Métodos Computacionales
Segundo Trabajo Práctico
Segundo Semestre 2024

Introducción

Los splines se utilizan en el ajuste de curvas para aproximar formas complejas, siendo populares en informática, especialmente en gráficos por computadora, debido a su simplicidad de representación y facilidad de cálculo. Un spline es una curva diferenciable por partes, definida mediante polinomios. Podemos definir una curva como una función $P(t)$, que mapea un intervalo $S \in \mathbb{R}$ a un plano $\mathbb{R}^2$, si elegimos $S = [0, 1]$, la función $P(t)$ toma la siguiente forma:

$$P : [0, 1] \to \mathbb{R}^2, \quad P(t) = \begin{bmatrix} x(t) \\ y(t) \end{bmatrix}, \ \ con \ \ t \in [0, 1]$$

Con esta formulación, es sencillo generar puntos en la curva y también describir su trayectoria en términos de la velocidad a lo largo de ella. En su formulación más general, podemos expresar los splines como:

$$P(t) = GM_BT(t)$$

Donde G representa los puntos de control ordenados en una matriz, M_B es la base del spline: define el tipo de spline, y T(t) es la base de potencias los monomios $(1, t, .., t^n)$.

Esta formulación general tiene varias ventajas: es compacta, facilita la conversión entre diferentes tipos de splines y es independiente de la dimensionalidad (ya sea $\mathbb{R}^2$ o $\mathbb{R}^3$). Por ejemplo, las curvas de Bézier, vistas en el trabajo práctico 1, son un caso particular de splines, que pueden representarse de la siguiente manera:

$$Q(t) = \begin{bmatrix} x_1 & x_2 & x_3 & x_4 \\ y_1 & y_2 & y_3 & y_4 \end{bmatrix} \begin{bmatrix} 1 & -3 & 3 & -1 \\ 0 & 3 & -6 & 3 \\ 0 & 0 & 3 & -3 \\ 0 & 0 & 0 & 1 \end{bmatrix} \begin{bmatrix} 1 \\ t \\ t^2 \\ t^3 \end{bmatrix} \tag{1}$$

En este segundo trabajo, utilizaremos curvas de Bézier para aproximar una curva .ºbjetivo"definida por un conjunto de puntos D_i. El método propuesto para el ajuste de la curva de Bézier cúbica a una nube de puntos $\{D_i \in \mathbb{R}^2, \text{ con } i = 0, 1, ..., N - 1, \}$ es el de cuadrados mínimos. Para ello mediremos el error total, como la suma de los errores al cuadrado de cada punto. Este es el error cuadrático total entre los puntos de la curva de Bézier y los puntos objetivo D_i:

$$E_{total} = \sum_{i=0}^{N-1} \|D_i - Q(t_i)\|^2 \tag{2}$$

Este enfoque nos permitirá ajustar con precisión la curva a los puntos objetivo mediante la minimización de este error cuadrático.

Ejercicios

Ejercicio 1. Mostrar que los polinomios de la forma $y(t) = a_0 + a_1 t + a_2 t^2 + ... + a_n t^n$, constituyen un espacio vectorial, y que los polinomios de grado 3 forman un subespacio. Calcular la matriz de cambio de base que permite convertir la base de polinomios de Bernstein $B_B = \{B_1(t), B_2(t), B_3(t), B_4(t)\}$ a la base canónica $B_C = \{1, t, t^2, t^3\}$. Finalmente, determinar las coordenadas del monomio t^2 en B_B.

Ejercicio 2. Una curva de Bézier cúbica, parametrizada por t en el intervalo $[0, 1]$, y definida por los puntos de control P_0, P_1, P_2, P_3, puede expresarse como $Q(t) = MP$[1]. Aquí, $P = \begin{bmatrix} P_0 \\ P_1 \\ P_2 \\ P_3 \end{bmatrix}$ es una matriz por bloques de 4×1, donde cada bloque tiene dimensión 2×1.

- ¿Cuál es la dimensión de la matriz M en esta representación?
- Hallar la matriz M.

Ejercicio 3. Demostrar que la función objetivo en su forma matricial puede expresarse como: $E_{total} = \|D - MP\|^2$. Donde D es el vector columna de los puntos D_i. Luego, establecer la relación entre el problema de minimizar el error total y la resolución del sistema $MP = D$.

Ejercicio 4. Dada la lista de puntos en el archivo: `archivo_01.npy`. Aproximar estos puntos utilizando una curva de Bézier cúbica. Para ello:

- Resolver el problema mediante las ecuaciones normales para encontrar la curva de mejor ajuste.
- Calcular el error de aproximación dado por $\|\mathbf{D} - M\hat{\mathbf{P}}\|^2$
- Graficar en una misma figura los datos originales D_i y la curva de Bézier ajustada.

Ejercicio 5. Repetir los pasos anteriores utilizando la lista de puntos del archivo: `archivo_02.npy`. Reflexionar sobre los resulados obtenidos: ¿Qué se observa en este caso? ¿Cómo podríamos extender este método para mejorar el ajuste de la curva?

Ejercicio 6. Implementar un ajuste por segmentos, dividiendo los datos originales en tramos y aproximando cada tramo mediante una curva de Bézier. ¿Como se conectan entre sí las curvas de Bézier obtenidas? Modifique el código para lograr transiciones más suaves entre segmentos. ¿Mejora la aproximación global?

[1] Ayuda: $B(t) = (1-t)^3 P_0 + 3(1-t)^2 t P_1 + 3(1-t) t^2 P_2 + t^3 P_3$

Observaciones

Los archivos de datos pueden leerse usando la función `x, y = np.load("archivo_XX.npy")`

Condiciones de Entrega

Se deberán implementar todas las funciones sin usar librerías. La entrega del trabajo debe consistir en los siguientes archivos:

1. `informe.pdf:` un informe o reporte en el que se describan las tareas realizadas, cómo fueron resueltas, los resultados de sus experimentos y conclusiones. Puede ser escrito en programas del tipo Word, Markdown o LaTeX, controlando que al exportar el archivo no haya errores de fórmulas ni de formato.
2. `Notebook:` un *Notebook* de `Jupyter` que contenga todo el código utilizado. El código debe ejecutar sin errores, es decir, debe ser reproducible sin la necesidad de cambios adicionales.

Además, se deben tener en cuenta las siguientes consideraciones:

- La fecha límite de entrega es el **Viernes 22 de Noviembre**.
- La entrega debe ser realizada a través del campus por sólo 1 de los integrantes del grupo (no entregar por duplicado).
- En cada uno de los archivos entregados deben especificarse, arriba de todo, los nombres y apellidos completos de los integrantes del grupo.
- Las consignas sirven como guía de trabajo, se valorará el uso de técnicas vistas en clase como operaciones de vectores, matrices, y demás.
- Mantener el mayor nivel de prolijidad posible, tanto en los desarrollos como en el código. Las resoluciones deben estar escritas de forma clara y precisa.