Universidad de Chile
Facultad de Cs. Físicas y Matemáticas
Departamento de Física
FI3104-1: Métodos Numéricos para la Ciencia e Ingeniería.

Tarea N°3

Interpolación de Polinomios.

José Ignacio Vines.
Profesor: Valentino González.
Auxiliares: Mario Aguilar.
Ignacio Armijo.
María Constanza Flores.
Fecha: 28 de septiembre de 2016

Índice

Índice de figuras

1. Introducción

El objetivo del presente informe es investigar cómo se comportan la interpolación de polinomios contra la de Spline, y utilizar el método de interpolación de Spline en 2 dimensiones para reparar una imagen dañada de una galaxia.

La interpolación de polinomios consiste en, dado un conjunto de puntos, encontrar un polinomio que pase por estos. Por otro lado, un Spline es una función definida por tramos de polinomios, con un grado de suavidad en los puntos, o nodos, donde se unen estos.

2. Procedimiento

2.1. Comparación de Métodos

Para comparar como funcionan los métodos, éstos se aplicarán a una función Gaussiana definida como sigue:

$$f(x) = e^{-x^2/0{,}05} \tag{1}$$

En primera instancia se divide el intervalo $[-1, 1]$ en 4 tramos equiespaciados con la función **linspace**, dejando un total de 5 puntos para llevar a cabo la interpolación, luego, haciendo uso del módulo **scipy.interpolate** y de la función **lagrange** y clase **UnivariateSpline** se hace la interpolación, aumentando en 5 el número de puntos para la interpolación hasta llegar a 20, siempre manteniendo el equiespaciado del intervalo.

La función **lagrange** recibe como argumentos dos vectores, x e $y = f(x)$: un vector con el intervalo de puntos a samplear, uno con la función evaluada en dichos puntos respectivamente, y retorna un polinomio de Lagrange que pasa por todos los puntos de x. Como nota adicional, la implementación de Spline es numéricamente inestable y el resultado sólo es confiable hasta un intervalo con veinte puntos[1].

La clase **UnivariateSpline** recibe como argumentos dos vectores, x e $y = f(x)$: un vector con el intervalo de puntos a samplear y uno con la función evaluada en dichos puntos respectivamente, y como parámetros adicionales recibe s, un factor de suavidad utilizado para elegir el número de nudos del Spline, y k, el orden del Spline. Se elige $s = 0$ para que pase por todos los puntos del intervalo y $k = 3$ para que sea un Spline cúbico. Esta clase instancia un objeto de tipo **UnivariateSpline**, que representa un Spline que pasa por todos los puntos de x, que luego se puede evaluar.

La clase **UnivariateSpline** es un wrapper de FITPACK, un paquete de subrutinas escrito en FORTRAN utilizado para calcular Splines suaves para distintos tipos de datos y geometrías[2]; en particular **UnivariateSpline** hace uso de la subrutina **fpcurf0**, que a su vez utiliza la subrutina **fpcurf**[3] para calcular el Spline. La condición de suavidad que se debe cumplir en los nodos es que la k-ésima derivada del Spline sea 0. La documentación de **fpcurf** no contiene información de cómo maneja la implementación los extremos del intervalo, sin embargo la implementación clásica del método de Spline es imponer

[1] http://docs.scipy.org/doc/scipy/reference/generated/scipy.interpolate.lagrange.html#scipy.interpolate.lagrange
[2] http://www.netlib.org/dierckx/
[3] El archivo fpcurf.f está incluido en la carpeta for_routines en el repositorio.

que la segunda derivada del Spline en los extremos sea 0, así el Spline sigue una línea recta desde los extremos.

2.2. Reparación de Imagen

Primero, usando el módulo **skimage**, **skimage.io** y la función **img_as_float**, se carga la imagen a reparar como un tensor de floats, compuesto por tres matrices que representan las capas R, G y B (red, green y blue) de la imagen. Luego de cargar la imagen se toma una estampilla de la imagen, es decir, se toma la sección de esta que requiere reconstrucción. Utilizando la función **scipy.where** se eligen los puntos de la estampilla no tiene errores para la interpolación, después de obtener los puntos en donde la estampilla no tiene pixeles muertos se hace una interpolación Spline en dos dimensiones para cada capa de la estampilla utilizando la clase **SmoothBivariateSpline**, luego se arreglan los puntos dañados por capa y se juntan todas para recrear la imagen original reconstruida.

La clase **SmoothBivariateSpline** también es un wrapper de FITPACK. Esta clase en particular utiliza la subrutina **surfit**. La clase recibe tres vectores de una dimensión; x, y, z; que representan las coordenadas de los puntos para la interpolación (es necesario notar que el orden de estos argumentos no es importante para el funcionamiento del algoritmo), y un parámetro adicional kx y ky, los órdenes del Spline para x e y respectivamente, éstos se eligen como $kx = ky = 3$.

A la clase **SmoothBivariateSpline** se le dio como argumentos los bordes de la estampilla y un vector con los valores de la estampilla en todos los pixeles no dañados.

3. Resultados

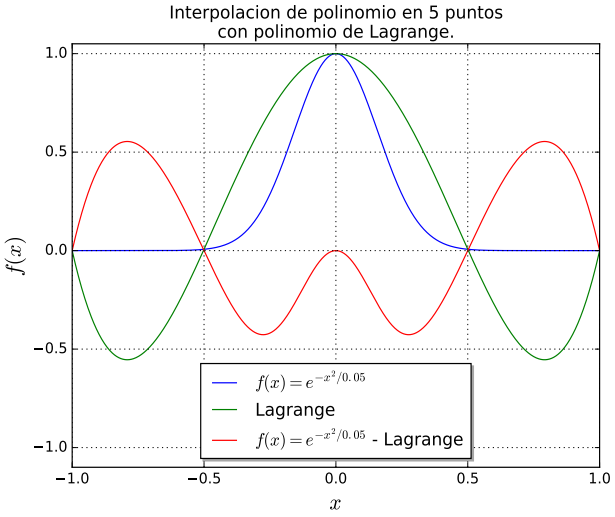
La figura 1 muestra la interpolación con polinomios de Lagrange. De la figura se aprecia que ésta aproximación es más fiel a la función original a medida se aumenta la cantidad de puntos utilizados para la interpolación, sin embargo a los extremos se observa un comportamiento oscilatorio de la aproximación, conocido como el Fenómeno de Runge[4].

La figura 2 muestra la interpolación con Spline. A mayor cantidad de puntos la interpolación con Spline es mejor aproximación que la interpolación con Lagrange. La interpolación con Spline no presenta el Fenómeno de Runge cuando se usa una cantidad grande de puntos.

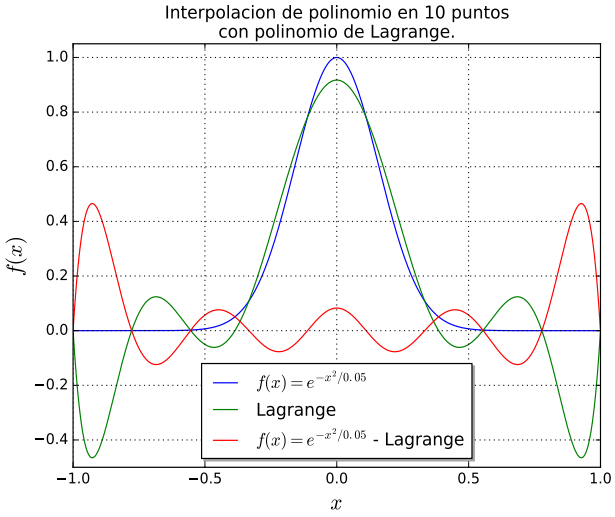
La figura 3 muestra la diferencia entre ambos métodos de interpolacion y la función original. Se puede apreciar de ella que la aproximación Spline es mejor que la aproximación con Lagrange.

La figura 4 muestra la imagen de la galaxia reconstruida.

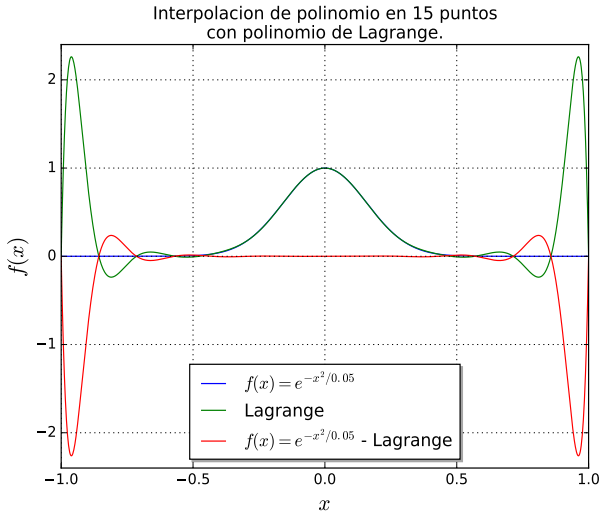
[4] https://en.wikipedia.org/wiki/Runge%27s_phenomenon



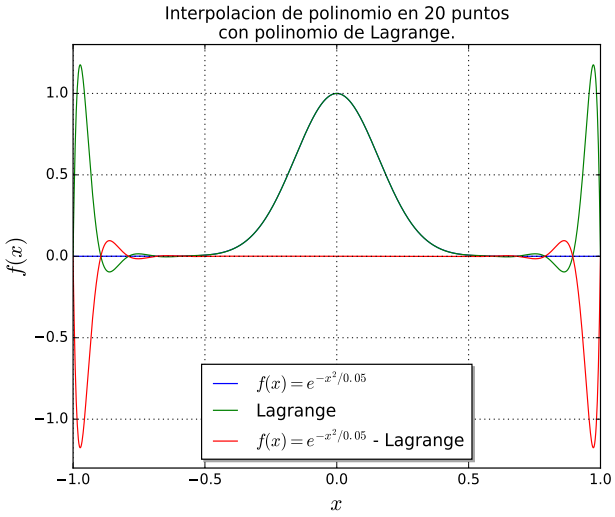
(a) Interpolación con Lagrange para un intervalo $[-1, 1]$ equiespaciado con 5 puntos.



(b) Interpolación con Lagrange para un intervalo $[-1, 1]$ equiespaciado con 10 puntos.

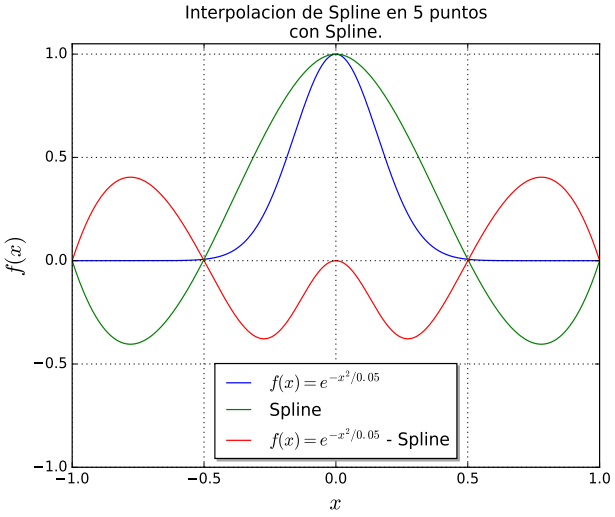


(c) Interpolación con Lagrange para un intervalo $[-1, 1]$ equiespaciado con 15 puntos.

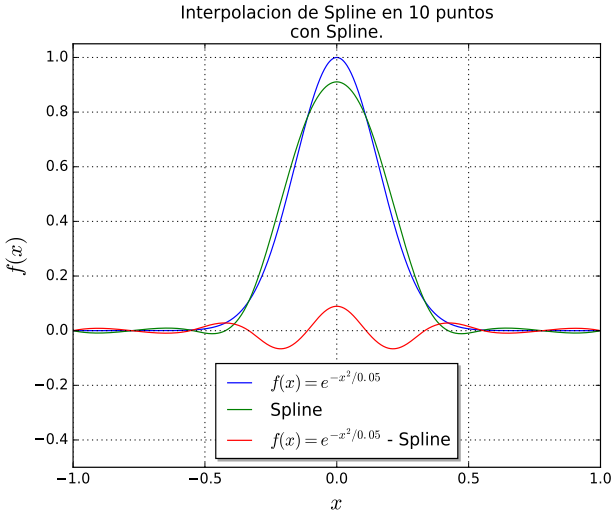


(d) Interpolación con Lagrange para un intervalo $[-1, 1]$ equiespaciado con 20 puntos.

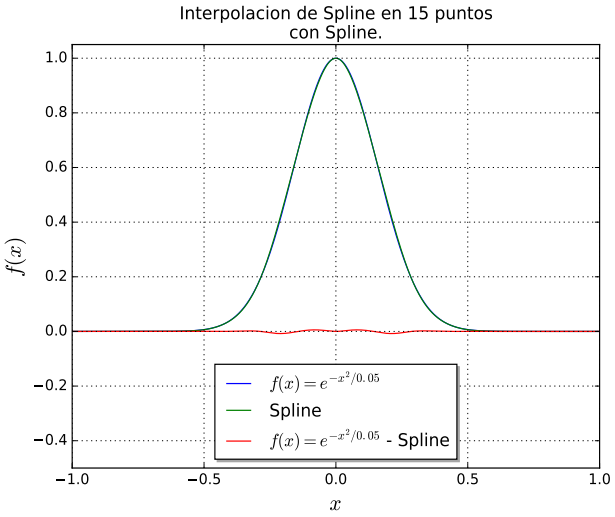
Figura 1: Distintas interpolaciones utilizando polinomios de Lagrange.



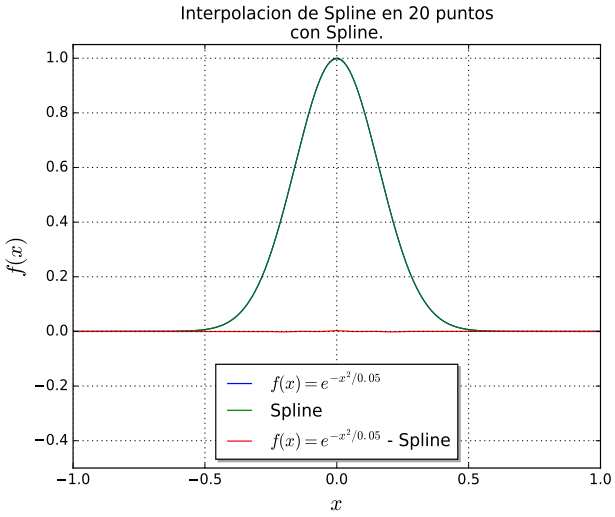
(a) Interpolación con Spline para un intervalo $[-1, 1]$ equiespaciado con 5 puntos.



(b) Interpolación con Spline para un intervalo $[-1, 1]$ equiespaciado con 10 puntos.



(c) Interpolación con Spline para un intervalo $[-1, 1]$ equiespaciado con 15 puntos.



(d) Interpolación con Spline para un intervalo $[-1, 1]$ equiespaciado con 20 puntos.

Figura 2: Distintas interpolaciones utilizando Spline.

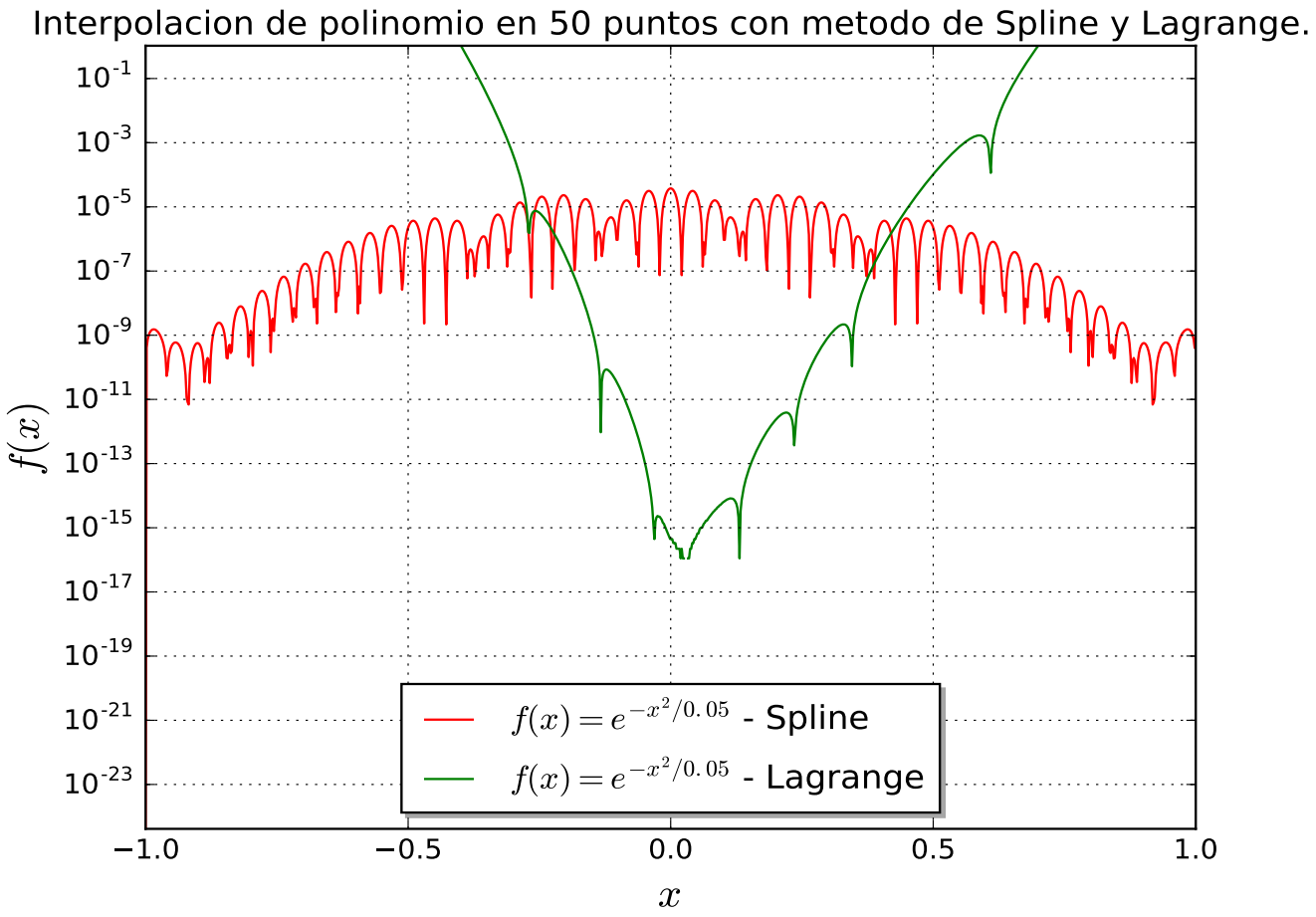


Figura 3: Diferencia entre la función original y las aproximaciones de Lagrnge y Spline para 50 puntos en el intervalo $[-1, 1]$.

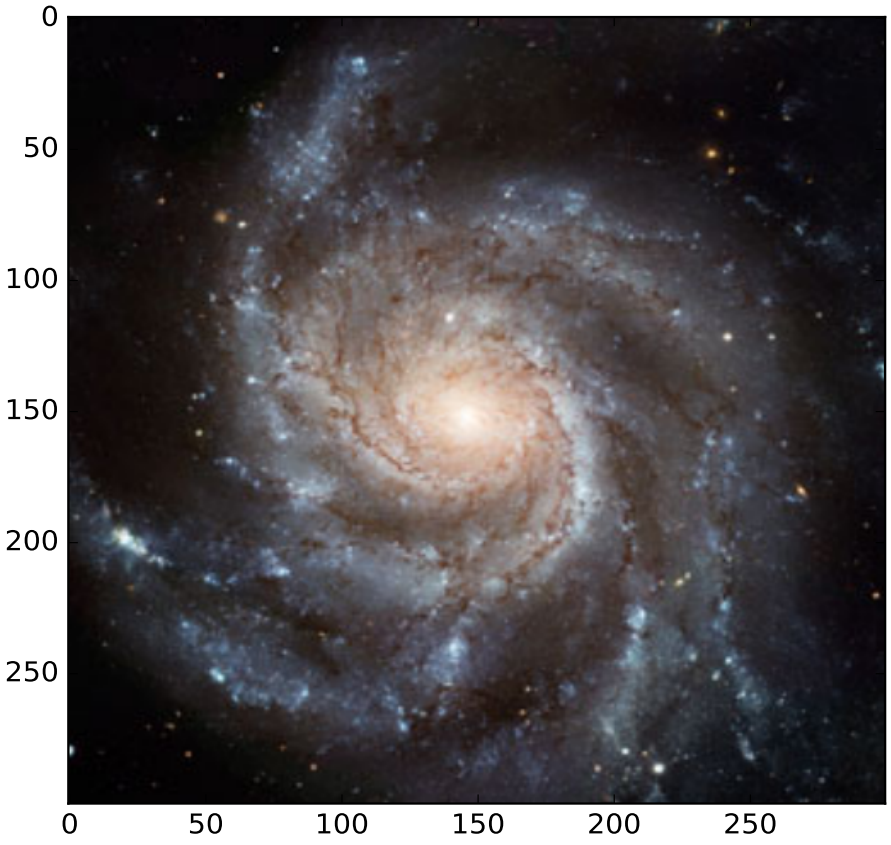


Figura 4: Imagen arreglada de la galaxia.

4. Conclusiones

Se concluye que una interpolación con Spline cúbico es una mejor aproximación a mayor cantidad de puntos que una interpolación con un polinomio de Lagrange, o si es de interés analizar los extremos de la función a interpolar ya que, a diferencia de ésta, Spline no presenta el Fenómeno de Runge.

Se concluye que la interpolación es una herramienta útil en el análisis de imágenes, en particular si se busca reparar una imagen dañada.