-1 do begin
        RealOut[i] := RealOut[i] / NumSamples;
        ImagOut[i] := ImagOut[i] / NumSamples;
    end;
end;

type
    doubleArray = array [0..0] of double;
var
    RealTemp, ImagTemp: ^doubleArray;
    TempArraySize: word;

procedure fft_integer (
    NumSamples: word;
    var Realln: array of integer;
    var Imagln: array of integer;
    var RealOut: array of double;
    var ImagOut: array of double );
var
    i: word;
begin
    if NumSamples > TempArraySize then begin
        fft_integer_cleanup; { free up memory in case we already have some. }
        GetMem ( RealTemp, NumSamples * sizeof(double) );
        GetMem ( ImagTemp, NumSamples * sizeof(double) );
        TempArraySize := NumSamples;
    end;
    for i := 0 to NumSamples-1 do begin
        RealTemp^[i] := Realln[i];
        ImagTemp^[i] := Imagln[i];
    end;
    FourierTransform (
        2*PI,
        NumSamples,
        RealTemp^, ImagTemp^,
        RealOut, ImagOut );
end;

procedure fft_integer_cleanup;
begin
    if TempArraySize > 0 then begin
        if RealTemp <> NIL then begin
            FreeMem ( RealTemp, TempArraySize * sizeof(double) );
            RealTemp := NIL;

```

```

    end;
    if ImagTemp <> NIL then begin
        FreeMem ( ImagTemp, TempArraySize * sizeof(double) );
        ImagTemp := NIL;
    end;
    TempArraySize := 0;
end;
end;

procedure CalcFrequency (
    NumSamples: word;    { must be integer power of 2 }
    FrequencyIndex: word; { must be in the range 0 .. NumSamples-1 }
    var Realln: array of double;
    var Imagln: array of double;
    var RealOut: double;
    var ImagOut: double );
var
    k: word;
    cos1, cos2, cos3, theta, beta: double;
    sin1, sin2, sin3: double;
begin
    RealOut := 0.0;
    ImagOut := 0.0;
    theta := 2*PI * FrequencyIndex / NumSamples;
    sin1 := sin ( -2 * theta );
    sin2 := sin ( -theta );
    cos1 := cos ( -2 * theta );
    cos2 := cos ( -theta );
    beta := 2 * cos2;
    for k := 0 to NumSamples-1 do begin
        { Update trig values }
        sin3 := beta*sin2 - sin1;
        sin1 := sin2;
        sin2 := sin3;
        cos3 := beta*cos2 - cos1;
        cos1 := cos2;
        cos2 := cos3;
        RealOut := RealOut + Realln[k]*cos3 - Imagln[k]*sin3;
        ImagOut := ImagOut + Imagln[k]*cos3 + Realln[k]*sin3;
    end;
end;

begin { Unit initialization code }
    TempArraySize := 0; {flag that buffers RealTemp, Reallmag not allocated}
    RealTemp := NIL;
    ImagTemp := NIL;
end.

```

Bibliografía

1. Estévez Báez M., Movsisyants S.A., Denisova V.V., Nikitina L.I. (1982) Particularidades de los procesos extralentos en algunas enfermedades del cerebro. Revista de Fisiología Humana, URSS, T.8, No. 5, pp. 851-860 (en idioma ruso).
2. Oppenheim A.V., Shafer R.W. (1989) Discrete-time signal processing. Englewood Cliffs, N.J.:Prentice-Hall.
3. Cooley J.W. and Tukey J.W. (1965) An algorithm for the machine computation of complex Fourier series. Mathematics of Computation Vol 19:297-301.
4. Merri M, Farden DC, Mottley JG, Titlebaum EL. Sampling frequency of the electrocardiogram for the spectral analysis of heart rate variability, IEEE Trans Biomed Eng 1990; 37:99–106.
5. Friesen GM, Jannett TC, Jadalloh MA, Yates SL, Quint SR, Nogle HT. A comparison of the noise sensitivity of nine QRS detection algorithms. IEEE Trans Biomed Eng 1990; 37:85–98.
6. Blackman R.B. and Tukey J.W. (1958) The measurement of power spectra. Dover Publications, New York.