Capítulo 11

Integração e Interpolação

(Veja Applets para este tema no site http://www.cidse.itcr.ac.cr/cursos-linea/NUMERICO/index.htm)

É bem conhecido que a maioria dos integrais definidas só podem ser calculadas numericamente. Neste capitulo, vamos estudar alguns métodos numéricos do tipo Newton-Côtes, que empregam os valores de f(x) para valores de x uniformemente espaçados. Dois métodos simples desse grupo são a regra dos trapézios e a regra de Simpson

Regra dos trapézios

Para obter a regra geral, dividimos a área debaixo da curva y = f(x) entre x = a = x_0 e x = b = x_n em n faixas da mesma largura h = (b-a)/n. A integral será aproximada pela seguinte soma de áreas de trapézios:

$$\int_a^b f(x)dx = \frac{h}{2}(f(x_0)+f(x_1))+\frac{h}{2}(f(x_1)+f(x_2))+\ldots$$
$$\frac{h}{2}(f(x_{n-1})+f(x_n))= \qquad (1)$$
$$\frac{h}{2}(f(x_0)+f(x_n))+h(f(x_1)+f(x_2)+\ldots+f(x_{n-1}))$$

A última expressão é a regra dos trapézios composta. Nas duas figuras seguintes criamos duas diferentes funções VBA para avaliar esta fórmula.

```
Function Trapezio(a As Double, b As Double, n As Integer) As Double
    Dim i As Integer, h As Double, soma As Double
    h = (b - a) / n ' largura das faixas
    soma = (f(a) + f(b)) / 2
    For i = 1 To n - 1 Step 1 ' n = número das faixas
      soma = soma + f(a + i * h)
    Next i
    Trapezio = h * soma

End Function
```

```
Function f(x) As Double

    f = Log(x) / (1 + x ^ 2)

End Function
```

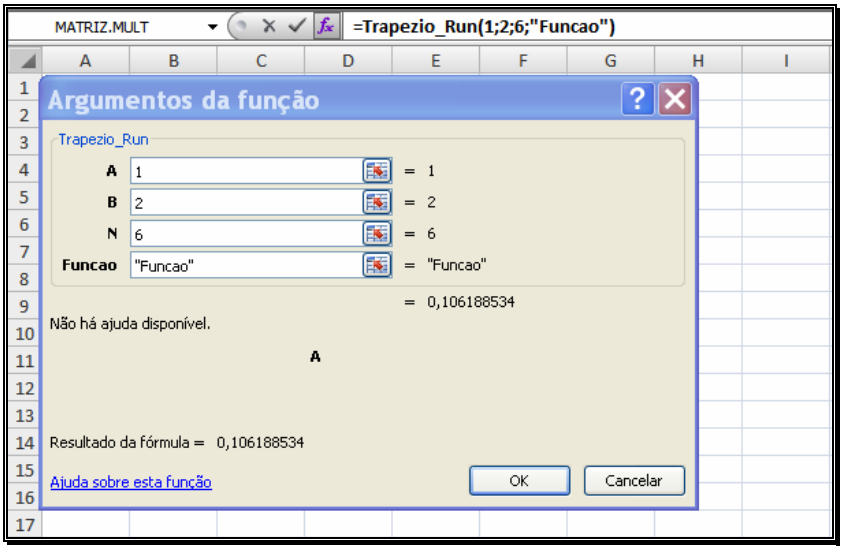
Para a =1, b = 2 , n = 6 obteremos o resultado 0,106189. Aumentando o valor de n, dará, evidentemente, melhores aproximações.

Na seguinte função utilizamos o método "Run", que requer um argumento nomeado: o nome da macro ou procedimento para ser executada, é em nosso caso "Funcao". O método Run retorna o que for retornado pela macro (pela Function Funcao) chamada. Os objetos passados são os valores de x.

```
Function Trapezio_Run(a, b, n, Funcao As String)
  Dim i As Integer, soma As Double, h As Double

    h = (b - a) / n

    soma = (Run(Funcao, a) + Run(Funcao, b)) / 2
    For i = 1 To (n - 1)
            soma = soma + Run(Funcao, a + i * h)
    Next

    Trapezio_Run = h * soma

End Function
```

```
Function Funcao(x)
   Funcao = Log(x) / (1 + x ^ 2) 'Sin(x) / x

End Function
```

O programa anterior parece ser preferível para a maioria dos usuários. Para calcular $\int_1^2 \frac{\ln(x)}{1+x^2}$, usando 6 faixas, a macro deve ser chamada como =Trapezio_Run(1;2;6;"Funcao"):



Regra de Simpson

Como foi feito com a regra dos trapézios, deve-se subdividir o intervalo de integração [a,b] em n subintervalos iguais de largura h. Com a regra de Simpson, o número n de subintervalos deve ser, porém, sempre par. A fórmula composta de Simpson é dada por

$$\int_a^b f(x)dx \approx \frac{h}{3}(y_0 + 4y_1 + 2y_2 + 4y_3 + 2y_4 + \ldots + 2y_{n-2} + 4y_{n-1} + y_n) \quad (2)$$

A implementação a seguir faz clara diferença entre os termos par e impar.

```
Function Simpson(a As Double, b As Double, n As Integer) As Double
    Dim i As Integer, h As Double
    Dim sompar As Double, somimpar As Double
    h = (b - a) / n ' largura das faixas
    somimpar = f(a) + f(b):
    For i = 1 To n - 1 Step 2 ' n = número das faixas
      somimpar = somimpar + 4 * f(a + i * h)
    Next i
     sompar = 0
    For i = 2 To n - 2 Step 2 ' n = número das faixas
      sompar = sompar + 2 * f(a + i * h)
    Next i
    Simpson = h * (sompar + somimpar) / 3

End Function
```

```
Function f(x) As Double

    f = Sin(x) / x ' Exp(x) * Log(x)

End Function
```

Mas, a seguinte macro é mais curta devido a um simples truco. Utiliza-se uma variável w = 6 - w que, no programa, somente pode tomar os valores 2, 4, 2, 4 ... se começa-se com w = 4.

Observe que o programa trabalha com uma soma só e que utilizamos somente um For To – Loop.

```
Function Simpson1(a As Double, b As Double, n As Integer) As Double
    Dim i As Integer, w As Integer, h As Double
    Dim soma As Double
    h = (b - a) / n ' largura das faixas
    soma = f(a) + f(b):
    w = 4 ' genera com w =6-w a série 4,2,4,2,4,...
    For i = 1 To n - 1 Step 1 ' n = número das faixas
      soma = soma + w * f(a + i * h)
      w = 6 - w
    Next i

    Simpson1 = h * soma / 3

End Function
```

```
Function f(x) As Double

    f = Sin(x) / x ' Exp(x) * Log(x)

End Function
```

Aplicação: Séries de Fourier

Uma série de senos e co-senos do tipo $\frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx + b_n sen\ nx)$ é chamada de *série trigonométrica*. Se esta série converge, ela representa então uma certa função f(x) que se pode escrever como

$$f(x) = \frac{a_0}{2} + \sum_{n=1}^{\infty} (a_n \cos nx + b_n sen\ nx) \quad (3)$$

Podemos calcular os coeficientes a_n e b_n, se conhecemos f(x). As fórmulas são

$$a_0 = \frac{1}{\pi} \int_{-\pi}^{+\pi} f(x)dx \quad (4)$$

$$a_n = \frac{1}{\pi} \int_{-\pi}^{+\pi} f(x)\cos nx\ dx \quad (n \geq 0)$$
$$b_n = \frac{1}{\pi} \int_{-\pi}^{+\pi} f(x)\text{sen}\ nx\ dx \quad (n > 0) \qquad (5),(6)$$

Na maioria dos casos, é impossível calcular as integrais analiticamente. Em tais situações devemos aplicar uma regra numérica, por exemplo a de Simpson.

Para poder fazer uma comparação entre integração analítica e numérica, usamos uma função simples, a saber $f(x) = x^2$. Queremos, então, desenvolver a função $f(x) = x^2$ em uma série de Fourier em co-senos no intervalo $(-\pi,+\pi)$. (Os coeficientes b_n se anulam todos, se f(x) for par. Isso é certo para $f(x) = x^2$.)

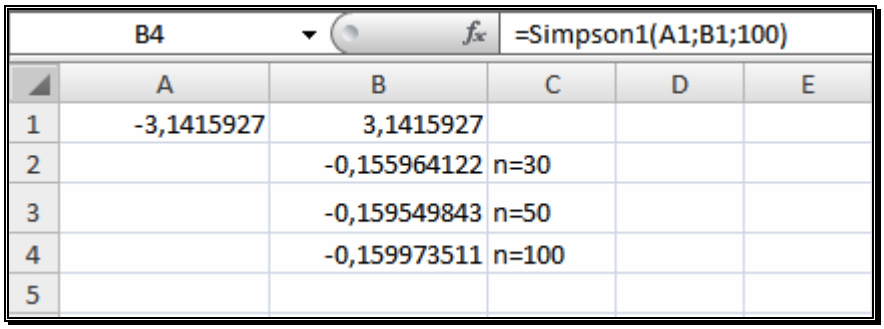
Utilizando o programa Simpson (ou Simpson1) podemos calcular os coeficientes a_n na tabela a seguir. Os resultados analíticos são $a_0 = 2\pi^2/3$ e $a_n = (-1)^n \cdot(4/n^2)$ para $n > 0$.

		numérico	**analítico**
A_0	=	3,2899	3,2899
A_1	=	-3,9999	-4,0000
A_2	=	0,9995	1,0000
A_3	=	-0,4432	-0,4444
A_4	=	0,2477	0,2500
A_5	=	-0,1560	-0,1600
A_6	=	0,1046	0,1111
A_7	=	-0,0715	-0,0816

Aqui temos um **exemplo**: queremos calcular com Simpson1 o coeficiente A_5:

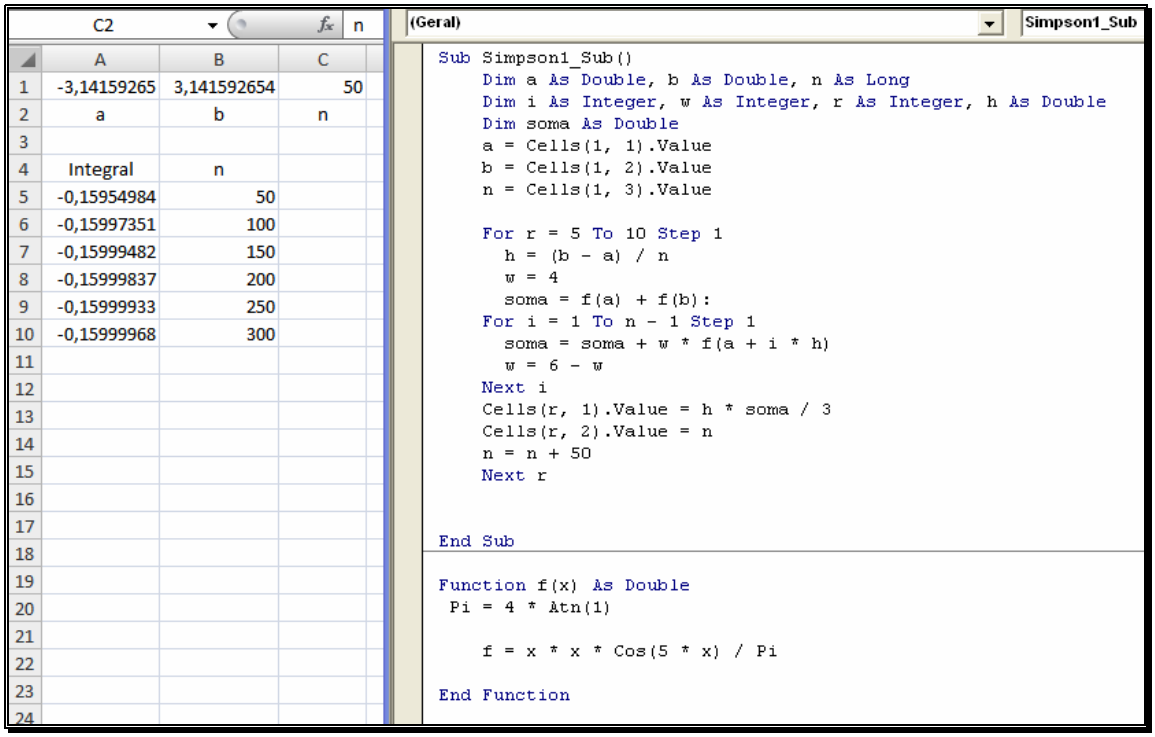
```
Function f(x) As Double
 Pi = 3.141592654

    f = x ^ 2 * Cos(5 * x) / Pi

End Function
```

Os valores para n = 30, 50, 100 podemos ver em B2:B4



Os valores numéricos na tabela foram calculados com n = 30.

A versão como sub-rotina permite também a produção de uma tabela. O programa lê os valores de a, b, n da planilha.



	A	B	C
1	-3,14159265	3,141592654	50
2	a	b	n
3			
4	Integral	n	
5	-0,15954984	50	
6	-0,15997351	100	
7	-0,15999482	150	
8	-0,15999837	200	
9	-0,15999933	250	
10	-0,15999968	300	

```
Sub Simpson1_Sub()
    Dim a As Double, b As Double, n As Long
    Dim i As Integer, w As Integer, r As Integer, h As Double
    Dim soma As Double
    a = Cells(1, 1).Value
    b = Cells(1, 2).Value
    n = Cells(1, 3).Value

    For r = 5 To 10 Step 1
      h = (b - a) / n
      w = 4
      soma = f(a) + f(b):
    For i = 1 To n - 1 Step 1
      soma = soma + w * f(a + i * h)
      w = 6 - w
    Next i
    Cells(r, 1).Value = h * soma / 3
    Cells(r, 2).Value = n
    n = n + 50
    Next r


End Sub

Function f(x) As Double
 Pi = 4 * Atn(1)

    f = x * x * Cos(5 * x) / Pi

End Function
```

O desenvolvimento da função $f(x) = x^2$ em uma série de Fourier em co-senos no intervalo (-π,+π) podemos escrever, então, assim

$$f(x) = x^2 \approx 3{,}2899 - 3{,}9999\cdot\cos(x) + 0{,}9995\cdot\cos(2x) - 0{,}4432\cdot\cos(3x) +$$

$$+ 0{,}2477\cdot\cos(4x) - 0{,}16\cdot\cos(5x)$$

O valor de π, podemos calcular no código VBA com Pi = 4*Atn(1), mas, na planilha, é 4*ATAN(1).

Planilha de Excel

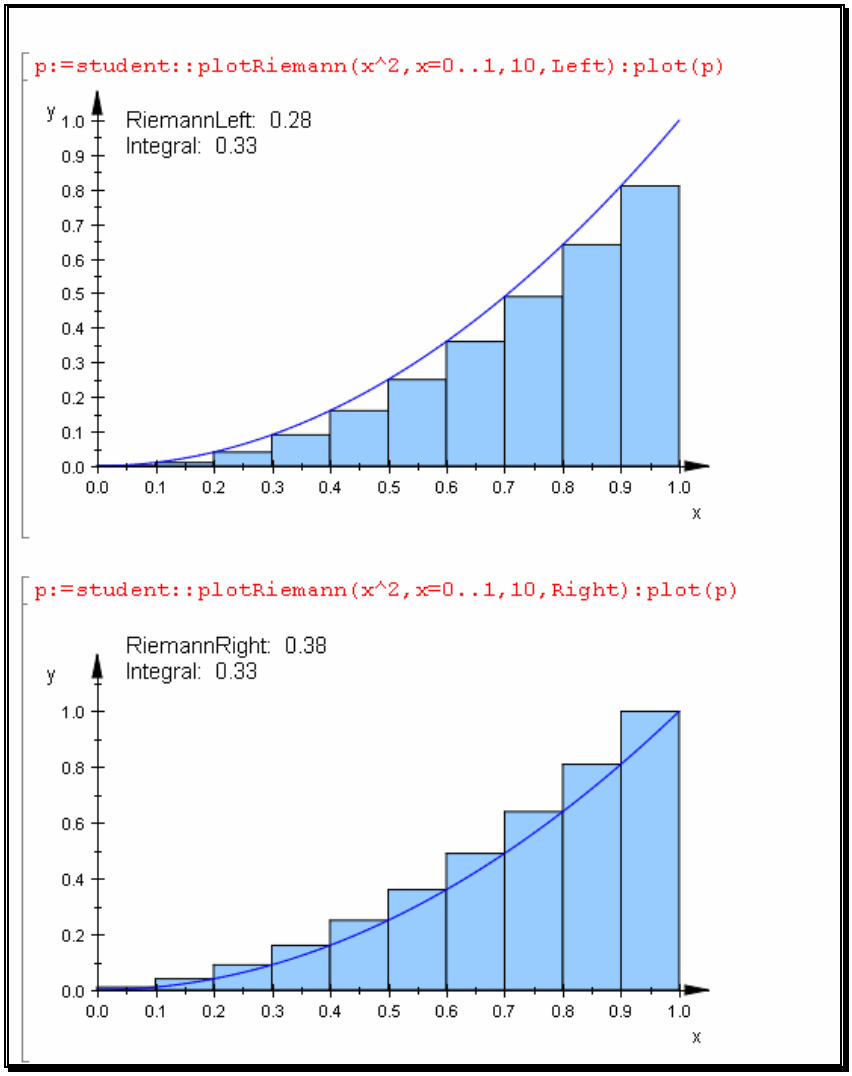
Também vale a pena criar uma planilha para três formulas (Trapézios, Tangentes, Simpson). Repetimos as bases teóricas utilizando as somas de Riemann. (A seguinte figura foi produzida em MuPAD cuja descrição pode-se encontrar no site do autor em http://www.geocities.com/Athens/Agora/6594/ .)

Na figura superior estão desenhados os retângulos que tocam a curva de por debaixo. A soma das áreas (soma inferior de Riemann) é

$$S_i = (y_0 + y_1 + y_2 + \ldots + y_{n-1})\cdot h$$

Na figura temos n = 10 e a função é a parábola $y = x^2$. $x_0 = 0$ e $x_{10} = 1.0$

Para S_i obtém-se o valor de 0.285 unidades de área.



A soma superior, S_u , é dada por

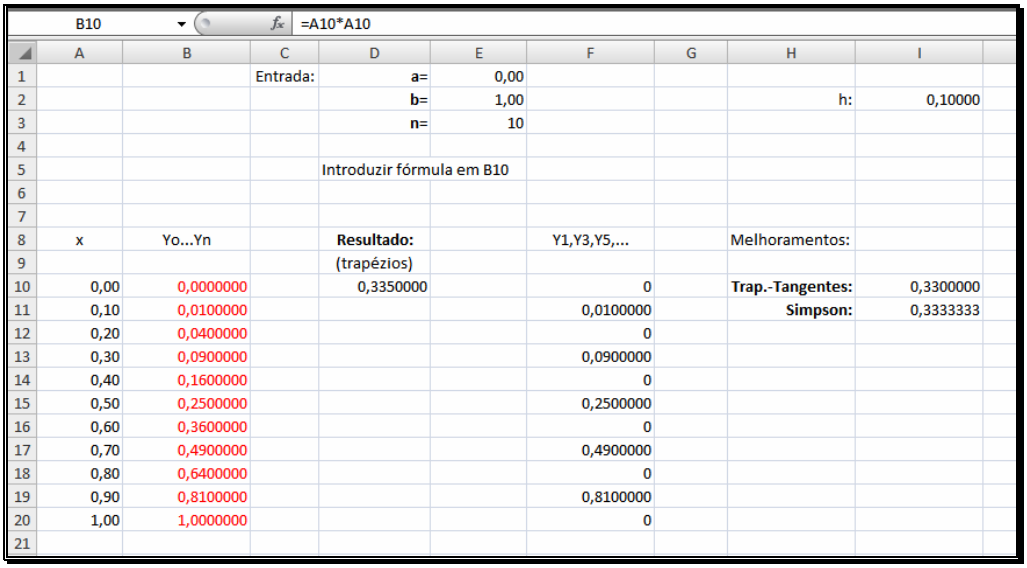
$$S_u = (y_1 + y_2 + y_3 + \ldots + y_n)\cdot h$$

Fazendo o cálculo, obtemos $S_u = 0.385$. O valor exato da integral é 1/3, e o valor médio $S_m = (S_i + S_u)/2 = 0{,}335$ é muito próximo ao valor exato. É fácil comprovar que este valor médio, S_m , coincide com a fórmula dos trapézios.

Uma melhoria obteremos também, se usamos sempre duas faixas e se traçamos por $y_{médio}$ a tangente à curva (será preciso, utilizar um número par de faixas!). A fórmula resultante será

$$S_t = 2\cdot h\cdot(y_1 + y_3 + y_5 + \ldots + y_{n-1})$$

Compare a planilha a seguir onde temos todas as fórmulas incorporadas.



B10 =A10*A10

	A	B	C	D	E	F	G	H	I
1			Entrada:	a=	0,00				
2				b=	1,00			h:	0,10000
3				n=	10				
4									
5				Introduzir fórmula em B10					
6									
7									
8	x	Yo...Yn		Resultado:		Y1,Y3,Y5,...		Melhoramentos:	
9				(trapézios)					
10	0,00	0,0000000		0,3350000		0		Trap.-Tangentes:	0,3300000
11	0,10	0,0100000				0,0100000		Simpson:	0,3333333
12	0,20	0,0400000				0			
13	0,30	0,0900000				0,0900000			
14	0,40	0,1600000				0			
15	0,50	0,2500000				0,2500000			
16	0,60	0,3600000				0			
17	0,70	0,4900000				0,4900000			
18	0,80	0,6400000				0			
19	0,90	0,8100000				0,8100000			
20	1,00	1,0000000				0			
21									

A fórmula de Simpson é dada pela seguinte expressão

$$S_{imp} = (2 \cdot S_m + S_t)/3$$

A função $y = x^2$ (=A10*A10) fica na célula B10.

Na célula D10, temos o resultado =I$2*(SOMA(B10:B20)- (B10+B20)/2). Em I10 fica =2*I$2*SOMA(F10:F20) e na I11 temos =(2*D10+I10)/3.

Um problema reside no fato de que a planilha deve ser modificada, se trocarmos o valor de n. No caso n = 20, temos de copiar as fórmulas até a linha 30 e as funções nas células D10, I10 devem conter B30 em vez de B20 e F30 em vez de F20.

Sem dúvida, nestes casos é preferível utilizar as macros que foram desenvolvidas mais acima.

Exercício: Calcule por meio da planilha a seguinte integral elíptica:

$$I = \int_0^{48} \sqrt{1+\cos^2 x}\ dx$$

utilizando n = 100. (Copiar até linha 110)

Resultados: Trapézios: 58,46239; Tangentes: 58,487679; Simpson: 58,4708211

Existem métodos melhores dos que utilizamos acima, por exemplo o método de Romberg, que dá o valor I ≈ 58,47047 com 4 casas decimais corretas.

Interpolação de Newton (1643-1727)

Vamos supor que, por meio de um experimento, temos uma tabela de dados:

x	-2	-1	0	1	2
y	3	0	-2	6	1

Suponhamos, além disso, que se queira conhecer o valor de y para um x não tabelado. O **método de Newton** resolve este problema determinando um polinômio, p(x), que passe pelos pontos dados e que permita determinar, aproximadamente, valores de y para pontos intermédios. (Devemos distinguir entre Interpolação e Regressão. Mais à frente, falando sobre métodos estadísticos, vamos estudar algumas técnicas de tratamento e análise de dados. Obviamente, a regressão vai ser uma destas técnicas. Dentre os processos matemáticos que resolvem problemas da regressão, com certeza, um dos mais utilizados é o *Método dos Mínimos Quadrados* de Gauss. Neste método trata-se de determinar uma função que passe o mais "próximo possível" dos pontos dados e não se pede que passe pelos pontos mesmos.)

Para não nos perdermos em considerações teóricas, apresentarei aqui a fórmula de Newton para a obtenção dum polinômio interpolador. (Fala-se de interpolação polinomial. Estes métodos, por exemplo os de Newton, Lagrange e Bernstein, diferem uns dos outros na táctica aplicada para determinar o polinômio interpolador.)

Consideremos uma tabela com n+1 pontos (x_0,y_0), (x_1,y_1), ... ,(x_n,y_n) e desejamos determinar um polinômio da forma

$$p(x) = a_0+a_1(x-x_0)+a_2(x-x_0)(x-x_1)+ \ldots +a_n(x-x_0)(x-x_1)\cdots(x-x_{n-1})$$

Pede-se determinar os coeficientes a_0, a_1, ..., a_n.

O polinômio interpolador da tabela acima tem, como demonstraremos, a forma

$$p(x) = -2 + (25x + 38x^2 - 7x^3 - 8x^4)/6$$

Para um programa como MuPAD, Maple, Mathematica, etc. não é nenhum problema calcular tal polinômio. Em MuPAD, por exemplo, temos

```
xList := [-2,-1,0,1, 2]:
yList := [3,0,-2,6,1]:
P := interpolate(xList, yList, X)
```

$$\mathrm{poly}\left(-\frac{4\cdot X^4}{3}-\frac{7\cdot X^3}{6}+\frac{19\cdot X^2}{3}+\frac{25\cdot X}{6}-2,\,[X]\right)$$

X é um valor não tabelado.

O método de Newton adapta-se particularmente bem à estrutura de uma planilha como o Excel, pois ele funciona segundo o seguinte esquema:

H2

	A	B	C	D	E	F	G	H	I	J	K	L	M
1													
2						xi	yi						
3													
4						**x0**	**y0** = a0						
5					x1-x0			[x0,x1,y]=a1					
6				x2-x0		**x1**	**y1**		[x0,x1,x2,y]=a2				
7			x3-x0		x2-x1			[x1,x2,y]		[x0,x1,x2,x3,y]=a3			
8		x4-x0		x3-x1		**x2**	**y2**		[x1,x2,x3,y]		[x0,x1,x2,x3,x4,y]=a4		
9			x4-x1		x3-x2			[x2,x3,y]		[x1,x2,x3,x4,y]			
10				x4-x2		**x3**	**y3**		[x2,x3,x4,y]				
11					x4-x3			[x3,x4,y]					
12						**x4**	**y4**						
13													
14													

As expressões do lado direito significam

$a_0 = y_0$
$a_1 = (y_1 - y_0)/(x_1 - x_0)$:= [x0,x1,y]
$a_2 = ((y_2 - y_1)/(x_2 - x_1) - (y_1 - y_0)/(x_1 - x_0))/(x_2 - x_0)$:= [x0,x1,x2,y]
.....
a_n = [x0,x1,...,xn,y]

I6 =(H7-H5)/D6

	A	B	C	D	E	F	G	H	I	J	K	L	M
1	**Valores dados**												
2	x	y				x	f			**Interpolação de Newton**			
3	1	3											
4	3	1				1	3						
5	4	6			2			-1					
6				3		3	1		2				
7					1			5					
8						4	6						
9													
10													
11					p(x) = 3 - 1(x-1) + 2(x-1)(x-3) = 2x^2 - 9x + 10								
12													

As entradas para esta planilha são

F4: x0 (=1); G4: y0 (=3)
F6: x1 (=3); G6: y1 (=1)
F8: x2 (=4); G8: y2 (=6)

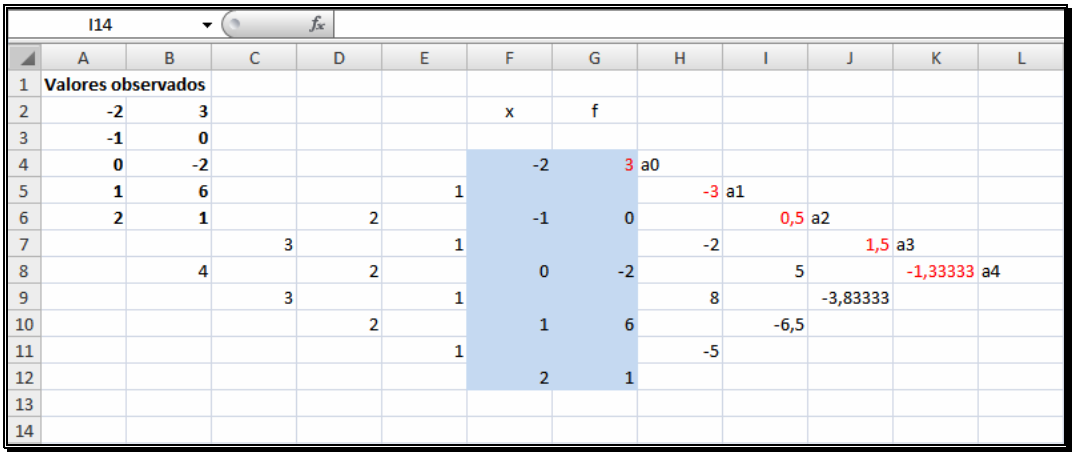
E5: =F6–F4; E7: =F8-F6; D6: =F8-F4

No lado direito (colunas H e I) calculamos as "diferenças divididas":

H5: =(G6-G4)/E5 (=a1); H7: =(G8-G6)/E7
I6: =(H7-H5)/D6 (=a2)

No numerador das frações temos a diferença de valores da coluna anterior, no denominador temos a diferença de valores x, que encontramos no lado esquerdo do esquema na posição correspondente (por exemplo: a célula D6 corresponde à célula I6).

A partir desta regra, podemos construir para qualquer número de valores observados o esquema das diferenças divididas. Compare a seguinte planilha que contém os 5 valores já considerados acima.



	A	B	C	D	E	F	G	H	I	J	K	L
1	Valores observados											
2	-2	3				x	f					
3	-1	0										
4	0	-2				-2	3 a0					
5	1	6			1			-3 a1				
6	2	1		2		-1	0		0,5 a2			
7			3		1			-2		1,5 a3		
8		4		2		0	-2		5		-1,33333 a4	
9			3		1			8		-3,83333		
10				2		1	6		-6,5			
11					1			-5				
12						2	1					
13												
14												

O polinômio resultante já foi calculado pelo MuPAD:

$$p(x) = 3 -3(x+2)+0.5(x+2)(x+1)+1.5(x+2)(x+1)(x)-4/3\cdot(x+2)(x+1)(x)(x-1)$$

$$p(x) = -2\,(25x + 38x^2 - 7x^3 - 8x^4)/6$$

O gráfico deste polinômio segue mais à frente.

Na realidade, podemos abrir mão do lado esquerdo do esquema e utilizar um esquema escalonado mais simples, veja a planilha a seguir, que vai servir para até 8 pares de valores observados (C4:D11).

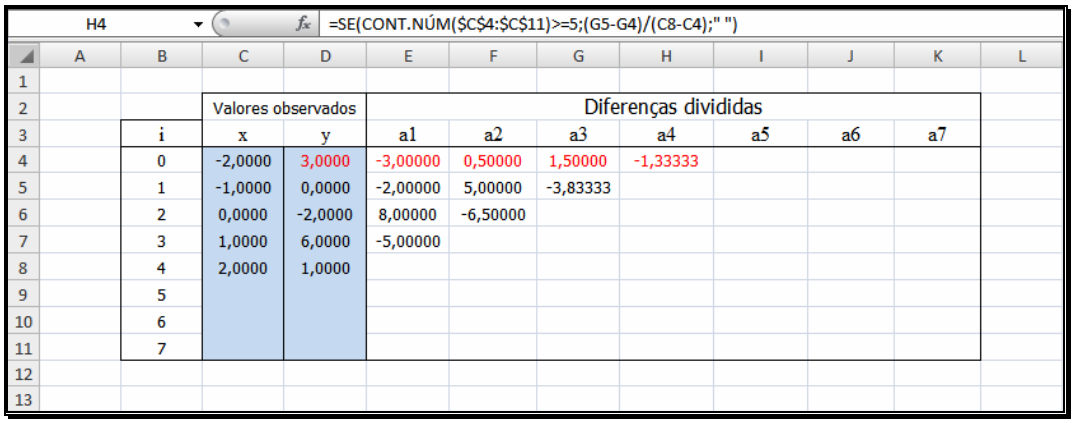
Na coluna E temos:

E4: =(D5-D4)/(C5-C4); E5: =SE(CONT.NÚM($C$4:$C$11)>=3;(D6-D5)/(C6-C5);" ") A função CONT.NÚM conta quantas células contêm números. No caso resulta CONT.NÚM(C4:C11) = 5, então colocamos na célula E5 o resultado de (D6-D5)/(C6-C5) = (-2-0)/(0-(-1))= -2.

E6: =SE(CONT.NÚM($C$4:$C$11)>=4;(D7-D6)/(C7-C6);" ")
E7: =SE(CONT.NÚM($C$4:$C$11)>=5;(D8-D7)/(C8-C7);" ")
E8: =SE(CONT.NÚM($C$4:$C$11)>=6;(D9-D8)/(C9-C8);" "), aqui escreve-se nada na célula E8, pois CONT.NÚM($C$4:$C$11)>=6 é falso. Na célula E10 temos a última fórmula nesta coluna: =SE(CONT.NÚM($C$4:$C$11)>=8;(D11-D10)/(C11-C10);" "))
F4: =SE(CONT.NÚM($C$4:$C$11)>=3;(E5-E4)/(C6-C4);" ")
F5: =SE(CONT.NÚM($C$4:$C$11)>=4;(E6-E5)/(C7-C5);" ").

Já podemos reconhecer o esquema detrás das fórmulas. É só necessário copiá-las para baixo e aumentar "manualmente" os números nas relações >= .

A última fórmula fica em K4 e reza =SE(CONT.NÚM($C$4:$C$11)=8;(J5-J4)/(C11-C4);" ")

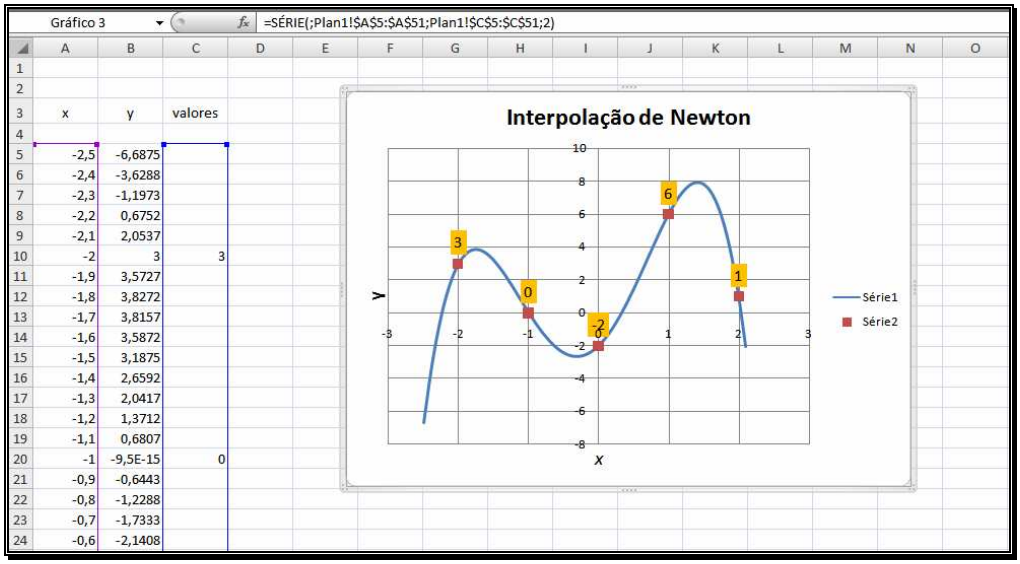


Obviamente, é muito desejável ter uma macro VBA que é capaz de fazer tudo isso mais rápido e praticamente com um só clique da mouse.
O programa VBA que segue baseia-se num artigo de R. Pfeifer no site http://www.arstechnica.de/computer/msoffice/vba/vba0094.html (em alemão). Ele se mantém ao esquema à pouco apresentado e trabalha com dois listas (Arrays) embutidas na macro. Desafortunadamente, o programa não pode determinar, explicitamente, o polinômio interpolador, como o vimos acima no caso do programa MuPAD. Porém, ele nos permite determinar o valor de p(x) para qualquer x fornecido por meio de uma InputBox. A sub-rotina "valor" contém os Arrays x, y dos dados observados e a função "InterNewton" calcula para cada valor x a ser interpolado primeiro o polinômio que depois é avaliado usando o método de **Horner**, que estudamos no último capítulo.

```
Sub valor()
 Dim x, y
 Dim z As Double
    z = InputBox("Valor de x?") 'com vírgula, p.ex. 2,5
    x = Array(0, 1, 2, -2, -1)
    y = Array(-2, 6, 1, 3, 0)
    MsgBox " f(" & z & ") = " & InterNewton(x, y, z)

End Sub


Function InterNewton(x, y, z)
    Dim i As Long, j As Long

    For i = LBound(y) To UBound(y)
       For j = UBound(y) To i + 1 Step -1
       y(j) = (y(j) - y(j - 1)) / (x(j) - x(j - i - 1))
       Next j
    Next i
    'Horner
    InterNewton = 0 ' inicia Horner
    For i = UBound(y) To LBound(y) Step -1
         InterNewton = InterNewton * (z - x(i)) + y(i)
    Next i

End Function
```

Para produzir uma tabela inteira de valores interpolados, que nos permitiria mostrar também a curva interpolador, precisamos de desenhar outra macro, assim como a nossa "Interpol", veja o próximo programa.

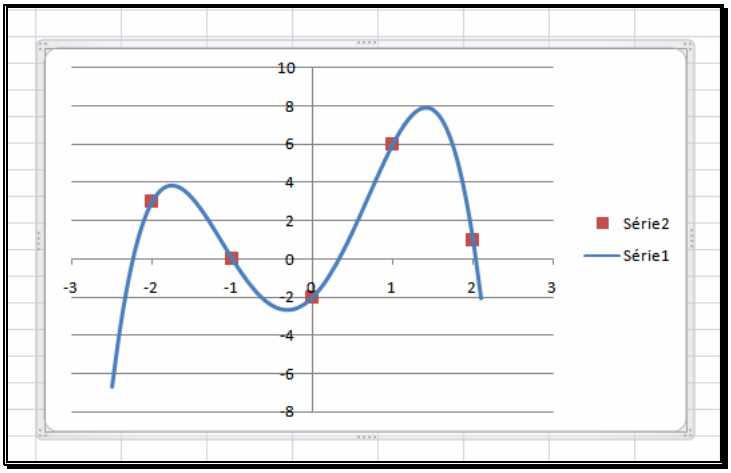
Na seguinte figura vemos o gráfico do polinômio p(x) = -2 (25x + 38x^2 – 7x^3 – 8x^4)/6, decorado com rótulos chamativos (veja embaixo para saber como se faz) nos marcadores dos pontos observados. Mas, o polinômio não foi utilizado. Os valores *y* na coluna B foram calculados pelo programa seguinte.



x	y	valores
-2,5	-6,6875	
-2,4	-3,6288	
-2,3	-1,1973	
-2,2	0,6752	
-2,1	2,0537	
-2	3	3
-1,9	3,5727	
-1,8	3,8272	
-1,7	3,8157	
-1,6	3,5872	
-1,5	3,1875	
-1,4	2,6592	
-1,3	2,0417	
-1,2	1,3712	
-1,1	0,6807	
-1	-9,5E-15	0
-0,9	-0,6443	
-0,8	-1,2288	
-0,7	-1,7333	
-0,6	-2,1408	

```
Private x, y

Public Sub Interpol(xx, ByVal yy)

    Dim i As Long, j As Long

    For i = LBound(yy) To UBound(yy)
       For j = UBound(yy) To i + 1 Step -1
       yy(j) = (yy(j) - yy(j - 1)) / (xx(j) - xx(j - i - 1))
       Next j
    Next i
    x = xx
    y = yy
    End Sub
Public Function InterHorn(z As Double) As Double
    Dim i As Long
    'Horner
    InterHorn = 0 ' inicia Horner
    For i = UBound(y) To LBound(y) Step -1
         InterHorn = InterHorn * (z - x(i)) + y(i)
    Next i

End Function

Sub valor()
 Dim x, y
 Dim k As Integer, z As Double
    'z = InputBox("Valor de x?") 'com vírgula, p.ex. 2,5
    x = Array(0, 1, 2, -2, -1)
    y = Array(-2, 6, 1, 3, 0)
    Interpol x, y
    'MsgBox " f(" & z & ") = " & InterHorn(z)
    z = -2.5
    k = 0
   Do While z >= -2.5 And z <= 2.5

    Cells(5 + k, 1) = z
    Cells(5 + k, 2) = InterHorn(z)

    k = k + 1
    z = z + 0.1
    Loop
End Sub
```

Uma vez calculadas as coordenadas, e colocadas automaticamente nas colunas A e B, inserimos os valores observados na coluna C. Em seguida selecionamos as células A5:C5 até C51 (com F5) e com *Inserir>Dispersão>Somente com Marcadores* fazemos um gráfico com pontos isolados. Clique, em seguida, sobre os pontos calculados e escolha *Alterar Tipo de Gráfico.* Deve-se eleger *Dispersão>Linhas Suaves* para obter algo parecido ao seguinte gráfico.



Finalmente clique em um dos marcadores e use *Adicionar Rótulos de Dados ...* e, depois, *Formatar Rótulos de Dados...* para moldá-los segundo o seu gosto.

Interpolação de Lagrange (1736-1813)

Se pode demonstrar que sempre existe o polinômio p(x) que interpola a função f(x), desconhecida, em x_0, x_1, ..., x_n e que é único. No entanto, existem várias formas para se obter tal polinômio. Ao lado do método de Newton é bem conhecida a forma de interpolação segundo Lagrange que consta de uma soma de polinômios especiais (os polinômios de Lagrange).

Buscamos, agora, um polinômio da forma

$$p(x) = y_0 L_0(x) + y_1 L_1(x) + ... + y_n L_n(x)$$

Calculam-se os polinômios de Lagrange, $L_i(x)$, pela seguinte expressão:

$$L_i(x) = \frac{(x-x_0)(x-x_1)...(x-x_{i-1})(x-x_{i+1})...(x-x_n)}{(x_i-x_0)(x_i-x_1)...(x_i-x_{i-1})(x_i-x_{i+1})...(x_i-x_n)}$$

Também podemos escrever $L_i(x) = \prod_{k=0}^{n} \frac{(x-x_k)}{(x_i-x_k)}; k \neq i \text{ para } i = 0,...,n$

As fórmulas têm um aspecto pouco amigável, mas a aplicação é bastante simples.

Na planilha utilizamos para cada $L_i(x)$ uma nova coluna e os coeficientes de p(x) são os valores y_j dos pares experimentais dados. Assim, para os três pares $(x_0;y_0)$ = (1;4), $(x_1;y_1)$ = (3;6) e $(x_2;y_2)$ = (4;12), o polinômio interpolador vai ser **$p(x) = 4L_0(x) + 6L_1(x) + 12L_2(x)$.**

Observe, que no cálculo de $L_i(x)$ fica excluído, no numerador, o valor de x_i. No denominador aparecem todos os x_i observados.

Exemplo: Dados os pontos experimentais (1;4), (3;6), (4;12)

Determine o polinômio de Lagrange para os pontos dados.

Resolução: $L_0 = (x-x_1)(x-x_2)/[(x_0-x_1)(x_0-x_2)] = (x-3)(x-4)/[(1-3)(1-4)$

$L_0 = (x^2-7x+12)/6$

$$L_1(x) = (x-1)(x-4)/[(3-1)(3-4)] = (x^2-5x+4)/(-2)$$

$$L_2(x) = (x-1)(x-3)/[(4-1)(4-3)] = (x^2-4x+3)/3$$

Assim obtemos $\mathbf{p(x) = 4L_0+6L_1+12L_2 = 5x^2/3 - 17x/3 + 8}$

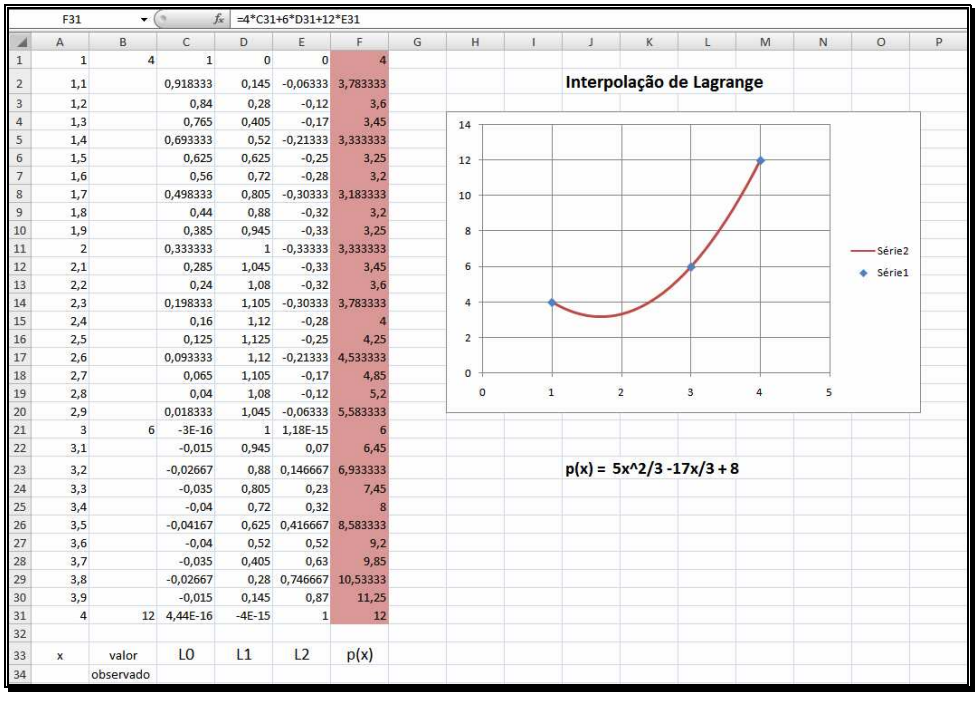
Na planilha, temos na coluna A os valores de x para os quais queremos calcular os valores y (A2: =A1+0,1). Em B1 fica $y_0 = 4$. A1 copiamos até A31 e na coluna B inserimos, nos lugares correspondentes, os valores y observados.

C1: =($A1-3)*($A1-4)/((1-3)*(1-4)) (=L_0). Esta fórmula copiamos na D1 e E1 e depois a editamos:
D1: =($A1-1)*($A1-4)/((3-1)*(3-4))
E1: =($A1-1)*($A1-3)/((4-1)*(4-3))

F1: 4*C1+6*D1+12*E1 (= p(x))

A representação no gráfico deve conter os 3 pontos observados e todos os pontos calculados na coluna F1. Visto que esta coluna não é adjacente, é preciso fazer a seleção das células com o nosso método **F8-F5**, que introduzimos no capítulo 5.

Em *Inserir* será preciso eleger, primeiro, XY *Linhas Suaves*, e depois XY *com Marcadores*, para ver os pontos experimentais. (Os valores calculados constituem, agora, a Série2 e não, como no caso da planilha de Newton, a Série1.) Não será difícil generalizar a planilha para acomodar mais pontos.



	A	B	C	D	E	F
1	1	4	1	0	0	4
2	1,1		0,918333	0,145	-0,06333	3,783333
3	1,2		0,84	0,28	-0,12	3,6
4	1,3		0,765	0,405	-0,17	3,45
5	1,4		0,693333	0,52	-0,21333	3,333333
6	1,5		0,625	0,625	-0,25	3,25
7	1,6		0,56	0,72	-0,28	3,2
8	1,7		0,498333	0,805	-0,30333	3,183333
9	1,8		0,44	0,88	-0,32	3,2
10	1,9		0,385	0,945	-0,33	3,25
11	2		0,333333	1	-0,33333	3,333333
12	2,1		0,285	1,045	-0,33	3,45
13	2,2		0,24	1,08	-0,32	3,6
14	2,3		0,198333	1,105	-0,30333	3,783333
15	2,4		0,16	1,12	-0,28	4
16	2,5		0,125	1,125	-0,25	4,25
17	2,6		0,093333	1,12	-0,21333	4,533333
18	2,7		0,065	1,105	-0,17	4,85
19	2,8		0,04	1,08	-0,12	5,2
20	2,9		0,018333	1,045	-0,06333	5,583333
21	3	6	-3E-16	1	1,18E-15	6
22	3,1		-0,015	0,945	0,07	6,45
23	3,2		-0,02667	0,88	0,146667	6,933333
24	3,3		-0,035	0,805	0,23	7,45
25	3,4		-0,04	0,72	0,32	8
26	3,5		-0,04167	0,625	0,416667	8,583333
27	3,6		-0,04	0,52	0,52	9,2
28	3,7		-0,035	0,405	0,63	9,85
29	3,8		-0,02667	0,28	0,746667	10,53333
30	3,9		-0,015	0,145	0,87	11,25
31	4	12	4,44E-16	-4E-15	1	12
32						
33	x	valor observado	L0	L1	L2	p(x)

O código VBA para a interpolação de Lagrange pode ter a seguinte forma:

```
Option Explicit

Sub valor() ' Interpolação de Lagrange
 Dim x, y
 Dim z As Double ' valor de x cujo f(x)=y é buscado

    z = InputBox("Valor de x?") 'com vírgula, p.ex. 2,5
    x = Array(0, 1, 2, -2, -1)
    y = Array(-2, 6, 1, 3, 0)
    'MsgBox " n = " & Application.Count(x) & "UBound= " & UBound(x)
    MsgBox " f(" & z & ") = " & InterLagrange(x, y, z)

End Sub


Function InterLagrange(x, y, z)
    Dim i As Long, j As Long
    Dim n As Integer, l As Double, px As Double


    'Lagrange
    n = Application.Count(x) - 1
    px = 0
    For i = 0 To n - 1
       l = 1
       For j = 0 To n Step 1
         If i <> j Then
           l = l * (z - x(j)) / (x(i) - x(j))
         End If
       Next j
       px = px + l * y(i)
    Next i
        InterLagrange = px

End Function
```

No caso de muitos dados (experimentais), é preferível tê-los numa planilha:

H3 | =Lagrange(E3;$A$3:$A$6;$B$3:$B$6)

	A	B	C	D	E	F	G	H	I
1	Valores observados								
2	x	y							
3	-1	-2		valor_x=	2		y=	2	
4	0	1							
5	1	0							
6	2	2		**Interpolação de Lagrange**					
7									
8									

Na célula E3 inserimos o valor de x buscado e na célula H3 colocamos a fórmula =Lagrange(...) com a qual calculamos o valor de y correspondente. O programa a seguir utiliza a função MATCH do Excel.

Na versão portuguesa do Excel, escreve-se CORRESP com a sintaxe

CORRESP(valor_procurado;matriz_procurada;tipo_correspondência)

Valor_procurado pode ser um valor (número, texto ou valor lógico) ou uma referência de célula de um número, texto ou valor lógico.

Matriz_procurada é um intervalo contíguo de células que contêm valores possíveis de procura. Matriz_procurada precisa ser uma matriz ou uma referência de matriz. A Matriz_procurada deve ser posicionada em ordem ascendente.

Tipo_correspondência é o número -1, 0 ou 1. Se tipo_correspondência for 1, CORRESP localizará o maior valor que for menor do que ou igual a valor_ procurado.

Para poder usar MATCH num macro VBA, devemos introduzi-lo pelo objeto *Application*. Veja o capítulo 7 onde isso foi usado e explicado.

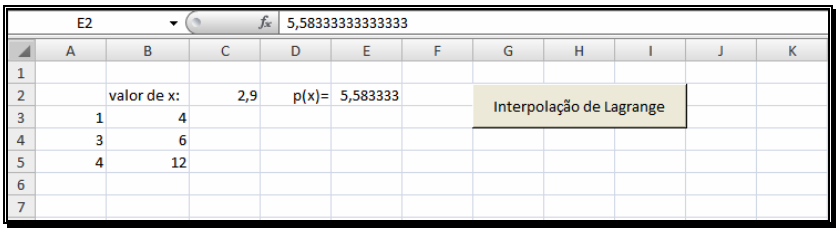
```
Option Explicit
Function Lagrange(valor_x, x, y)
'Basea-se no macro Intercep no livro "Excel for Scientists", p. 89, de E.J.Billo

Dim pos As Integer
Dim i As Integer, j As Integer
Dim l As Double, px As Double

'Evita extrapolação
If valor_x < Application.Min(x) Or valor_x > Application.Max(x) Then
    Lagrange = CVErr(xlErrRef): Exit Function
End If

pos = Application.Match(valor_x, x, 1) ' posição rel. de valor_x no array x

    If pos < 2 Then pos = 2
    If pos > x.Count - 2 Then pos = x.Count - 2

For i = pos - 1 To pos + 2
    l = 1
For j = pos - 1 To pos + 2
    If i <> j Then l = l * (valor_x - x(j)) / (x(i) - x(j)) ' pol. de Lagrange
Next j
    px = px + l * y(i)
Next i
Lagrange = px
End Function
```

Se buscarmos na lista x = {-2,-1,0,1,2,3,4} o valor 1,5, então a variável pos assumirá o valor 5 e i correrá de 4 até 7, ou seja, o programa utilizará os 4 valores 1, 2, 3, 4 para calcular um polinômio de terceiro grau. Trata-se de uma i**nterpolação cúbica**. (O programa não funciona com menos de 4 pontos, e sempre utiliza 4 pontos.)

Finalmente, aplicaremos o programa "Pontos" do último capítulo à interpolação de Lagrange:

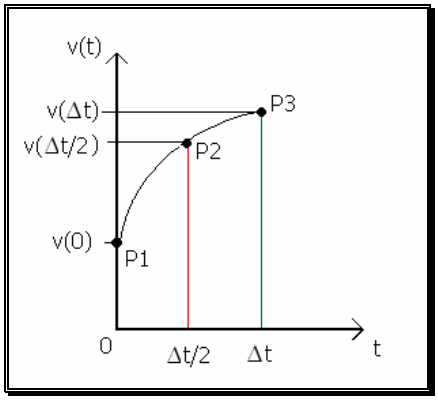
```
Private Sub InterLagrange()

    Dim R As Range '= objeto (aqui o conjunto de células selecionadas)
    Dim n As Integer, i As Integer, j As Integer, k As Integer
    Dim x() As Double   ' matriz dinâmica
    Dim y() As Double
    Dim px As Double, L As Double

    Set R = Selection
    n = R.Rows.Count     ' Número de linhas

    MsgBox "n= " & n
    ReDim x(n)
    ReDim y(n)

    valor_x = Cells(2, 3)

    If n > 1 Then  'tudo em orden, pois foram selecionadas as células
        For i = 1 To n
            x(i) = R(i, 1) 'lemos os dados das suas células
            y(i) = R(i, 2)
        Next i

        px = 0
        For k = 1 To n
            L = 1
            For j = 1 To n
               If j <> k Then
                 L = L * (valor_x - x(j)) / (x(k) - x(j))
               End If
            Next j
            px = px + L * y(k)
        Next k
        Cells(2, 5) = px ' p(valor_x) em D5

    Else   'janela de advertência: selecione dados!
        MsgBox ("Deve selecionar dados - ou faltam dados")
        Exit Sub  ' termina a sub-rotina se não tem dados
    End If
End Sub
```

Com os três pontos padrão (1;4), (3;6), (4;12) temos a seguinte planilha:



Algumas considerações teóricas

A regra de Simpson tem grande utilidade na matemática numérica. Mais à frente vamos usá-la, por exemplo, no contexto das equações diferenciais, discutindo os métodos de Runge-Kutta. Por isso será útil dar uma dedução da regra com a notação da mecânica.



Uma partícula se move em Δt segundos de um ponto x(0) até x(Δt). A figura mostra três valores da velocidade para três instantes 0, Δt/2 e Δt. A verdadeira curva da velocidade ignoramos, por isso a substituímos por uma parábola que passa por P1, P2, P3. A equação da parábola é

$$v(t) = At^2 + Bt + C$$

Devemos determinar os três coeficientes.

Para t = 0 temos $v(0) = C$ (1)

Para t = Δt/2 temos $v(\Delta t/2 = A\cdot(\Delta t/2)^2 + B\cdot\Delta t/2 + v(0)$ (2)

Para t = Δt temos $v(\Delta t) = A\cdot(\Delta t)^2 + B\cdot\Delta t + v(0)$ (3)

A seguinte função vai mostrar-se útil, a definimos da seguinte maneira

$$D := v(0) + 4v(\Delta t/2) + v(\Delta t) = 6C + 2A\cdot(\Delta t)^2 + 3B\cdot\Delta t \quad (4)$$

daí resulta $\Delta t\cdot D/6 = C\cdot\Delta t + A\cdot(\Delta t)^3/3 + B\cdot(\Delta t)^2/2$ (5)

Por outro lado temos

$$_0\int^{\Delta t} v(t)dt = {}_0\int^{\Delta t} (At^2+Bt+C)dt = A\cdot(\Delta t)^3/3 + B\cdot(\Delta t)^2/2 + C\cdot\Delta t$$

Fazendo uso de (4) e (5) resulta

$$_0\int^{\Delta t} v(t)dt = x(\Delta t) - x(0) = \Delta t\,[v(0) + 4v(\Delta t/2) + v(\Delta t)]/6 \quad (6)$$

Isto é a **primeira** regra de Simpson ou também conhecida como a regra do 1/6. (Às vezes fala-se da regra do 1/3, pois quando se utiliza h = Δt/2 como largura de um subintervalo, resulta $_0\int^{\Delta t} v(t)dt = h[v(0) + 4v(h) + v(2h)]/3$.)

A regra de Simpson que foi utilizada acima é a regra de Simpson **composta**, pois se subdivide o intervalo de integração [a,b] em n subintervalos iguais de largura h e *a cada par* de subintervalos aplica-se a 1ª regra de Simpson. (Como a regra de Simpson é aplicada em pares de subintervalos, o número n de subintervalos deve ser sempre par.)

Exemplo: Queremos calcular a integral da *seguridade estadística* definida por

$$S(x) = \int_{-x}^{x} f(t)dt \quad \text{com} \quad f(t) = \frac{1}{\sqrt{2\pi}} e^{\frac{-t^2}{2}}$$

por meio da regra de Simpson composta utilizando (6) repetidas vezes.

Solução:

Criamos uma planilha do Excel com o seguinte aspecto:

C10 =1/RAIZ(2*PI())*EXP(-A10*A10/2)

	A	B	C	D	E	F	G	H	I	J
1										
2	**Simpson**			a=	-2		n=	20		
3				b=	2		h=	0,2		
4							h/2=	0,1		
5										
6										
7										
8	**x**	**x-h/2**	**f(x)**	**f(x-h/2)**	**Área**					
9					**parcial**					
10	-2	-2,1	0,053991	0,043984			**Resultado:**			
11	-1,8	-1,9	0,07895	0,065616	0,01318015		0,95449961			
12	-1,6	-1,7	0,110921	0,094049	0,01886891					
13	-1,4	-1,5	0,149727	0,129518	0,02595729					
14	-1,2	-1,3	0,194186	0,171369	0,03431293					
15	-1	-1,1	0,241971	0,217852	0,04358552					
16	-0,8	-0,9	0,289692	0,266085	0,05320011					
17	-0,6	-0,7	0,333225	0,312254	0,06239773					
18	-0,4	-0,5	0,36827	0,352065	0,07032520					
19	-0,2	-0,3	0,391043	0,381388	0,07616214					
20	-2,8E-16	-0,1	0,398942	0,396953	0,07925984					

O mesmo resultado obtemos com o programa "Simpson". A fórmula do 1/6 fica em E11: =H$3*(C10+4*D11+C11)/6, copiar até E30. O resultado em G11 calculamos como =SOMA(E11:E30). B10: =A10-H$4, B11: =B10+H$3