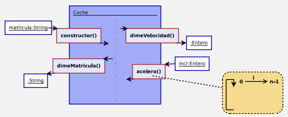
Grado en Ingeniería de Sistemas de Telecomunicación, Sonido e Imagen

Programación 2



Práctica 1

UNIVERSIDAD POLITECNICA DE VALENCIA

Escola Politècnica Superior de Gandia

DSIC

Departament de Sistemes Informàtics i Computació

Sesiones

- Grupos lunes: feb-11, feb-18, feb-25, mar-4
- Grupos viernes: feb-15, feb-22, mar-1, mar-8
- examen: mar-4 (10-12h)

Práctica 1

Objetivos

- Diseño de un programa con varias clases relacionadas mediante tenencia, delegación, herencia, polimorfismo.
- Ingeniería inversa del código de una clase.
- Utilización de clases de biblioteca
- Implementación de métodos dados su diseño y su algoritmo.
- Introducción a algoritmos elementales para el análisis de una señal.

¡ Atención !

▷ Se recuerda que las prácticas deben prepararse antes de acudir al aula informática. Esto incluye leer el código proporcionado y probarlo.

▷ La realización de las prácticas es un trabajo individual y original. En caso de plagio se excluirá al alumno de la asignatura. Por tanto es preferible presentar el trabajo realizado por uno mismo aunque éste tenga errores.

1 Introducción intuitiva al análisis en frecuencia de señales

Una señal es la variación en el tiempo de una magnitud medible. Por ejemplo, en el caso del sonido (fenómeno producido por la vibración del aire), se puede medir (entre otras magnitudes)

- la amplitud de la oscilación del aire que vibra (que llamamos intensidad sonora).
- la cantidad de oscilaciones que se producen por unidad de tiempo (que llamamos frecuencia).

Las señales se representan gráficamente dibujando la variación de la magnitud respecto al tiempo. Si la variación es continua, suave[1], y periódica[2], entonces, la señal se puede representar como una funcion matemática de la familia senoidal. En concreto, se dice que la señal es pura cuando su comportamiento es *exactamente* como *una única* función seno (o coseno). Es decir, además de ser periódica, cuando la magnitud medida sube, se alcanza siempre el mismo máximo, y cuando la magnitud desciende se alcanza el mismo mínimo. La expresión matemática de una señal pura (utilizando $cos()$) es

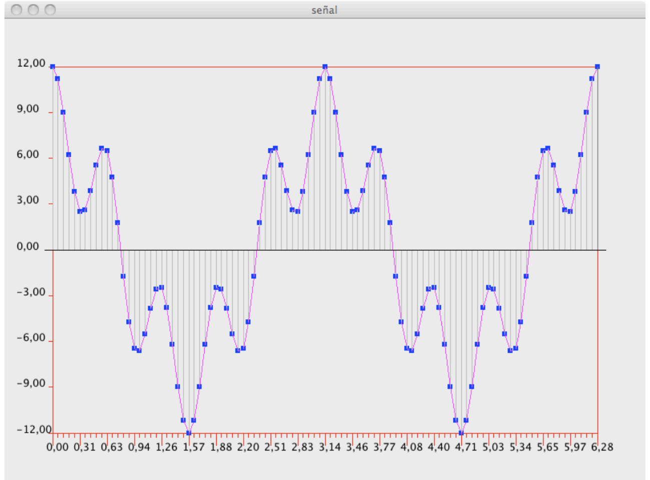
$$y(t) = Acos(\omega t + \phi)$$

Siendo

- $y(t)$ el valor de la magnitud en el instante t.
- A el valor máximo de la magnitud (la amplitud).
- ω el pulso (o velocidad angular), en radianes. Se utiliza para indicar la velocidad a la que cambia la magnitud medida. Si tenemos en cambio f, la frecuencia (cantidad de veces que la magnitud alcanza un máximo por unidad de tiempo), entonces simplemente $\omega = 2\pi f$.
- ϕ la fase (desplazamiento respecto el punto $x = 0$).

[1] Derivable en cada instante.
[2] Los valores medidos se repiten cada cierto intervalo de tiempo fijo.

Ningún fenómeno físico real se corresponde exactamente con una senoidal pura (sólo un seno o sólo un coseno). De hecho, aunque un fenómeno sea periódico, los máximos y mínimos *locales* alcanzados no son siempre los mismos, como por ejemplo:



Sin embargo, se puede demostrar que toda señal se puede obtener como la suma de senoidales. Es decir, como sumas de senos y cosenos. La anterior gráfica corresonde a 6.28 segundos[3] de esta señal:

$$f(t) = 8cos(2t) + 4cos(10t)$$

siendo, como hemos visto,

- 8 y 4 la amplitud de cada senoidal pura
- 2 y 10 el pulso ω de cada señal pura (y, equivalentemente, $1/\pi$ y $5/\pi$ sus frecuencias).

[3] Exactamente 2π segundos.

Es importante saber que los fenómenos físicos que nos conviene utilizar como señales de comunicación (para transmitir información) pueden ser representados como funciones senoidales -a su vez compuestas por sumas de senoidales puras. En particular, el sonido y los fenoménos electromagnéticos (como la eletricidad, la luz, las ondas de radio) tienen las propiedades explicadas.

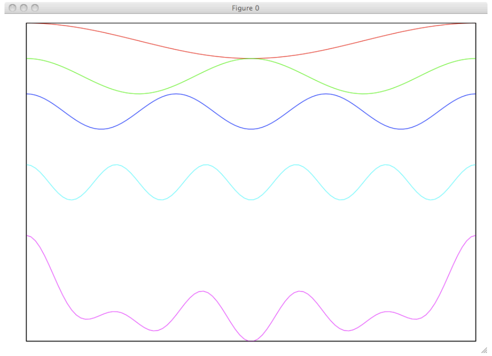
En el tratamiento de señales es muy importante su análisis para obtener su descomposición en senoidales puras, no sólo para conocer sus constituyentes, sino porque simplifica muchos procedimientos que hay que realizar sobre ella, como por ejemplo filtros. Con este análisis se obtiene el *espectro de frecuencias* de la señal.

Como ilustración de lo anterior podemos citar el caso del sonido de instrumentos musicales. Cuando un instrumento toca una nota, el sonido que produce contiene varios tonos puros. El tono que percibimos de la nota es aquel con mayor amplitud y que además es el más grave (tiene menor frecuencia). Ese tono percibido se llama *fundamental*. Pero además del tono fundamental, suenan simultáneamente otros tonos puros con una frecuencia múltiplo $2, 3, 4, 5, \ldots$ del fundamental. Estos otros tonos se llaman *armónicos* y cada uno suena con una intensidad distinta.

La combinación de armónicos del sonido producido por un instrumento en cada nota (junto con el *ataque*[4] y el *decaimiento*[5]) permite distinguir la misma nota emitida por dos instrumentos diferentes y se conoce con el nombre de *timbre*.

[4] El ruido que se produce al iniciar la emisión de la nota.
[5] Cómo se atenua el sonido de la nota en el tiempo.

1.1 Fundamentos matemáticos



La señal inferior es
la suma de las otras

Fue el matemático francés Joseph Fourier quien generalizó la idea de descomponer una función en una suma de senoidales al desarrollar un método de análisis llamado en su honor transformada de Fourier.

El razonamiento simplificado es el siguiente.

Supongamos que tenemos una señal en función del tiempo $y(t)$ definida de la siguiente forma[6]:

$$y(t) = a_1 cos(wt) + a_2 cos(2wt) + a_3 cos(3wt) + ... + a_N cos(Nwt)$$

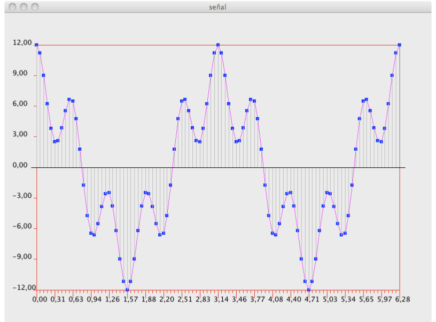
La anterior fórmula sirve para *sintetizar* la señal. Es decir, nos dice cómo calcular la amplitud de la señal en un momento dado: es la *receta de cocina* de la señal. Con esa expresión asumimos que la señal se sintetiza como una suma de frecuencias múltiplo de una fundamental, cada una con un peso. Es decir, la frecuencia fundamental es $cos(wt)$, los armónicos (frecuencias múltiplo) son $cos(wt)$, $cos(2wt)$, $cos(3wt)$, ..., y sus amplitudes respectivas a_1, a_2, a_3, ...

[6] Como simplificación utilizamos sólamente sumas de cosenos

Por ejemplo, ya hemos visto que si la expresión de síntesis de una señal es

$$y(t) = 8cos(2t) + 4cos(10t)$$

entonces la señal es



La fórmula de síntesis se puede escribir abreviadamente

$$y(t) = \sum_{i=1}^{N} a_i cos(iwt)$$

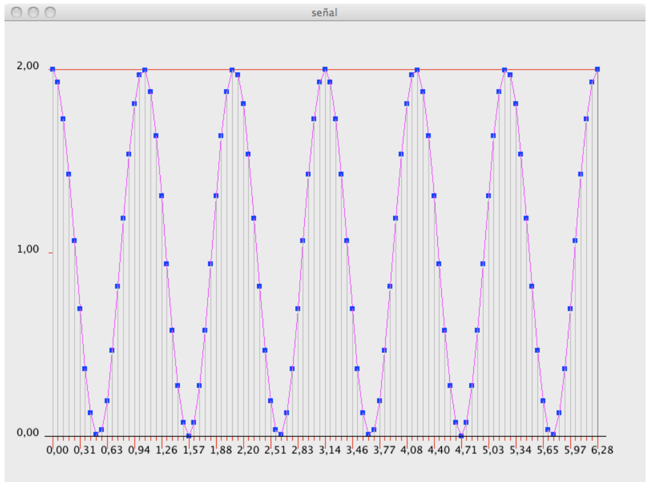
La cuestión es cómo obtener una fórmula de análisis, que solamente a partir de muestras de la señal (sin saber su fórmula de síntesis), nos diga qué armónicos tiene la señal y cuál es la amplitud de cada uno.

Para conseguirlo, en primer lugar hay que fijarse en la siguiente propiedad de las senoidales. En un periodo[7], como por ejemplo $[0, 2\pi]$, la integral del producto de dos cosenos

$$\int_0^{2\pi} a_1 cos(w_1 t) a_2 cos(w_2 t) dt$$

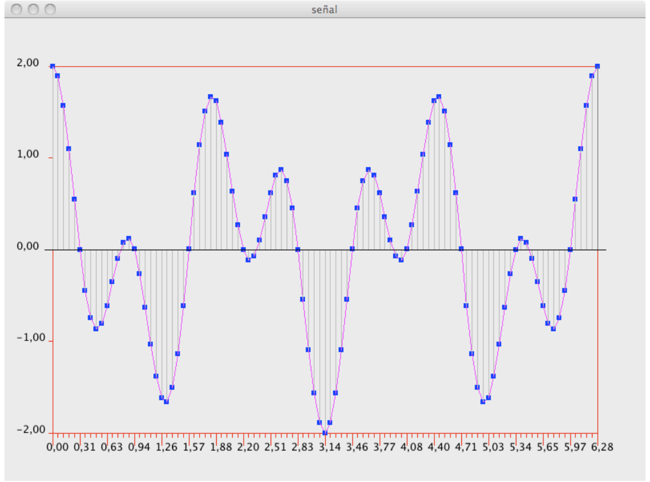
resulta que

- la integral es mayor que 0 si ambos cosenos tienen la misma frecuencia f ($w_1 = w_2 = 2\pi f$). Es más, la integral vale[8] $\pi a_1 a_2$.
  Por ejemplo, la siguiente gráfica muestra $f(t) = 2cos(3t)cos(3t)$ en $[0, 2\pi]$. Véase que la función está siempre sobre el eje 0. La integral vale 2π.



[7] Intervalo de tiempo en el que la señal se repite exactamente una cantidad entera de veces.
[8] Compruébese analíticamente.

- la integral es exactamente 0 si ambos cosenos tienen frecuencia distinta (y por tanto, $w_1 \neq w_2$). Por ejemplo, la siguiente gráfica muestra $f(t) = 2cos(2t)cos(5t)$ en $[0, 2\pi]$. Véase que el área sobre el eje X es la misma que bajo él.



Utilizando la anterior propiedad podremos calcular (en un periodo) la integral de una señal $y(t)$ multiplicada por un armónico concreto k ($cos(kwt)$):

$$\int_0^{2\pi} y(t)cos(kwt)dt$$

$=$ $\langle$explicación: $y(t) = \sum_{i=1}^{N} a_i cos(iwt)\rangle$

$$\int_0^{2\pi} (\sum_{i=1}^{N} a_i cos(iwt))cos(kwt)dt$$

$=$ $\langle$Por la propiedad del producto de cosenos, el único producto de cosenos con área no nula es en el que coincide i=k$\rangle$

$$\int_0^{2\pi} a_k cos^2(kwt)dt$$

$=$ $\langle$sencillo ejercicio de integración$\rangle$

$a_k\pi$

En definitiva hemos demostrado que

$$\int_0^{2\pi} y(t)cos(kwt)dt = a_k\pi$$

y, por tanto, que

$$a_k = \frac{1}{\pi}\int_0^{2\pi} y(t)cos(kwt)dt$$

Es decir, para calcular la amplitud del armónico k en una señal, basta con calcular la integral de la señal multiplicada por el armónico.

Hemos utilizado como periodo temporal [0,2π], en el cuál el coseno (o cualquier armónico suyo) es periódico. Si hay que analizar unas muestras tomadas en un intervalo de tiempo de T tendremos que

$$a_k = \frac{2}{T}\int_0^T y(t)cos(kwt)dt$$

Si disponemos de una señal de N muestras podremos calcular los armónicos de 1 a $N/2$.

En definitiva, disponemos de

- Una fórmula de síntesis para generar una señal

$$y(t) = \sum_{i=1}^{N} a_i cos(iwt) \tag{1}$$

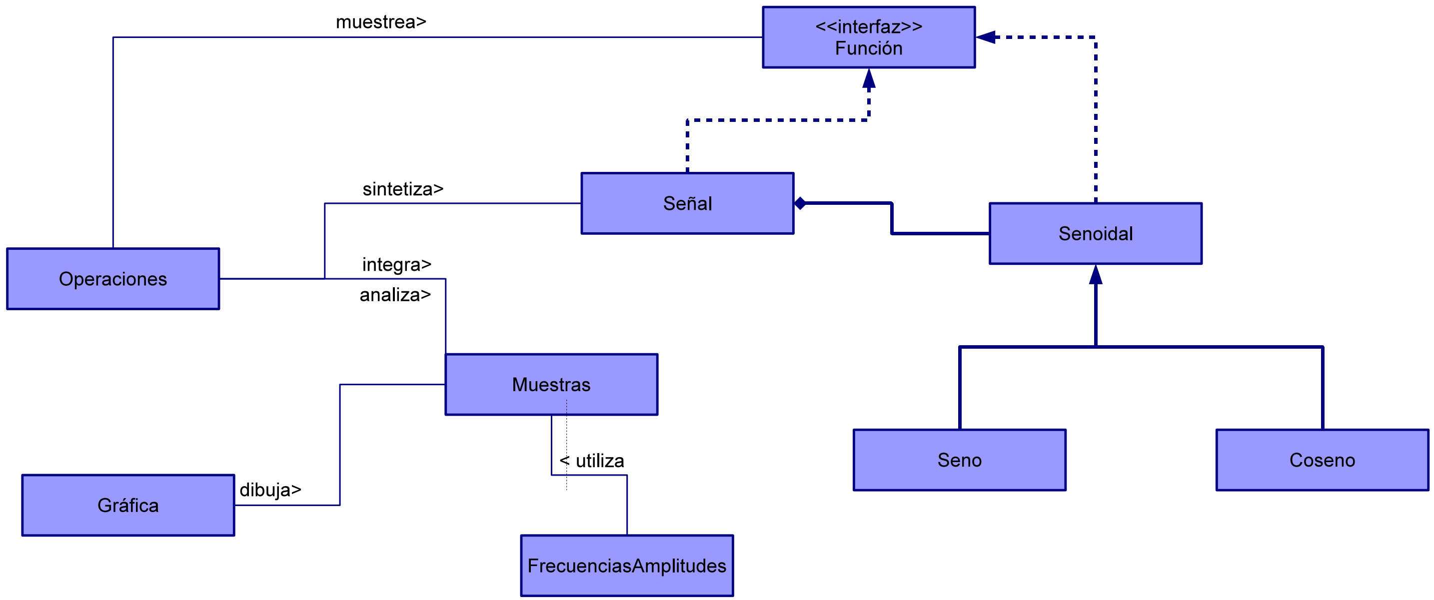
- Una fórmula análisis para obtener las frecuencias y su amplitud presentes en un señal dada

$$f(k) = \frac{2}{T}\int_0^T y(t)cos(kwt)dt \tag{2}$$

2 Diseño de un programa de análisis de señales

Para desarrollar el programa aprovecharemos algunas clases de la práctica 4 de programación 1 (ampliándolas) y añadiremos varias clases nuevas.

En un programa orientado a objeto, lo habitual es que haya bastantes clases (cada una representa un aspecto del problema a resolver) relacionadas entre sí. Por eso, antes de realizar el diseño detallado de cada clase, hay que establecer las relaciones que existen entre ellas. El siguiente diagrama muestra las relaciones entre las clases del programa.



¡ Trabajo a realizar !

En el anterior diseño, escribe al lado de cada relación (línea entre 2 clases) cómo se indica en Java esa relación. Por ejemplo, la relación entre señal y función es Senyal implements Funcion.

2.1 La interfaz Funcion

Una interfaz define un contrato. En este caso, cualquier clase que quiera ser una función tendrá que implementar la interfaz Funcion, que únicamente tiene el método double valor (double x); Es decir, quien quiera ser una función tendrá que tener un método valor(). Las clases Señal, Senoidal, Seno, Coseno (e incluso la clase Polinomio de la anterior práctica) son funciones e implementan la interfaz Funcion.

¿Por qué se hace esto? Porque simplifica y ahorra código. El método Operaciones.tomarMuestras() puede muestrear cualquier cosa que tenga un método valor() -es decir que sea una función. Es lo único que necesita, no le hace falta distinguir entre el tipo de función concreto. Si no lo hiciéramos así, habría que escribir una función distinta para muestrear cada tipo de función diferente (señal, seno, coseno, polinomio).

¡ Trabajo a realizar !

Lee, analiza y realiza la ingeniería inversa[9] de la interfaz Funcion.

[9] Saca su diseño a partir del código.

2.2 Las clases Senoidal, Seno y Coseno y su relacion con Senyal

Esencialmente un seno y un coseno es prácticamente la misma función: una senoidal. De una senoidal nos interesa guardar tres cosas: su amplitud, su pulso w[10] y su fase. Toda esta gestión es la misma para un seno y para un coseno, que sólo se diferencian en si finalmente el cálculo utiliza sin() o cos(). Por tanto, en primer lugar nos interesa escribir una clase (Senoidal) que tenga el código común de las clases Seno y Coseno.

Es importante darse cuenta que las clases Seno y Coseno no son las funciones sin() y cos(). Son contenedores de los parámetros amplitud, w y fase, con un método valor() que hace el cálculo correspondiente teniendo en cuenta esos parámetros.

Por otro lado, ya hemos visto que podemos representar una señal como una suma de senos y cosenos. Y como ambos son senoidales, en la clase Senyal podremos guardar a la vez en un mismo vector de senoidales, tanto objetos seno como objetos coseno, lo cual simplifica enormemente la implementación de Senyal.

¡ Trabajo a realizar !

1. Lee, analiza y realiza la ingeniería inversa del código de la clase Senoidal.
2. ¿Qué métodos tienen la misma implementación y cuáles no en las clases Seno y Coseno, teniendo en cuenta que heredan de Senoidal?
3. Lee, analiza y realiza la ingeniería inversa del código de las clases Seno y Coseno.
4. Implementa el método Coseno.valor() a semejanza del método Seno.valor().
5. Lee, analiza y realiza la ingeniería inversa de la clase Senyal.

[10] $2\pi frec$

6. El algoritmo del método Senyal.valor(). es
$$\sum_{i=1}^{N} senoidal_i.valor(t)$$
es decir la suma del valor de los objetos senoidales que Senyal almacena en un vector (¿cuál?).
7. Implementa el método Senyal.valor().

Ejercicio

Para probar la anterior implementación escribe una funcion Main.pruebaSenyal() donde crees objetos para representar

$$y(t) = 8cos(2t) + 4cos(10t)$$

A continuación saca 100 muestras de dicha señal en el intervalo temporal $[0, 2\pi]$ y guarda las muestras obtenidas en un fichero.

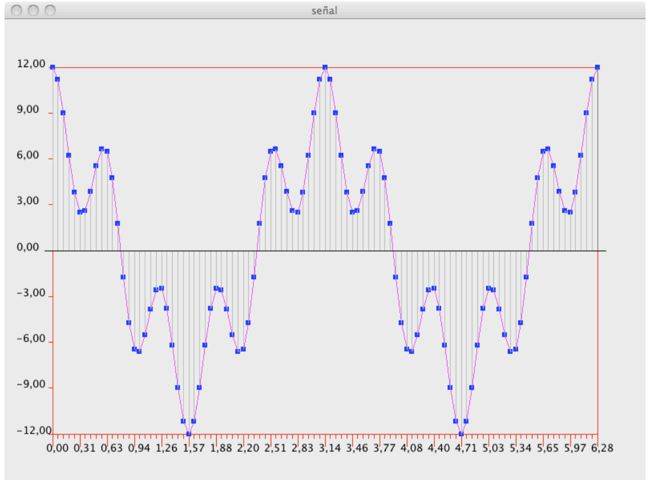
Este es el código de la prueba:

```
Coseno c1 = new Coseno (8.0, 2.0);
Coseno c2 = new Coseno (4.0, 10.0);
Senyal s = new Senyal (c1, c2);

double periodo  = 2.0 * Math.PI; // segundos
double ts  = periodo / 100; // 100 muestras en el periodo
Muestras m = Operaciones.tomarMuestras (s, 0.0, periodo, ts);

Utilidades.escribeEnNuevoFichero("senyal.dat", m.aTexto());
Grafica graf1 = new Grafica ();
graf1.dibuja(m, "prueba senyal");
```

Comprueba que aparece esta gráfica.



Voluntariamente: en matlab representa $y(t) = 8cos(2t)+4cos(10t)$ y también las muestras del fichero "senyal.dat". Compara ambas gráficas entre sí, y con la representación que realiza nuestro programa.

2.3 La clase Muestras

La clase Muestras es una ampliación de la que se utilizó en la práctica 4 de Programación 1.

A los métodos que ya tiene dicha clase añadiremos 2 más:

- Muestras multiplicar(Muestras otras);
  debe devolver un nuevo objeto muestras a partir del producto de las nuestras muestras y otras. De tal forma que la muestra i en el resultado sea el producto de nuestra muestra i por la muestra i en otras.
- boolean mismosIndices(Muestras otras);
  Este método es de utilidad y se da implementado. Devuelve verdadero si nuestros índices y los de otras son los mismos (requisito indispensable para poder multiplicarlas).

¡ Trabajo a realizar !

1. Realiza la ingeniería inversa de la clase Muestras
2. Escribe el algoritmo del método Muestras.multiplicar() y después impleméntalo.

Ejercicio

Para probar el anterior método, escribe una funcion Main.pruebaMultiplicar() que muestree $2cos(2t)$ por un lado y $cos(5t)$ por otro, calcule el producto de las muestras, las guarde en un fichero y las represente.

Es decir:

```
public static void pruebaMultiplicar () throws IOException  {
  double periodo  = 2.0 * Math.PI; // segundos
  double ts  = periodo / 100; // 100 muestras en el periodo
```

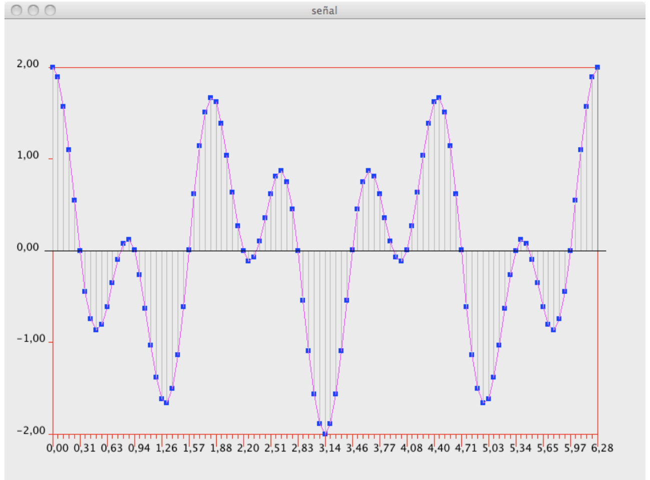
```
    Coseno c1 = new Coseno (2.0, 2.0);
    Coseno c2 = new Coseno (1.0, 5.0);

    Muestras m1 = Operaciones.tomarMuestras (c1, 0.0, periodo, ts);
    Muestras m2 = Operaciones.tomarMuestras (c2, 0.0, periodo, ts);
    Muestras m3 = m1.multiplicar(m2);

    Utilidades.escribeEnNuevoFichero("senyal.dat", m3.aTexto());

    Grafica graf1 = new Grafica ();
    graf1.dibuja(m3, "prueba multiplicar");
  }
```

Comprueba que aparece la siguiente gráfica.



Voluntariamente: en matlab, compara la representación de $y(t) = cos(2t)cos(5t)$ con la representación de las muestras obtenidas al realizar el producto (fichero "senyal.dat").

2.4 La clase FrecuenciasAmplitudes

La clase FrecuenciasAmplitudes sirve para guardar una serie de parejas de números reales frecuencia-amplitud (o análogamente w-amplitud)

Como la clase Muestras guarda ya una serie de pares de números reales (índice-muestra) resulta muy conveniente tratar de aprovecharla. Efectivamente, utilizando delegación, hacemos que FrecuenciasAmplitudes utilice Muestras para guardar los pares, haciendo una adaptación de frecuencia a índice, y de amplitud a muestra.

¡ Trabajo a realizar !

Lee y realiza la ingeniería inversa de la clase FrecuenciaAmplitudes

2.5 Métodos estáticos de la clase Operaciones

Vamos a añadir nuevos métodos estáticos a la clase Operaciones para realizar la síntesis y el análisi de una señal.

2.5.1 Sintetizar una señal

El método que realiza la síntesis es

```
public static Senyal sintetizarSenyal (FrecuenciasAmplitudes fa);
```

que dado un objeto FrecuenciasAmplitudes devuelva un objeto Senyal utilizando el siguiente algoritmo:

$cosenos$: vector de Coseno

0 $\xrightarrow{\text{i}}$ fa.talla()-1

  $a \leftarrow$ amplitud i en fa
  $f \leftarrow$ frecuencia i en fa
  $cos \leftarrow$ nuevo Coseno(a, f)
  $cosenos \leftarrow cosenos \cup \{cos\}$

devolver nueva Senyal($cosenos$)

Como acabamos de ver, para simplificar, la señal se sintetiza sólo como suma de cosenos.

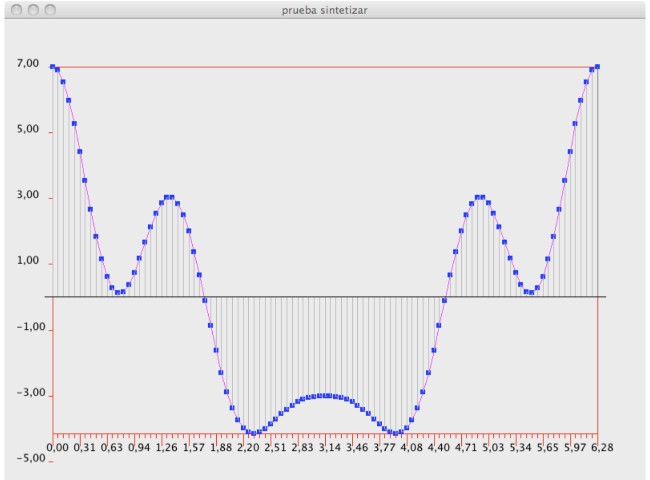
¡ Trabajo a realizar !

1. ¿Qué hace el anterior algoritmo?
2. Implementa el método Operaciones.sintetizarSenyal().

Ejercicio

Para probar el método de sintetizar, escribe una funcion Main.pruebaSenyal() en donde se sintetice un objeto Senyal a partir de un objeto FrecuenciasAmplitudes que contenga como frecuencias[11]: 1, 4 y 5; y como sus respectivas amplitudes: 4, 2 y 1. A continuación muestrea la senyal y representa las muestras.

Comprueba que la gráfica es como la siguiente.



[11] Mejor dicho, como w.

2.5.2 Analizar una señal

Para analizar una señal necesitamos un método

`public static double integral (Muestras m)`

que dado un objeto `Muestras m` devuelva su integral, utilizando el algoritmo:

$sum \leftarrow 0$

1 $\xrightarrow{\texttt{i}}$ m.talla()-1

  $base \leftarrow$ indice(i) - indice($i-1$) de m
  $sum \leftarrow sum$ + (muestra(i) de m) $\times$ $base$

devolver sum

Finalmente, escribiremos el método

`public static FrecuenciasAmplitudes analizarMuestras (Muestras m);`

que dado un objeto `Muestras m` con las muestras tomadas de una señal desconocida, devuelva un objeto con la amplitud de cada armónico presente en la señal.

Recordemos que la fórmula de análisis para averiguar la amplitud del armónico k se calcula

$$f(k) = \frac{2}{T}\int_0^T y(t)cos(kwt)dt$$

Así pues el algoritmo de analizarMuestras() será:

$vFrec \leftarrow$ nuevo vector de reales

$vAmpl \leftarrow$ nuevo vector de reales

$T \leftarrow$ m.periodo

$t_s \leftarrow$ m.indice(1) - m.indice(0) $\langle$ averiguamos el intervalo de muestreo $\rangle$

$\langle$ para todo armónico k posible $\rangle$

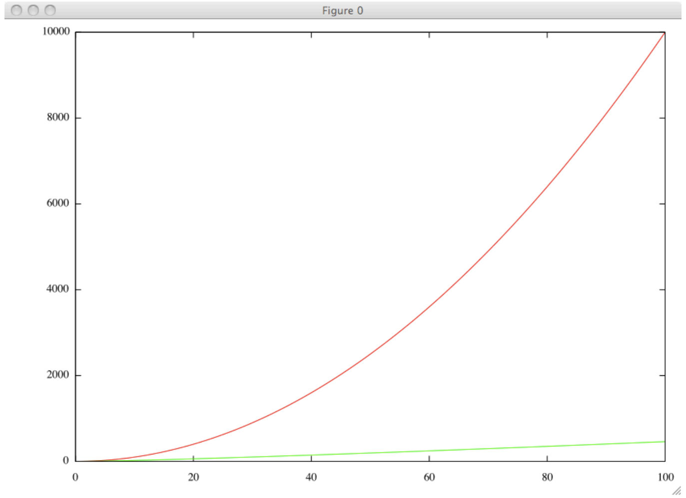
1 $\xrightarrow{\text{k}}$ m.talla()/2

$vFrec \leftarrow vFrec \cup \{k\}$
$c \leftarrow$ nuevo Coseno de amplitud 1 y $w = k$ $\langle$ suponemos que la fundamental es $cos(1t)$ $(w = 1)$ $\rangle$
$mc \leftarrow$ tomar muestras de c en el periodo $[0, T]$ cada t_s segundos
$mp \leftarrow m \times mc$
$a \leftarrow \frac{2}{T} \times$ integral de mp
$vAmpl \leftarrow vAmpl \cup \{a\}$

devolver nuevo FrecuenciasAmplitudes$(vFrec, vAmpl)$

El anterior algoritmo tiene un coste[12] N^2 (siendo $N =$ la cantidad de muestras (m.talla()), que lo hace inapropiado para realizar análisis de muestras en casos prácticos. Afortunadamente, existe un algoritmo llamado transformada rápida de Fourier con un coste notablemente menor: $\log_2(N)N$ gracias al cual se ha podido desarrollar prácticamente toda la tecnología digital de comunicaciones que conocemos actualmente.

Véase la comparación de costes:



¡ Trabajo a realizar !

1. Relaciona la fórmula de análisis de la señal con los pasos del anterior algoritmo.
2. Implementa el método `public static double integral (Muestras m)` de la clase `Operaciones`.
3. Implementa el método `public static FrecuenciasAmplitudes analizarMuestras (Muestras m);` de la clase `Operaciones`.

[12] ¿Por qué?

Ejercicio

Para probar el análisis de una señal, escribe una función Main.pruebaAnalizar() que

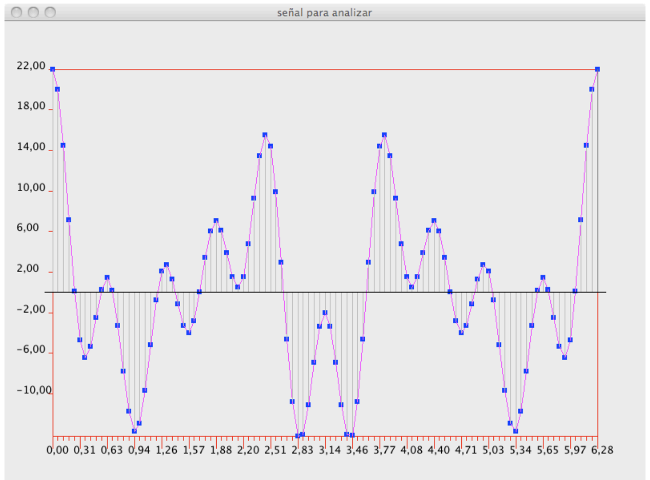
1. Obtenga un objeto Senyal para
$$y(t) = 8cos(3t) + 4cos(5t) + 3cos(8t) + 7cos(10t)$$
2. Tome 100 muestras en el periodo temporal $[0, 2\pi]$ y las represente.
3. Analice las anteriores muestras obteniendo el correspondiente objeto FrecuenciasAmplitudes, que deberá también representar.
4. Tomando el anterior objeto FrecuenciasAmplitudes sintetice una nueva señal y la represente, comprobando que vuelve a salir la gráfica original.

En concreto las gráficas que deberían aparecer al trabajar con

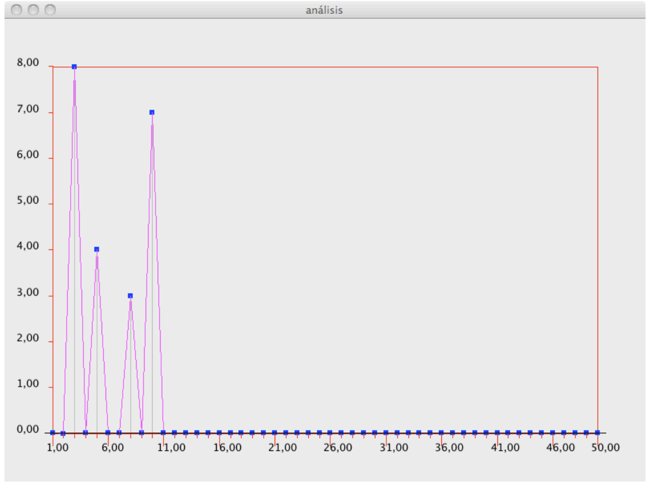
$$y(t) = 8cos(3t) + 4cos(5t) + 3cos(8t) + 7cos(10t)$$

son:

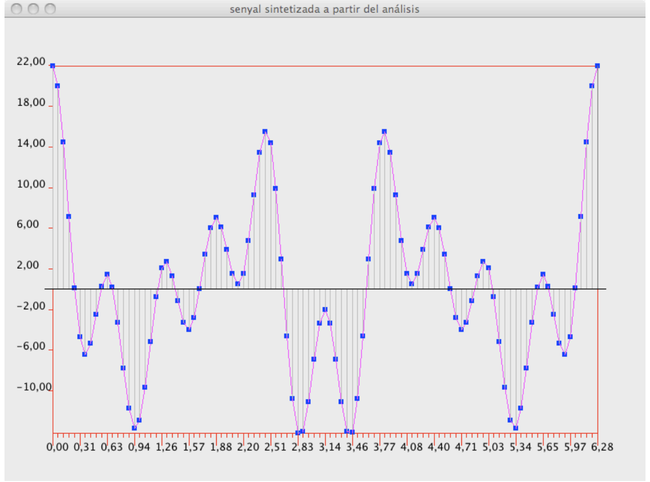
- La señal que hay que analizar (el eje X representa el tiempo):



- El resultado del análisis de la señal. El eje X representa las frecuencias (w en realidad) y así podemos ver qué frecuencias están presentes en la señal y en qué cantidad. Compara la gráfica con $y(t) = 8cos(3t) + 4cos(5t) + 3cos(8t) + 7cos(10t)$



- Si sintentizamos la señal a partir de su análisis vuelve a aparecer la gráfica original.



3 Entregas

3.1 Trabajo realizado en papel

- Para la primera sesión hay que tener preparado
  - El trabajo de la sección 2.1
  - El trabajo de la sección 2.2 y su ejercicio.
- Para la segunda sesión hay que tener preparado
  - El trabajo de la sección 2.3 junto con su ejercicio.
  - El trabajo de la sección 2.4.
- Para la tercera sesión hay que tener preparado
  - Los trabajos de la sección 2.5 y sus ejercicios.

3.2 Código completo de la práctica

El fichero p1.zip con el código completo se debe entregar electrónicamente y presentar oralmente al profesor antes del 8 de marzo.

28 enero 2013