PEC 4 Interpolación, derivación e integración numérica (I)

Alan Coila Bustinza

2022-03-22

1. Interpolación

Primero asignamos los valores de los intereses según la tabla y los intereses necesitados

```
ti1=1.44
ti2=1.47
ti3=1.50
ti4=1.51
ti5=1.54
ti6=1.57
ti7=1.60
ti8=1.61
ti9=1.73
ti10=1.79
T1=1.5
T2=10
T3=6.5*12
T4=9*12
```

Dado que contamos con 10 puntos dados en nuestra tabla podemos predecir que nuestro polinomio interpolador es de grado 10

1.1. Cálculo del polinomio interpolador

a. Método del determinante de Vandermonde

Dado un polinomio de interpolación

$$p(x) = a_n x^n + a_{n-1} x^{n-1} + ... + a_1 x + a_0$$

Para un conjunto de puntos (x_i, y_i) podemos obtener que:

$$y_i = a_n x_i^n + a_{n-1} x_i^{n-1} + ... + a_1 x_i + a_0$$

Esta ecuación puede ser escrita de la forma

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \dots & x_1^n \\ 1 & x_2 & x_2^2 & \dots & x_2^n \\ \vdots & \vdots & \vdots & \ddots & \vdots \\ 1 & x_n & x_n^2 & \dots & x_n^n \end{bmatrix} \begin{bmatrix} \beta_n \\ \beta_{n-1} \\ \vdots \\ \beta_0 \end{bmatrix} = \begin{bmatrix} y_n \\ y_{n-1} \\ \vdots \\ y_n \end{bmatrix}$$

Podemos crear una función auxiliar que mediante esta matriz de vandermonde nos devuelva los coeficientes del polinomio de interpolación

```
polyinterp=function(x,y)
{
  # comprobamos que la longitud de los vectores sea la misma
  if(length(x)!=length(y))
    stop ("La longitud de los vectores x e y debe ser la misma" )
  # calculamos el valor de n que es el grado del polinomio
  # a partir del numero de puntos menos 1
  n=length(x)-1
  # creamos la primera columna de la matriz de vandermonde
  vandermonde=rep(1,length(x))
  # iteramos para ir agregando las columnas siguientes según
  # el grado del polinomio
  for(i in 1:n)
  {
    # la matriz contiene columnas sucesivas de los valores de x
    # elevados a la nth potencia
    xi=x^i
    vandermonde=cbind(vandermonde,xi)
  }
  # resolvemos el sistema de ecuaciones
  beta=solve(vandermonde,y, tol=1e-22)

  # borramos los nombres de las columnas (xi)
  names(beta)=NULL
  # nos retorna los coeficientes del polinomio
  return(beta)
}
```

Aplicamos nuestra función a los puntos requeridos, esta nos devuelve los coeficientes del polinomio de interpolación

```
T=c(1,2,3,6,12,24,36,60,84,120)
r=c(1.44,1.47,1.50,1.51,1.54,1.57,1.60,1.61,1.73,1.79)
```

```
polyinterp(T,r)
```

```
##  [1]  1.436230e+00 -2.259148e-02  3.478131e-02 -9.425224e-03  1.062711e-03
##  [6] -5.872473e-05  1.700976e-06 -2.598483e-08  1.960488e-10 -5.700449e-13
```

Para evaluar los valores solicitados usaremos la regla de Horner

```
horner=function(x,coefs)
# esta función auxiliar recibe los argumentos:
#       x     :     puntos a evaluar
#       coefs :     los coeficientes del polinomio
{
  y=rep(0,length(x))
  for(i in length(coefs):1)
    y=coefs[i]+x*y

  return(y)
}
```

Calculamos los interés respectivos

```
interes<-c(T1,T2,T3,T4)

# realizamos la evaluación por el método de Horner
h=horner(interes,polyinterp(T,r))

# comprobamos los valores
for(i in 1:length(interes)){
  print(paste('Para ', interes[i], 'meses, el interes es', h[i]))
}
```

```
## [1] "Para  1.5 meses, el interes es 1.4537433022279"
## [1] "Para  10 meses, el interes es 1.47802305316286"
## [1] "Para  78 meses, el interes es -163.71151140132"
## [1] "Para  108 meses, el interes es 5348.83533153132"
```

b. Método del Polinomio interpolador de Lagrange

Dado que el polinomio interpolador de Lagrange se obtiene por la fórmula

$$p_m(x) = \sum_{k=0}^{m} f_k l_k(x), l_k(x) = \prod_{i \neq k} \frac{x - x_i}{x_k - x_i}, k = 0, ..., m$$

Dividiremos las operaciones de esta formula, por un lado calcularemos los polinomios de Lagrange, y posteriormente realizaremos la multiplicación y sumatoria.

Obtendremos primero los polinomios de Lagrange para un valor de x

```
polinomio_L<-function(x,T){

# Recibe los parámetros:
#   y : vector con las abscisas
#   L : un polinomio de Lagrange para un valor de x


  L=c(0)
  # iniciamos el polinomio para poder utilizarlo en la iteración
  for(i in 1:length(T)){
    L[i]=1
    for(j in 1:length(T)){
      # debemos tener en cuenta la condición de la formula de Lagrange
      # i debe ser diferente de j
      if (i!=j){
        # realizamos el productorio, con cada iteración se añade un factor
        L[i]=L[i]*((x-T[j])/(T[i]-T[j]))
      }
    }
  }
  return(L)
}
```

Toca realizar la sumatoria de los polinomios multiplicados por f_x , mediante otra función

```
p_interpolación<- function(y,L){
# Recibe los parámetros:
#   y : vector con las abscisas
#   L : un polinomio de lagrange para un valor de x
```

```
  p=0
  for(i in 1:length(y)){
    # realizamos la multiplicación de cada valor de los intereses en la tabla f(x)
    # con los polinomios de Lagrange obtenidos con la función
    p=p+(y[i]*L[i])
  }
  return(p)
}
```

Ahora calculamos los intereses solicitados para cada mes

```
for(i in interes){
    # para cada tiempo asignado a la variable interes en el ejercicio anterior
    t=i
    L=polinomio_L(t,T)
    pI=p_interpolación(r,L)
    print(paste('Para ',i , 'meses, el interes es', pI))
}
```

```
## [1] "Para  1.5 meses, el interes es 1.45374330222963"
## [1] "Para  10 meses, el interes es 1.47802305310533"
## [1] "Para  78 meses, el interes es -163.711511112579"
## [1] "Para  108 meses, el interes es 5348.83532235367"
```

1.2. Interpolación por tramos lineal

Creamos una función auxiliar para realizar la interpolación lineal

```
linterp=function(x1,y1,x2,y2){
  # calculamos la pendiente de la función lineal
  m=(y2-y1)/(x2-x1)
  # y el punto de corte en el eje de las ordenadas
  b=y2-m*x2
  return(c(b,m))
}
```

Ahora podemos aplicar nuestra función auxiliar para obtener los valores de la pendiente y el punto de corte y poder calcular el interes interpolado

```
p1=linterp(1,ti1,2,ti2)
# Sustituimos los valores obtenidos para obtener el valora de interes interpolado
# para un tiempo T1
r1=p1[[2]]*T1+p1[[1]]
r1
```

```
## [1] 1.455
```

```
# de forma similar para el resto de valores solicitados
p2=linterp(6,ti4,12,ti5)
r2=p2[[2]]*T2+p2[[1]]
r2
```

```
## [1] 1.53
```

```
p3=linterp(5*12,ti8,7*12,ti9)
r3=p3[[2]]*T3+p3[[1]]
r3
```

```
## [1] 1.7
```

```
p4=linterp(7*12,ti9,10*12,ti10)
r4=p4[[2]]*T4+p4[[1]]
r4
```

```
## [1] 1.77
```

1.3. Análisis de los resultados

Vemos que mediante los métodos de interpolación se obtiene algunos resultados extraños

```
for(i in interes){
    t=i
    L=polinomio_L(t,T)
    pI=p_interpolación(r,L)
    print(paste('Para ',i , 'meses, el interes es', pI))
}
```

```
## [1] "Para  1.5 meses, el interes es 1.45374330222963"
## [1] "Para  10 meses, el interes es 1.47802305310533"
## [1] "Para  78 meses, el interes es -163.711511112579"
## [1] "Para  108 meses, el interes es 5348.83532235367"
```

Inicialmente los valores podría considerarse aceptables, pero los valores negativos y extremadamente altos nos dan la sospecha de que, quizá el método de interpolación no es el correcto.

Esto sucede ya que nuestro polinomio es de alto grado por lo que esta presentando el fenómeno de Runge. El mejor método para el cálculo del polinomio para este ejercicio, es el método de interpolación por tramos.