CÀLCUL NUMÈRIC

Memòria de la pràctica 1:

Fractals generats pel mètode de Newton

1.- Plantejament del problema i objectius:

Una empresa requereix un disseny d'un programa que pugui crear fractals del mètode de Newton. Han comunicat que estan interessats principalment en unes rutines de càlcul. Tot i això, també necessiten algún mètode per obtenir una representació gràfica dels fractals amb el programa gnuplot.

Una vegada plantegen el problema, cal analitzar què és el que demanen. A partir d'un polinomi amb n arrels (aquesta n ja estarà prèviament definida) i un nombre complex z_0, es faran iteracions seguint la fórmula d'iteracions de Newton fins que la diferència entre el valor y l'arrel del polinomi sigui menor que la tolerància. Llavors, aquell punt es pintarà amb el color associat a l'arrel corresponent.

Fórmula del mètode de Newton: $z_{n+1} = z_n - \dfrac{p(z_n)}{p'(z_n)}$

2.- Procés d'implementació del treball

El treball s'ha dividit en dues parts, creant a cadascuna un arxiu de tipus C diferents:

2.1- nwtfr.c

Aquest arxiu ha fet la funció de llibreria, en la que s'han creat tres rutines diferents amb l'objectiu de poder fer el procés de Newton només passant-li arguments.

La primera rutina anomenada **avalp** avalua el polinomi p(z) a partir dels complexos z i w.

$$z = x + iy$$

$$w_j = u_j + iv_j$$

$$p(z) = (z - w_0)(z - w_1)...(z - w_{n-1})$$

La segona rutina anomenada **avaldp** avalua la derivada del polinomi p(z) a partir del complex z (que segueix la misma fórmula que a avalp).

L'última rutina anomenada **cnvnwt,** la qual aplica el mètode de Newton a partir de les dues altres rutines fetes prèviament, aproxima les arrels de p(z) per tal de calcular les conques d'atracció de z_i.

Es considera que un punt pertany a la conca d'atracció de l'arrel w_j quan es compleix que:

$$z_n \xrightarrow{n\to\infty} w_j$$

2.2- dibfr.c

Aquest arxiu constitueix el programa principal de la pràctica, i es que genera l'arxiu fractal.txt, que a partir de les conques d'atracció recull tots els punts, que conformaran el fractal, junt amb els seus colors associats.

Aquest arxiu importa les funcions de l'altre fent servir un arxiu header anomenat "**nwtfr.h**" que fent un include del mateix a tots dos arxius

2.3- Altres

Cal esmentar que tant l'arxiu de la llibreria com el programa principal inclouen i fan servir llibreries predeterminades de C:

Els dos arxius comparteixen (junt a la prèviament esmentada "**nwtfr.h**"):
- stdio.h
- math.h

A més, el programa principal també importa:
- stdlib.h
- assert.h

3.- Procés d'implementació del software

A les dues primeres rutines, el codi es basava en el càlcul d'operacions entre nombres complexos, cosa que ha facilitat bastant el procés de dissenyar-les i programar-les. En aquest cas la operació era la multiplicació, i només cal fer servir que donat dos nombres complexos **z = x + yi** i **c = a + bi,** la multiplicació és **zc = (xa - yb) + (xb - ya)**.

D'altre banda, l'última rutina es fa a partir de la fórmula de la divisió de dos nombres complexos, on fent servir els mateixos nombres d'abans, quedaría com:

$$\frac{x+yi}{a+bi} = \frac{xa+yb}{a^2+b^2} + \frac{ya-xb}{a^2+b^2}$$

4.- Manual del software

4.1 - avalp

La funció avalp és del tipus void i per tant no retorna cap valor, només es fa servir per guardar al punt ***px** la part real de la funció p(z) i al punt ***py** la part imaginària.

Els arguments que rep la funció són:

- double: x (és la part real del nombre complex z)
- double: y (és la part imaginària del nombre complex z)
- double: *px (és la part real del polinomi p(z))
- double: *py (és la part imaginària del polinomi p(z))
- int: n (és el grau del polinomi p(z))
- double: u[ ] (vector que conté les parts reals de les solucions)
- double: v[ ] (vector que conté les parts imaginàries de les solucions)

Aquesta funció itera n vegades sobre els valors de les parts reals i imaginàries de les solucions i fent servir la fórmula del producte de nombres complexos. Una vegada ha acabat guarda els valors als seus respectius apuntadors.

4.2 - avaldp

La funció avaldp és del tipus void i per tant, a l'igual que l'anterior, no retorna cap valor i només es fa servir per guardar al punt ***pdx** la part real de la funció p'(z) i al punt ***pdy** la part imaginària.

Els arguments que rep la funció són:

- double: x (és la part real del nombre complex z)
- double: y (és la part imaginària del nombre complex z)
- double: *pdx (és la part real del polinomi p'(z))
- double: *pdy (és la part imaginària del polinomi p'(z))
- int: n (és el grau del polinomi p(z))
- double: u[ ] (vector que conté les parts reals de les solucions)
- double: v[ ] (vector que conté les parts imaginàries de les solucions)

Aquesta funció actúa d'una manera realment semblant a l'anterior. També itera n vegades sobre els valors de les parts reals i imaginàries de les solucions i fent servir la fórmula del producte de nombres complexos, però aquesta vegada ho fa dins d'una altre bucle. Una vegada ha acabat guarda els valors als seus respectius apuntadors.

4.3 - cnvnwt

La funció cnvnwt és del tipus int i per tant retorna un valor. La funció retornarà -1 si no es troba cap valor que s'aproximi a l'arrel amb un error menor que tolcnv o el valor de la iteració en la que sí que ho compleix.

Els arguments que rep la funció són:

- double: x (és la part real del nombre complex z)
- double: y (és la part imaginària del nombre complex z)
- double: tolcnv (és el valor màxim que pot adquirir la distància entre el punt i l'arrel)
- int: maxit (és el nombre màxim d'iteracions)
- int: n (és el grau del polinomi p(z))
- double: u[ ] (vector que conté les parts reals de les solucions)
- double: v[ ] (vector que conté les parts imaginàries de les solucions)

Mitjançant un bucle, itera maxit vegades, i fent servir les dues altres funcions i la fórmula del quocient de nombres complexos, obté el valor a comprovar amb la tolcnv.

4.4 - dibfr

Aquest fitxer és l'encarregat de generar el fitxer fractal.txt amb totes les dades per tal de generar posteriorment amb el programa "gnuplot" el fractal desitjat.

Per tal d'aconseguir-ho, cal compilar-lo sabent que, com està explicat prèviament, l'arxiu està enllaçat al fitxer nwtfr.c (que conté les rutines necessàries) mitjançant un header (nwtfr.h).

Degut a això, es fa servir la comanda:

```
$ gcc dibfr.c nwtfr.c -o dibfr - Wall - lm
```

Tot i això, també es pot fer servir un arxiu Makefile que el crea directament amb la comanda:

```
$ make dibfr
```

Una vegada compilat, l'executable rep els 9 següents arguments (en aquest mateix ordre):

- int: narr (és el grau del polinomi p(z))
- double: xmn (és el valor mínim que es pot representar al fractal en l'eix de les x)
- double: xmx (és el valor màxim que es pot representar al fractal en l'eix de les x)
- double: ymn (és el valor mínim que es pot representar al fractal en l'eix de les y)
- double: ymx (és el valor màxim que es pot representar al fractal en l'eix de les y)
- int: nx (és el nombre de punts que es representen horitzontalment)
- int: ny (és el nombre de punts que es representen verticalment)
- double:tolcnv (és el valor màxim que pot prendre la distància del punt amb l'arrel)
- int: maxit (és el nombre màxim d'iteracions que es fan al mètode de Newton)

Un exemple de com quedaria les línies de comanda sería:

```
$ ./dibfr 3=n -2=xmn 2=xmx 640=nx -2=ymn 2=ymx 480=ny 1e-3=tolcnv 50=maxit
> fractal . txt
```

Posteriorment, per tal de fer servir l'executable, només caldrà afegir les arrels amb els seus respectius colors. Llavors, les línies de comandes que surten a la terminal fent-ho des del començament són tal que així:

```
$ gcc dibfr . c nwtfr . c -o dibfr - Wall - lm
$./dibfr 3=n -2=xmn 2=xmx 640=nx -2=ymn 2=ymx 480=ny 1e-3=tolcnv 50=maxit >
fractal . txt
1 0 1 0 0
-0.5 .8660254037844386 0 1 0
-0.5 -.8660254037844386 0 0 1
```

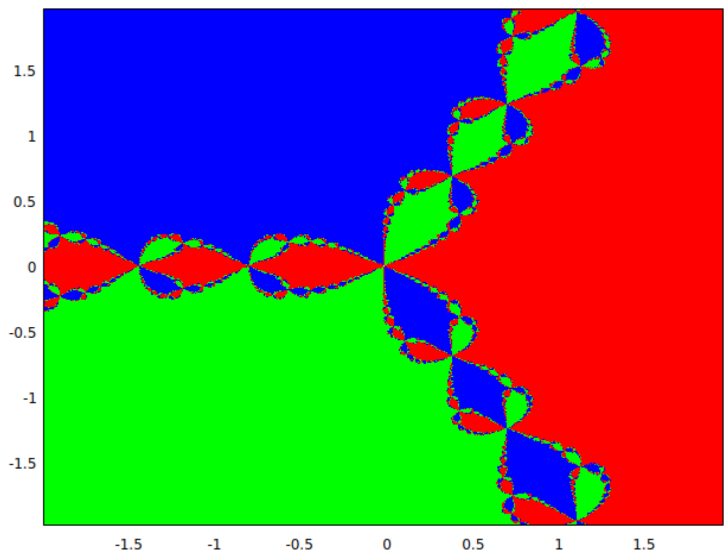
Una vegada generat l'arxiu fractal, només caldrà obrir el gnuplot i iniciar-lo:

```
$ gnuplot
unset key
gnuplot > plot “fractal.txt” w rgbimage
```

5 - Exemples gràfics

5.1 Exemple bàsic:

Fent servir els mateixos valors utilitzats a l'apartat 4.4 de la memòria, s'obté el següent fractal per representar la funció `p(z) = z³ - 1`.

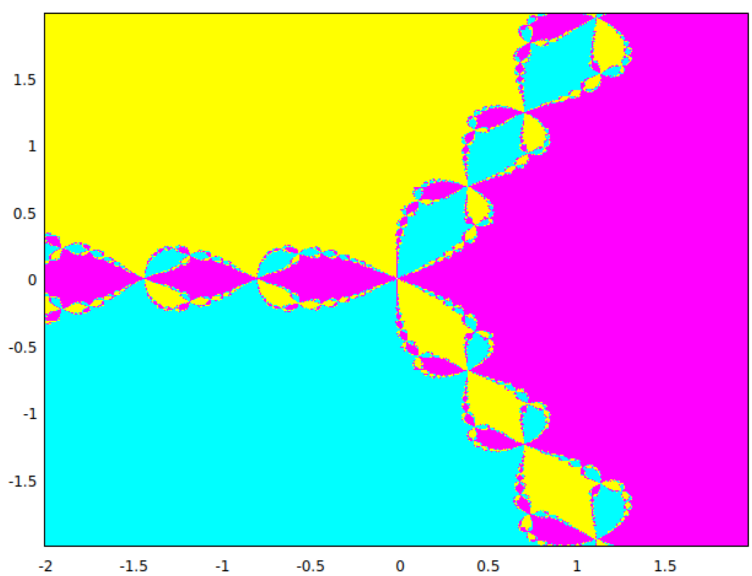


5.2 Canvi de colors:

Fent servir els mateixos valors que a l'exemple anterior pero canviant els valors dels colors pels següents:

- Primera arrel: 1 1 0
- Segona arrel: 1 0 1
- Tercera arrel: 0 1 1

S'obté el següent fractal:



5.3 Canvi de polinomi:

En aquest últim cas s'augmenta el nombre d'arrels de 3 a 5, i per tant s'avalua el polinomi $p(z) = z^5 - 1$.

```
$./dibfr 5=n -2=xmn 2=xmx 640=nx -2=ymn 2=ymx 480=ny 1e-3=tolcnv 50=maxit >
fractal . txt
-0.309017 -0.951057 0 0 1
0.809017 -0.587785 0 1 0
0.809017 0.587785 1 0 0
-0.309017 0.951057 1 0 1
-1 0 1 1 0
```

I d'aquí s'obté el següent fractal: