ESTABILIDAD NUMÉRICA. ERRORES DE REDONDEO Y TRUNCAMIENTO

TAREA NÚMERO UNO

ESTABILIDAD NUMÉRICA

Estamos interesados en escoger métodos computacionales que produzcan resultados confiables en su precisión. Un criterio que es aconsejable utilizar en los programas, es que cambios pequeños en los datos iniciales produzcan cambios pequeños en los resultados finales. Un algoritmo que satisface esta propiedad se llama estable. Es inestable cuando este criterio no se cumple. Para ejemplificar esto tenemos la siguiente explicación:
Para un sistema de ecuaciones es deseable incrementar el ancho de etapa *h* progresivamente tan rápido que la transición decaiga.El método debe ser numéricamente estable para valores largos de *h*. La formula de Euler a diferencia de la derivada y la regla trapezoidal ha sido reconocida como un procedimiento apropiado. Dahlquist (1963) estudio el problema y definió la propiedad de estabilidad en λ para un método de multi-etapas aplicado a la ecuación de prueba:

$$dy/dx = \lambda y$$

Donde λ es una cantidad compleja con una parte real negativa.
Un metodo de etapas k es escrito de la siguiente forma:

$$\sum_{j=0}^{k} \alpha_j y_{n+j} = h \sum_{j=0}^{k} \beta_j f_{n+j}$$

Y para la ecuación de prueba tenemos asociada la estabilidad polinomial:

$$\Pi(r) = \sum_{j=0}^{k} \alpha_j r^j - h\lambda \sum_{j=0}^{k} \beta_j r^j$$

El metodo es dicho ser absolutamente estable en una region R en el plano complejo si, para todos $h\lambda \in R$, todas las raices de la estabilidad polinomial satisfacen:

$$|r_s| < 1, \quad s = 1,2,\ldots,k$$

El método de etapas k, es estable en A si las regiones de estabilidad absoluta contienen el entero del semiplano izquierdo $Re(\lambda h)<0$
Dahlquist (1963) muestra que

- En método de multi-etapas linear no puede ser estable en λ.
- El método del orden en la estabilidad λ no puede ser excedido en dos.

- El método de la segunda orden en la estabilidad A con un error pequeño constante es una regla trapezoidal.

En la busqueda de un metodo de orden mas alto para una estabilidad apropiada, Widlund (1967) propone la estabilidad propia de $A(\alpha)$.
Un metodo numerico es $A(\alpha)$ estable, $\alpha E(0,\pi/2)$ si esta region de estabilidad absoluta contiene el pico infinito $| \arg(-h\lambda) | <\alpha$
Widlund mostro que el metodo de multi-etapas linear del orden 3 y 4 existen cuando $A(\alpha)$ es estable para cualquier $\alpha <\pi/2$.
Gear (1969,1971) combino las propiedades de la estabilidad y precisión en su definición de estabilidad robusta. Un método numérico es dicho de estabilidad robusta si es absolutamente estable en la región R1 ($Re(h\lambda) <$ D) y precisa en la región R2 ($D< Re (h\lambda) < a$, $| Im (h\lambda) | < q$). Esto permite un incremento del ancho de cada etapa el largo $(h\lambda)$ en R2 pero ocurren dificultades con los sistemas que contienen componentes de alta oscilación
Posteriormente algunos otros autores estudiaron la estabilidad, teniendo el propósito de darle mas rigidez a la estabilidad en A. Cuando el método de una etapa es aplicado a la ecuación de prueba,

$$Y_{n+1} = Q(h\lambda) y_n$$

La estabilidad en A demuestra que

$$| Q(h\lambda) | < 1, \ Re (h\lambda) < 0$$

Para muchos métodos de una etapa, la estabilidad en A, sin embargo, Q(hl) es mas que

$$| Q(h\lambda) | \rightarrow 1 \quad \text{así} \quad Re (h\lambda) \rightarrow a$$

De manera que ésta aproximación numérica decae rápidamente, con una largo $| \lambda |$ decaerá mas lentamente. La regla trapezoidal condujo en esta moda (Rosenbrock (1963)) . Para traer esta dificultad a una estabilidad mas fuerte como se requería. El método de una etapa es estable en L si es estable en A y $| Q(h\lambda) | \rightarrow 0$ así Re (hl) –>a
El método de la estabilidad L enmarca que la transición rápida puede ser corrompida en la solución numérica así como las etapa h es rápidamente incrementada.
En la aplicación de la estabilidad en A con el método de una etapa, el largo del sistema de una ecuación no linear, Portero y Robinson (1974) encontraron que :

- Varios métodos de estabilidad en A dan soluciones altamente inestables.
- La precisión de las soluciones, frecuentemente aparecen con el orden del método usado.

Esto permitió estudiar la estabilidad y precisión de las soluciones numéricas a la ecuación de prueba mas general

$$y' = g' (x) + l (y-g(x))$$

La solución de la ecuación por el método de una etapa nos da una ecuación diferente con la forma:

$$e_{n+1} = y_{n+1} - g(x_{n+1}) = a(h\lambda)\, e_n + \beta\,(h, h\lambda, g(x))$$

Prothero y Robinson (1974) definen la estabilidad en S, que generaliza la estabilidad en A de la ecuación 2.6 y la examinan detalladamente por el método de una etapa en el límite $|h\lambda| \rightarrow a$. Entonces el error en la regla trapezoidal esta en el límite de la forma

$$\beta = -\ 1/6\ h^3\ (h\lambda)^{-1}\ g_n'''$$

Mientras, queda cerrada implícitamente la regla del punto medio

$$\beta = -\ 1/4\ h^2\ g_n''$$

El método de una etapa para la cual $b \rightarrow 0$ como $|hl| \rightarrow a$ se dicen que son precisas y robustas.

Prothero y Robinson demostraron que para el método de una etapa la cuál no era precisa ni robusta era considerablemente deteriorada en el orden que se incrementaba Re ($h\lambda$), en contraste con el error de estos métodos los cuales tendían a cero así Re ($-\lambda$) $\rightarrow$ a.

Para la perspectiva de Fuller en la estabilidad numérica del objeto nos referimos a Lambert (1973) y a Watt (1976). Comparando los diferentes métodos de los sistemas que han sido echos por Enright, Hull y Lindberg (1975), Ehle y Lawson (1975) quienes generalizaron el proceso de Runge-Kutta para hacer rigidos los sistemas y mas recientemente por Alexander (1977) quien hizo comentarios favorables en el uso acerca de la diagonal implícita en el método de Runge-Kutta. El coeficiente de la matriz de este método es el triángulo mas pequeño con todos los elementos de la diagonal iguales, de manera que las ecuaciones implícitas quedan resueltas en la forma que la misma iteración de la matriz es usada. El método de la estabilidad S son derivados y confirmados por Prothero y Robinson, que con sus observaciones le dieron una gran importancia a la estabilidad robusta.

El método tiene ventajas para funciones altamente oscilatorias y requerimientos de baja precisión.

ERROR DE REDONDEO

Los errores de redondeo se deben a que las computadoras sólo guardan un número finito de cifras significativas durante un calculo. Las computadoras realizan esta función de maneras diferentes. Por ejemplo, si sólo se guardan siete cifras significativas, la computadora puede almacenar y usar π como $\pi = 3.141\ 592$,omitiendo los términos restantes y generando un error de redondeo el cual se ilustra con la ecuación siguiente:

E = valor verdadero - valor aproximado

p - 3.141 592

E = 0.00000065. . .

La anterior es una de las varias formas que utiliza una computadora para redondear números. Esta técnica de retener sólo los primeros siete términos se le llamó "truncamiento" en el ambiente de computación. De preferencia se le llamará de corte para distinguirlo de los errores de truncamiento. Un corte ignora los términos restantes de la representación decimal completa.

Por ejemplo, el octavo dígito significativo en este caso es 6. Por lo tanto p se representa de manera más exacta como 3.141 593 que como 3.141 592 obtenido mediante un corte, ya que el valor está más cercano del valor verdadero. Esto se puede visualizar de la siguiente manera, si p se aproxima por p = 3.141 593, el error de redondeo se reduce a:

E = 0.000 000 35. . .

Las computadoras se pueden desarrollar para redondear números de acuerdo a reglas de redondeo, como la que se acaba de mencionar. Sin embargo, esto agrega costo computacional por lo que algunas computadoras usan el corte directo. Este enfoque se justifica bajo la suposición de que el número de cifras significativas en la mayor parte delas computadoras es mucho mayor que el error de redondeo dado por un corte usualmente insignificante. Ya que la mayor parte de las computadoras tienen entre 7 y 14 cifras significativas, los errores de redondeo parecerían no ser muy importantes. Sin embargo, hay dos razones del porqué pueden resultar críticos en algunos métodos numéricos:

1.- Ciertos métodos requieren cantidades extremadamente grandes para obtener una respuesta. Además, estos cálculos a menudo dependen entre sí. Esto es, los cálculos posteriores son dependientes de los anteriores. En consecuencia, aunque un error de redondeo individual puede ser muy pequeño, el efecto de acumulación en el transcurso de la gran cantidad de cálculos puede ser significativo.

2.- El efecto del redondeo puede ser exagerado cuando se llevan a cabo operaciones algebraicas que emplean números muy pequeños y muy grandes al mismo tiempo. Ya que este caso se presenta en muchos métodos numéricos, el error de redondeo puede resultar de mucha importancia.

Ejemplo: Determínese la diferencia de dos números grandes: 32 981 108.123 4 y 32 981 107.998 9. En seguida, repítanse los cálculos pero incrementando el minuendo en un 0.001%.

Solución:

La diferencia de los números es

$$\begin{array}{r} 32\ 981\ 108.123\ 4 \\ \underline{-32\ 981\ 107.998\ 9} \\ 0.124\ 5 \end{array}$$

Ahora, incrementado el minuendo en un 0.001% se obtiene el número 32 981 437.934 5, y la diferencia es:

$$\begin{array}{r} 32\ 981\ 437.934\ 5 \\ \underline{-\ 32\ 981\ 107.998\ 9} \\ 329.935\ 6 \end{array}$$

que es considerablemente diferente de la primera. De aquí que una modificación en el minuendo, aparentemente insignificante, provoca una gran diferencia en el resultado. A fin de analizar en detalle el error por redondeo es necesario comprender cómo las cantidades numéricas se representan en las computadoras. En casi todos los casos, los números se almacenan como cantidades de punto flotante(números reales), que se parecen mucho a la

notación científica. Por ejemplo, el número de punto fijo 13.524 es lo mismo que el número de punto flotante .13524*102, que suele representarse como .13524E2. Otro ejemplo: -0.0442 es lo mismo que -.442E-1. Y así se elimina en gran medida dicho error.

ERROR DE TRUNCAMIENTO

Este tipo de error ocurre cuando un proceso que requiere un número infinito de pasos se detiene en un número finito de pasos.Generalmente se refiere al error involucrado al usar sumas finitas o truncadas para aproximar la suma de una serie infinita. Note que el error de truncamiento, a diferencia del error de redondeo, no depende directamente del sistema numérico que se emplee.

$$Pn(x) = f(x_o) + f'(x_o)(x - x_o) + \frac{f''(x_o)}{2!}(x - x_o)^2 + \ldots\ldots + \frac{f^n(x_o)}{n!}(x - x_o)^n$$

$$Pn(x) = \sum_{k=0}^{n} \frac{f^{(k)}(x_o)}{k!}(x - x_o)^k$$

Que es el polinomio de Taylor de grado n para la función f alrededor de xo.

$$Rn(x) = \frac{f^{(n+1)}(g(x))}{(n+1)!}(x - x_o)^{n+1}$$

Que es el residuo o error de truncamiento asociado con Pn. f(x) = Pn(x) + Rn(x) En el caso específico de que xo = 0 el polinomio de Taylor se conoce como el polinomio de Maclaurin y la serie de Taylor se conoce como la serie de Maclaurin.

Ejemplo 1:

Determine el polinomio de Taylor de segundo grado y también el de tercer grado para f(x) = cos(x) respecto a xo=0 y use este polinomio para aproximar cos(0.01)

Solución:

Polinomio de Taylor de segundo orden.

$$P_2(x) = \sum_{k=0}^{2} \frac{f^k(x_o)}{k!}(x - x_o)^k$$

$$P_2(x) = \frac{f^o(x_o)}{0!}(x - x_o)^0 + \frac{f'(x_o)}{1!}(x - x_o)^1 + \frac{f''(x_o)}{2!}(x - x_o)^2$$

$$con\_x_o = 0$$

$$P_2(x) = f^o(0) + \frac{f'(0)}{1!}x + \frac{f''(0)}{2!}x^2$$

$$f(x) = \cos x$$

$$f'(x) = \frac{d(\cos x)}{dx} = -\operatorname{sen} x$$

$$f''(x) = \frac{d(-\operatorname{sen} x)}{dx} = -\cos x$$

$$f'''(x) = \frac{d(-\cos x)}{dx} = \operatorname{sen} x$$

$$con\_x = 0.01$$

Calculando derivadas:

$$P_2(x) = \cos(0) + (-\text{sen}(0)x) - \frac{\cos(0)}{2}x^2$$

$$P_2(x) = 1 - x(0) - \frac{1}{2}x^2(1)$$

$$P_2(x) = 1 - \frac{1}{2}x^2$$

$$R_2(x) = \frac{f'''(g(x))}{3!}(x-0)^3$$

$$R_2(x) = \frac{\text{sen}(g(x))}{6}x^3$$

$$f(x) = Pn(x) + Rn(x)$$

$$f(x) = \cos x = 1 - \frac{1}{2}x^2 + \frac{x^3}{6}\text{sen}(g(x))$$

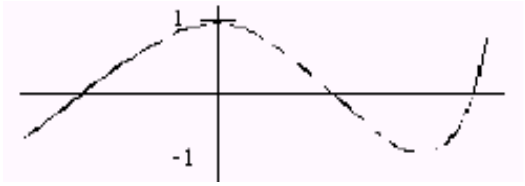
$$\cos(0.01) = 1 - \frac{1}{2}(0.01)^2 + \frac{\text{sen}(g(x))}{6}(0.01)^3$$

$$\cos(0.01) = 0.99995 + 0.16x10^{-6}\text{sen}(g(x))$$

$$\cos(0.01) - 0.99995 = 0.16x10^{-6}\text{sen}(g(x))$$

$$|\cos(0.01) - 0.99995| = |0.16x10^{-6}\text{sen}(g(x))|$$

$$|\cos(0.01) - 0.99995| = 0.16x10^{-6}|\text{sen}(g(x))|$$



donde $|\text{sen}(g(x))|$ a lo más es 1 por lo que

$$|\cos(0.01) - 0.99995| <= 0.16x10^{-6}\ Error\_de\_Truncamiento$$

PROGRAMA Ventana principal del programa:

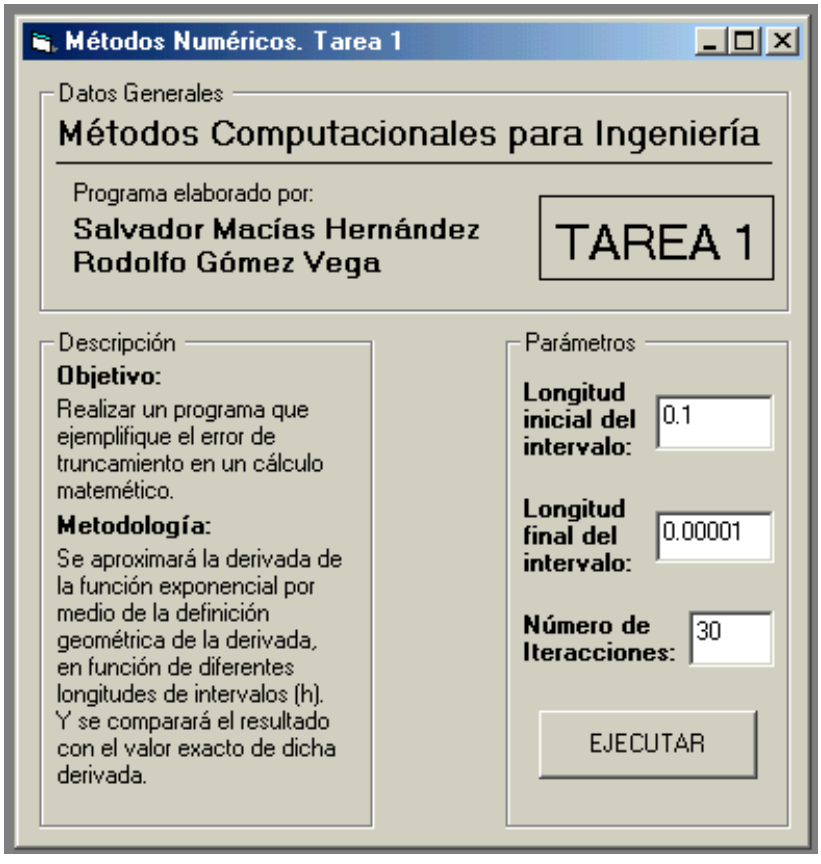
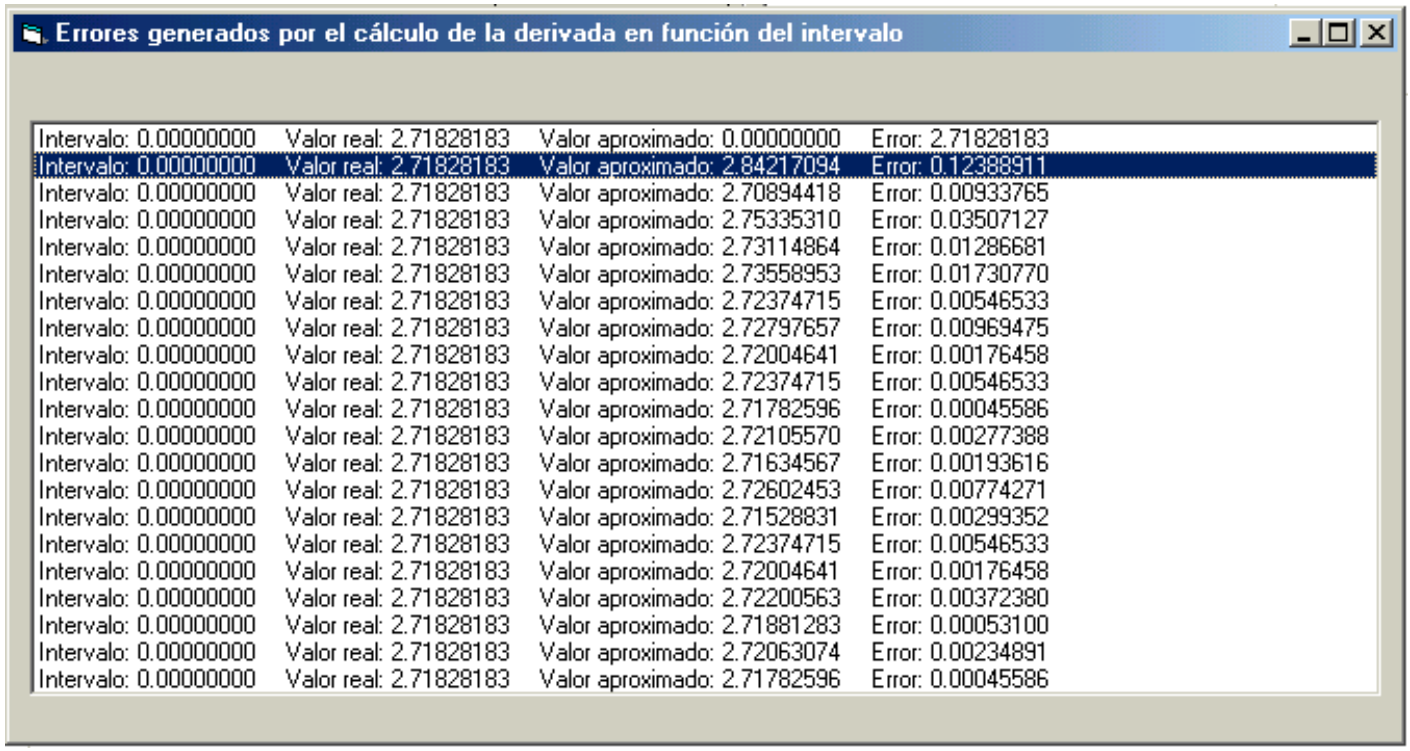


Figura 15

Pantalla de resultados:

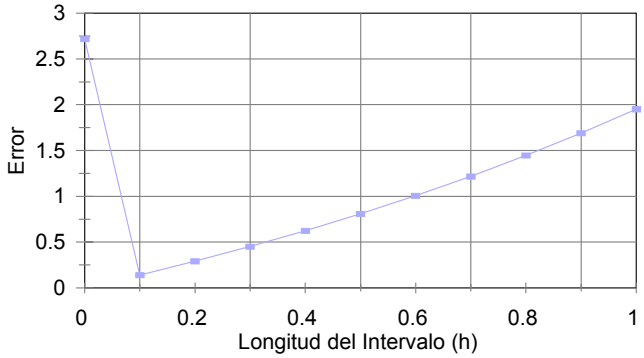


Para la prueba del programa se corrieron los siguientes parámetros:

Longitud inicial del intervalo:**1**
Longitud final del intervalo: **0.000000000000000000000001**
Número de iteracciones: **10**

Se obtuvieron los siguientes resultados:

Intervalo	Error
1E-24	2.71828182845905
0.1	0.140560126414838
0.2	0.290893642928466
3.00E-01	4.52E-01
0.4	0.624013517505028
5.00E-01	0.808532655298994
0.6	1.00630249810107
0.7	1.21838326192403
0.8	1.44592521648333
0.9	1.6901766313412
1	1.95249244201256



Código:

```
Public Function INTERVALO(ITERACCION)
  INTERVALO = frmTAREA1.txtINTERVALO2 + ITERACCION * (Val(frmTAREA1.txtINTERVALO0) - Val(frmTAREA1.txtINTERVALO2)) / Val(frmTAREA1.txtITERACCIONES)
End Function

Public Function VALORAPROXIMADO(INTERVALO)
'    D()=(Y(x+h)-X(x))/h)
' La función es e^x
' Y el valor de x es siempre uno
  VALORAPROXIMADO = (Exp(INTERVALO + 1) - Exp(1)) / INTERVALO
End Function

Private Sub Form_Load()
 For CUENTA = 0 To Val(frmTAREA1.txtITERACCIONES)
  intervalo1 = INTERVALO(CUENTA)
  Vaprox = VALORAPROXIMADO(intervalo1)
  lstDERIVADA.AddItem ("Intervalo: " & FormatNumber(intervalo1, 8) & "    Valor real: " & FormatNumber(Exp(1), 8) & "    Valor aproximado: " & FormatNumber(Vaprox, 8) & "     Error: " & FormatNumber(Abs(Exp(1) - Vaprox), 8))
 Next CUENTA
End Sub
```

BIBLIOGRAFÍA

Análisis Numérico
Richard L. Buden; J. Douglas Faire

http://mailweb.pue.udlap.mx/~ccastane/Analisis_Numerico_html/Unidad1_html/Sub1.html

Numerical methods for scientists and engineers
R.W. Hamming

Metodos Numericos y Programacion Fortran
Mccraken, Daniel D

Visual Basic 6, How to program
Deitel & Deitel T.R. Nieto