Trabajo práctico Nº1

Introducción – Errores   
  


Metodos de computación científica -2013

Brenda Soledad Dilschneider L.U. 92774

Profesora: Dra. Nélida Beatriz Brignole

**Ejercicio 1**  
Encuentre el Epsilon machine de su computadora trabajando con Matlab.

Desarrollo:

♦ Codigo en Matlab:

fprintf('Ejercicio 1')

%Realizo el calculo del epsilon machine

s=1.0;

j=0;

while (1==1)

j=j+1;

s=s/2.0;

t=1.0+s;

if(t==1.0)

epsilon = s\*2.0

iteraciones = j-1

break

end

end

♦ Salida en pantalla:

Ejercicio 1

epsilon =

2.220446049250313e-016

iteraciones =

52

Otra posibilidad es utilizar el comando “eps” de Matlab, el cual muestra el épsilon machine de la computadora.

♦ Codigo en Matlab:

fprintf('Ejercicio 1')

%Realizo el calculo del epsilon machine

epsilon=eps

♦ Salida en pantalla:

Ejercicio 1

epsilon =

2.220446049250313e-016

**Ejercicio 2**

La constante de amortiguación c de la suspensión mostrada en la figura está dada por:

Donde µ es la viscosidad del fluido.  
Sugiera un procedimiento para encontrar la influencia de pequeños errores en a,h,r y l sobre c para los valores de referencia: µ= 0.3445 Pa.s, l=10 cm, h=0.1 cm, a=2cm, y r=0.5 cm. Utilice el procedimiento para predecir el valor de c bajo estas condiciones:

1. l=9.999 cm, h=0.09 cm, a=1.999 cm y r=0.499 cm
2. l=10.001 cm, h=0.101 cm, a=2.001 cm y r=0.501 cm
3. l=9.999 cm, h=0.101 cm, a=2.001 cm y r=0.501 cm
4. Extraiga conclusiones



Desarrollo:

Codigo Matlab:

Previamente, realizo la función de amortiguación “c”

‘amortiguacion.m’

function [c] = amortiguacion(l, h, a, r)

c = 6\*pi\*0.3445\*l/h^3\*((a-h/2)^2-r^2)\*((a^2-r^2)/(a-h/2)-h);

end

Luego, realizo el cálculo de la misma utilizando los valores de referencia y los valores bajo las condiciones de los insisos a) , b) y c).

%calculo "c" utilizando los valores de referencia

referencia=amortiguacion(10,0.1,2,0.5);

%calculo "c" tomando los valores con errores

a=amortiguacion(9.999,0.09,1.999,0.499);

b=amortiguacion(10.001,0.101,2.001,0.501);

c=amortiguacion(9.999,0.101,2.001,0.499);

%obtengo los errores absolutos

ea = a-referencia;

eb = b-referencia;

ec = c-referencia;

%obtengo los errores relativos

ear = (a-referencia)/referencia;

ebr = (b-referencia)/referencia;

ecr = (c-referencia)/referencia;

%presento los resultados para realizar la comparacion

Valores = [referencia;a;b;c]

ErroresAbs = [0;ea;eb;ec]

ErroresRel = [0;ear;ebr;ecr]

Salida en pantalla:

Valores =

1.0e+005 \*

4.205614192664587 %con valores de referencia

5.809814254317265 %punto a)

4.083527730113591 %punto b)

4.087306523181474 %punto c)

ErroresAbs =

1.0e+005 \*

0 %con valores de referencia

1.604200061652678 % punto a)

-0.122086462550996 % punto b)

-0.118307669483114 % punto c)

ErroresRel =

0 %con valores de referencia

0.381442516636623 % punto a)

-0.029029401404422 % punto b)

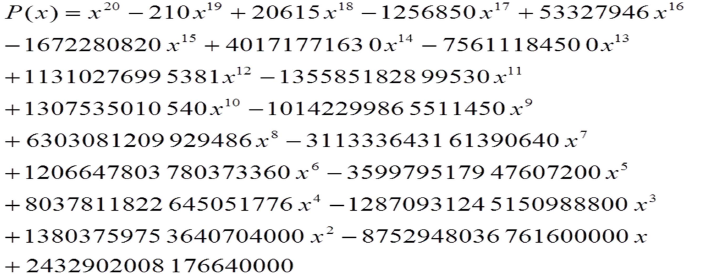
-0.028130889820913 % punto c)

Conclusión:

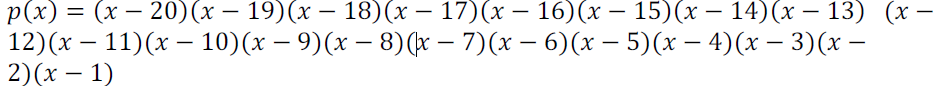
Se observa en los errores relativos de cada caso que la mayor modificación de la variable *h* es la que presenta un mayor impacto en el cálculo de *c*. Esto ocurre porque la variable *h* es la más cercana al 0, y además dicha variable se eleva al cubo en la fórmula.

**Ejercicio 3**

Usando una rutina para encontrar raíces, compute las 20 raices computadas para k=1,2…20 del polinomio P donde:



Use una rutina que sea capaz de construir raíces complejas de doble precisión. La fórmula dada para P es la forma expandida del polinomio de Wilkinson p(x):



Chequee la calidad de las raíces computadas para k=1,2…20 calculando:  
 . Explique.

Desarrollo:

Con el comando “roots” de Matlab calculo las raíces del polinomio desarrollado, P(x).

Codigo Matlab:

%Genero el polinomio desarrollado P(x) y luego calculo sus raices

p=[1 -1];

for n=2:20, p=[p 0]-[0 p\*n];end

format long;

raices = roots(p)

Salida en pantalla:

raices =

20.000324878101402

18.997159990805148

18.011221676102622

16.971132243131219

16.048274533412592

14.935355760260174

14.065272732694492

12.949055715498519

12.033449121964885

10.984041283443625

10.006059681252784

8.998394492431256

8.000284343435283

6.999973481009383

5.999999755869895

5.000000341909659

3.999999967630562

3.000000001049189

1.999999999997379

0.999999999999841

Se puede comparar en la siguiente tabla, que las raíces obtenidas son diferentes a las del polinomio sin desarrollar p(x).

|  |  |
| --- | --- |
| Raices computadas | |
| Raices del polinomio p(x) | Raices computadas con Matlab de la forma expandida P(x) () |
| 20 | 20.000324878101402 |
| 19 | 18.997159990805148 |
| 18 | 18.011221676102622 |
| 17 | 16.971132243131219 |
| 16 | 16.048274533412592 |
| 15 | 14.935355760260174 |
| 14 | 14.065272732694492 |
| 13 | 12.949055715498519 |
| 12 | 12.033449121964885 |
| 11 | 10.984041283443625 |
| 10 | 10.006059681252784 |
| 9 | 8.998394492431256 |
| 8 | 8.000284343435283 |
| 7 | 6.999973481009383 |
| 6 | 5.999999755869895 |
| 5 | 5.000000341909659 |
| 4 | 3.999999967630562 |
| 3 | 3.000000001049189 |
| 2 | 1.999999999997379 |
| 1 | 0.999999999999841 |

Realizo los cálculos de .

|  |  |  |  |  |
| --- | --- | --- | --- | --- |
|  |  |  |  |  |
| Z20 | 20.000324878101402 | 4.093834113718904e+028 | 3.956540207918145e+013 | 3.248781014022484e-004 |
| Z19 | 18.997159990805148 | 1.961062198795864e+028 | 1.805422127058150e+013 | 0.002840009194852 |
| Z18 | 18.011221676102622 | 8.970179353340975e+027 | 8.157011713765164e+012 | 0.011221676102622 |
| Z17 | 16.971132243131219 | 3.617999265161218e+027 | 3.461391431569775e+012 | 0.028867756868781 |
| Z16 | 16.048274533412592 | 1.460830424179883e+027 | 1.602499385073991e+012 | 0.048274533412592 |
| Z15 | 14.935355760260174 | 3.926366822874965e+026 | 6.312099816011450e+011 | 0.064644239739826 |
| Z14 | 14.065272732694492 | 9.286054615380475e+025 | 3.049334859849435e+011 | 0.065272732694492 |
| Z13 | 12.949055715498519 | 2.095499523787772e+025 | 1.193536834504933e+011 | 0.050944284501481 |
| Z12 | 12.033449121964885 | 3.179290549654640e+025 | 5.428732482131066e+010 | 0.033449121964885 |
| Z11 | 10.984041283443625 | 2.184002194421331e+025 | 2.097297657358964e+010 | 0.015958716556375 |
| Z10 | 10.006059681252784 | 1.165819063980438e+025 | 7.974216864037723e+009 | 0.006059681252784 |
| Z9 | 8.998394492431256 | 5.067994544412036e+024 | 2.585219715284541e+009 | 0.001605507568744 |
| Z8 | 8.000284343435283 | 1.874584789508096e+024 | 6.863531474568951e+008 | 2.843434352826080e-004 |
| Z7 | 6.999973481009383 | 5.755937488452425e+023 | 1.188990024602716e+008 | 2.651899061678620e-005 |
| Z6 | 5.999999755869895 | 1.401812892745079e+023 | 2.553942045378088e+006 | 2.441301045763566e-007 |
| Z5 | 5.000000341909659 | 2.395972573776952e+022 | 1.073055139717602e+007 | 3.419096588075377e-007 |
| Z4 | 3.999999967630562 | 1.723959558521846e+021 | 4.063553878431277e+006 | 3.236943779683088e-008 |
| Z3 | 3.000000001049189 | 3.390828683044084e+020 | 7.463668005077952e+005 | 1.049189179980203e-009 |
| Z2 | 1.999999999997379 | 8.782569861287019e+019 | 1.678071442310527e+004 | 2.621014516535070e-012 |
| Z1 | 0.999999999999841 | 3.181966722132622e+018 | 1.933961699513385e+004 | 1.589839371263224e-013 |

Conclusión:

Las raíces múltiples son mal condicionadas, esto es, son sensibles a pequeñas perturbaciones tales como errores de redondeo. Un pequeño cambio en los datos provoca grandes cambios en los resultados.  
Por lo tanto, es natural esperar que el mal condicionamiento ocurra cuando el polinomio tenga ceros que están muy cercanos. Sin embargo, el problema puede ser mal condicionado por polinomios con ceros bien separados. Wilkilson uso el polinomio p(x) para ilustrar este punto.