Práctica 1: Introducción a la programación FORTRAN

Joaquim Lloberas Valls

Programación y Cálculo Matricial.
Curso Julio 2011

Métodos Numéricos para Cálculo y Diseño en Ingeniería

Índice

Resumen:

En este trabajo se pretende abordar la programación FORTRAN poniendo en relieve el trabajo con distintas variables FORTRAN.
También se analizará los posibles errores numéricos que pueden surgir en algoritmos de programación debido a la precisión de las variables y los errores acumulados en las operaciones que se realizan.
También se estudian los tipos de salidas y entradas que podemos realizar en FORTRAN y plataformas gráficas*
*Nota: las representaciones se han realizado utilizando la plataforma Matlab 6.5 con su correspondiente código anexado.

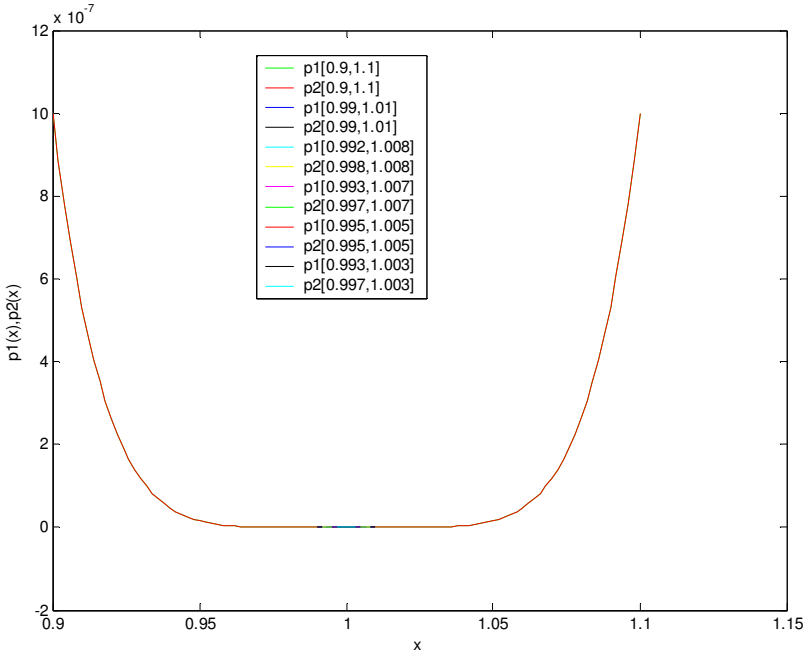
Práctica 1: Introducción a la programación FORTRAN

1) Desde el punto de vista algebraico es conocido que $p_6(x)=(x-1)^6= x^6-6\cdot x^5+15\cdot x^4-20\cdot x^3+15\cdot x^2-6\cdot x+1$.
Dibujar en una única gráfica las curvas correspondientes a las expresiones del polinomio anterior. En esta gráfica el eje de abscisas será el intervalo I=[1-δ,1+δ], siendo δ un valor conocido. Presentar las gráficas que se obtienen para δ=0.1,0.01,0.008,0.007,0.005,0.003, en un único dibujo y justificar los resultados obtenidos.

Planteamiento del problema:

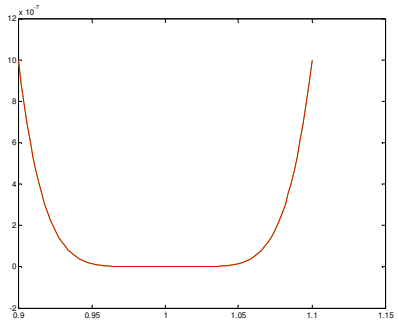
Para realizar este algoritmo debemos disponer de un vector para cada polinomio con las particiones necesitadas. En este caso 100. vectores (p1 y p2)
Además dispondremos de otro vector que nos dará el paso del intervalo (vector delta)
Realizaremos dos bucles iterativos: el más profundo nos calculará cada una de las imágenes que le pertocarán al polinomio p1 y p2. Mientras que el bucle superior nos dará el intervalo donde queremos estas imágenes. Consecuentemente habrá que calcular el paso de cada intervalo mediante el vector pas.

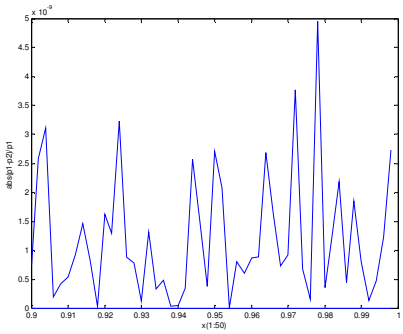
Presentación de resultados



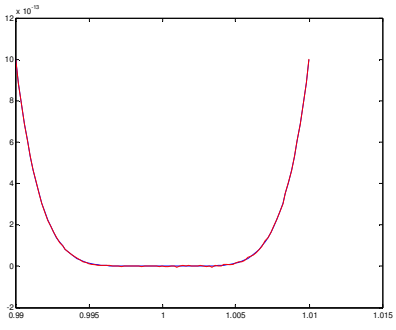
Para mayor facilidad de interpretación se representa cada uno de los intervalos con las dos formulaciones para el polinomio y los errores relativos entre ellos según cada paso.

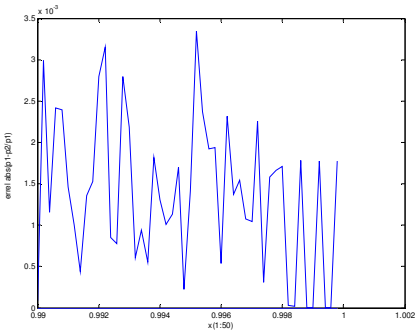
I=[0.9,1.1]



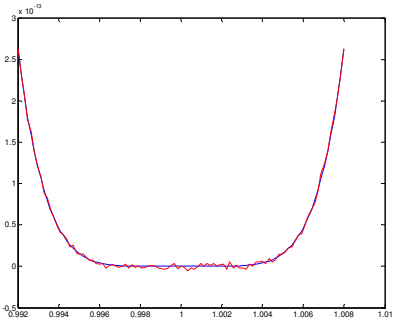


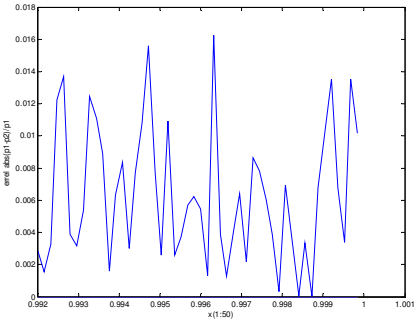
I=[0.99,1.01]



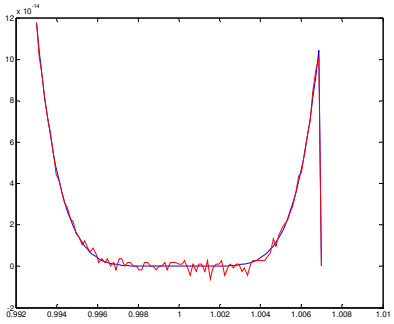


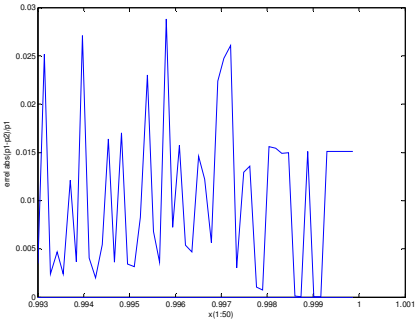
I=[0.992,1.008]



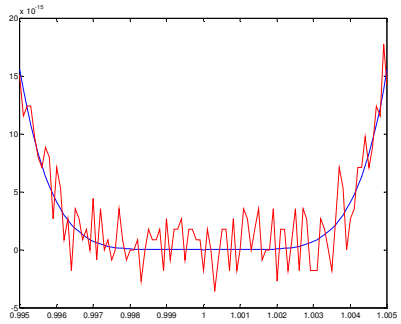


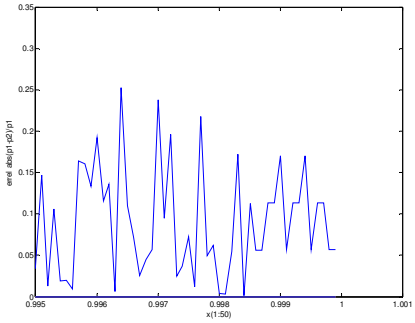
I=[0.993,1.007]



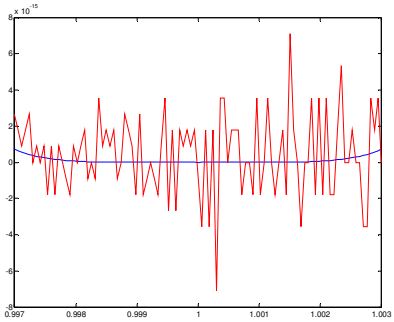


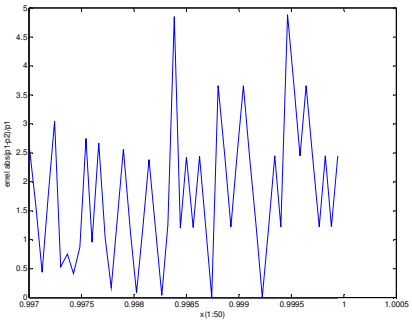
I=[0.995,1.005]





I=[0.997,1.003]





Conclusiones

Debido al número de operaciones realizadas para calcular el polinomio según la segunda expresión, se cometen muchos más errores de redondeo debido a la cantidad de operaciones realizadas. Al realizar un paso más pequeño, entran en juego más cifras decimales y consecuentemente se puede cometer un error mayor como se representa.

2) Escribir un programa FORTRAN que dado un cierto valor x tabule:

a) Los valores de la aproximación de la derivada de la función: f(x)=sin(x)

$$f'(x) \approx \frac{\sin(x+h)-\sin(x-h)}{2 \cdot h}$$

Variando h desde 1e-1 hasta 1e-18 dividiendo el valor de h entre 10 en cada paso
b) El valor de cos(x)
c) El error absoluto y el error relativo cometidos en la aproximación
Justificar razonadamente los resultados obtenidos considerando primero x=0 y después x=π
d) Reproducir las tablas anteriores para las aproximaciones a la derivada de la función f(x)=sin(x):

$$f'(x) \approx \frac{\sin(x+h)-\sin(x)}{h}$$

$$f'(x) \approx \frac{\sin(x)-\sin(x-h)}{h}$$

Justificar razonadamente los resultados obtenidos considerando primero x=π/4
e) Representar el logaritmo del error absoluto respecto al logaritmo de h para las tres aproximaciones anteriores. Cual es la pendiente de las curvas obtenidas? Qué indica el resultado obtenido?

Planteamiento del problema

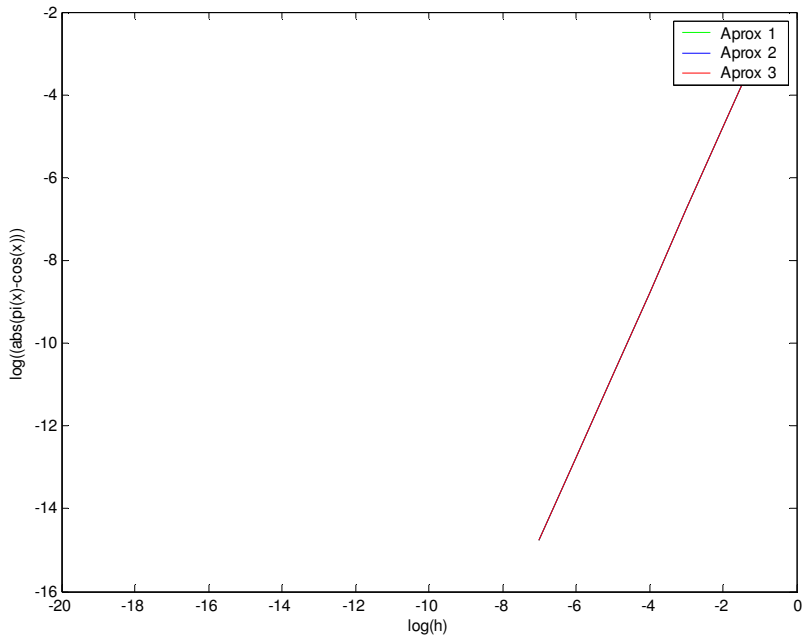
Mediante un algoritmo que tabule cada aproximación del coseno utilizando distintos incrementos podremos analizar los errores cometidos en cada algoritmo y la precisión de cada iteración.

Presentación de resultados

Para el valor x = 0

Inc.	Aprox 1	Error abs 1	Error relativo Aprox 1	Aprox 2	Error abs Aprox 2	Error relativo Aprox 2	Aprox 3	Error abs Aprox 3	Error realtivo Aprox 3
1e-1	0.99833416646828	0.00166583353172	0.00166583353172	0.99833416646828	0.00166583353172	0.00166583353172	0.99833416646828	0.00166583353172	0.00166583353172
1e-2	0.99998333341667	0.00001666658333	0.00001666658333	0.99998333341667	0.00001666658333	0.00001666658333	0.99998333341667	0.00001666658333	0.00001666658333
1e-3	0.99999983333334	0.00000016666666	0.00000016666666	0.99999983333334	0.00000016666666	0.00000016666666	0.99999983333334	0.00000016666666	0.00000016666666
1e-4	0.99999999833333	0.00000000166667	0.00000000166667	0.99999999833333	0.00000000166667	0.00000000166667	0.99999999833333	0.00000000166667	0.00000000166667
1e-5	0.99999999998333	0.00000000001667	0.00000000001667	0.99999999998333	0.00000000001667	0.00000000001667	0.99999999998333	0.00000000001667	0.00000000001667
1e-6	0.99999999999983	0.00000000000017	0.00000000000017	0.99999999999983	0.00000000000017	0.00000000000017	0.99999999999983	0.00000000000017	0.00000000000017
1e-7	1.00000000000000	0.00000000000000	0.00000000000000	1.00000000000000	0.00000000000000	0.00000000000000	1.00000000000000	0.00000000000000	0.00000000000000
1e-8	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-9	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-10	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-11	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-12	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-13	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-14	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-15	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-16	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-17	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0
1e-18	1.00000000000000	0	0	1.00000000000000	0	0	1.00000000000000	0	0

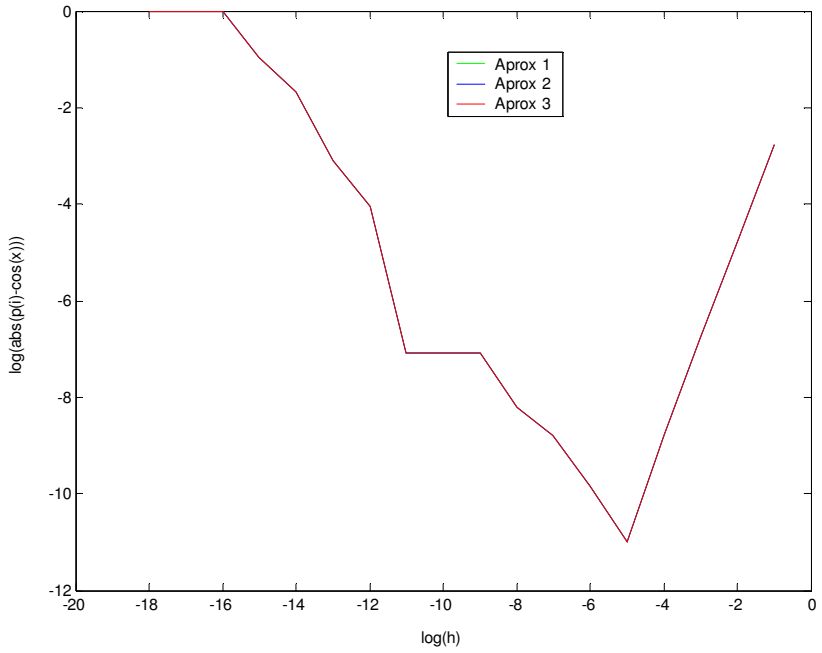
Representación gráfica del logaritmo del error absoluto obtenido frente al logaritmo de los incrementos para X=0



Para el valor x=π

Inc.	Aprox 1	Error abs 1	Error relativo Aprox 1	Aprox 2	Error abs Aprox 2	Error relativo Aprox 2	Aprox 3	Error abs Aprox 3	Error realtivo Aprox 3
1e-1	-0.99833416646828	0.00166583353172	-0.00166583353172	-0.99833416646828	0.00166583353172	-0.00166583353172	-0.99833416646828	0.00166583353172	-0.00166583353172
1e-2	-0.99998333341664	0.00001666658336	-0.00001666658336	-0.99998333341665	0.00001666658335	-0.00001666658335	-0.99998333341664	0.00001666658336	-0.00001666658336
1e-3	-0.99999983333323	0.00000016666677	-0.00000016666677	-0.99999983333323	0.00000016666677	-0.00000016666677	-0.99999983333323	0.00000016666677	-0.00000016666677
1e-4	-0.99999999833544	0.00000000166456	-0.00000000166456	-0.99999999833544	0.00000000166456	-0.00000000166456	-0.99999999833544	0.00000000166456	-0.00000000166456
1e-5	-0.99999999998988	0.00000000001012	-0.00000000001012	-0.99999999998988	0.00000000001012	-0.00000000001012	-0.99999999998988	0.00000000001012	-0.00000000001012
1e-6	-1.00000000013961	0.00000000013961	-0.00000000013961	-1.00000000013961	0.00000000013961	-0.00000000013961	-1.00000000013961	0.00000000013961	-0.00000000013961
1e-7	-0.99999999836342	0.00000000163658	-0.00000000163658	-0.99999999836342	0.00000000163658	-0.00000000163658	-0.99999999836342	0.00000000163658	-0.00000000163658
1e-8	-0.99999999392253	0.00000000607747	-0.00000000607747	-0.99999999392253	0.00000000607747	-0.00000000607747	-0.99999999392253	0.00000000607747	-0.00000000607747
1e-9	-1.00000008274037	0.00000008274037	-0.00000008274037	-1.00000008274037	0.00000008274037	-0.00000008274037	-1.00000008274037	0.00000008274037	-0.00000008274037
1e-10	-1.00000008274037	0.00000008274037	-0.00000008274037	-1.00000008274037	0.00000008274037	-0.00000008274037	-1.00000008274037	0.00000008274037	-0.00000008274037
1e-11	-1.00000008274037	0.00000008274037	-0.00000008274037	-1.00000008274037	0.00000008274037	-0.00000008274037	-1.00000008274037	0.00000008274037	-0.00000008274037
1e-12	-1.00008890058234	0.00008890058234	-0.00008890058234	-1.00008890058234	0.00008890058234	-0.00008890058234	-1.00008890058234	0.00008890058234	-0.00008890058234
1e-13	-0.99920072216264	0.00079927783736	-0.00079927783736	-0.99920072216264	0.00079927783736	-0.00079927783736	-0.99920072216264	0.00079927783736	-0.00079927783736
1e-14	-1.02140518265514	0.02140518265514	-0.02140518265514	-1.02140518265514	0.02140518265514	-0.02140518265514	-1.02140518265514	0.02140518265514	-0.02140518265514
1e-15	-0.88817841970012	0.11182158029988	-0.11182158029988	-0.88817841970012	0.11182158029988	-0.11182158029988	-0.88817841970012	0.11182158029988	-0.11182158029988
1e-16	0	1.00000000000000	-1.00000000000000	0	1.00000000000000	-1.00000000000000	0	1.00000000000000	-1.00000000000000
1e-17	0	1.00000000000000	-1.00000000000000	0	1.00000000000000	-1.00000000000000	0	1.00000000000000	-1.00000000000000
1e-18	0	1.00000000000000	-1.00000000000000	0	1.00000000000000	-1.00000000000000	0	1.00000000000000	-1.00000000000000

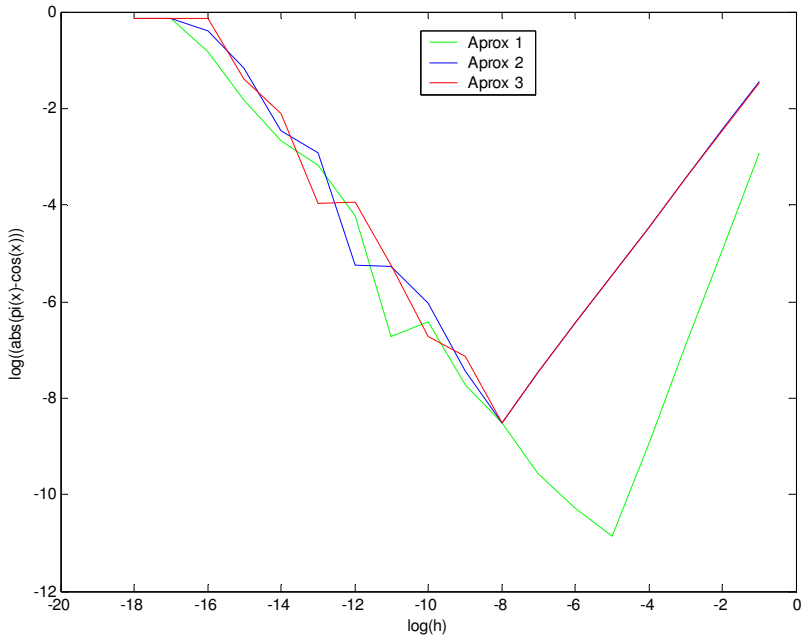
Para x=π errores obtenidos con las 3 aproximaciones



Para el valor x=π/4

Inc.	Aprox 1	Error abs 1	Error relativo Aprox 1	Aprox 2	Error abs Aprox 2	Error relativo Aprox 2	Aprox 3	Error abs Aprox 3	Error realtivo Aprox 3
1e-1	0.70592885899994	0.00117792218661	0.00166583353172	0.67060297290399	0.03650380828256	0.05162418075146	0.74125474509589	0.03414796390935	0.04829251368802
1e-2	0.70709499613245	0.00001178505410	0.00001666658334	0.70355949168921	0.00354728949734	0.00501662491680	0.71063050057569	0.00352371938915	0.00498329175013
1e-3	0.70710666333545	0.00000011785109	0.00000016666661	0.70675310997437	0.00035367121218	0.00050016662489	0.70746021669654	0.00035343550999	0.00049983329166
1e-4	0.70710678000796	0.00000000117859	0.00000000166677	0.70707142466930	0.00003535651724	0.00005000166621	0.70714213534662	0.00003535416007	0.00004999833266
1e-5	0.70710678117258	0.00000000001396	0.00000000001975	0.70710324564516	0.00000353554139	0.00000500001058	0.70711031670001	0.00000353551346	0.00000499997109
1e-6	0.70710678123920	0.00000000005265	0.00000000007446	0.70710642774419	0.00000035344236	0.00000049984298	0.70710713473421	0.00000035354766	0.00000049999190
1e-7	0.70710678090613	0.00000000028042	0.00000000039657	0.70710674648922	0.00000003469733	0.00000004906944	0.70710681532304	0.00000003413650	0.00000004827630
1e-8	0.70710678423680	0.00000000305025	0.00000000431371	0.70710678423680	0.00000000305025	0.00000000431371	0.70710678423680	0.00000000305025	0.00000000431371
1e-9	0.70710676203234	0.00000001915421	0.00000002708814	0.70710681754349	0.00000003635694	0.00000005141648	0.70710670652119	0.00000007466536	0.00000010559276
1e-10	0.70710715061040	0.00000036942385	0.00000052244422	0.70710770572191	0.00000092453536	0.00000130749045	0.70710659549888	0.00000018568766	0.00000026260201
1e-11	0.70710659549888	0.00000018568766	0.00000026260201	0.70711214661401	0.00000536542746	0.00000758786028	0.70710104438376	0.00000573680279	0.00000811306430
1e-12	0.70704553323253	0.00006124795402	0.00008661768724	0.70710104438376	0.00000573680279	0.00000811306430	0.70699002208130	0.00011675910525	0.00016512231017
1e-13	0.70776717819854	0.00066039701199	0.00093394241091	0.70832228971085	0.00121550852430	0.00171898864025	0.70721206668622	0.00010528549968	0.00014889618156
1e-14	0.70499162063697	0.00211516054957	0.00299128873580	0.71054273576010	0.00343595457355	0.00485917355762	0.69944050551385	0.00766627567270	0.01084175102922
1e-15	0.72164496600635	0.01453818481980	0.02056009814445	0.77715611723761	0.07004933605106	0.09906472107864	0.66613381477509	0.04097296641145	0.05794452478974
1e-16	0.55511151231258	0.15199526887397	0.21495377065811	1.11022302462516	0.40311624343861	0.57009245868377	0	0.70710678118655	1.00000000000000
1e-17	0	0.70710678118655	1.00000000000000	0	0.70710678118655	1.00000000000000	0	0.70710678118655	1.00000000000000
1e-18	0	0.70710678118655	1.00000000000000	0	0.70710678118655	1.00000000000000	0	0.70710678118655	1.00000000000000

Para x=π/4 errores obtenidos con las 3 aproximaciones



Conclusiones

Si nos fijamos en el valor x=0 y x=pi los tres algoritmos tienen el mismo comportamiento mientras que en el valor pi/4, el primer algoritmo alcanza una precisión más alta.

Las pendientes de las representaciones nos indican la variación absoluta del error en función de incremento escogido.

3.- El objetivo de este ejercicio es poner de manifiesto los efectos producidos por la propagación del error de redondeo. Para ello se propone calcular las sucesivas potencias del número áureo:

$$\varphi = \frac{\sqrt{5}-1}{2}$$

Mediante diferentes algoritmos, todos ellos teóricamente plausibles pero que presentan comportamientos dispares al implementarse en un ordenador. Los dos algoritmos propuestos son:

ALGORITMO 1: $x_0=1$; $x_1=\varphi$; $x_k=x_{k-1}*\varphi$

ALGORITMO 2: $x_0=1$; $x_1=\varphi$; $x_k=x_{k-2}- x_{k-1}$

Se pide:

Realizar un programa que calcule las sucesivas potencias del número áureo mediante los dos algoritmos anteriores utilizando exclusivamente variables REAL*4. Este programa debe cumplir los siguientes puntos:

a) el programa calculará las sucesivas potencias del número áureo para cada uno de los anteriores algoritmos
b) El programa presentará en una tabla las cien primeras potencias del número áureo

Repetir el programa anterior usando variables REAL*8

Justificar los resultados obtenidos

Nota: A fin de resaltar las diferencias existentes entre trabajar en simple y doble precisión se debe operar siempre con constantes, variables y funciones del mismo tipo. Por ejemplo si se trabaja con variables REAL*4 se debería utilizar:

φ=(sqrt(5.0)-1.0)/2.0

Mientras que en variables REAL*8

φ=(dsqrt(5.DO)-1.DO)/2.DO

Planteamiento y resultados

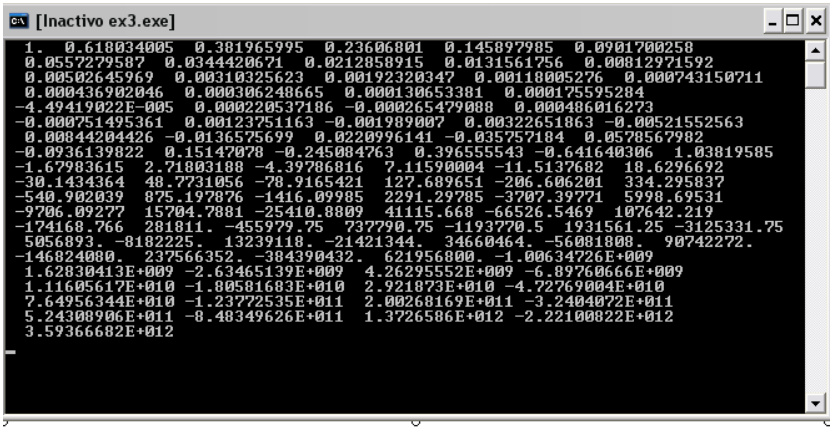
En este ejercicio podremos analizar la convergencia del número áureo con dos algoritmos de programación. Veremos también el uso de variables REAL*4 y REAL*8 como difieren los resultados

Presentación de resultados

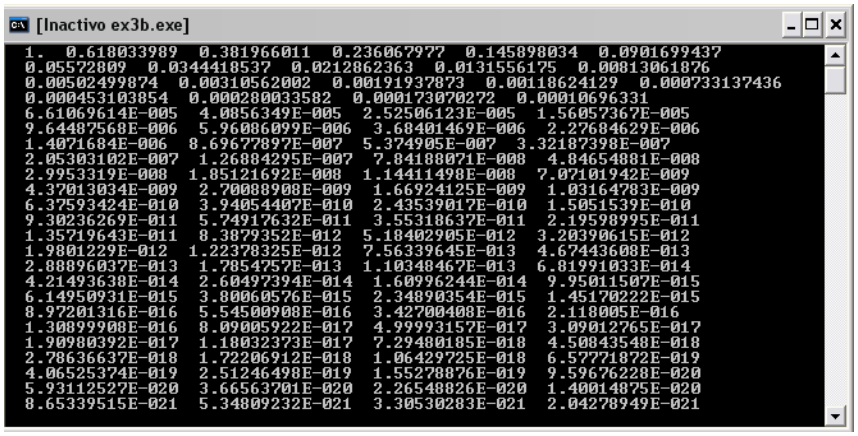
Tabla con precisión real*4 para los dos algoritmos de computación

REAL*4. Algoritmo 1

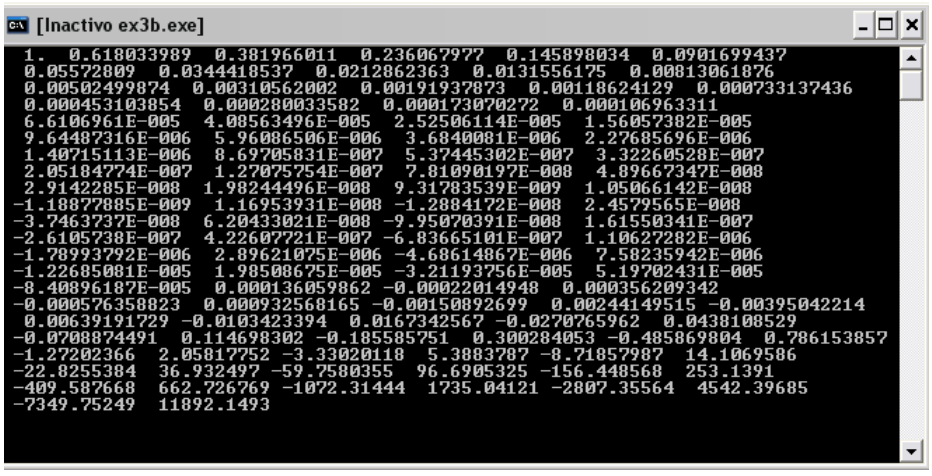
REAL*4 Algoritmo 2



REAL*8 Algoritmo 1



REAL*8 Algoritmo 2



Salidas DE MATLAB 6.5 con doble precisión

	Algoritmo 1	Algoritmo 2
1	1.000000000000000e+000	1.000000000000000e+000
2	6.180339887498949e-001	6.180339887498949e-001
3	3.819660112501052e-001	3.819660112501051e-001
4	2.360679774997898e-001	2.360679774997898e-001
5	1.458980337503155e-001	1.458980337503153e-001
6	9.016994374947429e-002	9.016994374947451e-002
7	5.572809000084125e-002	5.572809000084078e-002
8	3.444185374863305e-002	3.444185374863373e-002
9	2.128623625220821e-002	2.128623625220705e-002
10	1.315561749642485e-002	1.315561749642669e-002
11	8.130618755783357e-003	8.130618755780361e-003
12	5.024998740641495e-003	5.024998740646325e-003
13	3.105620015141862e-003	3.105620015134036e-003
14	1.919378725499634e-003	1.919378725512289e-003
15	1.186241289642229e-003	1.186241289621748e-003
16	7.331374358574060e-004	7.331374358905407e-004
17	4.531038537848228e-004	4.531038537312071e-004
18	2.800335820725832e-004	2.800335821593336e-004
19	1.730702717122397e-004	1.730702715718735e-004
20	1.069633103603436e-004	1.069633105874601e-004
21	6.610696135189610e-005	6.610696098441338e-005
22	4.085634900844749e-005	4.085634960304674e-005
23	2.525061234344862e-005	2.525061138136664e-005
24	1.560573666499888e-005	1.560573822168010e-005
25	9.644875678449740e-006	9.644873159686540e-006
26	5.960860986549141e-006	5.960865061993559e-006
27	3.684014691900600e-006	3.684008097692981e-006
28	2.276846294648543e-006	2.276856964300578e-006
29	1.407168397252057e-006	1.407151133392404e-006
30	8.696778973964855e-007	8.697058309081740e-007
31	5.374904998555718e-007	5.374453024842296e-007
32	3.321873975409138e-007	3.322605284239444e-007
33	2.053031023146580e-007	2.051847740602852e-007
34	1.268842952262559e-007	1.270757543636591e-007
35	7.841880708840217e-008	7.810901969662609e-008
36	4.846548813785372e-008	4.896673466703305e-008
37	2.995331895054845e-008	2.914228502959304e-008
38	1.851216918730528e-008	1.982444963744001e-008
39	1.144114976324318e-008	9.317835392153029e-009
40	7.071019424062098e-009	1.050661424528698e-008
41	4.370130339181083e-009	-1.188778853133954e-009
42	2.700889084881016e-009	1.169539309842094e-008
43	1.669241254300068e-009	-1.288417195155489e-008
44	1.031647830580948e-009	2.457956504997583e-008
45	6.375934237191192e-010	-3.746373700153072e-008
46	3.940544068618291e-010	6.204330205150654e-008
47	2.435390168572902e-010	-9.950703905303726e-008
48	1.505153900045390e-010	1.615503411045438e-007
49	9.302362685275129e-011	-2.610573801575811e-007
50	5.749176315178771e-011	4.226077212621249e-007
51	3.553186370096359e-011	-6.836651014197059e-007
52	2.195989945082413e-011	1.106272822681831e-006
53	1.357196425013946e-011	-1.789937924101537e-006
54	8.387935200684670e-012	2.896210746783368e-006
55	5.184029049454796e-012	-4.686148670884904e-006
56	3.203906151229874e-012	7.582359417668272e-006
57	1.980122898224923e-012	-1.226850808855318e-005
58	1.223783253004951e-012	1.985086750622145e-005
59	7.563396452199718e-013	-3.211937559477462e-005
60	4.674436077849796e-013	5.197024310099607e-005
61	2.888960374349924e-013	-8.408961869577070e-005
62	1.785475703499873e-013	1.360598617967668e-004
63	1.103484670850051e-013	-2.201494804925375e-004
64	6.819910326498218e-014	3.562093422893042e-004
65	4.214936382002292e-014	-5.763588227818417e-004
66	2.604973944495928e-014	9.325681650711459e-004
67	1.609962437506365e-014	-1.508926987852988e-003
68	9.950115069895624e-015	2.441495152924134e-003
69	6.149509305168032e-015	-3.950422140777121e-003
70	3.800605764727593e-015	6.391917293701255e-003
71	2.348903540440439e-015	-1.034233943447838e-002
72	1.451702224287155e-015	1.673425672817963e-002
73	8.972013161532847e-016	-2.707659616265801e-002
74	5.545009081338700e-016	4.381085289083764e-002
75	3.427004080194147e-016	-7.088744905349564e-002
76	2.118005001144554e-016	1.146983019443333e-001
77	1.308999079049594e-016	-1.855857509978289e-001
78	8.090059220949597e-017	3.002840529421622e-001
79	4.999931569546347e-017	-4.858698039399911e-001
80	3.090127651403251e-017	7.861538568821533e-001
81	1.909803918143096e-017	-1.272023660822145e+000
82	1.180323733260155e-017	2.058177517704298e+000
83	7.294801848829409e-018	-3.330201178526442e+000
84	4.508435483772148e-018	5.388378696230740e+000
85	2.786366365057262e-018	-8.718579874757182e+000
86	1.722069118714886e-018	1.410695857098792e+001
87	1.064297246342377e-018	-2.282553844574510e+001
88	6.577718723725088e-019	3.693249701673302e+001
89	4.065253739698684e-019	-5.975803546247813e+001
90	2.512464984026405e-019	9.669053247921116e+001
91	1.552788755672280e-019	-1.564485679416893e+002
92	9.596762283541252e-020	2.531391004209004e+002
93	5.931125273181550e-020	-4.095876683625897e+002
94	3.665637010359703e-020	6.627267687834901e+002
95	2.265488262821847e-020	-1.072314437146080e+003
96	1.400148747537857e-020	1.735041205929570e+003
97	8.653395152839911e-021	-2.807355643075650e+003
98	5.348092322538656e-021	4.542396849005219e+003
99	3.305302830301255e-021	-7.349752492080869e+003
100	2.042789492237402e-021	1.189214934108609e+004

Conclusiones:

El primer algoritmo es el más fiable debido a que no contiene operaciones de sustracción sucesivas que hacen cancelar residuos por redondeo. Es por ello que llega una iteración en algoritmo 2 (39 aprox. en real*8 y 17 aprox. en real*4) donde los errores de redondeo de la máquina se disparan; claro está, que utilizando variables de mayor precisión es preciso realizar más iteraciones para llegar a esta cancelación.

4.- Un rollo de papiro escrito hacia 1650 A.C. contiene la evidencia de que los antiguos egipcios ya tenían plena conciencia de que el cociente entre la longitud de la circunferencia y su diámetro es un número constante (π) que llegaron a estimar en π=3.16045. Los babilónicos también obtuvieron estimaciones groseras (π=3.125) al igual que sus contemporáneos chinos (π=3). Algunos autores sostienen que incluso un pasaje del Antiguo testamento (concretamente del libro de los reyes) indica la aproximación π=3. Pero, sin lugar a dudas, la mayor contribución de la antigüedad en referencia al tema se debe a los griegos y es en el año 250 A.C. por el chino Liu Hui (π=3.14159), el mérito de Arquímedes radica en que además sugiere un procedimiento iterativo sencillo para el cálculo de este número trascendente.
En realidad, el mérito de Arquímedes es puramente geométrico y consiste en aproximar una circunferencia de diámetro unidad (es decir, de longitud π) por sucesiones de polígonos regulares inscritos y circunscritos. Como se desprende de la figura adjunta, el perímetro de un polígono regular inscrito proporciona una cota inferior de π, mientras que el perímetro de un polígono regular circunscrito establece una cota superior. De este modo, partiendo de un hexágono circunscrito, doblando sucesivamente el número de lados hasta llegar a un polígono de 96 lados y repitiendo el proceso para un hexágono inscrito, Arquímedes fue capaz de demostrar que 3+10/71<π<3+1/7. Una ligera modificación respecto al método original de Arquímedes consiste en partir de un cuadrado inscrito e ir doblando sucesivamente el número de lados. De esta forma, si p_n es el perímetro del polígono inscrito de 2^n lados, se obtiene la fórmula recursiva:

$$p_2 = 2\cdot\sqrt{2}$$

$$p_{n+1} = 2^n \cdot \sqrt{2\cdot\left(1-\sqrt{1-\left(\frac{p_n}{2^n}\right)^2}\right)}$$

que teóricamente satisface la propiedad lim p_n=π
Sin embargo en la práctica, la implementación de la sucesión definida por las expresiones (1) y (2) reserva algunas sorpresas.
Se pide:

a) Calcular el perímetro p_n del polígono regular de 2^n lados inscrito en la circunferencia según la expresión (2) hasta n=30 así como el error relativo cometido en la estimación de π es decir: $E^n=(\pi-p_n)/\pi$

Representar gráficamente la evolución del error relativo E^n en función de n

Justificar el fenómeno observado (conocido habitualmente como cancelación catastrófica) en base a la propagación de los errores.

b) Una medida para evitar la cancelación catastrófica es rescribir la fórmula recursiva (2) en una forma más apropiada. Para ello se empieza por denotar

$$p_{n+1} = 2^n \cdot \sqrt{r_{n+1}}$$ Para n>=2

Donde obviamente

$$r_{n+1} = 2\cdot\left(1-\sqrt{1-\left(\frac{p_n}{2^n}\right)^2}\right)$$ Para n>=2

y tras algunas manipulaciones algebraicas se demuestra fácilmente que la sucesión r_n viene dada por

$$r_3 = 2 - \sqrt{2}$$

$$r_{n+1} = \frac{r_n}{2 + \sqrt{4 - r_n}}$$ Para n>=3

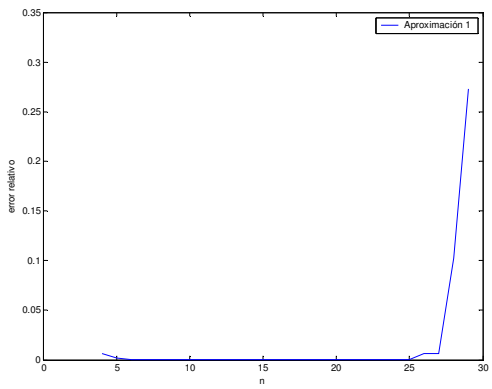
Calcular nuevamente el perímetro p_n del polígono regular inscrito de 2^n lados, así como el error cometido en la estimación de π, utilizando la fórmula recursiva anterior.
Representar gráficamente la evolución del error relativo e^n en función de n y comprobar que efectivamente el efecto indeseable de la cancelación catastrófica queda eliminado.

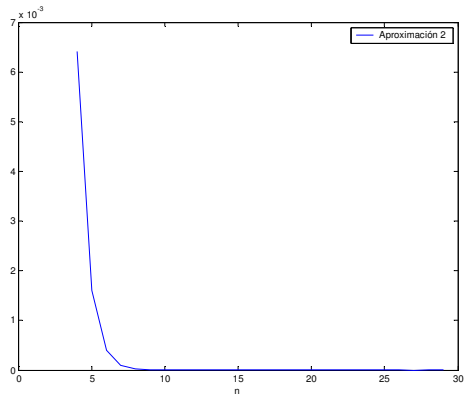
Planteamiento y resolución

Se van a realizar un bucle recursivo para obtener cada una de las aproximaciones del número pi. Primero mediante la expresión al completo y la segunda mediante la división de la expresión con variable r. Se utilizarán variables REAL*8.

It.	Apartado A	Error relativo A	Apartado B	Error relativo B
1	0	0	0	0
2	2.828427124746190e+000	0	0	0
3	3.061467458920719e+000	2.550464159556715e-002	3.061467458920718e+000	2.550464159556743e-002
4	3.121445152258053e+000	6.413148855794013e-003	3.121445152258052e+000	6.413148855794295e-003
5	3.136548490545941e+000	1.605606964381162e-003	3.136548490545939e+000	1.605606964381728e-003
6	3.140331156954739e+000	4.015468503252505e-004	3.140331156954753e+000	4.015468503210097e-004
7	3.141277250932757e+000	1.003957838632654e-004	3.141277250932773e+000	1.003957838581766e-004
8	3.141513801144146e+000	2.509951299941639e-005	3.141513801144301e+000	2.509951294994109e-005
9	3.141572940367883e+000	6.274913422618814e-006	3.141572940367091e+000	6.274913674660102e-006
10	3.141587725279961e+000	1.568729741775773e-006	3.141587725277159e+000	1.568730633603306e-006
11	3.141591421504635e+000	3.921848863793808e-007	3.141591421511199e+000	3.921827969669921e-007
12	3.141592345611077e+000	9.803267013483490e-008	3.141592345570118e+000	9.804570786458515e-008
13	3.141592576545004e+000	2.452411794475727e-008	3.141592576584872e+000	2.451142767293622e-008
14	3.141592633463248e+000	6.406478208483288e-009	3.141592634338563e+000	6.127856953573550e-009
15	3.141592654807589e+000	3.876365335917220e-010	3.141592648776985e+000	1.531964344411877e-009
16	3.141592645321215e+000	2.631970065560944e-009	3.141592652386591e+000	3.829912274609551e-010
17	3.141592607375720e+000	1.471039646217161e-008	3.141592653288992e+000	9.574784220473524e-011
18	3.141592910939673e+000	8.191701089404810e-008	3.141592653514593e+000	2.393699589068027e-011
19	3.141594125195191e+000	4.684265467399403e-007	3.141592653570993e+000	5.984390330655911e-012
20	3.141596553704820e+000	1.241445170185552e-006	3.141592653585093e+000	1.496132922160438e-012
21	3.141596553704820e+000	1.241445170185552e-006	3.141592653588618e+000	3.740332305401096e-013
22	3.141674265021758e+000	2.597772561988120e-005	3.141592653589500e+000	9.343762864210598e-014
23	3.141829681889202e+000	7.544845100703004e-005	3.141592653589720e+000	2.332406766406579e-014
24	3.142451272494134e+000	2.733068857159364e-004	3.141592653589775e+000	5.795677419555742e-015
25	3.142451272494134e+000	2.733068857159364e-004	3.141592653589789e+000	1.272221872585407e-015
26	3.162277660168380e+000	6.584242089740800e-003	3.141592653589792e+000	2.827159716856459e-016
27	3.162277660168380e+000	6.584242089740800e-003	3.141592653589793e+000	0
28	3.464101615137754e+000	1.026577908435841e-001	3.141592653589794e+000	1.413579858428230e-016
29	4.000000000000000e+000	2.732395447351627e-001	3.141592653589794e+000	1.413579858428230e-016
30	0	1.000000000000000e+000	3.141592653589794e+000	1.413579858428230e-016

Presentación de resultados





Conclusiones:

Como es evidente el segundo algoritmo converge totalmente y no difiere a cierta iteración como el primer algoritmo. La cancelación catastrófica queda eliminada.

ANEXOS

Ejercicio 1

Programa FORTRAN 77

```
integer j,i
REAL x
DIMENSION p1(100,6),p2(100,6),delta(6),pas(6)
delta(1)=0.1
delta(2)=0.01
delta(3)=0.008
delta(4)=0.007
delta(5)=0.005
delta(6)=0.003
i=1
x=1-delta(i)
for n=1:1:6
pas(n)=2*delta(n)/100;
end

DO while (i.LE.6)
j=1
x=1-delta(i)
DO while (x.LE.1+delta(i))
p1(j,i)=(x-1)**6
p2(j,i)=x**6-6*x**5+15*x**4-20*x**3+15*x**2-6*x+1
x=x+pas(i)
j=j+1
ENDDO
i=i+1
ENDDO
WRITE(6,FMT=*) p1,p2
END
```

Código Matlab6.5 para la representación:

```
p1=zeros(101,6);
p2=zeros(101,6);
delta=zeros(6);
x=zeros(101);
delta(1)=0.1;
delta(2)=0.01;
delta(3)=0.008;
delta(4)=0.007;
delta(5)=0.005;
delta(6)=0.003;
for n=1:6
pas(n)=2*delta(n)/100;
end
i=1;

x(1)=1-delta(i);
while (i<=6)
j=1;
x(j,i)=1-delta(i);
while (x(j,i)<=1+delta(i))
p1(j,i)=(x(j,i)-1)^6;
p2(j,i)=x(j,i)^6-6*x(j,i)^5+15*x(j,i)^4-20*x(j,i)^3+15*x(j,i)^2-6*x(j,i)+1;
x(j+1,i)=x(j,i)+pas(i);
j=j+1;
end
i=i+1;
end
```

```
plot(x(1:101,1),p1(1:101,1),'g',x(1:101,1),p2(1:101,1),'r',x(1:101,2),p1(1:101,2),'b',x(
1:101,2),p2(1:101,2),'k',x(1:101,3),p1(1:101,3),'c',x(1:101,3),p2(1:101,3),'y',x(1:101,4
),p1(1:101,4),'m',x(1:101,4),p2(1:101,4),'g',x(1:101,5),p1(1:101,5),'r',x(1:101,5),p2(1:
101,5),'b',x(1:101,6),p1(1:101,6),'k',x(1:101,6),p2(1:101,6),'c')
plot(x(1:101,1),p1(1:101,1),'g',x(1:101,1),p2(1:101,1),'r')
plot(x(1:50,1),abs(p1(1:50,1)-p2(1:50,1))/p1(1:50,1))
plot(x(1:50,2),p1(1:50,2),'b',x(1:50,2),p2(1:50,2),'r')
```

```
plot(x(1:50,2),abs(p1(1:50,2)-p2(1:50,2))/p1(1:50,2))
plot(x(1:50,3),p1(1:50,3),'b',x(1:50,3),p2(1:50,3),'r')
plot(x(1:50,3),abs(p1(1:50,3)-p2(1:50,3))/p1(1:50,3))
plot(x(1:50,4),p1(1:50,4),'b',x(1:50,4),p2(1:50,4),'r')
plot(x(1:50,4),abs(p1(1:50,4)-p2(1:50,4))/p1(1:50,4))
plot(x(1:50,5),p1(1:50,5),'b',x(1:50,5),p2(1:50,5),'r')
plot(x(1:50,5),abs(p1(1:50,5)-p2(1:50,5))/p1(1:50,5))
plot(x(1:50,6),p1(1:50,6),'b',x(1:50,6),p2(1:50,6),'r')
plot(x(1:50,6),abs(p1(1:50,6)-p2(1:50,6))/p1(1:50,6))
```

Ejercicio 2

Programa FORTRAN 77

```
      integer j,i
      REAL x, tab(19), p1(19), p2(19), p3(19), errabs1(19), errrel1(19),
      errabs2(19), errrel2(19), errabs3(19), errrel3(19), coseno
      PARAMETER(f1='ESCRIBE EL NUMERO A EVALUAR:')
      write(6,*)
      read(3,*) x
      i=1
      coseno=cos(x)
      tab(1)=1.0e-1
      DO while (tab(i).LE.1e-18)
      p1(i)=(sin(x+tab(i))-sin(x-tab(i)))/tab(i)
      errabs1(i) =p1(i)-coseno
      errrel1(i)=(p1(i)-coseno)/coseno
      p2(i)=(sin(x+tab(i))-sin(x))/tab(i)
      errabs2(i)=p2(i)-coseno
      errrel2(i)=(p2(i)-coseno)/coseno
      p3(i)=(sin(x+tab(i))-sin(x))/tab(i)
      errabs3(i)=p3(i)-coseno
      errrel3(i)=(p3(i)-coseno)/coseno
      tab(i+1)=tab(i)*1.0e-1
      i=i+1
      ENDDO
      WRITE(6,FMT=*) p,coseno
      END
```

Código para las representaciones realizado con Matlab 6.5

```
tab=zeros(19,1);
p1=zeros(19,1);
p2=zeros(19,1);
p3=zeros(19,1);
errabs1=zeros(19,1);
errrel1=zeros(19,1);
errabs2=zeros(19,1);
errrel2=zeros(19,1);
errabs3=zeros(19,1);
errrel3=zeros(19,1);
i=1;
coseno=cos(x);
tab(1,1)=1.0e-1
while (tab(i,1)>=1e-18)
p1(i,1)=(sin(x+tab(i,1))-sin(x-tab(i,1)))/tab(i,1);
errabs1(i,1) =abs(p1(i,1)-coseno);
errrel1(i,1)=errabs1(i,1)/coseno;
p2(i,1)=(sin(x+tab(i,1))-sin(x))/tab(i,1);
errabs2(i,1)=abs(p2(i,1)-coseno);
errrel2(i,1)=errabs2(i,1)/coseno;
p3(i,1)=(sin(x+tab(i))-sin(x))/tab(i,1);
errabs3(i,1)=abs(p3(i,1)-coseno);
errrel3(i,1)=errabs3(i,1)/coseno;
tab(i+1,1)=tab(i,1)*1.0e-1;
i=i+1;
end

plot(log(tab(:)),log(errabs1(:)),'g',log(tab(:)),log(errabs2(:)),'b',log(tab(:)),
log(errabs3(:)),'r')
```

Ejercicio 3

*Programa FORTRAN 77 para real*4*

```
      integer i
      REAL x1(100),x2(100)
C     Programa con variables real4

      i=3
      x1(1)=1
      x1(2)=(sqrt(5.0)-1.0)/2.0
      x2(1)=1
      x2(2)=(sqrt(5.0)-1.0)/2.0
      DO while (i.LE.100)
      x1(i)=x1(i-1)*(sqrt(5.0)-1.0)/2.0
      x2(i)=x2(i-2)-x2(i-1)
      i=i+1
      ENDDO
      WRITE(6,FMT=*) x1,x2
      END
```

*Para real*8*

```
      integer i
      DOUBLEPRECISION x1(100),x2(100)
C     Programa con variables real8

      i=3
      x1(1)=1
      x1(2)=(dsqrt(5.DO)-1.DO)/2.DO
      x2(1)=1
      x2(2)=(dsqrt(5.DO)-1.DO)/2.DO
      DO while (i.LE.100)
      x1(i)=x1(i-1)*(dsqrt(5.DO)-1.DO)/2.DO
      x2(i)=x2(i-2)-x2(i-1)
      i=i+1
      ENDDO
      WRITE(6,FMT=*) x1,x2
      END
```

Ejercicio 4

Programa FORTRAN 77 apartado a

```
      integer i
      DOUBLEPRECISION p(30),errel(30)
      n=2
      p(1)=0
      p(2)=2*(dsqrt(2.DO))
      DO while (n.LE.30)
      p(n+1)=(2.DO)**n*dsqrt((2.DO)*((1.DO)-dsqrt((1.DO)-
(p(n)/((2.DO)**n))**(2.DO))
      errel(n+1)=abs(π-p(n+1))/π;
      n=n+1
      ENDDO
      WRITE(6,FMT=*) p,errel
      END
```

Código Matlab6.5

```
      format long e
      DIGITS(10);
      p=zeros(30,1);
      errel=zeros(30,1);
      n=2;
      p(1,1)=0;
      p(2,1)=2*(sqrt(2));
```

```
while (n<=30)
p(n+1,1)=2^n*(sqrt(2*(1-(sqrt(1-(p(n,1)/(2^n))^2)))));
errel(n+1,1)=abs(p(n+1,1)-pi)/pi;
n=n+1;
end
```

Programa FORTRAN 77 apartado b

```
   integer n
   DOUBLEPRECISION p(30),errel(30),r(30)
   PARAMETER(pi=3.14159265358979323846)
n=2
r(3)=2.DO-dsqrt(2.DO)
p(n)=2**2*(dsqrt(r(n)))
DO while (n.LE.30)
p(n+1)=(2.DO)**n*dsqrt(r(n+1))
errel(n+1)=abs(pi-p(n+1))/pi;
n=n+1
r(n+1)=r(n)/((2.DO)+(dsqrt(4.DO-r(n))))
ENDDO
WRITE(6,FMT=*) p,errel
END
```

Programa Matlab 6.5 Apartabo b

```
   format long e;
   DIGITS(10);
   p=zeros(30,1);
   r=zeros(31,1);
   errel=zeros(30,1);
n=2;
r(3,1)=2-sqrt(2);
p(3,1)=2^2*(sqrt(r(n+1)));
while (n<=30)
p(n+1,1)=2^n*(sqrt(r(n+1,1)));
errel(n+1,1)=abs(pi-p(n+1,1))/pi;
n=n+1;
r(n+1)=r(n)/(2+sqrt(4-r(n,1)));
end
```