Universitat Autònoma de Barcelona

Facultat de Ciències

Pràctica 3

Autors:

Gerard Lahuerta & Ona Sánchez

1601350 — 1601181

13 de Maig del 2022

Índex

1 Presentació de la llibreria i informació d'interés

La llibreria de funcions utilitzada es tracta d'una modificació de la llibreria ja existent aleatori.h, programada en llenguatge *C*.
Consta d'un fitxer capcelera i un amb el codi de les funcions.
Per veure el funcionament de la llibreria aleatori.h és recomana accedir a la documentació.
La modificació feta es basa únicament en ampliar la llibreria amb 1 funció més: muller().
L'explicació i implementació de la funció s'expliquen als apartats 2.1.1 i 2.2 respectivament.

Per altra banda, el fitxer capcelera aleatori.h també s'ha modificat, concretament s'ha afegit la següent constant:

- E_MULLER: missatge d'error per longitud del vector coeficients errònia.

Recalcar a més, que cada funció conté el seu *CDocs* propi que conté la informació més rellevant de manera resumida: *input, output, descripció* i *informació rellevant.*
Acabar comentant que a aleatori.h no se li han afegit altres llibreries.

2 Presentació dels programes

2.1 Fitxer *aleatori.c*

2.1.1 muller

- Entrada:
  - Vector a omplir de coeficients: **double*** coef
  - Nombre de paràmetres a generar: **int** n
- Sortida: No retorna cap valor, **void**.
- Funcionament: Crida la funció **Normal()** n cops per generar n coeficients de forma aleatòria, i els normalitza.
- Informació rellevant: El valor del paràmetre n ha de coincidir amb la llargada del vector coef.

2.2 Fitxer *arrels.c*

- Entrada:
  - *(Opcional)* Informació extra: **unsigned char** extra
  - *(Opcional)* Nombre de mostres: **int** n
  - *(Opcional)* Valor de la llavor: **double** seed
- Sortida: Variable de control: **int**
- Funcionament: Genera n vegades 5 valors (mitjançant la funció **muller()**) que representen els coeficients d'un polinomi de quart grau sobre $\mathbb{S}^4$ i els classifica segons les seves arrels (4 reals, 4 complexes o 2 reals i 2 complexes).
- Informació rellevant: La classificació dels polinomis es mostra per `stdout`.

3 Control dels errors

3.1 Llibreria *aleatori.c*

3.1.1 Normal i Uniforme

Per la informació sobre el control d'errors d'aquestes funcions consultar la documentació de la llibreria.

3.1.2 muller

Per tal d'assegurar el correcte funcionament la funció, es confirmarà la correcta introducció dels paràmetres.
Es comprovarà que:

1. El valor del paràmere n és major que 0.

En cas d'inclompir aquesta condició el programa actuarà de la següent manera:

1. Es mostrarà per pantalla el missatge d'error corresponent, guardat a la variable E_MULLER (explicada anteriorment a la secció **1**).

2. No modificarà el vector introduït i finalitzarà inmediatament la funció.

3.2 Programa *arrels.c*

En cas d'introduïr algun dels paràmetres opcionals, es faran les següents comprovacions:

1. En cas d'introduïr un sol paràmetre, es llegirà com a **unsigned char** i es guardarà a la variable *extra* (tot valor que no sigui 0 es llegirà com a "True", solicitant informació extra).

2. En cas d'introduïr un segon paràmetre, es llegirà com a **int** i es guardarà a la variable n, en cas que el valor introduït sigui major a 0, sinó es guardarà el valor per defecte $2 \cdot 10^7$.

3. En cas d'introduïr un tercer paràmetre, es llegirà com a **double** i es guardarà a la variable *seed*.

4 Compilació i execució

4.1 Compilació del Programa

4.1.1 Makefile

Per facilitar la compilació del programa s'ha creat un fitxer $makefile$ que inclou tant les comandes per crear l'executable com altres comandes associades (*clean* i $clean\_all$ que explicarem més endavant) per la correcta gestió dels fitxers que s'obtenen de l'execució del makefile.
Per compilar el programa i generar l'executable (*arrels*) només cal escriure a la terminal on es troben tots els fitxers (inclós el $makefile$) la comanda make:

```
$make
```

Com hem comentat anteriorment, el fitxer makefile també conté comandes per la correcta gestió dels fitxers resultants d'obtenir l'executable, aquestes comandes són:

```
$make clean
$make clean_all
```

La comanda *clean* elimina tots el fitxers *.o* del directori mentre que la comanda $clean\_all$ elimina tant l'executable com els fitxers amb terminació *.o*.
Mencionar que no es poden utilitzar les dues comandes de manera seguida ja que donarà error.
Important: en cas de no utilitzar el sistema operatiu *Linux* (o semblants) o IOS cal modificar la variable $DELETE$ de l'arxiu $makefile$ per a poder utilitzar-lo (substituir per la comanda *del* en cas d'utilitzar Windows).

4.1.2 Compilació manual

En cas de voler compilar el programa de manera manual (comanda a comanda) utilitzar en ordre les següents comandes al terminal (una vegada ubicat al directori on es troben els fitxers).

```
$gcc -c aleatori.c aleatori.h -lm
$gcc -c arrels.c  aleatori.c -lm
$gcc arrels.o  aleatori.o -lm -o arrels
```

4.2 Execució del Programa

Comanda	**Descripció**
`./arrels`	Classifica 20 milions de polinomis de grau 4 segons les seves arrels i es mostra el percentage de cada tipus.
`./arrels` *extra*	Classifica 20 milions de polinomis de grau 4 segons les seves arrels i dona tota la informació del programa.
`./arrels` *extra n*	Classifica n polinomis de grau 4 segons les seves arrels i dona tota la informació del programa.
`./arrels` *extra n seed*	Classifica n polinomis de grau 4, generats usant la llavor seed, segons les seves arrels i dona tota la informació del programa.

Qualsevol argument de més que s'introdueixi al programa serà omés.

La informació que es mostra en introduir el paràmetre *extra* és el percentatge de cada tipus de polinomi, nombre de polinomis de cada tipus, nombre de polinomis que no segueixen cap dels casos contemplats, el seu percentatge, i el temps que triga en executar-se els programa.

5 Desenvolupament del programa

Partint d'un polinomi de grau 4 amb coeficients reals $ax^4 + 4bx^3 + 6cx^2 + 4dx + e$, que han estat generats de forma aleatòria, hem de classificar el polinomi segons els següents casos:

- Té 4 arrels reals
- Té 4 arrels complexes
- Té 2 reals i 2 complexes

Per tal de classificar el polinomi generat, tenim els discriminants:

$$P = ae - 4bd + 3c^2 \qquad Q = (b^2 - ac)e + ad^2 + (c^2 - 2bd)c$$
$$D = 27Q^2 - P^3 \qquad R = b^2 - ac$$
$$S = 12R^2 - a^2P \qquad T = 3aQ - 2PR$$
$$U = 2d^2 - 3ce$$

Aquests es relacionen amb les arrels de polinomis de quart grau segons la següent taula:

Cas $a \neq 0$	Nombre d'arrels reals diferents	Multiplicitats	Condicions
1	4	1-1-1-1	$D < 0,\ R > 0,\ S > 0$
2	3	1-1-2	$D = 0,\ T < 0$
3	2	1-1	$D > 0$
4	2	2-2	$D = T = 0,\ P \cdot R > 0$
5	2	1-3	$D = P = 0,\ R \neq 0$
6	1	4	$D = P = R = 0$
7	1	2	$D = 0,\ T > 0$
8	0	–	$D = T = 0,\ P \cdot R < 0$
9	0	–	$D < 0,\ R \leq 0$
10	0	–	$D < 0,\ S < 0$

Cas $a = 0$	Nombre d'arrels reals diferents	Multiplicitats	Condicions
1	3	1-1-1	$D < 0,\ R \neq 0$
2	2	1-1	$R = 0,\ P > 0,\ U > 0$
3	2	1-2	$D = 0,\ P \cdot R \neq 0$
4	1	3	$D = P = 0,\ R \neq 0$
5	1	2	$R = U = 0,\ P \neq 0$
6	1	1	$D > 0,\ R \neq 0$
7	1	1	$R = P = 0,\ U \neq 0$
8	0	–	$R = 0,\ P \neq 0,\ U < 0$
9	0	–	$R = P = U = 0,\ e \neq 0$
10	∞	–	$R = P = U = e = 0$

Com que el programa genera polinomis amb coeficients de forma aleatòria, segons una normal amb paràmetres (0,1), hi ha alguns casos que no cal contemplar (és a dir, que tenen probabilitat 0). També es pot veure que, en la taula, tenim informació redundant, de manera que es poden resumir les condicions i els càlculs dels discriminants per calcular només els imprescindibles.

Per tal de simplificar els càlculs, tenim en compte:

- El cas $a = c$ (per a un c donat) té probabilitat 0, per lo que no cal usar les condicions de classificació de la taula $a = 0$.
- En no contemplar el cas $a = 0$, no cal calcular el discriminant U mencionat anteriorment.
- Els casos 1, 2, 4, 5 i 6 donen 4 arrels reals. El 5 i 6 es poden fusionar en la condició $D = P = 0$.
- Els casos 3 i 7 donen 2 arrels reals i 2 complexes.
- Els casos 8, 9 i 10 donen 4 arrels complexes.

Tenint en compte aquests punts, el primer esborrany del classificador va ser:

```
if(disc[1]>0 || !disc[1]*disc[5]>0) goto both;
if(disc[1]<0){
    if (disc[2] > 0 && disc[4] > 0) goto real;
    if(disc[4]<=0 || disc[2]<0) goto complex;
    goto out;
}
if(!disc[1]*!disc[5]){
    if(disc[0]*disc[4]<0) goto complex;
    if(disc[0]*disc[4]>0) goto real;
}
if(!disc[1]*!disc[0] || !disc[1]*disc[5]<0) goto real;
```

On `disc[0]` $= P$, `disc[1]` $= D$, `disc[2]` $= S$, `disc[3]` $= Q$, `disc[4]` $= R$ i `disc[5]` $= T$.
Durant el procés d'obtenció del primer model de filtratge vam tenir problemes classificant funcions que tenien 4 arrels reals degut a una equivocació a l'hora de redactar les condicions (en comptes de posar `disc[2] > 0 && disc[4] > 0` vam posar `!disc[2] && disc[4] > 0`) pel que tots els càlculs desquadraben.
Al adonar-nos de l'error i corretgir-lo ens vam adonar que aquella condició de manera exclusiva classificaba tots el polinomis de grau quatre amb arrels reals.
Deduírem que hi havia casos amb probabilitat 0 que no haviem identificat com a innecesaris, pel que una vegada ja teniem poques condicions, vam determinar, tant a prova i error com amb relacions lògiques) quines de les condicions que teniem eren innecesàries.
L'output que obteniem amb la primera versió correcte del classificador era similar a:

```
Fraction of polynomials quadratic with 4 real roots: 0.245215
Fraction of polynomials quadratic with 4 complex roots: 0.085308
Fraction of polynomials quadratic with 2 real roots and 2 complex roots: 0.669477
```

Una vegada vam provar quines eren condicions necesàries (les que mantenien el resultat correcte) vam finalitzar obtenint la següent cadena de `if` per classificar:

```
if(discriminants[1] > 0) n_types[2]++;
else{
  if (discriminants[4] > 0 && discriminants[2] > 0) n_types[0]++;
  else n_types[1]++;
}
```

On `discriminants[1] = D`, `discriminants[2] = S` i `discriminants[4] = R`.
Per tant, acabem concluïnt que les úniques restriccions necessàries per classificar són:

- $D > 0$: 2 arrels reals i dos arrels complexes
- $D \leq 0$, $S > 0$ i $R > 0$: 4 arrels reals
- $D \leq 0$, $S \leq 0$ o $R \leq 0$: 4 arrels complexes

Degut a aquesta reducció de les condicions, només ha sigut necessari calcular els discriminants P, D, S, Q i R (estalviant-nos així el càlcul de la U i la T).
També comentar, que s'ha decidit posar aquesta distribució de les comparacions degut a que és més probable obtenir 2 arrels reals i 2 complexes que només 4 reals; i és més probable obtenir 4 reals que 4 complexes.

6 Comprovacions de funcionament

Per asegurar el correcte funcionament del programa, aplicant el que hem analitzat en l'apartat 5, adjuntem ara un seguit de taules amb tota la informació obtinguda (aplicant varies vegades el programa generant polinomis de manera aleatòria):

Usant la comanda `./arrels`

Generació	4 **arrels reals**	4 **arrels complexes**	2 **arrels reals i** 2 **complexes**
1	0.245181	0.085204	0.669615
2	0.244983	0.085351	0.669666
3	0.245044	0.085185	0.669771
4	0.245058	0.085133	0.669808
5	0.245034	0.085303	0.669663

Usant la comanda `./arrels 1` i definint com **Tipus 1** els polinomis amb 4 arrels reals, **Tipus 2** els polinomis amb 4 arrels complexes i **Tipus 3** els polinomis amb 2 arrels de cada (reals i complexes); obtenim la següent taula de generacions:

Gen.	**Tipus 1**		**Tipus 2**		**Tipus 3**		**Others**	
	#Creats	Fracció	#Creats	Fracció	#Creats	Fracció	#Creats	Fracció
1	4898321	0.244916	1705573	0.085279	13396106	0.669805	0	0.000000
2	4896770	0.244838	1704272	0.085214	13398958	0.669948	0	0.000000
3	4900629	0.245031	1702771	0.085139	13396600	0.669830	0	0.000000
4	4900973	0.245049	1703550	0.085178	13395477	0.669774	0	0.000000
5	4902916	0.245146	1704086	0.085204	13392998	0.669650	0	0.000000

Per tal de comprovar que el vostre programa funcioni correctament, podeu usar la comanda `./arrels 1 10 0` i comprovar que l'output que us surt és igual a:

```
Extra information has been requested.
Number of samples changed.
Values of the seed changed.
Calculating...
End of the calculations.

--------------------------------------------------------------------------------
Number of polynomial functions of degree 4 generated: 10

Number of which has 4 real root: 1
Fraction of polynomials quadratic with 4 real roots: 0.100000

Number of which has 4 complex root: 2
Fraction of polynomials quadratic with 4 complex roots: 0.200000

Number of which has 2 real roots and 2 complex roots: 7
Fraction of polynomials quadratic with 2 real roots and 2 complex roots: 0.700000
................................................................................
Number of other cases: 0
Fraction of other cases: 0.000000
--------------------------------------------------------------------------------
Time elapsed from the start to the end of the program: 0 ms
```

7 Conclusions

Finalitzem l'informe confirmant (a partir de les comprovacions de funcionament dutes a terme a l'apartat 6) que el programa *arrels.c* funciona de manera correcta, proporcionant les fraccions de polimonis quadràtics amb 4 arrels reals, 4 arrels complexes, i 2 reals 2 complexes.
Per tant, podem afirmar també que la funció **muller()** funciona correctament, ja que depen únicament del correcte funcionament de la funció **Normal()** a l'hora de generar els coeficients aleatoris.

7.1 Informació adicional

- Responem a la pregunta: **En les taules hi ha tots els casos possibles que es poden donar. Els necessiteu tots per la implementació? Per què? En cas negatiu, quins són supèrfluos?** Responem aquesta pregunta a l'apartat 5.
- Responem a la pregunta: **Què vol dir la fila 10 de la taula cas $a = 0$ quan diu que hi ha infinites arrels reals?**
  Si suposem $R = P = U = e = 0$ obtenim que:

  1. $R = P = U = e = 0$ ocurreix en el cas $a = 0$
  2. $U = 0 \Rightarrow 2 \cdot d^2 - 3c \cdot 0 \Rightarrow d = 0$
  3. $P = 0 \Rightarrow a \cdot 0 - 4b \cdot 0 + 3c^2 = 0 \Rightarrow c = 0$
  4. $R = 0 \Rightarrow b^2 - a \cdot 0 = 0 \Rightarrow b = 0$

  Per tant, observem que tots els coeficients són 0; es a dir, $\forall x \in \mathbb{C}, f(x) = 0$,
  essent $f(x) := ax^4 + 4bx^3 + 6cx^2 + 4dx + e \in \mathbb{R}_4[X]$ (hi ha infinites solucions).
- La complexitat dels mètodes són d'ordre lineal; és a dir, $O(n)$ (on n és el nombre de mostres que volem que generi el programa).
- Aconsellem llegir la Pràctica 2 per informació complementària sobre l'ús i les propietats del programa.