Matemática: Una publicación del ICM – ESPOL
2016, Vol. 14, No. 1

EL TRAZADOR CÚBICO PARAMÉTRICO CERRADO Y EL CÁLCULO DE ÁREAS

Rodríguez Ojeda Luis [1]

Resumen. *En este artículo se propone una forma especial del trazador cúbico en forma paramétrica para describir figuras planas cerradas. Los polinomios por segmentos resultantes permiten calcular el área de la figura usando el teorema de Green. Como soporte para esta investigación se instrumentó una aplicación computacional para obtención de resultados. Estos resultados se comparan con figuras conocidas y se define un criterio de convergencia y precisión.*
Palabras clave: Interpolación paramétrica. Trazador cúbico. Teorema de Green.

Abstract. *In this article a special form of the cubic spline in parametric form is proposed to describe closed plane figures. The resulting polynomials by segments allow to calculate the area of the figure using the Green's theorem. As support for this research a computational application was implemented for obtaining results. These results are compared with known figures and a convergence criterion and precision is defined.*
Keywords: Parametric interpolation. Cubic spline. Green's theorem.
Recibido: Marzo 2016.
Aceptado: Abril 2016.

1. CURVAS PARAMÉTRICAS

Las curvas paramétricas se usan para expresar y graficar una relación entre dos variables que puede no ser de tipo funcional. Utilizando otra variable denominada parámetro, esta definición permite conocer información adicional acerca de la relación entre las variables y trazar gráficos más generales que los gráficos de funciones, proporcionando además su orientación.

2. INTERPOLACIÓN PARAMÉTRICA

Los métodos de interpolación permiten aproximar una función, de la cual se conocen algunos puntos (x_i, y_i), mediante otra función, típicamente un polinomio. Estos métodos no son aplicables si los datos no tienen una relación de tipo funcional y(x). Por lo tanto estos métodos no se pueden usar si los puntos pertenecen a una curva que tiene una forma general.

Sin embargo, si las coordenadas x_i, y_i se expresan como funciones de otra variable t denominada parámetro, entonces los puntos $x(t_i)$, $y(t_i)$ si tienen una relación funcional y con ellos se pueden construir polinomios de interpolación paramétricos: $\mathbf{T}_x(t)$, $\mathbf{T}_y(t)$. Estos polinomios separados no son de interés, pero en cambio el gráfico de puntos de estos polinomios paramétricos permite describir de manera adecuada a la curva.

Hay alguna libertad en la asignación de valores al parámetro t. Es suficiente que pertenezcan a un subconjunto ordenado de los reales **R**, con la misma cardinalidad que el conjunto de datos, y que sean asociados en forma consecutiva para cada punto $x(t_i)$, $y(t_i)$.

Después de construir los polinomios de interpolación paramétricos $\mathbf{T}_x(t)$ y $\mathbf{T}_y(t)$ y para que el gráfico se muestre como una curva contínua, se evalúan paralelamente los polinomios con valores de t muy cercanos, en el mismo dominio de t. Estos puntos se conectan para trazar la curva.

2.1 El trazador cúbico cerrado

El procedimiento de interpolación paramétrica se puede instrumentar con el trazador cúbico para obtener una aproximación cercana a la curva propuesta. En esta sección se propone una forma especial del trazador cúbico que permitirá construir el trazador cúbico paramétrico para modelar curvas cerradas.

Sean (t_i, u_i), i=0,1,2,3,...,n-1 puntos de una función real u: **R→R**, contínua y diferenciable en el dominio de los puntos dados.

Si se supone que u(t) es una función simple, se la puede representar aproximadamente mediante otra función **T(t)** definida en segmentos delimitados por los puntos dados:

$$\mathbf{T(t)} = \begin{cases} \mathbf{T_0(t)}, & \mathbf{t_0 \le t \le t_1} \\ \mathbf{T_1(t)}, & \mathbf{t_1 \le t \le t_2} \\ & \mathbf{...} \\ \mathbf{T_{n-2}(t)}, & \mathbf{t_{n-2} \le t \le t_{n-1}} \end{cases}$$

Por las propiedades de continuidad es conveniente que las funciones $\mathbf{T_i}$ sean polinomios segmentados de tercer grado. Para la formulación, se les asignan la forma general:

$$\mathbf{T_i(t) = a_i(t-t_i)^3 + b_i(t-t_i)^2 + c_i(t-t_i) + d_i}$$

; $t \in [t_i, t_{i+1}]$; i=0,1,2,...,n-2

En los puntos interiores deben coincidir la pendiente y la curvatura de los polinomios de intervalos adyacentes:

$$\mathbf{T_{i-1}^{(k)}(t_i) = T_i^{(k)}(t_i)}; \qquad i=1,2,...,n-2; \quad k=1,2$$

Adicionalmente, para instrumentar la forma cerrada del trazador cúbico, es necesario que la pendiente y curvatura en el punto final del polinomio en el último intervalo, coincidan con

[1] *Rodriguez Ojeda Luis., Profesor, Departamento de Matemáticas, Facultad de Ciencias Naturales y Matemáticas, ESPOL. (e_mail: lrodrig@espol.edu.ec).*

la pendiente y curvatura en el primer punto del polinomio en el primer intervalo.

$$T^{(k)}_{n-2}(t_{n-1}) = T^{(k)}_{0}(t_0);\quad k=1,2$$

Esta condición especial es parte de nuestra contribución en este artículo. Debido a esta condición, hemos designado a este conjunto de polinomios con el nombre de **trazador cúbico cerrado.**

2.2 Formulación del trazador cúbico cerrado

Dados los puntos (t_i, u_i), i=0,1,2,...,n-1. Sea $h_i = t_{i+1}-t_i$ el espaciamiento entre puntos adyacentes

Los polinomios segmentados del trazador cúbico y sus dos primeras derivadas:

$$T_i(t) = T_i = a_i(t-t_i)^3 + b_i(t-t_i)^2 + c_i(t-t_i) + d_i;$$

$t\in[t_i, t_{i+1}]$; i==0,1,2,...,n-2

$$T_i'(t) = D_i = 3a_i(t-t_i)^2 + 2b_i(t-t_i) + c_i$$

$$T_i''(t) = S_i = 6a_i(t-t_i) + 2b_i$$

Los polinomios del trazador cúbico y sus derivadas deben incluir a los puntos en cada intervalo:

$$T_i(t_i) = u(t_i) = u_i = a_i(t_i-t_i)^3 + b_i(t_i-t_i)^2 + c_i(t_i-t_i) + d_i \quad (1)$$
$$= d_i \quad \Rightarrow \quad d_i = u_i$$

$$T_i(t_{i+1}) = u(t_{i+1}) = u_{i+1} \quad (2)$$
$$= a_i(t_{i+1}-t_i)^3 + b_i(t_{i+1}-t_i)^2 + c_i(t_{i+1}-t_i) + d_i$$
$$= a_ih_i^3 + b_ih_i^2 + c_ih_i + d_i$$
$$\Rightarrow \quad u_{i+1} = a_ih_i^3 + b_ih_i^2 + c_ih_i + d_i$$

$$T_i''(t_i) = S_i = 6a_i(t_i-t_i) + 2b_i = 2b_i \quad \Rightarrow \quad b_i = \frac{S_i}{2} \quad (3)$$

$$T_i''(t_{i+1}) = S_{i+1} = 6a_i(t_{i+1}-t_i) + 2b_i = 6a_ih_i + 2b_i \quad (4)$$
$$\Rightarrow \quad a_i = \frac{S_{i+1}-S_i}{6h_i}$$

Sustituir (1), (3), y (4) en (2)

$$u_{i+1} = \frac{S_{i+1}-S_i}{6h_i}h_i^3 + \frac{S_i}{2}h_i^2 + c_ih_i + u_i \quad (5)$$
$$\Rightarrow \quad c_i = \frac{u_{i+1}-u_i}{h_i} - \frac{2h_iS_i + h_iS_{i+1}}{6}$$

Las fórmulas (1), (3), (4) y (5) definen a los coeficientes a_i, b_i, c_i, d_i, i=0,1,..., n-2 de cada polinomio segmentado mediante valores de las coordenadas de los datos y de valores de S_i. En la siguiente sección se desarrolla un dispositivo para encontrar los valores de S_i con los cuales se determinarán los coeficientes.

Primera derivada en el intervalo $[t_i, t_{i+1}]$:

$$T_i'(t_i) = 3a_i(t_i-t_i)^2 + 2b_i(t_i-t_i) + c_i = c_i$$

Primera derivada en el intervalo $[t_{i-1}, t_i]$:

$$T_{i-1}'(t_i) = 3a_{i-1}(t_i-t_{i-1})^2 + 2b_{i-1}(t_i-t_{i-1}) + c_{i-1}$$
$$= 3a_{i-1}h_{i-1}^2 + 2b_{i-1}h_{i-1} + c_{i-1}$$

Continuidad en la primera derivada entre polinomios adyacentes, en el punto común t_i:

$$T_{i-1}'(t_i) = T_i'(t_i) \quad \Rightarrow \quad 3a_{i-1}h_{i-1}^2 + 2b_{i-1}h_{i-1} + c_{i-1} = c_i$$
(6)

Sustituyendo (1), (3), (4) y (5) en (6) y simplificando:

$$h_{i-1}S_{i-1} + 2(h_{i-1}+h_i)S_i + h_iS_{i+1} = 6\left(\frac{u_{i+1}-u_i}{h_i} - \frac{u_i-u_{i-1}}{h_{i-1}}\right)$$

; i=1,2,3,...,n-3 (7)

Esta fórmula genera un sistema de n-3 ecuaciones con las variables S_0, S_1, ..., S_{n-2}

Dos ecuaciones adicionales se obtienen de las condiciones especiales del trazador cúbico cerrado:

Condición de continuidad de la primera derivada entre los segmentos inicial y final:

$$T_{n-2}'(t_{n-1}) = T_0'(t_0)$$

Primera derivada del primer segmento evaluada en el punto inicial:

$$T_0'(t_0) = 3a_0(t_0-t_0)^2 + 2b_0(t_0-t_0) + c_0 =$$
$$c_0 = \frac{u_1-u_0}{h_0} - \frac{2h_0S_0 + h_0S_1}{6}$$

Primera derivada del último segmento evaluada en el punto final:

$$T_{n-2}'(t_{n-1}) = 3a_{n-2}(t_{n-1}-t_{n-2})^2 + 2b_{n-2}(t_{n-1}-t_{n-2}) + c_{n-2} =$$
$$3a_{n-2}h_{n-2}^2 + 2b_{n-2}h_{n-2} + c_{n-2}$$
$$= 3\left(\frac{S_{n-1}-S_{n-2}}{6h_{n-2}}\right)h_{n-2}^2 + 2\left(\frac{S_{n-2}}{2}\right)h_{n-2} + \frac{u_{n-1}-u_{n-2}}{h_{n-2}} - \frac{2h_{n-2}S_{n-2} + h_{n-2}S_{n-1}}{6}$$

Igualando derivadas y simplificando, con $S_{n-1} = S_0$:

$$-\frac{1}{3}(h_0+h_{n-2})S_0 - \frac{1}{6}h_0S_1 - \frac{1}{6}h_{n-2}S_{n-2} \quad (8)$$
$$= -\frac{u_1-u_0}{h_0} + \frac{u_{n-1}-u_{n-2}}{h_{n-2}}$$

Condición de continuidad de la segunda derivada entre los segmentos inicial y final:

$$T_{n-2}''(t_{n-1}) = T_0''(t_0)$$

La ecuación (7) correspondiente al intervalo final n-2, con $S_{n-1} = S_0$

$$h_{n-3}S_{n-3} + 2(h_{n-3}+h_{n-2})S_{n-2} + h_{n-2}S_0 = \quad (9)$$
$$6\left(\frac{u_{n-1}-u_{n-2}}{h_{n-2}} - \frac{u_{n-2}-u_{n-3}}{h_{n-3}}\right)$$

Las ecuaciones (7), (8), (9) conforman un sistema completo de n-1 ecuaciones. Estas ecuaciones se pueden desarrollar y expresar en forma matricial mediante el sistema: **AS=B** en donde **S** es el vector de las variables: S_0, S_1, ..., S_{n-2}, mientras que la matriz **A** y el vector **B** se definen de la siguiente manera:

$$A = \begin{bmatrix} -\frac{1}{3}(h_0+h_{n-2}) & -\frac{1}{6}h_0 & 0 & 0 & .\ . & 0 & 0 & 0 & -\frac{1}{6}h_{n-2} \\ h_0 & 2(h_0+h_1) & h_1 & 0 & .\ . & 0 & 0 & 0 & 0 \\ 0 & h_1 & 2(h_0+h_1) & h_2 & .\ . & 0 & 0 & 0 & 0 \\ . & . & . & . & .\ . & . & . & . & . \\ . & . & . & . & .\ . & . & . & . & . \\ 0 & 0 & 0 & 0 & .\ . & h_{n-5} & 2(h_{n-5}+h_{n-4}) & h_{n-4} & 0 \\ 0 & 0 & 0 & 0 & .\ . & 0 & h_{n-4} & 2(h_{n-4}+h_{n-3}) & h_{n-3} \\ h_{n-2} & 0 & 0 & 0 & .\ . & 0 & 0 & h_{n-3} & 2(h_{n-3}+h_{n-2}) \end{bmatrix}$$

$$S = \begin{bmatrix} S_0 \\ S_1 \\ S_2 \\ . \\ . \\ S_{n-4} \\ S_{n-3} \\ S_{n-2} \end{bmatrix},$$

$$B = \begin{bmatrix} -\frac{u_1-u_0}{h_0} + \frac{u_{n-1}-u_{n-2}}{h_{n-2}} \\ 6(\frac{u_2-u_1}{h_1} - \frac{u_1-u_0}{h_0}) \\ 6(\frac{u_3-u_2}{h_2} - \frac{u_2-u_1}{h_1}) \\ . \\ . \\ 6(\frac{u_{n-3}-u_{n-4}}{h_{n-4}} - \frac{u_{n-4}-u_{n-5}}{h_{n-5}}) \\ 6(\frac{u_{n-2}-u_{n-3}}{h_{n-3}} - \frac{u_{n-3}-u_{n-4}}{h_{n-4}}) \\ 6(\frac{u_{n-1}-u_{n-2}}{h_{n-2}} - \frac{u_{n-2}-u_{n-3}}{h_{n-3}}) \end{bmatrix}$$

La solución de este sistema entrega los valores de S_0, S_1, ..., S_{n-2}, que al sustituir en las definiciones (1), (3), (4), y (5) proporcionan los coeficientes de cada polinomio segmentado del trazador cúbico cerrado:

$$T_i(t) = a_i(t-t_i)^3 + b_i(t-t_i)^2 + c_i(t-t_i) + d_i\,;$$

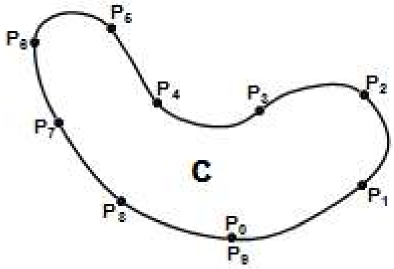
$t \in [t_i, t_{i+1}]$; i=0,1,2,...,n-2

En algunos coeficientes se hace referencia al valor de S_{n-1}, el cual con la definición establecida, debe sustituirse con el valor de S_0

2.3 El trazador cúbico paramétrico cerrado

Esta es una aplicación importante del trazador cúbico cerrado como dispositivo para modelar figuras cerradas en el plano y para calcular el valor del área de la región que encierra, la cual será tratada en una siguiente sección.

Sean P_0, P_1, P_2, ..., P_{n-1} puntos tomados en sentido antihorario de una figura cerrada **C** en el plano, con $P_{n-1}=P_0$, con coordenadas (x_i, y_i), i=0, 1, 2,..., n-1,



Estos puntos ya no pueden expresarse mediante una relación funcional $y(x)$, por lo que se debe usar un enfoque paramétrico.

Si se supone que **C** es una figura simple y regular por segmentos, se la puede representar aproximadamente en forma paramétrica mediante funciones x(t), y(t) definidas en segmentos delimitados por los puntos expresados con el parámetro t

$$P_i = \begin{cases} (t_i, x_i) \\ (t_i, y_i) \end{cases};\quad i=0, 1,2,..., n-1$$

En donde los valores de **t**: ti, i=0, 1, 2, ...,n-1 pertenecen a algún subconjunto ordenado de **R**.

Para aproximar la curva **C** en forma paramétrica una opción es elegir como funciones x(t), y(t), los polinomios del trazador cúbico cerrado. Siendo estos polinomios de tercer grado, la continuidad entre los segmentos puede llegar hasta la segunda derivada y el gráfico final será simple y suave:

$$C = \begin{cases} T_x(t) \\ T_y(t) \end{cases}$$

$T_x(t)$ se lo construye con los puntos base (t_i, x_i), i=0, 1, 2, ...,n-1

$$T_x(t) = \begin{cases} T_{x,0}(t), & t_0 \le t \le t_1 \\ T_{x,1}(t), & t_1 \le t \le t_2 \\ \quad ... & \\ T_{x,n-2}(t), & t_{n-2} \le t \le t_{n-1} \end{cases}$$

$T_y(t)$ se lo construye con los puntos base (t_i, y_i), i=0, 1, 2, ...,n-1

$$T_y(t) = \begin{cases} T_{y,0}(t), & t_0 \le t \le t_1 \\ T_{y,1}(t), & t_1 \le t \le t_2 \\ \quad ... & \\ T_{y,n-2}(t), & t_{n-2} \le t \le t_{n-1} \end{cases}$$

El gráfico de puntos de $T_x(t)$ y $T_x(t)$, $t\in[t_0, t_{n-1}]$, será una aproximación para la curva **C**.
El uso manual de estas funciones matemáticas involucra muchos cálculos y sería muy laborioso y susceptible a errores numéricos especialmente si la cantidad de puntos es grande, por esto se requiere un tratamiento computacional. Para este artículo se usó como soporte el lenguaje computacional Python disponible como software libre y por ofrecer facilidades para este tipo de aplicaciones.

2.4 Instrumentación computacional del trazador cúbico paramétrico cerrado

La función **trazador_cerrado** desarrollada en lenguaje Python se muestra al final de este documento. Esta función recibe separadamente los vectores **x, y** conteniendo las coordenadas de los puntos de la curva **C** que se desea modelar y los valores del parámetro t, y entrega cada uno de los polinomios segmentados del trazador cúbico paramétrico cerrado $\mathbf{T_x}$(t) y $\mathbf{T_x}$(t).

Si se incluye un vector adicional con puntos del parámetro, el resultado que entrega es un vector con puntos evaluados con los polinomios paramétricos segmentados. Si estos puntos son muy cercanos, se pueden usar para la graficación. Los polinomios que entrega $\mathbf{T_x}$**(t) y** $\mathbf{T_x}$(t) también pueden ser usados para calcular el área de la región que encierra, como se verá en otra sección de este artículo.

Para probar este método de aproximación se usará la figura de una circunferencia. Se realizarán varios intentos con diferentes cantidades de puntos para aproximar la forma de la circunferencia con polinomios segmentados. Posteriormente se comparará el área del círculo con el área de la región delimitada por los polinomios.

Ejemplo. Dados 4 puntos equidistantes de la circunferencia de un círculo de radio unitario centrado en el origen, encontrar los polinomios segmentados del trazado cúbico parametrizado y obtener un vector con puntos evaluados en los polinomios segmentados para dibujar y aproximar el gráfico del círculo.

Puntos de la circunferencia: (0, -1), (1, 0), (0, 1), (-1, 0). Para cerrar la figura se agregará al final el primer punto.

Interacción en la ventana de Python con la función **trazador_cerrado**. Se muestran las instrucciones y los resultados obtenidos. A la derecha se escriben comentarios acerca de las instrucciones y resultados.

`>>> from trazador_cerrado import*`	Carga del módulo con el trazador cúbico:
`>>> x=[0,1,0,-1,0]`	Coordenadas x
`>>> y=[-1,0,1,0,-1]`	Coordenadas y
`>>> t=[1,2,3,4,5]`	Puntos del parámetro
`>>> Tx=trazador_cerrado(t,x)`	Obtención de los polinomios $T_x(t)$
`>>> Ty=trazador_cerrado(t,y)`	Obtención de los polinomios $T_y(t)$
`>>> Tx[0]`	
`-0.5*t**3 + 1.5*t**2 - 1.0`	Polinomio $T_{x,0}(t)$
`>>> Ty[0]`	
`-0.5*t**3 + 3.0*t**2 - 4.5*t + 1.0`	Polinomio $T_{y,0}(t)$
`>>> Tx[1]`	
`0.5*t**3 - 4.5*t**2 + 12.0*t - 9.0`	Polinomio $T_{x,1}(t)$
`>>> Ty[1]`	
`-0.5*t**3 + 3.0*t**2 - 4.5*t + 1.0`	Polinomio $T_{y,1}(t)$
`>>> Tx[2]`	
`0.5*t^3 - 4.5*t^2 + 12.0*t - 9.0`	Polinomio $T_{x,2}(t)$
`>>> Ty[2]`	
`0.5*t**3 - 6.0*t**2 + 22.5*t - 26.0`	Polinomio $T_{y,2}(t)$
`>>> Tx[3]`	
`-0.5*t**3 + 7.5*t**2 - 36.0*t + 55.0`	Polinomio $T_{x,3}(t)$
`>>> Ty[3]`	
`0.5*t**3 - 6.0*t**2 + 22.5*t - 26.0`	Polinomio $T_{y,3}(t)$
`>>> from pylab import*`	Librería gráfica
`>>> s=arange(1,5,0.01)`	Puntos para evaluar al trazador cúbico
`>>> Txs=trazador_cerrado(t,x,s)`	Abscisas del trazador cúbico
`>>> Tys=trazador_cerrado(t,y,s)`	Ordenadas del trazador cúbico
`>>> plot(Txs,Tys,'-')`	Gráfico de los puntos del trazador
`>>> plot(x,y,'o')`	Gráfico de los 4 puntos base del círculo
`>>> grid(True)`	Motrar cuadrículas
`>>> show()`	Mostrar gráfico

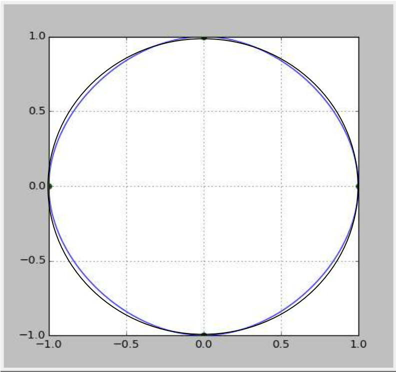


Gráfico de la circunferencia y del trazador cúbico paramétrico cerrado con 4 puntos de la circunferencia. Existen diferencias significativas entre los gráficos.

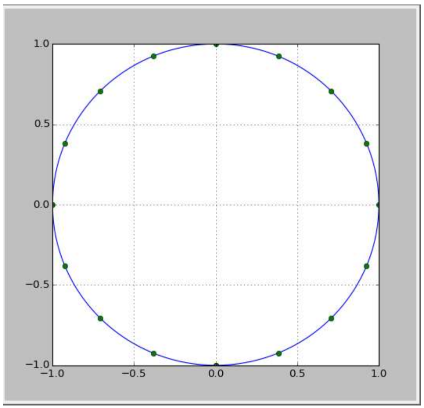


Gráfico del trazador cúbico paramétrico cerrado con 16 puntos de la circunferencia. El gráfico del trazador cúbico es muy cercano al gráfico de la circunferencia

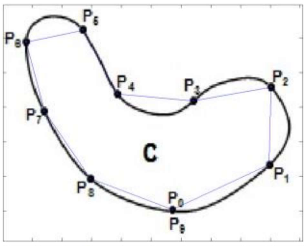
Ejemplo. Dados 16 puntos equidistantes de la circunferencia de un círculo de radio unitario centrado en el origen, usar el trazador cúbico parametrizado para obtener un vector con puntos evaluados en los polinomios segmentados y dibujarlo para aproximar el gráfico del círculo. Como en el ejemplo anterior se agrega al final el primer punto para cerrar la figura.

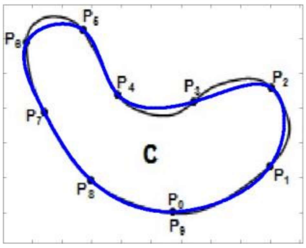
```
>>> from trazador_cerrado import*
>>> from pylab import*
>>> a1=cos(3*pi/8); a2=cos(pi/4);
        a3=cos(pi/8)
>>> b1=sin(pi/8); b2=sin(pi/4);
        b3=sin(3*pi/8)
>>>
 x=[0,a1,a2,a3,1,a3,a2,a1,0,-a1,-a2,-a3,-1,-a3,-
        a2,-a1,0]
>>>
 y=[-1,-b3,-b2,-b1,0,b1,b2,b3,1,b3,b2,b1,0,-
        b1,-b2,-b3,-1]
>>>
t=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17]
>>> s=arange(1,17,0.01)
>>> Txs=trazador_cerrado(t,x,s)
>>> Tys=trazador_cerrado(t,y,s)
>>> plot(Txs,Tys,'-')
>>> plot(x,y,'o')
>>> grid(True)
>>> show()
```

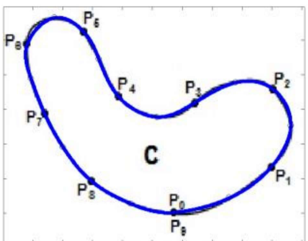
2.5 Aplicación del trazador cúbico paramétrico cerrado al modelado de curvas cerradas planas

El trazador cúbico paramétrico cerrado puede usarse para describir curvas de las cuales solamente se conocen algunos puntos de referencia. Estos puntos pueden ser resultados de observaciones o datos de diseño.

Para este artículo se usó un dispositivo computacional de captura de datos para registrar los 9 puntos del gráfico de la figura de la Sección 2.3 de este artículo. Se muestran los resultados obtenidos







En el primer gráfico se han unido los puntos con segmentos de recta como una primera aproximación
En el segundo gráfico se ha colocado el trazador paramétrico cerrado con los 9 puntos disponibles.
En el tercer gráfico se ha colocado el trazador paramétrico cerrado usando los 9 puntos disponibles y tomado 9 puntos intermedios adicionales de la curva. La aproximación es aceptable.

3. CÁLCULO DE ÁREAS DE REGIONES PLANAS

El trazador cúbico paramétrico cerrado es un dispositivo para modelar figuras cerradas en el plano y puede ser usado para calcular en forma aproximada el valor del área de la región que encierra.

3.1 Cálculo del área de una región poligonal en el plano cartesiano

Sea P un polígono de n lados cuyos vértices numerados en sentido antihorario son:

$P_0(x_0, y_0)$, $P_1(x_1, y_1)$, $P_2(x_2, y_2)$, . . ., $P_{n-1}(x_{n-1}, y_{n-1})$

Entonces el área de la región poligonal S correspondiente, está dada por la siguiente expresión atribuida al matemático Gauss :

$$S = \frac{1}{2}\left(\begin{vmatrix} x_0 & x_1 \\ y_0 & y_1 \end{vmatrix} + \begin{vmatrix} x_1 & x_2 \\ y_1 & y_2 \end{vmatrix} + \ldots + \begin{vmatrix} x_{n-1} & x_0 \\ y_{n-1} & y_0 \end{vmatrix} \right)$$

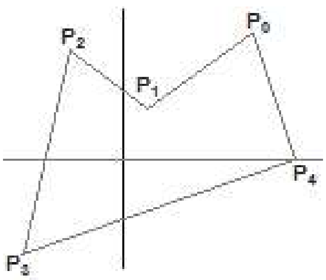
$$= \frac{1}{2}\left| \sum_{i=0}^{n-2} x_i y_{i+1} + x_{n-1} y_0 - \sum_{i=0}^{n-2} x_{i+1} y_i - x_0 y_{n-1} \right|$$

En donde
S es el área de la región poligonal
n es la cantidad de lados del polígono
(x_i, y_i), $i = 0, 1, 2, \ldots, n-1$ son los n vértices del polígono.

Ejemplo. Calcular el área de la región poligonal cuyos vértices son:
$P_0(8,7)$, $P_1(1,3)$, $P_2(-2,6)$, $P_3(-5, -4)$, $P_4(9,0)$:
Con la fórmula de Gauss:

$$S = \frac{1}{2} \mid (8)(3) + (1)(6) + (-2)(-4) + (-5)(0) + (9)(7) - (1)(7) - (-2)(3) - (-5)(6) - (9)(-4) - (8)(0) \mid$$

$$= 83.0$$



3.2 Cálculo computacional del área de una región poligonal

La función **agauss** desarrollada al final de este documento recibe los vectores **x, y** conteniendo las coordenadas de los vértices de un polígono en el plano y entrega el área de la región poligonal calculada con la fórmula de Gauss.

Ejemplo. Calcular con la función **agauss** el área de la región poligonal definida con los vértices:
$P_0(8,7)$, $P_1(1,3)$, $P_2(-2,6)$, $P_3(-5, -4)$, $P_4(9,0)$:

```
>>> from agauss import*
>>> x=[8,1,-2,-5,9]
>>> y=[7,3,6,-4,0]
>>> s=agauss(x,y)
>>> s
83.0
```

3.3 Cálculo de áreas de regiones planas con el teorema de Green

El teorema de Green puede utilizarse para calcular un integral de línea mediante un integral doble o para realizar el cálculo del área de una región mediante el integral de línea. Esta última aplicación es de interés en este artículo.

Sea C una curva parametrizada en el plano, cerrada y simple. Sea S la región del plano determinada por C y su interior. Si $M, N : R^2 \to R$ son funciones reales continuas y que tienen derivadas parciales continuas en toda la región S, entonces el siguiente enunciado es el teorema de Green:

$$\oint_C M(x,y)dx + N(x,y)dy = \iint_S \left(\frac{\partial N(x,y)}{\partial x} - \frac{\partial M(x,y)}{\partial y}\right)dA$$

Para aplicar este teorema en el cálculo del área de una región cerrada se toman las funciones simples: $M(x,y) = -y$, $N(x,y) = x$ en el enunciado del teorema. Entonces se obtienen:

$$\frac{\partial M(x,y)}{\partial y} = -1, \quad \frac{\partial N(x,y)}{\partial x} = 1$$

Reemplazando en el teorema de Green se obtiene una fórmula para calcular el área de una región en el plano mediante el integral de línea de la curva parametrizada que la encierra:

$$A = \frac{1}{2}\oint_C x dy - y dx$$

3.4 Cálculo del área de una región delimitada con el trazador cúbico paramétrico cerrado usando el teorema de Green.

Si C está definida con los polinomios del trazador cúbico paramétrico cerrado:

$$C = \begin{cases} T_x(t) \\ T_y(t) \end{cases}$$

Entonces se puede calcular el área de la región C delimitada por el trazador cúbico paramétrico cerrado con la fórmula de Green:

$$A = \frac{1}{2}\oint_C T_x(t)\frac{dT_y(t)}{dt} - T_y(t)\frac{dT_x(t)}{dt}$$

En donde $T_x(t)$ y $T_y(t)$ son los polinomios segmentados del trazador cúbico paramétrico cerrado de la Sección 2.3

3.5 Cálculo computacional del área de una región cerrada con el teorema de Green

La función **green** desarrollada al final de este documento recibe los polinomios segmentados $T_x(t)$ y $T_y(t)$ del trazador cúbico paramétrico cerrado y los puntos del parámetro **t** que definen los extremos del intervalo de cada polinomio segmentado y entrega el área de la región cerrada calculada con la fórmula de Green. En el siguiente ejemplo se compara el resultado del **método propuesto, con el valor exacto de una figura conocida.**

Ejemplo. Calcular aproximadamente con la función **green** el área de un círculo unitario centrado en el origen tomando 16 puntos equidistantes de la circunferencia en sentido antihorario. Para cerrar la figura se agrega al final el primer punto.

```
>>> from trazador_cerrado import*
>>> a1=cos(3*pi/8); a2=cos(pi/4);
a3=cos(pi/8)
>>> b1=sin(pi/8); b2=sin(pi/4);
b3=sin(3*pi/8)
>>> x=[0,a1,a2,a3,1,a3,a2,a1,0,-a1,-a2,-a3,-1,-
a3,-a2,-a1,0]
>>> y=[-1,-b3,-b2,-b1,0,b1,b2,b3,1,b3,b2,b1,
0,-b1,-b2,-b3,-1]
>>>
t=[1,2,3,4,5,6,7,8,9,10,11,12,13,14,15,16,17]
>>> Tx=trazador_cerrado(t,x)
>>> Ty=trazador_cerrado(t,y)
>>> from green import*
>>> A=green(Tx,Ty,t)
>>> A
3.14137
```

Este resultado difiere del valor exacto π, en aproximadamente 0.0002

Si los puntos provienen de una curva cerrada arbitraria, entonces el resultado de la aplicación de la fórmula de Green dependerá de la buena aproximación a la curva que proporciona el trazador cúbico paramétrico cerrado.

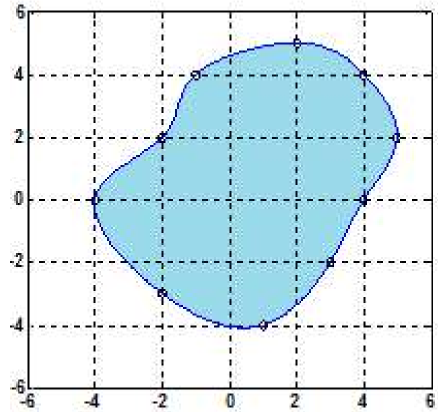
En la práctica, el área de una curva cerrada puede hacerse mediante un dispositivo denominado planímetro, pero este método no se puede aplicar si solamente se conocen puntos de la curva. En este caso primero debe construirse la curva para lo cual el trazador cúbico paramétrico cerrado es una opción adecuada.

Ejemplo. Colocar el trazador cúbico paramétrico cerrado sobre los siguientes diez puntos y calcular con el teorema de Green el valor del área de la región encerrada:

$P_0(4,0)$, $P_1(5,2)$, $P_2(4,4)$, $P_3(2,5)$, $P_4(-1,4)$, $P_5(-2,2)$, $P_6(-4,1)$, $P_7(-2,-3)$, $P_8(1,-4)$, $P_9(3,-2)$

El punto inicial se debe repetir al final para cerrar la figura

```
>>> from trazador_cerrado import*
>>> x=[4,5,4,2,-1,-2,-4,-2,1,3,4]
>>> y=[0,2,4,5,4,2,0,-3,-4,-2,0]
>>> t=[1,2,3,4,5,6,7,8,9,10,11]
>>> from pylab import*
>>> s=arange(1,11,0.01)
>>> Txs=trazador_cerrado(t,x,s)
>>> Tys=trazador_cerrado(t,y,s)
>>> plot(x,y,'o')
>>> plot(Txs,Tys,'-')
>>> show()
>>> from green import*
>>> Tx=trazador_cerrado(t,x)
>>> Ty=trazador_cerrado(t,y)
>>> A=green(Tx,Ty,t)
>>> A
54.0402   Valor del área de la región encerrada
```



4. CONVERGENCIA Y PRECISIÓN

Una figura poligonal cerrada puede usarse como una aproximación para una figura plana cerrada. En el límite, con la cantidad suficiente de vértices, ambas figuras tienden a coincidir. Por lo tanto la fórmula geométrica del área de Gauss pudiera usarse como una aproximación para calcular el área de la figura cerrada plana. El trazador cúbico paramétrico cerrado es una aproximación mucho mejor debido a que usa segmentos curvos para aproximar los segmentos curvos de la curva de la cual provienen los datos manteniendo una conexión suave entre segmentos. Por lo tanto se requerirá una cantidad menor de puntos para llegar a una precisión aceptable.

Los puntos o el gráfico que se van a estudiar deben digitalizarse para facilitar el uso de algún dispositivo computacional para captura de datos. La precisión de la representación de la curva, y consecuentemente del cálculo del área encerrada dependerá de la precisión y la cantidad de puntos disponibles.

Para calcular el área, las coordenadas de los puntos deben estar en la escala real. Si se desea calcular el área de una figura de la cual se toman puntos, para llevar a la escala real, se puede usar como referencia algún cuadro del dibujo que tenga la escala real. Con la fórmula del área de una región poligonal se puede calcular el área de este cuadro. Posteriormente, con la fórmula de Green se calcula el valor del área de la región de interés y con una relación directa se lo puede llevar al valor del área en la escala real.

5. CONCLUSIONES

En este artículo se ha tratado un problema clásico en el que se han vinculado enunciados y fórmulas matemáticas con instrumentos computacionales de libre acceso. Esta combinación tiene mucha importancia pues permite utilizar el desarrollo matemático en la investigación y en la resolución práctica de problemas de aplicación.

Los resultados que se obtuvieron en las pruebas son satisfactorios y permiten establecer criterios para la precisión.

La instrumentación computacional usa como soporte el lenguaje Python, el cual por ser software de uso libre no requiere licencia. Este lenguaje tiene características adecuadas para manejo matemático numérico, simbólico y gráfico siendo además muy simple de usar.

6. BIBLIOGRAFÍA BÁSICA

[1]. Análisis Numérico Básico Rodríguez Ojeda, Luis Libro digital disponible en la FCNM, ESPOL, 2014 http://www.fcnm.espol.edu.ec/publicaiones

[2]. Python Programación Rodríguez Ojeda, Luis Libro digital disponible en la FCNM, ESPOL, 2014 http://www.fcnm.espol.edu.ec/publicaciones

[3]. Teorema de Green https://es.khanacademy.org/math/multivariable-calculus/line-integrals-topic/greens_theorem/

[4]. Parametric Spline Curves http://folk.uio.no/in329/nchap6.pdf

[5]. Parametric curves and surfaces http://www.robots.ox.ac.uk/~ian/Teaching/CompGeom/lec4.pdf

Bibliografía especializada de la red internet

[6]. An approach to Data Parametrization in Parametric Cubic Spline Interpolation Problems,
Samuel P. Marin
Journal of Aproximation Theory, 64-86 (1984)

[7]. On the deviation of parametric cubic spline interpolant from its data polygon
Michael S. Floater
Computer Aided Geometric Design 25 148-156 (2008)

[8]. Choosing nodes in parametric curve interpolation
E.T.Y. Lee
Computer Aided Design, 363-370 (1989)

[9]. Mathematics Behind Planimeters
Osman Yardimci, 2013
https://etd.auburn.edu/bitstream/handle/10415/3667/Mathematics%20Behind%20Planimeters.pdf?sequence=2

7. INSTRUMENTACIÓN COMPUTACIONAL

7.1 Instrumentación computacional en Python del trazador cúbico cerrado

```
import numpy as np
from sympy import*
def trazador_cerrado(z,u,s=[]):
# Trazador cúbico cerrado para modelar una función u(z)
# Para uso paramétrico deben ingresar separadamente las funciones  x(t), y(t)
# El punto inicial debe coincidir con el punto final para cerrar la figura
# Entrega los polinomios segmentados en una lista de celdas T(t)
# Si hay un tercer parámetro conteniendo un vector de valores
# entrega un vector con los resultados evaluados con el trazador
    n=len(z)
    h=np.zeros([n-1])
    A=np.zeros([n-1,n-1]);B=np.zeros([n-1]);S=np.zeros([n])
    a=np.zeros([n-1]);b=np.zeros([n-1]);c=np.zeros([n-1]);d=np.zeros([n-1])
    if n<3:
        T=[]
        return
    for i in range(n-1):
        h[i]=z[i+1]-z[i]
    A[0][0]=-1/3*(h[0]+h[n-2])                   #Construir el sistema de ecuaciones
    A[0][1]=-1/6*h[0]
    A[0][n-2]=-1/6*h[n-2]
    B[0]=-(u[1]-u[0])/h[0]+(u[n-1]-u[n-2])/h[n-2]
    for i in range(1,n-2):
        A[i][i-1]=h[i-1]
        A[i][i]=2*(h[i-1]+h[i])
        A[i][i+1]=h[i]
        B[i]=6*((u[i+1]-u[i])/h[i]-(u[i]-u[i-1])/h[i-1])
    A[n-2][0]=h[n-2]
    A[n-2][n-3]=h[n-3]
    A[n-2][n-2]=2*(h[n-3]+h[n-2])
    B[n-2]=6*((u[n-1]-u[n-2])/h[n-2]-(u[n-2]-u[n-3])/h[n-3])
    r=np.linalg.solve(A,B)                       #Resolver el sistema
    for i in range(n-1):
        S[i]=r[i]
    S[n-1]=r[0]
    for i in range(n-1):                         #Coeficientes de los polinomios
        a[i]=(S[i+1]-S[i])/(6*h[i])
        b[i]=S[i]/2
        c[i]=(u[i+1]-u[i])/h[i]-(2*h[i]*S[i]+h[i]*S[i+1])/6
        d[i]=u[i]
    try:
        if len(s)==0:                            #Detecta si es un vector
            pass
    except TypeError:
        s=[s]
    if len(s)==0:                                #Construir el trazador
        t=Symbol('t')
        T=[]
        for i in range(n-1):
            p=expand(a[i]*(t-z[i])**3+b[i]*(t-z[i])**2+c[i]*(t-z[i])+d[i])
            T=T+[p]
        return T                                 #Retorna los polinomios
    else:                                        #Evaluar el trazador
        m=len(s)
        q=np.zeros([m])
```

```
    for k in range(m):
       t=s[k]
       for i in range(n-1):
          if t>=z[i] and t<=z[i+1]:
            q[k]=a[i]*(t-z[i])**3+b[i]*(t-z[i])**2+c[i]*(t-z[i])+d[i]
    if m>2:
      k=m-1
      i=n-2
      q[k]=a[i]*(t-z[i])**3+b[i]*(t-z[i])**2+c[i]*(t-z[i])+d[i]
    if len(q)==1:
      return q[0]                         #Retorna un valor
    else:
      return q                            #Retorna un vector
```

7.2 Instrumentación computacional en Python de la fórmula de Gauss para calcular el área de una región poligonal cerrada

```
def agauss(x,y):
# Cálculo del área de una región poligonal
# con la fórmula de Gauss
   n=len(x)
   s=0
   for i in range(n-1):
      s=s+x[i]*y[i+1]
   s=s+x[n-1]*y[0]
   for i in range(n-1):
      s=s-x[i+1]*y[i]
   s=s-x[0]*y[n-1]
   return abs(0.5*s)
```

7.3 Instrumentación computacional en Python del teorema de Green para calcular el área de una figura plana cerrada

```
from sympy import*
def green(Tx,Ty,s):
# Cálculo del área de una región cerrada
# con el teorema de Green
   t=Symbol('t')
   n=len(Tx)
   r=0
   for i in range(n):
      x=Tx[i]
      dx=diff(x,t)
      y=Ty[i]
      dy=diff(y,t)
      r=r+integrate(x*dy-y*dx,(t,s[i],s[i+1]))
   return 0.5*r
```