UNIVERSIDAD DE MURCIA

DEPARTAMENTO DE MATEMÁTICAS

ANTONIO J. PALLARÉS Y SALVADOR SÁNCHEZ-PEDREÑO

18 de diciembre de 2017.

Crea en tu proyecto el paquete `practica7` donde incluir los ejercicios.

Ejercicio 1

1. *Abre en la Zona de Recursos los ficheros* `ParaClase_Polinomio.txt` *y copia los métodos que incluyen en la clase* `Polinomio` *del paquete auxiliar de tu proyecto.*
2. *Incluye los ficheros* `FuncionDeriv.java` *y* `MetodosEcuacionNoLineal.java` *en el paquete* `auxiliar` *de tu proyecto.*
3. *Descarga de la Zona de Recursos en el Aula Virtual el fichero* `cn1vejemplosUnidad4.zip` *y descomprímelo en tu escritorio.*

   Copia la carpeta `cn1vejemplosUnidad4` *en la carpeta* `src` *de tu proyecto.*
4. *Comprueba el resultado de todo este movimiento de ficheros, ejecutando las distintas clases principales incorporadas en este paquete con los ejemplos mostrados en clase.*

Ejercicio 2 *Considera las funciones $f(x) = \operatorname{sen}(x)\pi/(2(1+x^2))$ y $g(x) = \exp(-x)$. Representa en un PanelDibujo las gráficas de las dos funciones con $x \in [0, 10]$ utilizando una escala adecuada en el eje OY. Observa los puntos de corte de las dos gráficas que se corresponden con las abscisas x_i que son soluciones de la ecuación $f(x) == g(x)$*

Utiliza el/los métodos de resolución de ecuaciones que consideres adecuados para encontrar las abscisas x_i referidas antes y añade los puntos donde se cortan las dos gráficas al PanelDibujo inicial.

Ejercicio 3 *Los métodos de Newton y de la secante no garantiza orden de convergencia superior a uno cuando el cero de f es múltiple.*

Consideramos la función $f(x) = (e^x - x^2 + 2x - 1 - e)^k$ que tiene un cero de multiplicidad k en $x = 1$. En este ejercicio tratamos de comparar la velocidad de aproximación en este caso para distintos valores de $k = 2, 3, 4y5$:

1. *con el método de Newton ($x_0 = 0.8$).*
2. *con el método de la secante ($x_0 = 0.7$ y $x_1 = 0.8$).*
3. *Con el método de Newton aplicado a la función $g(x) = f(x)/f'(x)$. Es fácil comprobar que si c es un cero de f, c es un cero simple de g. Puesto que se ha de derivar g y esto necesita la derivada segunda de f, realizaremos el cálculo utilizando la aproximación de las derivadas mediante tres puntos ($g'(x) \approx \frac{g(x+h)-g(x-h)}{2h}$).*
4. *Utilizando el método de punto fijo de Steffensen para la función del método de Newton $F(x) = x - g(x)$.*

Para analizar los resultados tengase en cuenta que $g'(1) = \frac{1}{k}$ y $F'(1) = \frac{k-1}{k}$.

Ejercicio 4 *Incorpora a la clase* `Polinomio` *los algoritmos contenidos en el fichero* `ParaClasePolinomio_2.txt`*, e importa en la clase Polinomio la clase ArrayList que necesita el método de la bisección de Sturm.*

Los métodos a incorporar son:

1. ***Sturm**, es un conjunto de algoritmos para aproximar simultáneamente todas las raíces reales de un polinomio:*
   (a) `public static int cambioSigno(double[] lista)`
   determina el número de cambios de signo que hay en una lista de números reales.

   (b) `public Polinomio[] sucesionSturm()` *devuelve la lista de polinomios con la sucesión de Sturm asociada a un polinomio.*

   (c) `public void biseccionSturm(double precision, ArrayList intervalos, Polinomio[] sturm, double a, double b)`
   genera una lista de intervalos de longitud menor que `precisiom` *que contienen una sóla raiz real.*

   (d) `public double[] bisecSturm(double precision, double a, double b)`
   devuelve una lista de aproximaciones a todas las raíces reales del polinomio (los puntos medios de los intervalos que proporciona el método anterior).
2. *Los métodos*
   `public Complex[] buscaRaicesAleatorio(boolean real, int nmaxintentos)`
   que aplicado a un Polinomio, devuelva una lista de sus raíces o un mensaje de Error.
   `public Complex[] buscaRaicesAleatorioPCR(boolean real, int nmaxintentos)`
   Si además el polinomio es real, cada vez que encuentre una raíz compleja, también añadirá a la lista de raíces la raíz conjugada.

Comprueba

y Bisección de Sturm aplicándolos en la busqueda de raíces del polinomio $p(x) = x^4 - 4x^3 + 7x^2 - 5x - 2$ de los ejemplos 4.6.2-6 y 13, de las guías de clase para aproximar sus raíces:

1. *Usa Laguerre con $z0 = -2$, $z0 = 3$ y $z0 = 1 + i$.*

2. *Aproxima simultáneamente los dos ceros reales con el método de la bisección de Sturm.*

Compruébalos aplicándoseloa al polinomio

$$p(x) = x^{11} - 12x^{10} - 3x^9 + 25x^8 - x^7 - 3x^5 - 12x^3 + 12x^2 - x + 3 * Math.PI$$

Obteniendo primero una lista de de sus ceros reales. Después deflacionando el esos ceros se reduce el grado del polinomio y se buscan el resto de raices de manera aleatoria.

Reune todas las raices en una lista `Complex[] raices` *y comprueba la bondad de los resultados reconstruyendo el polinomio con el método* `Polinomio.constrido` *y midiendo la suma de los cuadrados de las diferencias de los coeficientes*

$$\sum |p.coeficientes()[k] - consturido.coeficientes()[k]|^2$$

(Como el proceso es aleatorio, una vez obtenida copia la lista de las raíces en un comentario dentro del mismo fichero java donde se calculan, y vuelve a ejecutar la clase.)

Ejercicio 5 *Comprobad la efectividad de los métodos implementados combinándolos para determinar las raíces de los polinomios, buscando primero las raíces reales con el método de Sturm, deflacionando el polinomio en éstas, y buscando las raíces complejas con el método*

1. $p(x) = x^5 - 5.6x^2 + 10x - 32$
2. $p(x) = 3x^{11} - 12x^{10} - x^9 + 23x^8 - x^7 + 32x^5 - 123x^3 + 12x^2 - x + 15$
3. $p(x) = 5x^{14} - 0.5x^11 + 2x^{10} + 0.8x^9 + 2x^8 - x^7 + 32x^5 - 13x^3 + 2x^2 - x - 1$

y compara los polinomios con los reconstruidos a partir de sus raíces.

Ejercicio 6 *El objetivo de este ejercicio es calcular todos los ceros de un polinomio real del que conocemos que todos los ceros son reales.*

Los polinomios de Legendre se definen de forma recurrente como:

$$\begin{aligned} L_0(x) &= 1 \\ L_1(x) &= x \\ (n+1)L_{n+1}(x) &= (2n+1)xL_n(x) - nL_{n-1}(x) \quad n \geq 1 \end{aligned}$$

Los ceros de estos polinomios están en el intervalo $[-1, 1]$ y son los puntos utilizados en uno de los métodos de cuadratura de Gauss que hemos estudiado.

Implementa en la clase `Polinomio` *un método*

`Polinomio[] legendre(int n)`

que devolverá todos los polinomios de Legendre, desde el de orden 0 al de orden n.

Calculad todos los ceros de L_n, para $n = 3, 4, 5, ...$

Podemos verificar las aproximaciones comparando los valores de la tabla incluida al final de la práctica.

TABLE

	$n=2$	
$x_1=0.577350269189626$		$a_1=1.000000000000000$
	$n=3$	
$x_0=0.000000000000000$		$a_0=0.888888888888889$
$x_1=0.774596669241483$		$a_1=0.555555555555556$
	$n=4$	
$x_1=0.339981043584856$		$a_1=0.652145154862546$
$x_2=0.861136311594053$		$a_2=0.347854845137454$
	$n=5$	
$x_0=0.000000000000000$		$a_0=0.568888888888889$
$x_1=0.538469310105683$		$a_1=0.478628670499366$
$x_2=0.906179845938664$		$a_2=0.236926885056189$
	$n=6$	
$x_1=0.238619186083197$		$a_1=0.467913934572691$
$x_2=0.661209386466265$		$a_2=0.360761573048139$
$x_3=0.932469514203152$		$a_3=0.171324492379170$
	$n=7$	
$x_0=0.000000000000000$		$a_0=0.417959183673469$
$x_1=0.405845151377397$		$a_1=0.381830050505119$
$x_2=0.741531185599394$		$a_2=0.279705391489277$
$x_3=0.949107912342759$		$a_3=0.129484966168870$
	$n=8$	
$x_1=0.183434642495650$		$a_1=0.362683783378362$
$x_2=0.525532409916329$		$a_2=0.313706645877887$
$x_3=0.796666477413627$		$a_3=0.222381034453374$
$x_4=0.960289856497536$		$a_4=0.101228536290376$
	$n=9$	
$x_0=0.000000000000000$		$a_0=0.330239355001260$
$x_1=0.324253423403809$		$a_1=0.312347077040003$
$x_2=0.613371432700590$		$a_2=0.260610696402935$
$x_3=0.836031107326636$		$a_3=0.180648160694857$
$x_4=0.968160239507626$		$a_4=0.081274388361574$
	$n=10$	
$x_1=0.148874338981631$		$a_1=0.295524224714753$
$x_2=0.433395394129247$		$a_2=0.269266719309996$
$x_3=0.679409568299024$		$a_3=0.219086362515982$
$x_4=0.865063366688985$		$a_4=0.149451349150581$
$x_5=0.973906528517172$		$a_5=0.066671344308688$
	$n=11$	
$x_0=0.000000000000000$		$a_0=0.272925086777901$
$x_1=0.269543155952345$		$a_1=0.262804544510247$
$x_2=0.519096129110681$		$a_2=0.233193764591990$
$x_3=0.730152005574049$		$a_3=0.186290210927734$
$x_4=0.887062599768095$		$a_4=0.125580369464905$
$x_5=0.978228658146057$		$a_5=0.055668567116174$