Preliminares matemáticos

Facultad de Ciencias de la Electrónica

Benemérita Universidad Autónoma de Puebla

Licenciatura en Ingeniería Mecatrónica

Dr. Fernando Reyes Cortés

Robótica

ftp://ece.buap.mx/pub/profesor/FernandoReyes/Robotica/

Primavera 2020

Parte VI

Integración numérica: $I(t) = \int_0^t f(\sigma)d\sigma$

Contenido

Robótica FCE

BUAP

Figura 1: Gottfried Wilhelm von Leibniz (1646-1716).

Figura 2: Firma de Leibniz.

Notación matemática:

$$I(t) = \int_0^t f(\sigma) d\sigma$$

$$\dot{\boldsymbol{x}} = \frac{d}{dt}\boldsymbol{x}$$

$$\dot{V}(\boldsymbol{x}) = \left[\frac{\partial V(\boldsymbol{x})}{\partial \boldsymbol{x}}\right]^T \dot{\boldsymbol{x}}$$

Integración numérica

Integración numérica

Es la técnica de aproximar integrales de funciones.

- La integración de funciones es un tópico especial y de interés en aplicaciones prácticas para ingeniería mecatrónica y robótica.
- Desde el punto de vista analítico y experimental las integrales representan una herramienta fundamental en el análisis y diseño de algoritmos de control, desarrollo y construcción de robots manipuladores.
- La aplicación inmediata de integración numérica se encuentra en el diseño de simuladores, la calidad de los resultados depende en gran medida de la exactitud para aproximar a una integral de función.

Esquemas de control bien conocidos que se emplean en robots manipuladores como es el caso del proporcional integral derivativo (PID) involucra la integral del error de posición para mejorar las características operativas de la estrategia de control

$$\boldsymbol{\tau} \quad = \quad K_p\tilde{\boldsymbol{q}}(t) + K_i \int_{t_0}^{t} \tilde{\boldsymbol{q}}(\sigma)\, d\sigma - K_v\dot{\boldsymbol{q}}(t)$$

La acción de control integral almacena energía (área bajo la curva del error de posición), y mediante una adecuada sintonía en la ganancia integral influye sobre la cantidad de energía aplicada al robot, repercutiendo en la respuesta transitoria y reduciendo el error en estado estacionario.

Robótica FCE

BUAP

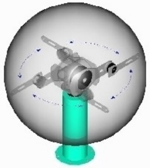
Es necesario remarcar que en el caso del esquema PID, la acción de control integral se realiza en cada instante de tiempo conforme el tiempo evoluciona. Es decir, para cada valor de t se calcula la integral, de tal forma que no significa realizar la integral una sola vez, más bien es un cálculo continuo, uno tras otro como el tiempo evolucione.

En contraste con otros métodos, el cálculo de la integral sólo se requiere una vez, tal es el caso de la medición de desempeño de reguladores usando la norma $\mathcal{L}_2[\tilde{\boldsymbol{q}}]$, cuya expresión matemática está dada por:

$$\mathcal{L}_2[\tilde{\boldsymbol{q}}] = \sqrt{\frac{1}{T}\int_0^T \|\tilde{\boldsymbol{q}}(\sigma)\|^2 \, d\sigma}$$

donde T representa el tiempo de experimentación o simulación (intervalo de integración).

Una magnitud pequeña en la norma $\mathcal{L}_2$ significa alto desempeño del esquema de control. Esto significa que el área bajo la curva del error de posición no fue significativa debido a que el robot se posicionó de manera inmediata, logrando la convergencia hacia cero del error de posición $\tilde{\boldsymbol{q}}(t) \to \boldsymbol{0}$. Por lo tanto, en el periodo de integración el transitorio fue rápido, sin sobretiros y el desempeño del algoritmo de control es muy bueno. En contra parte, un alto valor de la norma $\mathcal{L}_2$ representa pobre desempeño. La norma $\mathcal{L}_2$ es muy útil cuando se compara varios algoritmos de control, entonces se determina cuál de ese conjunto de esquemas tiene mejores prestaciones y por lo tanto determina la selección para una aplicación específica.



Robótica FCE

BUAP

La adecuada interpretación de resultados experimentales, analíticos y funcionamiento cualitativo de estrategias de control requiere del buen entendimiento de tópicos específicos de cálculo integral y diferencial. Más aún, cómo implementar una integral, o seleccionar el método numérico más adecuado son conocimientos fundamentales que se ven reflejados en la exactitud de posicionamiento de un robot manipulador y en el desempeño del esquema de control.

La integral definida de una función $f(x)$ sobre un intervalo finito $[a, b]$ es interpretada como el área sobre la curva de $f(x)$ como se muestra en la figura 3. Para varias funciones la integral se puede obtener en forma analítica. Sin embargo, para un tipo de funciones su integral no se puede obtener por medios analíticos, y por lo tanto se requiere de métodos numéricos para estimar su valor.

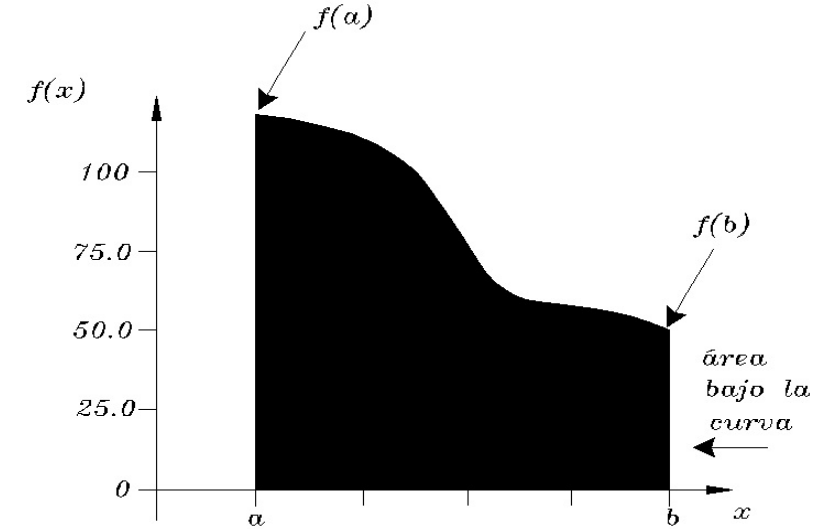


Figura 3: La integral de $f(x)$ significa el área bajo la curva.

Cómo realizar el procedimiento de integración numérica del sistema dinámico?

$$\dot{\boldsymbol{x}} \;=\; \boldsymbol{f}(\boldsymbol{x}) \;\Longrightarrow\; \boldsymbol{x}(t) = \int_0^t \boldsymbol{f}(\boldsymbol{x}(\sigma))d\sigma$$

Hay varias técnicas para aproximar la integral de una función $\boldsymbol{f}(\boldsymbol{x})$; la evaluación numérica de una integral se le denomina ***cuadratura*** cuyo nombre proviene de un problema geométrico ancestral.

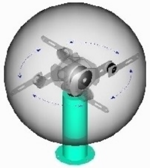
A continuación se presentan dos métodos geométricos ampliamente conocidos: método trapezoidal y regla de Simpson.

Regla trapezoidal

Cuando el área bajo la curva de la función $f(x)$ es representada por trapezoides y el intervalo $[a, b]$ es dividido en n secciones del mismo tamaño, entonces el área de $f(x)$ puede ser aproximada por la siguiente expresión:

$$I_T \;=\; \frac{b-a}{2n}\left[f(x_0) + 2f(x_1) + 2f(x_2) + \cdots + 2f(x_{n-1}) + f(x_n)\right] \qquad (1)$$

donde los valores x_i representan los valores finales de los trapecios, para $i = 1, 2, \cdots, n-1$; $x_0 = a$ y $x_n = b$.



Robótica FCE

BUAP

MATLAB tiene la función `trapz` para aproximar la integral de una función $f(x)$ por el método trapezoidal, cuya sintaxis es la siguiente:

```
y=trapz(f)
```

```
y=trapz(x,f)
```

```
y=trapz(x,f,dim)
```

- La función `trapz(f)` calcula la aproximación numérica de la integral de la función `f` por el método trapezoidal.
- Si `f` es un vector, entonces `y=trapz(f)` es la integral de `f`. Cuando `f` representa a una matriz `y=trapz(f)` retorna un vector renglón con la integral sobre cada columna.
- Cuando la función `trapz` tiene la sintaxis `y=trapz(x,f)` realiza la integral de f con respecto a `x`.
- Para el caso `y=trapz(x,f, dim)`, `dim` es un escalar que indica la dimensión de la función `f`. En ambos casos, el vector `x` debe tener la misma dimensión `dim`.

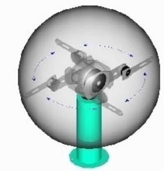
Mayor información en la ventana de comandos de **MATLAB**:

fx `>> help trapz` ↩

Ejemplo 1.1:

Calcular la integral de la función $\text{sen}(x)$ por el método trapezoidal para el intervalo $x \in [0, \pi]$

$$I_{\text{trapz}} = \int_0^{\pi} \text{sen}(x)\, dx.$$



Código MATLAB 1: ejemplo14

Robótica, Período de primavera (21 de enero de 2020). Licenciatura en Ingeniería Mecatrónica.
Dr. Fernando Reyes Cortés. Facultad de Ciencias de la Electrónica, BUAP.

ejemplo14.m — MATLAB 2019a

```
clc; clear all; close all;
format short
tini=0; tinc=pi/1000; tfinal=pi;
x=tini:tinc:tfinal;
f=sin(x);
I_trapz= trapz(x,f);
I_a= 1-cos(pi);
disp('Resultado')
 %despliega los resultados comparativos del método trapezoidal con el analítico.
disp([I_trapz I_a])
```

Regla de Simpson

Cuando el área bajo la curva de la función $f(x)$ es representada sobre áreas de secciones cuadráticas y si el intervalo $[a, b]$ es dividido en $2n$ secciones iguales, entonces el área de $f(x)$ puede ser aproximada por la regla de Simpson cuya expresión está dada de la siguiente manera:

$$\begin{aligned} I_{\text{simp}} &= \frac{b-a}{6n}\left[f(x_0) + 4f(x_1) + 2f(x_2) + 4f(x_3) + \cdots + 2f(x_{2n-2})\right. \\ &\quad \left. + 4f(x_{2n-1}) + f(x_{2n})\right] \end{aligned} \tag{2}$$

donde x_i representan los valores finales de cada sección, para $i = 1, 2, \cdots, n-1$; $x_0 = a$, y $x_{2n} = b$.

En **MATLAB**, la regla de Simpson se implementa en forma adaptiva por medio de la función `quad`; como se muestra a continuación:

: Funciones de cuadratura

`quad('nombre_funcion',a,b)`	Retorna el área de la función en el intervalo comprendido entre a y b usando la regla de Simpson.
`q = quad(fun,a,b,tol)`	Emplea un error absoluto de tolerancia *tol* en lugar del valor por default 1e-06 sobre el intervalo $[a, b]$. Valores grandes de *tol* producen integración numé-

`quad(f,a,b)` aproxima la integral de una función $f(x)$ dentro de los límites finitos $[a,b]$, usando un algoritmo recursivo adaptable de cuadratura de Simpson, donde $a,b \in \mathbb{R}$. El error de integración numérica que usa es 1e-06. La función $f(x)$ puede ser escalar o vectorial. En el caso vectorial `quad` retorna un vector de salida donde cada componente tiene la integral de la correspondiente componente de la función vectorial $\boldsymbol{f}(\boldsymbol{x})$. La función `quad` puede ser menos eficiente (pobre exactitud) con funciones no suaves.

Ejemplo 2.1:

Calcular la integral por el método de Simpson de la función raíz cuadrática $f(x) = \sqrt{x}$ para intervalos no negativos $[a,b]$.

La función raíz cuadrada $f(x) = \sqrt{x}$ puede ser integrada analíticamente sobre el intervalo $[a,b]$, $a,b \in \mathbb{R}_+$ de la siguiente forma:

$$I = \int_a^b \sqrt{x}\, dx \;=\; \frac{2}{3}(b^{\frac{3}{2}} - a^{\frac{3}{2}}). \tag{3}$$

Este procedimiento sirve de referencia para comparar el valor numérico que calcula el método de Simpson.

Robótica FCE

BUAP

Código MATLAB 2: ejemplo15

Robótica, Período de primavera (21 de enero de 2020). Licenciatura en Ingeniería Mecatrónica.
Dr. Fernando Reyes Cortés. Facultad de Ciencias de la Electrónica, BUAP.

ejemplo15.m MATLAB 2019a

```
clc; clear all; close all; format short
disp('Método de Simpson')
 % los valores del intervalo [a, b] son proporcionados por el usuario desde el teclado
a=input('Introduzca el valor inicial del intervalo de integración: ');
b=input('Introduzca el valor final del intervalo de integración: ');
if a>=0 && b>= 0
    if a>b
        disp('Error: el intervalo positivo [a,b] debe cumplir a<b.')
        return
    end
     % regla de Simpson. La función raíz cuadrada (sqrt) es indicada como cadena de caracteres en la función quad
    I_simp=quad('sqrt',a,b);
     % método analítico
    I_a=2/3*(b^(3/2)-a^(3/2));
    fprintf('Valor analítico= %f \n Simpson: %f \n',I_a,I_simp)
    else % Para el caso de valores negativos
    disp('Error en los valores del intervalo de integración.')
    disp('El intervalo [a,b] con a<b debe contener únicamente valores positivos.')
end
```

Robótica
FCE
BUAP

Funciones de cuadratura

MATLAB tiene varias funciones que calculan la integral de funciones por métodos numéricos conocidas como funciones de ***cuadratura***. Adicional a la regla de Simpson `quad`, existen más opciones que se presentan a continuación:

quad8

La función `quad8('nombre_funcion',a,b)` retorna el área de la función en el intervalo $[a, b]$ usando la regla de Newton-Cotes 8 panel.

quadl

`q = quadl(f,a,b)` aproxima la integral de la función f en un intervalo finito $[a, b]$ dentro de un error 1e-06 usando el algoritmo recursivo adaptable de cuadratura de Lobatto. La sintaxis de `quadl` requiere que la función f sea una función manejador.

`quadl` acepta funciones vectoriales y retorna un vector con la integral de cada componente. `quadl` puede ser más eficiente (alta exactitud) que la función `quad` con funciones suaves.

Robótica FCE

BUAP

quadgk

La función `quadgk` tiene buena eficiencia para funciones oscilatorias, y soporta intervalos de integración infinitos, así como manejar moderadamente valores singulares. Una ventaja de esta función es que soporta integración de contorno a lo largo de trayectorias continuas por trozos. Si el intervalo de integración es infinito, es decir $[a, \infty]$, entonces para que la integral de $f(x)$ exista, $f(x)$ debe caer como $x \to \infty$. Esta es una condición para poderse utilizar `quadgk`, particularmente para funciones oscilatorias sobre intervalos infinitos, $f(x)$ debe caer muy rápida.

quadv

La función `quadv` vectoriza a `quad` para una función vectorial.

Robótica FCE
BUAP

Método de Euler

Un método ampliamente utilizado en robótica y sistemas mecatrónicos es la integración numérica discreta, la cual se puede establecer directamente del método de Euler.

Por ejemplo, al derivar con respecto al tiempo la integral $I(t)$ de la función $f(t)$ se obtiene lo siguiente:

$$I(t) = \int_{t_1}^{t_2} f(\sigma)d\sigma \quad \Rightarrow \quad \dot{I}(t) = f(t) \;\; \forall t \in [t_1, t_2]. \tag{4}$$

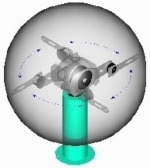
Partiendo de la definición matemática de la derivada se tiene:

$$\dot{I}(t) \quad = \quad \lim_{\triangle t \to 0} \frac{I(t + \triangle t) - I(t)}{\triangle t} \tag{5}$$

$$\tag{6}$$

donde $\triangle t = t_2 - t_1$, representa un infinitésimo intervalo de tiempo.

Si aproximamos a la derivada $\dot{I}(t)$ en su forma discreta $\dot{I}(t) \simeq \dot{I}(t_k)$ entonces el tiempo discreto está dado por $t_k = kh$, $k = 1, 2, \cdots, n$ y h es el periodo de muestreo. De esta forma se cumple $t_{k-1} = (k-1)h$ y $\triangle t_k = t_k - t_{k-1} = kh - (k-1)h = h$.



Robótica FCE

BUAP

Por lo tanto, la derivada discreta $\dot{I}(t_k)$ se puede realizar por diferenciación numérica de la posición $I(t_k)$ de la siguiente forma:

$$\dot{I}(t_k) = \frac{I(t_k) - I(t_{k-1})}{t_k - t_{k-1}} = \frac{I(t_k) - I(t_{k-1})}{h} = f(t_k). \tag{7}$$

La expresión 7 se conoce como método de Euler y sirve para estimar la velocidad por diferenciación numérica de la posición.

Despejando $I(t_k)$ de la expresión (7), la integral discreta adquiere la siguiente forma:

$$\boxed{I(t_k) = I(t_{k-1}) + hf(t_k) \tag{8}}$$

La expresión (8) es muy simple, se convierte en una sumatoria y es adecuada para poderse implementar como algoritmo recursivo.

Robótica FCE

BUAP

Ejemplo 4.1:

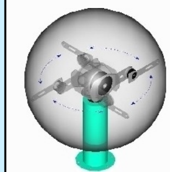
Calcular la integral de la función $f(t) = \text{sen}(t)$ por el método de Euler en forma recursiva para el periodo de integración $t \in [0, 10]$ segundos. El incremento del tiempo es por pasos de un milisegundo.

Código MATLAB 3: ejemplo16

Robótica, Período de primavera (21 de enero de 2020). Licenciatura en Ingeniería Mecatrónica.
Dr. Fernando Reyes Cortés. Facultad de Ciencias de la Electrónica, BUAP.

ejemplo16.m — MATLAB 2019a

```
clc; clear all; close all; format short
h=0.001; % periodo de muestreo
t=0:h:10; % intervalo de integración
f=sin(t); % función a integrar
I_e=0; % condición inicial de la integral por
 % método de Euler
 % método comparativo trapezoidal
I_trap=trapz(t,f);
 % método comparativo regla de Simpson
I_simp=quad('sin',0,10);
[m, n]=size(t); % Orden del vector de tiempo
for k=1:n
    I_e=I_e+h*f(k);
end
fprintf('Método Euler= %f \n Método Trapezoidal= %f \n Regla de Simpson = %f \n',I_e,I_trap, I_simp)
```



La salida del programa produce el siguiente resultado.

Método Euler= 1.838799

Método Trapezoidal= 1.839071

Regla de Simpson= 1.839072

Los resultados corresponden al intervalo $[0, 10]$, con incrementos de 0.001.
Obsérvese que existe un mayor error de integración del método de Euler con respecto a los métodos trapezoidal y el de Simpson. Sin embargo, en robótica se utiliza el método de Euler para estimar la velocidad de movimiento del robot por diferenciación numérica de la posición.

Los anteriores ejemplos ilustran el cálcuo de la integral numérica como un valor total. ¿Cómo sería su implementación recursiva en función del tiempo?

Algoritmo de integración de Euler

Ejemplos de programación en **MATLAB** y aplicaciones prácticas con el método de integración de Euler.

- Implementar en **MATLAB** el método de integración numérica de Euler.
- Calcular la integral numérica de los siguientes casos de estudio $f(t) = \operatorname{sen}(t), t^2, 8$ y $\cos(t)$:
  - $I(t) = \int_0^t \operatorname{sen}(\sigma)d\sigma.$
  - $I(t) = \int_0^t \sigma^2 d\sigma.$
  - $I(t) = \int_0^t 8 d\sigma.$
  - $I(t) = \int_0^t \cos(\sigma)d\sigma.$

  considere el intervalo de integración de $t \in [0,\ 10]$; además el paso de integración $h = 1$ mseg.
- Realizar una comparación técnica entre el método numérico de Euler con la solución analítica (presentar las gráficas en función del tiempo).

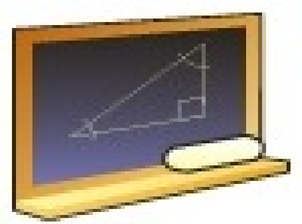


Figura 4: Ejemplos prácticos del algoritmo de integración numérica de Euler.