Pràctica 1 – Enunciat B
Computació Numèrica

David Moreno Borràs

2017-18 Q2

1 | Representació de nombres

Les dues expressions següents:

$$f(x) = 1.01e^{4x} - 4.62e^{3x} - 3.11e^{2x} + 12.2e^{x} - 1.99\,,$$

$$F(x) = (((1.01z - 4.62)z - 3.11)z + 12.2)z - 1.99\,, \quad z = e^{x}\,.$$

són duess fórmules diferents per a calcular la mateixa funció.

1. Cerca documentació sobre l'ús de la regla de Horner per avaluar polinomis. Escriu un breu resum del que has entès (màxim 1/2 full).

   Dóna les teves fonts bibliogràfiques.

2. Fent ús de l'aritmètica de **tres xifres** arrodonint calculeu el valor de les dues expressions per a $x = 1.53$. Per què donen diferent $f(1.53)$ i $F(1.53)$? **(No matlab)**

3. Fent ús de l'aritmètica de **quatre xifres** arrodonint calculeu el valor de les tres expressions per a $x = 0.925$. Per què donen diferent $f(0.925)$ i $F(0.925)$? **(No matlab)**

4. Calculeu en cada cas l'error relatiu percentual. Quina expressió dona una millor aproximació?

1) Per polinomis que són difícils d'avaluar, la regla de Horner ens permet avaluar-los convertint-los des d'una forma de grau n a una lineal/monomial.

En el cas d'aquest exercici, per exemple, tenim f(x) i F(x), que és la forma monomial obtinguda després d'aplicar la regla de Horner.

Tenir aquesta forma ens permet determinar si un valor és o no una arrel de l'expressió.

Fonts:
- https://goo.gl/bhu5y9
- https://goo.gl/YPfgkM

2) f(x) amb x = 1,53 i aritmètica de tres xifres arrodonint tenim:

$1.01e^{4x} = 459.4133$, $4.62e^{3x} = 455.04426$, $3.11e^{2x} = 66.3287$, $12.2e^{x} = 56.34175$

Per tant tenim 459 – 455 = 4; 4 – 66.3 = -62.3; -62.3 + 56.3 = -6; -6 – 1.99 = **-7.99 = f(x)**

En el cas de F(x) amb $z = e^{x}$ amb tres xifres tenim: 1.01*z = 4.66; 4.66 – 4.62 = 0.04; 0.04 * z = 0.18; 0.18 – 3.11 = -2.92; -2.92*z = -13.5; -13.5 + 12.2 = -1.31; -1.31 * z = -6.05;
- 6.05 – 1.99 = **-8.04 = F(x)**

Donen resultats diferents perquè al anar multiplicant en el cas de F(x) l'error es propaga més.

3) f(x) amb x = 0.925 i aritmètica de quatre xifres arrodonint tenim:

$1.01e^{4x}$ = 40.852, $4.62e^{3x}$ =74.098, $3.11e^{2x}$ = 19.779, $12.2e^{x}$ = 30.767

Per tant tenim 40.85 – 74.10 = -33.25; -33.25 – 19.78 = -53.03; -53.03 – 30.77 = -22.26; -22.26 – 1.99 = **-24.25 = f(x)**

En el cas de F(x) i z = 2.5219 tenim:

1.01*z = 2.547; 2.547 – 4.62 = -2.073; -2.073 * z = -5.227; -5.227 – 3.11 = -8.337; -8.337*z = -21.03; -21.03 + 12.2 = -8.826; -8.826 * z = -22.26; -22.26 – 1.99 = **-24.25 = F(x)**

En aquest cas els dos donen el mateix valor (arrodonint a quatre xifres, de fet, si fèiem els dos resultats amb Matlab la diferencia és als últims decimals que amb aquesta aritmètica de quatre xifres no es pot apreciar).

4) Amb 3 xifres:

errorRelf3 = 5.02281 amb f(x), **errorRelF3 = 5.64059** amb F(x). En aquest cas f(x) dona una millor aproximació perquè com hem vist abans l'error es propaga menys.

Amb 4 xifres com hem vist tenen el mateix resultat (ja que la diferència no s'aprecia als primers decimals) però l'error és diferent ja que el càlcul de f(x) i F(x) amb l'aritmètica de Matlab és diferent:

errorRelf4 = 0.004432133817226 amb f(x), **errorRelF4 = 0.004432133817358** amb F(x). En aquest cas mostro més decimals per veure la diferència, F(x) té un error relatiu una mica més gran que f(x), com en el primer cas.

I òbviament, com es pot apreciar, en els dos casos (f i F), treballar amb quatre xifres dona un resultat amb un error relatiu percentual molt menor que amb només tres xifres, on l'error es va propagant molt més per l'arrodoniment.

Annex

Codis Matlab Exercici 1: Errors relatius percentuals

```
%% Funcions
format long
f=@(x)1.01*exp(4*x)-4.62*exp(3*x)-3.11*exp(2*x)+12.2*exp(x)-1.99;
F=@(x)(((1.01*exp(x)-4.62)*exp(x)-3.11)*exp(x)+12.2)*exp(x)-1.99;

%% x = 1.53

fCorrecte3 = f(1.53);
FCorrecte3 = F(1.53);
fTresXifres = -7.99;
FTresXifres = -8.037;

errorRelf3 = (abs(fTresXifres-fCorrecte3)/abs(fCorrecte3))*100
errorRelF3 = (abs(FTresXifres-FCorrecte3)/abs(FCorrecte3))*100

%% x = 0.925

fCorrecte4 = f(0.925);
FCorrecte4 = F(0.925);

fQuatreXifres = -24.25;
FQuatreXifres = -24.25;

errorRelf4 = (abs(fQuatreXifres-fCorrecte4)/abs(fCorrecte4))*100
errorRelF4 = (abs(FQuatreXifres-FCorrecte4)/abs(FCorrecte4))*100
```

2 | Algoritmes

Considereu el següent algoritme per calcular el nombre π : *"Genereu n parelles de nombres aleatoris $\{(x_k, y_k)\}_{k=1\div n}$ de l'interval $[0, 1]$. Compteu el nombre m dels que es troben dins del primer quadrant del cercle unitat. Resulta que π és el límit de la successió $\pi_n = \frac{4m}{n}$."*

1. Construïu un programa en `Matlab` per calcular el termes la successió π_n.
2. Feu un joc de proves per a valors de $n = 7^k$, per exemple $1 \leq k \leq 12$. El resultat ha d'èsser una taula de la forma:

n	Valor π_n	Error abs.	Error rel.

3. A partir dels valors de la taula, l'exactitud creix o decreix en funció de n? Quants decimals iguals obteniu? Quantes xifres significatives obteniu? Els resultats del teu càlcul es corresponen amb el concepte *límit d'una successió*? Raona totes les teves respostes.

1) Codi de Matlab a l'annex.

Per aquest algorisme vaig tenir problemes al principi ja que generava totes les parelles de cop i l'ordinador no podia amb l'algorisme. Ara les genera seqüencialment

També vaig tenir problemes a l'hora de mostrar els valors amb prou exactitud ja que per mostrar les variables amb **disp** he necessitat fer servir num2str. Amb *num2str(nom_var,'%6f')* puc escollir com mostra les variables.

Per comprovar si el punt està dins del quadrant miro si $r = x^2+y^2$ és més petit o igual que 1.

Per generar els nombres aleatoris he usat r = a + (b-a).*rand(100,1); Això genera els nombres entre a i b però com en aquest cas a = 0 i b = 1 l'expressió queda així: x = rand(1,1);

Per fer el càlcul més eficient enlloc de generar 7^k parelles i fer la comprovació, generar $7^{(k+1)}$ parelles i fer la comprovació, etc el nou codi (**effPI.m**) les va generant fins el valor màxim i cada 7^k parelles fa la comprovació de m.

Les funcions per fer les proves es criden des de **tests.m**

2) Resultats joc de proves des de n = 7^1 a 7^{12}:

n	Valor pi	Error abs	Error rel
7	3,428571	0,286979	0,09134818
49	3,265306	0,123713	0,03937922
343	3,276968	0,135375	0,04309129
2401	3,173678	0,032085	0,01021297
16807	3,151306	0,009713	0,00309186
117649	3,139202	0,00239	0,0007609
823543	3,143802	0,002209	0,00070321
5764801	3,140391	0,001202	0,00038256
40353607	3,141751	0,000158	0,00005039
282475249	3,141815	0,000223	0,00007092
1977326743	3,141636	0,000043	0,00001371
13841287201	3,141567	0,000026	0,00000817

3) Com es pot veure a la taula l'exactitud creix en funció de n ja que cada vegada s'apropa més al valor real de pi (i els errors disminueixen).

El nombre de decimals correctes el sabem gràcies al valor de l'error absolut. En aquest cas, per l'últim valor aproximat de pi, 3.141567, l'error absolut es 0.000026, com és menor que $0.5x10^{-4}$ (0.00005), llavors 4 decimals són correctes.

Respecte les xifres significatives, com l'error relatiu es 0.00000817 i és més petit que $0.5x10^{-4}$ (0.00005) llavors té 4 xifres significatives.

Els resultats del càlcul es corresponen amb el concepte de *límit d'una successió* ja que com podem veure la successió tendeix al valor de pi conforme més gran és la n.

Annex

Codis Matlab Exercici 2: Pi

test.m des d'on es criden les funcions:

```
%% Pi efficient
% El for amb el codi eficient està dins de la funció, aquí li diem la k
màxima
    k = 5;
    effPi(k);

%% Golden Ratio
format long
phiCorrect = (1 + sqrt(5))/2;

% %% Using Fibonacci
for n = 5:1000
    Orfib(n);
end

% % %% Using continued fraction
for n = 5:1000
   Orfract(n);
end
```

effPi.m on està la funció per calcular pi:

```
%% Pi efficient
function [ pi_val ] = effPi(kMax)
k = 1;
m = 0;
nMax = 7^kMax;
    for i = 1:nMax
        x = rand(1,1);
        y = rand(1,1);

        r = x^2+y^2;
        % If x^2 + y^2 is less than or equal to 1,
        % then the point given by x,y is inside the quadrant.
        if (r <= 1)
                m = m + 1;
        end
        ex = 7^k;
        if (i == ex)
            pi_val = (4*m)/i;
            errAbs = abs(pi-pi_val);
            errRel = errAbs/pi;
            disp([num2str(i), '; ', num2str(pi_val,'%6f'), '; ',
num2str(errAbs,'%6f'), '; ', num2str(errRel,'%6f')]);
            k = k + 1;
        end
    end
end
```

3 | Expressions recurrents

Calcular valors aproximats del nombre iracional ϕ, conegut com a nombre d'or o proporció àuria, el valor del qual és $\phi = \dfrac{1+\sqrt{5}}{2}$.

Llegiu l'apartat 1.1 The Golden Ratio del llibre de Cleve Moler ([4]) fundador i principal promotor de Matlab.

En aquest apartat Moler proposa dos mètodes per a calcular ϕ:

Primer La fracció contínua; $\phi = \cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cfrac{1}{1+\cdots}}}}}$

Segon El límit del quocient de termes consecutius de la successió de Fibonacci;

$$\phi = \lim_{n\to\infty} \frac{F_n}{F_{n-1}},$$

on F_n es tal que $F_{-1} = F_0 = 1$ i la recurrència $F_{n+1} = F_n + F_{n-1}$, si $n \in \mathbb{N}$.

Es demana:

1. Cerca documentació sobre els conceptes *nombre d'or, fracció contínua, sucessió recurrent, successió de Fibonacci.* Escriu un breu resum del que has entès (màxim 1 full). Dóna les teves fonts bibliogràfiques.

2. Escriure dues funcions en Matlab, (`Orfract` i `Orfib`) per avaluar l'exactitud dels dos mètodes d'aproximació citats fent ús de n termes en la fracció contínua i n termes de la successió de Fibonnaci respectivament.

3. Feu un joc de proves per a valors de n, per exemple $1 \leq n \leq 1000$. Comenta els resultats obtinguts. El resultat ha d'èsser una taula de la forma

n	Valor Orfract	Error abs.	Error rel.	Valor Orfib	Error abs.	Error rel.

4. A partir dels valors de la taula, l'exactitud creix o decreix en funció de n? Quants decimals iguals obteniu? Quantes xifres significatives obteniu? Els resultats del teu càlcul es corresponen amb el concepte *límit d'una successió*? Raona totes les teves respostes.

1) El nombre d'or (Φ), també anomenat raó àuria, és un nombre molt especial ja que té una sèrie de característiques úniques i apareix a molts àmbits de les matemàtiques.

Per exemple una de les seves propietats deduïda a partir del rectangle d'or (un rectangle al que al eliminar-li un quadrat s'obté un rectangle amb la mateixa forma però més petit) és que el recíproc de Φ s'obté simplement restant 1.

$$\frac{1}{\phi} = \frac{\phi - 1}{1}$$

El valor del nombre d'or (nombre irracional, ja que la seva representació decimal no té cap període) es pot expressar així:

$$\phi = \frac{1 \pm \sqrt{5}}{2}$$

El nombre d'or no es va trobar directament com una expressió en l'antiguitat, sinó com una relació o proporció entre dos segments d'una recta.

Aquesta proporció és coneguda perquè apareix a molts llocs de la natura, com ara a la closca d'un cargol, als gira-sols, etc.

Una fracció infinita és una fracció de la forma:

$$a_0 + \cfrac{1}{a_1 + \cfrac{1}{a_2 + \cfrac{1}{a_3 + \cdots}}}. \qquad \phi = 1 + \cfrac{1}{1 + \cfrac{1}{1 + \cfrac{1}{1 + \cdots}}}.$$

Aquest tipus de fracció es relaciona amb el nombre d'or ja que si totes les ak són 1s, llavors tenim un altre representació del nombre d'or.

Una successió es recurrent si per calcular un terme específic es necessiten termes anteriors de la recurrència. Un exemple és la successió infinita de Fibonacci:

$$f_n = f_{n-1} + f_{n-2}.$$

0, 1, 1, 2, 3, 5, 8, 13, 21, 34... Els valors pels termes 1 i 0 d'aquesta successió estan fixats per poder calcular la recurrència correctament.

Aquesta successió també té moltes característiques especials, entre elles que el límit de la raó entre nombres de Fibonacci successius s'apropa a la raó àuria:

$$\lim_{n \to \infty} \frac{f_{n+1}}{f_n} = \phi.$$

Fonts:

- https://goo.gl/suQ8gD
- https://goo.gl/1kb6c6
- https://goo.gl/Z8KV2r

2) Funcions *Orfib.m* i *Orfrac.m* de Matlab a l'annex. Com amb el càlcul de pi, he necessitat *num2str(nom_var,'%10.17f')* per mostrar els valors amb exactitud.

Per Orfib també he creat una funció addicional *fibonnaci.m* que pren com a paràmetre una n i retorna dos valors consecutius de la successió de Fibonnaci, F_n i F_{n+1}.

Les funcions pels jocs de proves es criden, com amb el càlcul de pi, des de **tests.m**

3) Resultat joc de proves per valors d'n entre 1 i 1000:

n	Valor Orfract	Error abs.	Error rel.	Valor Orfib	Error abs.2	Error rel.3
5	1,625000000000000	0,006966	0,43052	1,600000000000000	0,018034	1,1146
6	1,615384615384610	0,0026494	0,16374	1,625000000000000	0,006966	0,43052
7	1,619047619047610	0,0010136	0,062646	1,615384615384610	0,0026494	0,16374
8	1,617647058823520	0,00038693	0,023914	1,619047619047610	0,0010136	0,062646
9	1,618181818181810	0,00014783	0,0091364	1,617647058823520	0,00038693	0,023914
10	1,617977528089880	5,65E-05	0,0034895	1,618181818181810	0,00014783	0,0091364
11	1,618055555555550	2,16E-05	0,0013329	1,617977528089880	5,65E-05	0,0034895
12	1,618025751072960	8,24E-06	0,00050912	1,618055555555550	2,16E-05	0,0013329
[...]	[...]	[...]	[...]	[...]	[...]	[...]
36	1,618033988749890	8,88E-16	5,49E-14	1,618033988749890	2,00E-15	1,24E-13
37	1,618033988749890	2,22E-16	1,37E-14	1,618033988749890	8,88E-16	5,49E-14
38	1,618033988749890	2,22E-16	1,37E-14	1,618033988749890	2,22E-16	1,37E-14
39	1,618033988749890	0	0	1,618033988749890	2,22E-16	1,37E-14
40	1,618033988749890	0	0	1,618033988749890	0	0
41	1,618033988749890	0	0	1,618033988749890	0	0
42	1,618033988749890	0	0	1,618033988749890	0	0
43	1,618033988749890	0	0	1,618033988749890	0	0
44	1,618033988749890	0	0	1,618033988749890	0	0
[...]	[...]	[...]	[...]	[...]	[...]	[...]
95	1,618033885370380	1,03E-07	6,39E-06	1,618033988749890	0	0
96	1,618034626172280	6,37E-07	3,94E-05	1,618033988749890	0	0
97	1,618033449641130	5,39E-07	3,33E-05	1,618033988749890	0	0
98	1,618033737889570	2,51E-07	1,55E-05	1,618033988749890	0	0
99	1,618034084570020	9,58E-08	5,92E-06	1,618033988749890	0	0
100	1,618034126624920	1,38E-07	8,52E-06	1,618033988749890	0	0
[...]	[...]	[...]	[...]	[...]	[...]	[...]
992	1,618033637728120	3,51E-07	2,17E-05	1,618033988749890	0	0
993	1,618034122828310	1,34E-07	8,29E-06	1,618033988749890	0	0
994	1,618033937536500	5,12E-08	3,17E-06	1,618033988749890	0	0
995	1,618033692938070	2,96E-07	1,83E-05	1,618033988749890	0	0
996	1,618034589019060	6,00E-07	3,71E-05	1,618033988749890	2,22E-16	1,37E-14
997	1,618033759467550	2,29E-07	1,42E-05	1,618033988749890	2,22E-16	1,37E-14

Valor de phi en format long per Matlab = 1.618033988749895
Com es pot veure a la taula en els dos casos s'arriba al valor aproximadament amb n = 40, com als exemples del llibre de Moler.

En els dos casos hi ha alguns canvis i sembla que dona problemes per a valors molt grans, probablement perquè es més difícil fer el càlcul i es va propagant l'error.

4) L'exactitud de l'aproximació creix en funció de n en el cas d'Orfib i Orfrac tot i que donen alguns problemes per valors molt grans.

En els dos casos el nombre de decimals correctes i xifres significatives és el mateix que el valor "original", el de Matlab. Aquest valor no és el correcte del nombre d'Or, però, ja que és un nombre periòdic i Matlab està limitat a una certa precisió.

Els resultats del càlcul es corresponen amb el concepte de *límit d'una successió* ja que com podem veure la successió tendeix al valor de phi conforme més gran és la n, tal com passava amb el càlcul de pi.

Annex

Codis Matlab Exercici 3: Phi

Funcions de http://www.mathworks.es/moler/chapters.html

Orfib.m:

```
%% Calcul nombre d'or
function [ phi ] = Orfib(n)
    phiCorrect = (1 + sqrt(5))/2;

    [fn1, fn] = fibonacci(n);
    phi = fn1/fn;

    errAbs = abs(phiCorrect-phi);
    errRel = errAbs/phiCorrect*100;

    disp([num2str(n), '; ', num2str(phi,'%15.18f'), '; ', num2str(errAbs), '; 
', num2str(errRel)]);
end
```

Orfract.m:

```
%% Calcul nombre d'or
function [ phi ] = Orfract(n)
    phi = '1';
    for k = 1:n
        phi = ['1+1/(' phi ')'];
    end

    phi = 1;
    q = 1;
    for k = 1:n
        s = phi;
        phi = phi + q;
        q = s;
    end
    phi = sprintf('%d/%d',phi,q);
    phi = eval(phi);

    phiCorrect = (1 + sqrt(5))/2;

    errAbs = abs(phiCorrect-phi);
    errRel = errAbs/phiCorrect*100;
    disp([num2str(n), '; ', num2str(phi,'%15.18f'), '; ', num2str(errAbs), '; 
', num2str(errRel)]);
end
```

4 | Solucions d'equacions no lineals

Calcular valors aproximats de l'arrel positiva de l'equació

$$(5-x)e^x = 5.$$

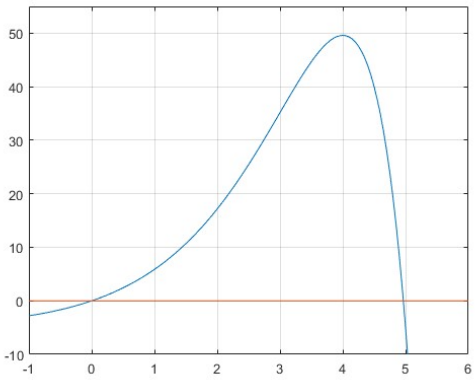
Es demana:

1. Quantes solucions diferents de $x = 0$ té l'equació $(5-x)e^x = 5$? Doneu intervals que separin les arrels. Justifica les teves respostes.
2. Calculeu la **arrel positiva no nul·la** (mínim 6 decimals correctes) per cadascun dels següents mètodes:
   (a) Mètode de la bisecció. Presenteu els resultats en una taula.
   (b) Mètode de la secant. Presenteu els resultats en una taula.
   (c) Mètode de Newton. Presenteu els resultats en una taula.
   Per cada mètode, doneu els punts inicials i el criteri d'aturada.
3. Considereu els mètodes iteratius següents:
   i) $x_{n+1} = 5 - \frac{5}{e^{x_n}}$,
   ii) $x_{n+1} = \ln\left(\frac{5}{5-x_n}\right)$,
   (a) Demostreu la convergència dels mètodes a l'arrel no nul·la de $(5-x)e^x = 5$ fent ús del teorema de convergència ()sense calcular les iteracions en `Matlab`). Doneu un interval que asseguri la convergència dels mètodes de la iteració simple. **("a priori")**
   (b) Obteniu el punt fix amb la mateixa tolerància prèvia. Doneu el punt inicial i el criteri d'aturada (fins a 6 decimals correctes). Presenteu els resultats en una taula.
4. Representeu en un gràfic **els logaritmes dels valors absoluts** dels errors relatius aproximats:
   $$r^{n+1} = \frac{x^{n+1} - x^n}{x^{n+1}}.$$
   Cada mètode un color diferent. A partir de les gràfiques realitzades, quin seria el millor procediment per obtenir la solució positiva de l'equació $(5-x)e^x = 5$. Raona les teves respostes.

1) Si mirem el plot de la funció **$f(x) = (5-x)e^x - 5$** (per valors d'x entre -1 i 6) veiem que té dues arrels, una a 0 i l'altre prop de 5. (plot a la següent pàgina)

Un interval per la primera pot ser [-1, 1], que per Teorema de Bolzano veiem que f(-1)*f(1) = -16.4 per tant hi ha una arrel entre -1 i 1

I pel cas de la segona arrel podem tenir l'interval [4,5], que veiem que f(4)*f(5) = -247.99, per tant hi ha una arrel entre 4 i 5.



Plot amb Matlab de f(x)

2) Als tres casos les funcions tenen un nou paràmetre *valorCorrecte* (del fzero de matlab) per calcular l'error absolut i parar l'algorisme una vegada arribi a 6 decimals correctes.

L'fzero de matlab diu que l'arrel és **4.965114231744276**

Amb **el mètode de la bisecció** començant amb l'interval [4,6] a l'iteració 18 obtenim el valor **4.965114593505859**, l'error absolut és 3.6176e-07, per tant té 6 decimals correctes.

Per motius d'espai la taula es mostra amb format short g:

iter	**a**	**x**	**b**	**f(x)**	**|b-a|**
0	4	5	6	-5	2
1	4	4.5	5	40.009	0.5
2	4.5	4.75	5	23.896	0.25
3	4.75	4.875	5	11.372	0.125
4	4.875	4.9375	5	3.7138	0.0625
5	4.9375	4.9688	5	-0.50478	0.03125
6	4.9375	4.9531	4.9688	1.6383	0.015625
7	4.9531	4.9609	4.9688	0.57529	0.0078125
8	4.9609	4.9648	4.9688	0.037404	0.0039063
9	4.9648	4.9668	4.9688	-0.23315	0.0019531
10	4.9648	4.9658	4.9668	-0.097739	0.00097656
11	4.9648	4.9653	4.9658	-0.030134	0.00048828
12	4.9648	4.9651	4.9653	0.0036435	0.00024414
13	4.9651	4.9652	4.9653	-0.013243	0.00012207
14	4.9651	4.9651	4.9652	-0.0047992	6.1035e-05
15	4.9651	4.9651	4.9651	-0.00057771	3.0518e-05
16	4.9651	4.9651	4.9651	0.0015329	1.5259e-05
17	4.9651	4.9651	4.9651	0.00047763	7.6294e-06
18	4.9651	4.9651	4.9651	-5.0041e-05	3.8147e-06

Amb el **mètode de la secant** començant amb l'interval [4,6] a l'iteració 11 ja obtenim el valor **4.965114045726**, l'error absolut és 1.8602e-07, per tant té 6 decimals correctes.

iter	a	c	x	f(x)	|b-a|
1	6	4	4.2166	48.117	1.7834
2	4.2166	6	4.4045	43.722	0.18796
3	4.4045	4.2166	6.2742	-681.28	1.8697
4	6.2742	4.4045	4.5173	39.21	1.757
5	4.5173	6.2742	4.6129	34.01	0.095616
6	4.6129	4.5173	5.2383	-49.878	0.62537
7	5.2383	4.6129	4.8664	12.343	0.37183
8	4.8664	5.2383	4.9402	3.359	0.073763
9	4.9402	4.8664	4.9678	-0.37044	0.027578
10	4.9678	4.9402	4.965	0.0094752	0.0027393
11	4.965	4.9678	4.9651	2.5731e-05	6.8318e-05

Finalment, amb el **mètode de Newton** he necessitat la regla de Fourier per determinar el punt inicial ja que em donava problemes:

Tenim $f'(x) = e^x(4-x)$, $f''(x) = e^x(3-x)$,

Regla de Fourier

1 Interval [a,b] = [4.5,5], f(4.5)*f(5) < 0

2 Si fem el plot veiem que f'(x)=0 a x=4, f''(x)=0 a x=3, per tant no a l'interval

3 a = 4.5: f(4.5)*f''(4.5) = -5.4022e+03

b = 5: f(5)*f''(5) = 1.4841e+03

Com amb b = 5: f(b)*f''(b) > 0, escollim x0 = 5.

Així amb el mètode de Newton tenim la següent taula (format long):

iter	x	f(x)	|b-a|
0	5.000000000000000	-5.000000000000000	5.000000000000000
1	4.966310265004573	-0.165642777610917	0.033689734995427
2	4.965115686301458	-0.000201201806099	0.001194578703114
3	4.965114231746430	-0.000000000297890	0.000001454555028

Com podem veure en aquest cas l'algorisme s'atura a l'iteració 3 ja que ja té 6 decimals correctes: **4.965114231746430** té un error absolut de 2.1538e-12.

3)
a) Pel primer cas, g1(x) és derivable, $g1'(x) = 5e^{-x}$ i per k = 5 que està a l'entorn de l'arrel tenim g1'(k) = 0.0336 < 1, per tant sí, és convergent a l'arrel.

Pel segon cas g2(x) també es derivable: g2'(x) = 1/(5-x) i per k = 4.9 tenim g2'(k) = 10 > 1 per tant és divergent.

Respecte l'interval que ens asseguri la convergència, podem trobar-lo amb una inequació: $5e^{-x} < 1$ [...] **x > 1.609.** Per tant podríem dir que un interval [2,5] assegura la convergència. (Al menys en el primer cas, el segon ja hem vist que és divergent i a més l'interval no podria ser 5 ja que anul·la el denominador).

b) La taula pel primer mètode iteratiu, amb punt inicial x0 = 5 i criteri d'aturada no superar el nombre màxim d'iteracions ni una tolerància tol = 0.0000005:

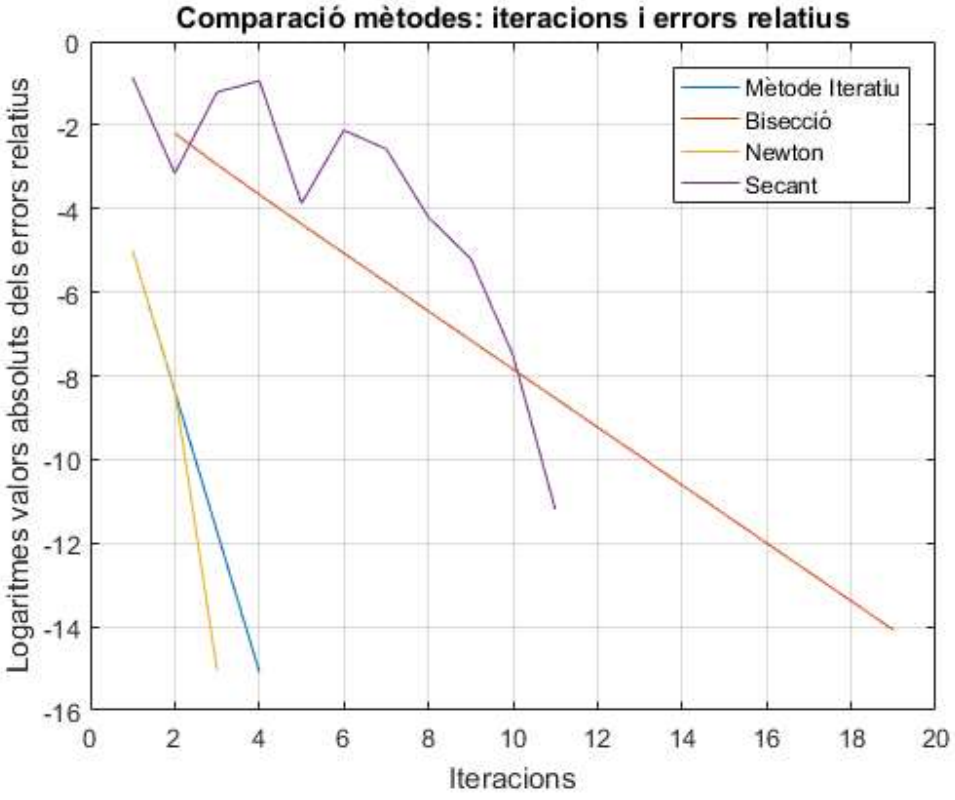
iter	**x**	**f(x)**	$\mathbf{x_n - x_{n-1}}$
0	5.000000000000000	-5.000000000000000	0
1	4.966310265004573	-0.165642777610917	-0.033689734995427
2	4.965155931341398	-0.005768338381813	-0.001154333663175
3	4.965115686436428	-0.000201220475782	-0.000040244904970
4	4.965114282492293	-0.000007019715721	-0.000001403944135

L'ultima iteració té un error absolut de 5.0748e-08, per tant té 6 decimals correctes (ja que 5.0748e-08 > 0.5e-06)

4) Per aquest apartat a cada mètode he afegit la següent línia:

```
logs = [logs, log(abs((x - xprev)/x))];
```

El plot de cada un dels mètodes és el següent:



Al plot de Matlab podem veure que el mètode de la bisecció té una convergència lineal però molt lenta. El de la secant, tot i que és millor que aquest, és una mica irregular i no tan ràpid com el mètode iteratiu (el primer, ja que el segon no convergeix).

I finalment, podem veure que, com hem pogut apreciar amb les taules anteriors, el millor mètode per obtenir la solució positiva de l'equació $\mathbf{(5-x)e^x = 5}$ és Newton, ja que és el que convergeix més ràpidament, tot i que hem de tenir en compte que és el que té un major cost computacional.

Annex

Codis Matlab Exercici 4: Solucions d'equacions no lineals

Main.m on es criden totes les funcions:

```
%% Plot
t=-1:0.05:6;
f=@(x)exp(x).*(5-x)-5;

plot(t,f(t),t,zeros(size(t))),axis([-1 6 -10 55]),grid

f(-1)*f(1)
f(4.5)*f(5)

%% FZero Matlab
% Arrel no nul·la
format long;
interval = [4,6];
valorCorrecte = fzero(f,interval)

%% Mètode de la bisecció
format long;
a = 4; b = 6; tol = eps; % 0.5*10^(-10)
[ root, taula, logsBi ] = new_bisec(f,a,b,tol,20,valorCorrecte)
plot(logsBi)
errAbs = abs(root-valorCorrecte)

%% Newton's method
df=@(x)exp(x).*(4-x);
df2=@(x)exp(x).*(3-x);

%plot(t,df(t),t,zeros(size(t))),axis([-1 6 -10 55]),grid
%plot(t,df2(t),t,zeros(size(t))),axis([-1 6 -10 55]),grid

% Regla de Fourier
% 1. f(4.5)*f(5) < 0 -> ok
% 2. df(x)=0 a x=4, df2(x)=0 a x=3
% 3. f(4.5)*df2(4.5) = -5.4022e+03
%    f(5)*df2(5) = 1.4841e+03
% Com amb = 5 f(b)*df2(b) > 0, x0 = 5.

x0 = 5; tol = eps;
[ root, taula, logsNew ] = new_new(f,df,x0,tol,10,valorCorrecte)
plot(logsNew)
errAbs = abs(root-valorCorrecte)

%% Mètode de la secant
a = 4; b = 6; tol = eps; % 0.5*10^(-10)
[ root, taula, logsSec ] = new_secant(f,a,b,tol,20,valorCorrecte)

errAbs = abs(root-valorCorrecte)

%% Mètode iteratiu #1

g1 = @(x)5-5/(exp(x));
dg1 = @(x)5*exp(-x);
```

```
x0 = 5;
tol = 0.00005;
N = 20;
if (abs(dg1(x0)) < 1)
    [ root, x_sol, logsIt1 ] = new_fixPoint(f,g1,x0,tol,N);
    errAbs = abs(root-valorCorrecte)
    plot(logsIt1)
else
    disp('Divergent!')
end

%% Mètode iteratiu #2

g1 = @(x)log(5/(5-x));
dg1 = @(x)1/(5-x);
x0 = 4.9;
tol = 0.0005;
N = 20;
if (abs(dg1(x0)) < 1)
    [ root, x_sol, logsIt2] = new_fixPoint(f,g1,x0,tol,N)
else
    disp('Divergent!')
end

%% 4
plot(logsIt1), hold on
plot(logsBi), hold on
plot(logsNew), hold on
plot(logsSec)
title('Comparació mètodes: iteracions i errors relatius'), grid
xlabel('Iteracions')
ylabel('Logaritmes valors absoluts dels errors relatius')
legend('Mètode Iteratiu','Bisecció', 'Newton','Secant')
```

new_secant.m:

```
function [ arrel , taula, logs ] = new_secant(f,a,b,tol,M, valorCorrecte)
%Mètode de la bisecció
%   f: function, a:dada, b:dada, opció f(a)*f(b)<0,
%   epsilon: cota error, M:cota iteracions
k = 0;
tolx=abs(b-a);
tolf=max(abs(f([a,b])));
era=max(tolx,tolf);
taula=[];
sisDecimals = 0.0000005;
errAbs = 1;
xprev = b
logs = [];
while (k<M && era>tol && sisDecimals < errAbs) %abs(b-a) > eps*abs(b)
    c = a;
    a = b;
    b = b + (b - c)/(f(c)/f(b)-1);
    k = k + 1;

    tolx=abs(b-a);
    tolf=max(abs(f(b)));
    era=max(tolx,tolf);
    taula=[taula;k,a,c,b,f(b),tolx];
    errAbs = abs(b-valorCorrecte);

    x = b
    xprev = a
    logs = [logs, log(abs((x - xprev)/x))];
end
arrel = b;
disp('            iter         a            c           x         f(x)
|b-a|')
format short g, taula, format
end
```

new_fixPoint.m:

```
function [ arrel, x_sol, logs ] = new_fixPoint(f, g, x, tol, N)
% f: equation
% g: iterative functon
% x = initial point
% tol: tol error
% N = number of iterations
    k = 0;
    tolx = 1;
    tolf=abs(f(x));
    err=max(tolx,tolf);
    x_sol = [k, x, f(x), 0]; %Per la taula de la pagina 16: Lab4-Tema2.pdf
    logs = [];
    while (err > tol && k < N)
       xprev = x;
       x = g(x);
       k = k + 1;
       tolx=abs(x - xprev);
       tolf=abs(f(x));
       err=max(tolx,tolf);
       x_sol = [x_sol; k, x, f(x), x-xprev];
       logs = [logs, log(abs((x - xprev)/x))];
    end
    arrel = x;
    disp('            iter           x           f(x)      xn - xn-1')
    format long, x_sol, format
end
```

new_new.m:

```
function [ arrel , taula, logs ] = new_new(f,df,x0,tol,M,valorCorrecte)
%Mètode de Newton o de la tangent
%   f: function, df:derivada funció, x0:dada,
%   tol: cota error, M:cota iteracions
k=0;
xprev = x0;
tolx=abs(xprev);
tolf=(abs(f(xprev)));
era=max(tolx,tolf);
taula=[k,xprev,f(xprev),tolx];
sisDecimals = 0.0000005;
errAbs = 1;
logs = [];
while (k < M && era > tol && sisDecimals < errAbs) %abs(b-a) > eps*abs(b)
    y = f(xprev); z=df(xprev); s=y/z;
    x = xprev-s;
    k = k + 1;
    tolx=abs(s);
    tolf=max(abs(f(x)));
    era=max(tolx,tolf);
    taula=[taula;k,x,f(x),tolx];
    errAbs = abs(x-valorCorrecte);
    logs = [logs, log(abs((x - xprev)/x))];
    xprev = x;
end
arrel = x;
disp('            iter           x           f(x)       |b-a|')
format long, taula, format
end
```

new_bisec.m:

```
function [ arrel , taula, logs ] = new_bisec(f,a,b,tol,M,valorCorrecte)
%Mètode de la bisecció
%   f: function, a:dada, b:dada, opció f(a)*f(b)<0,
%   epsilon: cota error, M:cota iteracions
k = 0;
tolx=abs(b-a);
tolf=max(abs(f([a,b])));
era=max(tolx,tolf);
taula=[];
sisDecimals = 0.0000005;
errAbs = 1;
xprev = (a + b)/2;
logs = [];
while (k<M && era>tol && sisDecimals < errAbs) %abs(b-a) > eps*abs(b)
    x = (a + b)/2;
    taula=[taula;k,a,x,b,f(x),tolx];
    if sign(f(x)) == sign(f(b))
        b = x;
    else
        a = x;
    end
    k = k + 1;
    tolx=abs(b-a)/2;
    tolf=max(abs(f(x)));
    era=max(tolx,tolf);
    errAbs = abs(x-valorCorrecte);
    logs = [logs, log(abs((x - xprev)/x))];
    xprev = x;
end
arrel = x;
disp('            iter      a      x      b      f(x)       |b-a|')
format short g, disp(taula), format
end
```

Tots aquests codis estan també a la carpeta **Codis**