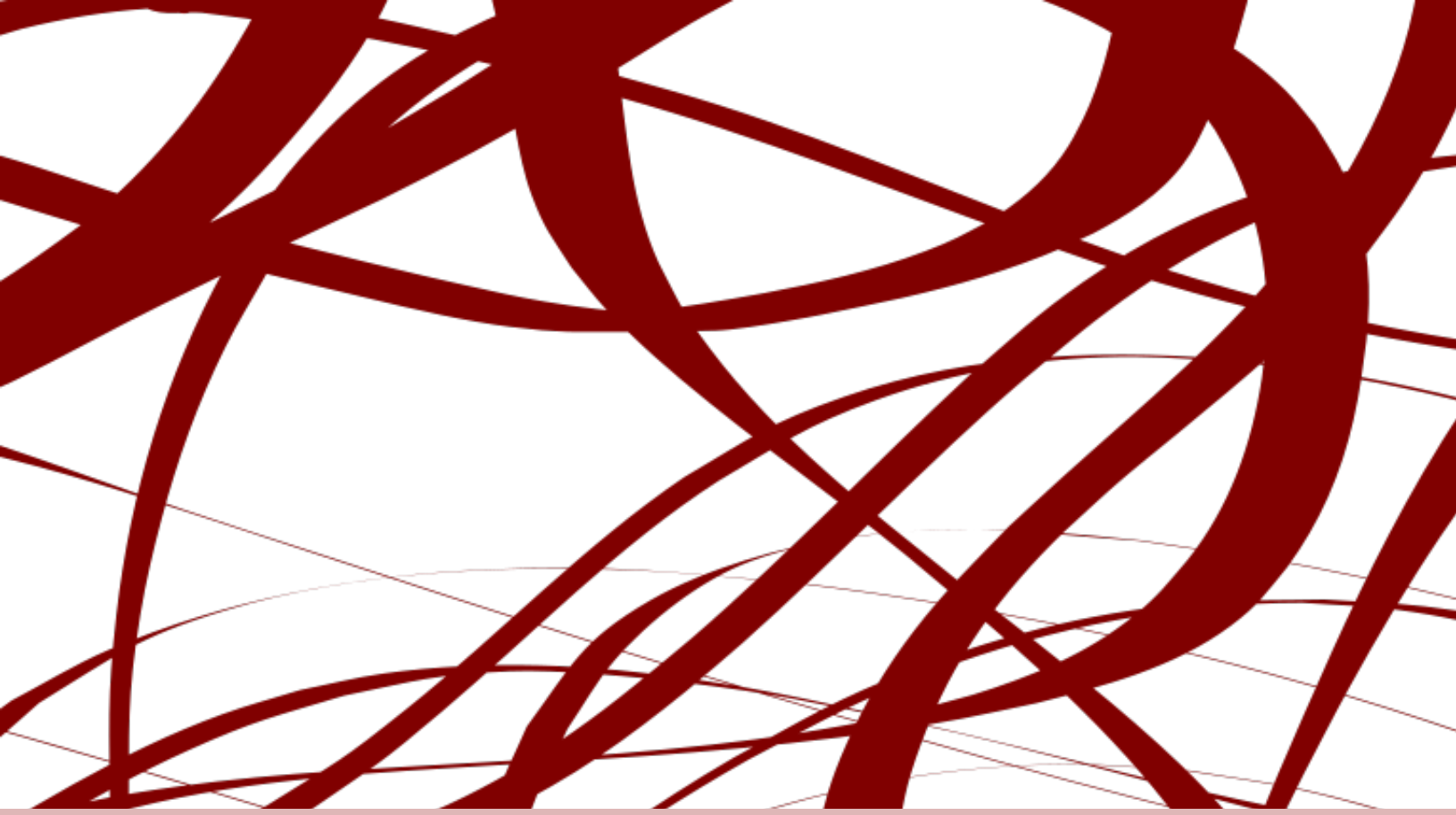
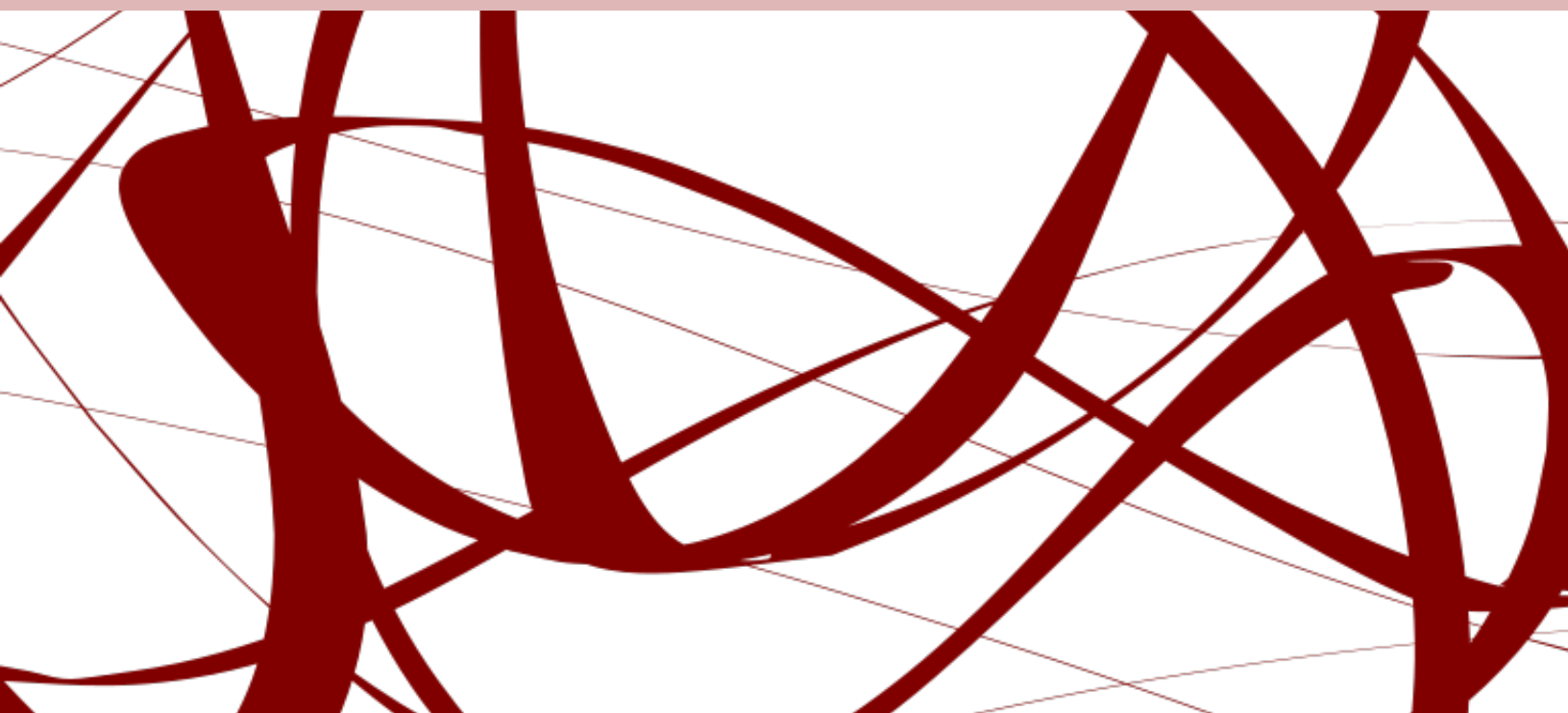


Análisis Numérico

Notas del curso

Dr. Jonathan Montalvo-Urquizo



HTTP://WWW.CIMAT.MX/~JONATHAN.MONTALVO/

Última actualización: 13.2.2018

Contenido

1. Introducción

1.1 Motivación e historia del cómputo

Muchos problemas científicos y tecnológicos que deben ser resueltos actualmente en áreas diversas del conocimiento como los procesos de producción, el entendimiento de los sistemas biológicos, o la exploración de fenómenos astronómicos o meteorológicos tiene un nivel alto de complejidad. La resolución de muchos de estos problemas requiere un proceso detallado de estudio que puede dividirse en las siguientes etapas:

1. Formulación del problema
2. Acotamiento del problema en un área científica
3. Abstracción conceptual
4. Modelo matemático del problema
5. Solución analítica o numérica
6. Interpretación de la solución

La obtención de una solución analítica (matemáticamente) depende no sólo de la existencia de una solución sino también de la posibilidad técnica de calcularla. Además, en muchos de los casos, a pesar de que las soluciones a los problemas existen, es imposible calcularlas.

La matemática numérica se encarga de encontrar soluciones aproximadas a problemas cuya complejidad supera las capacidades de calcular soluciones de manera analítica. Actualmente, el concepto de "Solución Numérica" se refiere a soluciones a problemas matemáticos obtenidas a través de cálculos realizados en una computadora. Este concepto lleva implícita la noción de que las soluciones obtenidas de esta manera

son, por su naturaleza, aproximaciones a las soluciones exactas del problema planteado.

Dentro de las funciones actuales que cumple la matemática numérica dentro del proceso científica de resolución de problemas aplicados son:

- el desarrollo de métodos para la construcción de soluciones numéricas,
- la implementación computacional de los métodos de manera eficiente,
- la selección de los parámetros adecuados para el uso de los métodos implementados,
- el análisis acerca de la calidad de las soluciones obtenidas en términos de nivel de exactitud, tiempo de cómputo y estabilidad de los resultados.

Aunque en la actualidad los métodos numéricos son comúnmente diseñados para su uso en computadoras digitales, las primeras metodologías de la matemática numérica pueden ser atribuídas a las prácticas de cálculo de poblaciones de la antigüedad.

Si solamente consideramos la era de computadoras basadas en algún tipo de maquinaria capaz de realizar cálculos de manera automatizada, algunos dispositivos de cómputo mejor documentados son:

1623 Máquina de cálculos (Rechenuhr) diseñada (y construida?) por Wilhelm Schikarden en Tübingen, Alemania.

1645 Blaise Pascal presenta su máquina de cálculo "Pascaline".

1673 Gottfried Wilhelm Leibniz presenta su máquina de cálculo basada en la hoy llamada rueda/cilindro de Leibniz (Staffelwalze).

1938 Konrad Zuse completa su máquina Z1 (originalmente llamada V1 como abreviatura del nombre Versuchsmodell-1). Ésta es considerada la primer máquina de cálculo que utilizó la aritmética de punto flotante y que utilizaba sistema de numeración binario.

1941 Konrade Zuse presenta la Z3. Por primera vez los casos excepcionales son considerados (NaN, $+\infty$, $-\infty$). Mejora sustancial desde la Z1.

1945 En marzo, Konrad Zuse presenta la Z4, primera computadora programable a través del uso de cintas perforadas similares a la cinta fílmica.

1946 En febrero, J. Presper Eckert y John W. Mauchly presentaron la computadora ENIAC (Electronic Numerical Integrator and Computer). Ésta es conocida como la primer computadora totalmente electrónica.

1951 UNIVAC, construida por J. P. Eckert y J. W. Mauchly y fue la primer computadora disponible para la venta comercial. UNIVAC existió hasta 1981.

En particular, las máquinas diseñadas por Konrad Zuse están bien documentadas a través del portal `http://zuse.zib.de/`, donde se exponen las versiones digitales de

planos, fotografías e incluso simulaciones de reconstrucciones de las máquinas Z1 y Z3.

A partir de la década de los 1950's, el uso de computadoras se incrementó en todas las áreas de actividad humana. En términos de cálculo, las capacidades computacionales empezaron a crecer impulsadas por los desarrollos de procesadores cada vez más potentes. En una necesidad de predecir el crecimiento acelerado de la industria de la computación, algunos pioneros se aventuraron a definir el nivel de crecimiento de las capacidades de cómputo. La predicción hasta ahora más certera es la famosa Ley de Moore[1], postulada en 1965 y que afirma que el número de transistores (y por ende la capacidad de cómputo) se duplica aproximadamente cada 2 años.

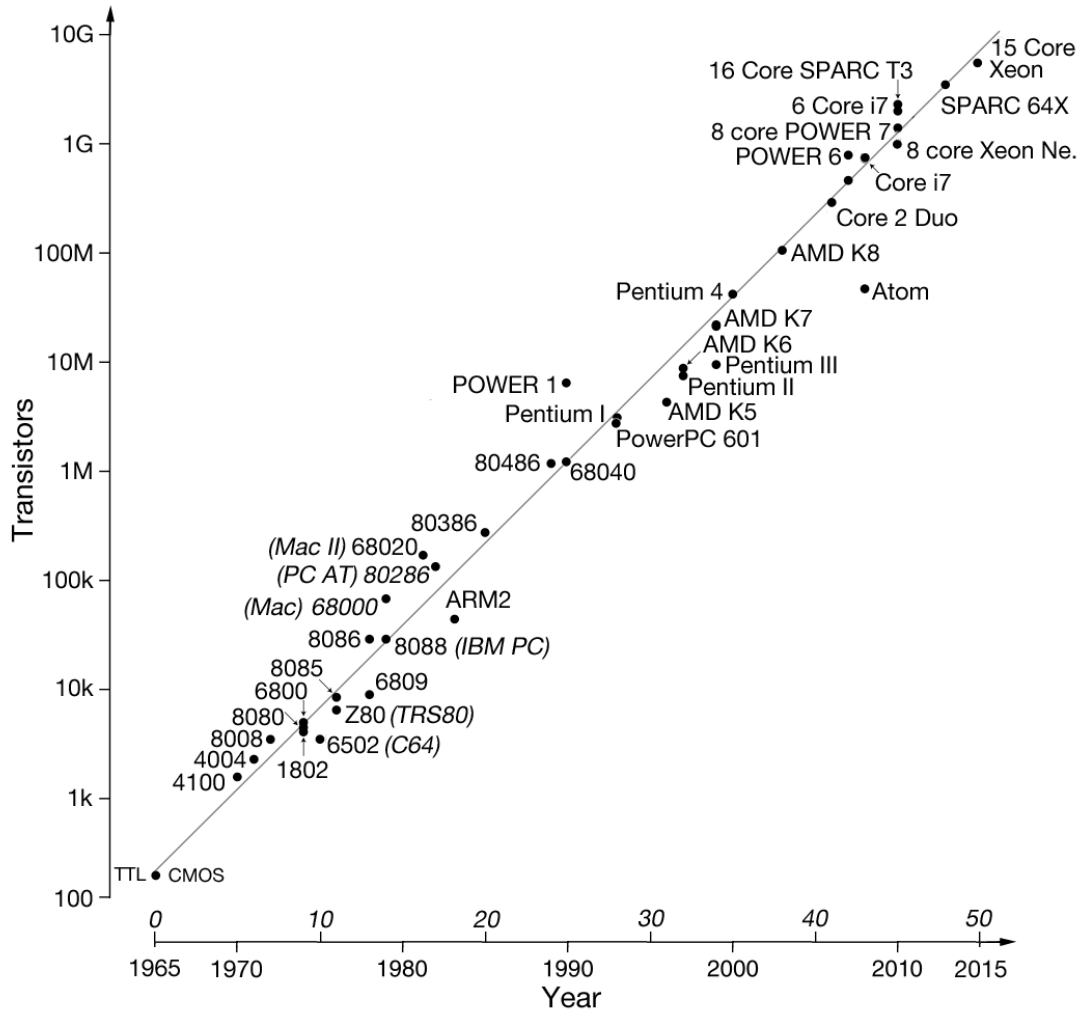


Figura 1.1: Cincuenta años de la Ley de Moore.

Como ya se dijo, la matemática numérica depende en gran medida de la capacidad de cómputo para calcular soluciones a modelos matemáticos, por lo que esta ley asegura que

[1] Atribuída a Gondor E. Moore (1929-), cofundador del Intel Corporation.

la eficiencia con que la matemática numérica puede resolver problemas se incrementa en un 100 % cada dos años, como puede ser apreciado en la Figura 1.1.

Veamos ahora algunos ejemplos básicos de problemas que pueden ser resueltos mediante el uso de la matemática numérica.

■ **Ejemplo 1.1 — Corte de 3 planos.** Usado por ejemplo en la construcción de techos de dos aguas con cortes laterales

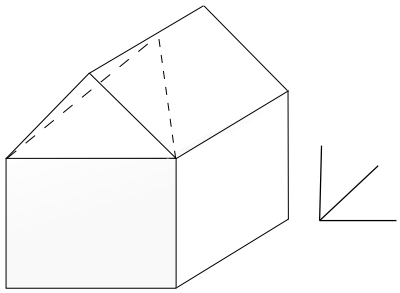


Figura 1.2: Corte de múltiples planos en $\mathbb{R}^3$.

Esto se traduce fácilmente en una tarea de resolver el sistema lineal

$$\begin{aligned} ax \quad\quad\quad +bz &= c, \\ -ax \quad\quad\quad +bz &= d, \\ ex \quad +fy \quad +gz &= h, \end{aligned} \tag{1.1}$$

y por ser pocas ecuaciones puede ser resuelto de manera analítica, pero esto no está automatizado directamente, a menos que se resuelva de manera numérica. Un simulador para resolver este tipo de problemas no podría considerar todas los posibles valores de las constantes $a,\ldots,h$, por lo que la resolución a través de una simulación numérica sería necesaria. ■

■ **Ejemplo 1.2 — Descripción de procesos dinámicos de primer orden.** La ecuación $y(t) = y'(t)$ cuya solución es $y(t) = e^t$ puede usarse como ejemplo de una dinámica temporal para describir la expansión bacteriana o el contagio de una enfermedad. Sin embargo, aunque la solución analítica sea conocida, saber exactamente cuántas bacterias se encuentran presentes en el modelo requiere de la evaluaión de una exponencial, lo cual solamente es asequible si se utiliza una cálculo numérico utilizando el tiempo deseado t. Evidentemente, este tipo de evaluaciones son imposibles de realizar utilizando cálculos hechos a mano o por medio de tablas precalculadas. ■

■ **Ejemplo 1.3 — Descripción de procesos dinámicos de segundo orden.** La ecuación $y(t) = -y''(t)$ es resuelta (entre otras) por la función $y(t) = \cos(t)$ y es usada para describir el movimiento ideal de un sistema formado por una masa colgada de un resorte (sin considerar los efectos de la gravedad). Esta función puede ser calculada

utilizando tablas preestablecidas hace muchos años, sin embargo, el nivel de precisión que estas tablas tienen no supera las 6 u 8 cifras decimales, por lo que la precisión del cálculo estaría limitada.

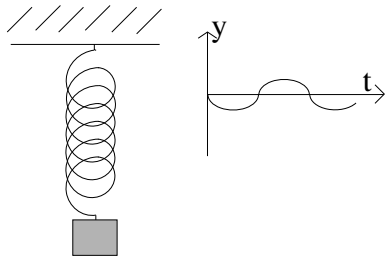


Figura 1.3: Oscilador armónico.

Por otro lado, los sistemas dinámicos de muchos problemas de interés, como son las enfermedades poblacionales están descritas por modelos dinámicos mucho más complejos (Dengue, Ébola, Zika, etc.). Otro ejemplo modelado por este tipo de osciladores son los sistemas de cuerpos articulados como el sistema de amortiguación de un automóvil, un brazo robótico, o incluso los planetarios. Todos estos sistemas son evidentemente mucho más complejos y requieren de una alta precisión para ser calculados. Por tanto, las ecuaciones diferenciales correspondientes a este tipo de problemas no pueden ser resueltas de manera analítica. ■

■ **Ejemplo 1.4 — Encontrar ceros de funciones.** El método más utilizado para funciones $f : \mathbb{R} \to \mathbb{R}$ consiste en buscar utilizando la información de la derivada para generar, a partir de una aproximación inicial x_0 una sucesión de parejas $(x_0, f(x_0)), (x_1, f(x_1)), \ldots$ a través del uso de la línea tangente a la curva, es decir

$$y(x) = f(x_0) + (x - x_0) f'(x_0), \tag{1.2}$$

cuyo corte con la línea del cero es

$$x_1 = x_0 - \frac{f(x_0)}{f'(x_0)}. \tag{1.3}$$

En general, al construir las aproximaciones de la forma

$$x_{i+1} = x_i - \frac{f(x_i)}{f'(x_i)} \tag{1.4}$$

se tendría un proceso iterativo como el ilustrado en la Figura 1.4 en la que repetir el mismo procedimiento converge al punto donde la función vale cero. Al construir este proceso iterativo se espera que $f(x_i) \to 0$ y por lo tanto $|x_{i+1} - x_i| \to 0$.

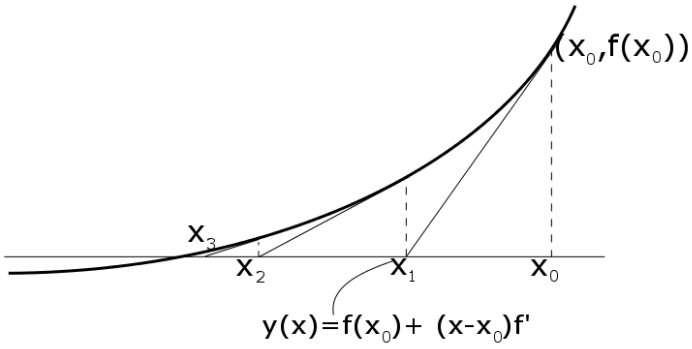


Figura 1.4: Encontrando el cero de una función a través de aproximaciones lineales.

Esto puede calcularse analíticamente algunas veces, por ejemplo cuando la función f es un polinomio o alguna otra función bien conocida analíticamente. Sin embargo, el caso general requerirá de un proceso iterativo cuya convergencia es desconocida y por lo tanto no se puede saber a priori cuántos cálculos requerirá. En este tipo de problemas, la única opción es considerar una solución numérica en la que puedan calcularse muchas iteraciones del proceso iterativo. ■

■ **Ejemplo 1.5 — Integración de funciones.** Es otro proceso que puede ser sencillamente aproximado de manera analítica a través de una partición del dominio $[a,b) = \cup_{i=1}^{n}[x_i, x_{i+1})$ y el uso de la idea original de la integral de Riemann, donde se consideraba un conjunto de barras de altura igual a la evaluación de la función y su respectiva área, tal como se ilustra en la Figura 1.5. Si se aproxima la integral de esta

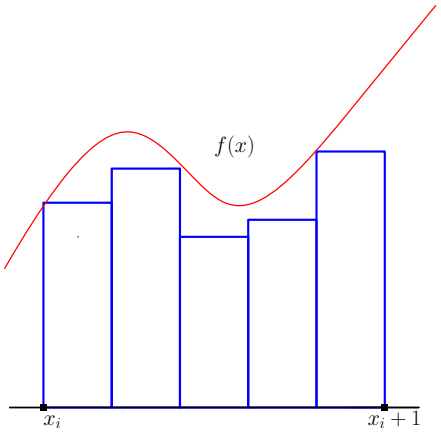


Figura 1.5: Aproximación de la integral de $f(x)$ en el intervalo $[x_i, x_{i+1}]$.

forma, pueden calcularse analíticamente una buena cantidad de áreas para anchos de las

barras relativamente pequeños. Otra opción de mejores resultados es considerar la regla del trapecio dada por

$$\int_{x_i}^{x_{i+1}} f(x) \approx \frac{f(x_i)+f(x_{i+1})}{2}(x_{i+1}-x_i) \tag{1.5}$$

que en general da buenas aproximaciones al valor de la integral (como veremos más adelante en el curso).

Sin embargo, el proceso de tomar barras cada vez más pequeñas solamente podrá ser realizado si se considera una gran cantidad de operaciones aritméticas, pues en cada barra considerada se tendrán que realizar evaluaciones de $f(x_i)$, las cuales pueden llegar a ser muy complicadas de calcular de forma analítica.

Existen ejemplos de aplicaciones en todas las áreas de la matemática donde el tratamiento de este tipo de problemas es necesario debido a la complejidad o a las dimensiones del espacio (euclidiano o no) en el que el modelo sea planteado. En particular, muchos métodos de solución numérica para resolver ecuaciones diferenciales están basados en calcular un gran número de integrales en varias dimensiones. Piense por ejemplo en el cálculo del volumen de almacenamiento de un tanque de gasolina moderno (3D, geometría rebuscada, asimetrías, etc.) ∎

1.2 Aritmética de punto flotante

El uso de cálculos numéricos tiene limitantes debido a que los números representables computacionalmente son finitos y, por muchos que parezcan, nunca será lo mismo que tener el continuo de números en $\mathbb{R}$.

Definición 1.1 Los números computacionales son llamados "de punto flotante" y forman, junto con el sistema aritmético para realizar operaciones básicas entre ellos, lo que se denomina "Aritmética de punto flotante" (APF).

Un número de punto flotante en base $b \in \mathbb{N} \setminus \{1\}$ es un número real de la forma

$$x = \pm m \cdot b^{\pm e} \tag{1.6}$$

donde los diferentes factores son

- El signo de x definido como $+$ ó $-$
- La mantisa, que es representable en términos de la base b y r números adicionales que forman la expresión

$$m = m_1 b^{-1} + m_2 b^{-2} + \cdots + m_r b^{-r}$$

- El exponente representable con s números

$$e = e_{s-1} b^{s-1} + \cdots + e_0 b^0$$

Tanto para la mantisa como el exponente, los números que actúan como coeficientes de la base cumplen con $m_i, e_i \in \{0, \cdots, b-1\}$.

Además, con el fin de normalizar la aritmética, se requiere que $m_1 \neq 0$, lo cual evita que los números puedan tener diversas representaciones, haciendo posible una única representación para cada número de la APF. Note que esta representación es cercana (aunque no exactamente igual) a la notación científica y permite considerar números de muy diversos tamaños como pudiera ser

- La velocidad de la luz $c = 0{,}299792458 \cdot 10^9 m/s$
- La masa de un electrón $M_o = 0{,}910938 \cdot 10^{-29} g$

Para guardar un número computacional hacen falta:

(a) r cifras para la mantisa más un signo;
(b) s cifras para el exponente más un signo;
(c) espacio para guardar los siguientes números:

- Infinito (positivo y negativo), que será cualquier número mayor al más grande representable $\rightarrow$ Región de *Overflow*. El infinito se representa usando $e = \text{máx}(posibles)$, $m = 0$
- NaN (not a number) que se obtiene al realizar operaciones no definidas en la aritmética. Este número se representa usando $e = \text{máx}(posibles)$, $m > 0$
- Números en la Región de *Underflow*, definida en $(0, \text{mín}_{x \neq 0}(|x|))$ y se representa por $e = \text{mín}(posibles)$, $m > 0$
- Cero, definido como el número en el centro de todos los reales y representado por $e = 0$, $m = 0$

Los números se guardan en formato

$$(\pm)[m_1 m_2 \cdots m_r](\pm)[e_{s-1} e_{s-2} \cdots e_0] \tag{1.7}$$

donde solamente es necesario guardar los coeficientes m_i y e_j. Debido a razones técnicas de construcción, las computadoras utilizan numeración binaria, es decir que las cifras pertenecen a $\{0,1\}$ y la base que se toma es $b = 2$. De esta forma, tiene sentido también declarar que $m_1 = 1$.

Otro recurso utilizado comúnmente es la representación del exponente como una traslación de un orden mayor, es decir en lugar de usar exponentes con signos y "desperdiciar" una cifra para el signo se puede usar un exponente siempre positivo y trasladarlo como

$$(\pm)[e_{s-1} e_{s-2} \cdots e_0] \longrightarrow [e_s e_{s-1} \cdots e_0] - T \tag{1.8}$$

donde T es el factor de traslación. Veamos porqué se hace esto en utilizando el ejemplo binario en que $s = 4$.

■ **Ejemplo 1.6 — APF con base $b = 2$, $s = 4$.** Con base en los valores de la base y el número de coeficientes del exponente se tiene que:

- En la primera representación $(\pm)e_3e_2e_1e_0$:
  Se tendrán exponentes mínimo absoluto de 0 (usando $e = \pm(0\cdot 2^3 + 0\cdot 2^2 + 0\cdot 2^1 + 0\cdot 2^0)$) y máximo en valor absoluto de $e = 8+4+2+1 = 15$.
  Pero como el máximo exponente deberá ser usado para guardar NaN y los Infinitos, sólo los exponentes en $\{-15,-14,\cdots,0,1,2,\cdots,14\}$ serán de utilidad real, introduciendo una asimetría exponencial.
- En la segunda representación $e_4e_3e_2e_1e_0$:
  Se tendrán exponentes con mínimo absoluto de 0 y máximo absoluto de $e = 2^4+2^3+2^2+2^1+2^0 = 31$. Trasladando $\{0,\cdots,31\}$ con $T = 15$ se obtiene $\{-15,\cdots,0,1,\cdots,16\}$, lo que permite usar el exponente 16 para NaN/Infinito y mantener la simetría exponencial.

■

■ **Ejemplo 1.7 — Sistema binario.** Veamos el ejemplo concreto de un sistema binario, con límite para guardar números en 1 Byte=8 bits.

De los 8 lugares podemos usar 1 lugar para el signo, 4 para la mantisa y 3 para el exponente. Los números $m_i, e_i \in \{0,1\}$, $m_1 = 1$ tal que

$$m = 2^{-1} + m_2 2^{-2} + m_3 2^{-3} + m_4 2^{-4} + m_5 2^{-5} \quad \text{(4 cifras)} \tag{1.9}$$

$$e = e_2 2^2 + e_1 2^1 + e_0 2^0 \quad \text{(3 cifras)} \tag{1.10}$$

Trasladando e con $T = 3$ se obtiene $e \in \{-3,-2-,1,0,1,2,3\}$. De esta manera, el mayor número en el sistema, será dado por la mantisa con $m = \underline{1}1111$ y el exponente 3, es decir

$$\begin{aligned} x_{max} &= (2^{-1}+2^{-2}+2^{-3}+2^{-4}+2^{-5})\cdot 2^3 \\ &= 2^2+2^1+2^0+2^{-1}+2^{-2} \\ &= 7+\frac{1}{2}+\frac{1}{4} \\ &= 7\frac{3}{4}. \end{aligned} \tag{1.11}$$

El número positivo más pequeño es dado por la menor mantisa $m = 10000$ y el menor exponente posible (-3), y es entonces

$$\begin{aligned} x_{min} &= 2^{-1}\cdot 2^{-3} = 2^{-4} \\ &= \frac{1}{16} \end{aligned} \tag{1.12}$$

Esto quiere decir que la región de Underflow es

$$\left(-\frac{1}{16},0\right) \cup \left(0,\frac{1}{16}\right) \tag{1.13}$$

y la de Overflow es

$$\left(-\infty,-7\frac{3}{4}\right)\cup\left(7\frac{3}{4},\infty\right). \tag{1.14}$$

En este sistema hay un total de 16 posibles mantisas y 7 exponentes, es decir que sólo pueden representarse 112 números positivos y 112 negativos. ∎

∎ **Ejemplo 1.8 — IEEE-754.** Esta APF es definida por el Institute of Electrical and Electronic Engineers y es la norma más utilizada para los cálculos en los sistemas de hoy en día. Es un sistema binario con mantisa de 52 cifras y exponente de 11 cifras (sin signo). El bit restante de los 64 que usa es el signo del número. Trasladando los exponentes con $T = 1023$ se obtiene un rango de exponentes $e \in \{-1023,\cdots,1023\}$.

Nuevamente las cifras $m_i, c_i \in \{0,1\}$ y análogamente

$$m = 2^{-1} + m_2 2^{-2} + m_3 2^{-3} + \cdots + m_{53} 2^{-53} \tag{1.15}$$

$$e = e_{10} 2^{10} + e_9 2^9 + \cdots + e_0 2^0 \tag{1.16}$$

El mayor número representable es entonces dado por el mayor exponente (1023) y la mantisa

$$m = 1\underbrace{11\cdots1}_{52 \text{ veces}} \tag{1.17}$$

$$\begin{aligned} \Rightarrow\ x_{max} &= (2^{-1}+2^{-2}+\cdots+2^{-53})\cdot 2^{1023} \\ &= 2^{1022}+2^{1021}+\cdots+2^{970} \\ &\approx 1{,}8\times 10^{308} \end{aligned} \tag{1.18}$$

El menor número positivo tiene el menor exponente (-1023) y la mantisa
$m = 1\underbrace{00\cdots0}_{52 \text{ veces}} \Rightarrow 2^{-1}\cdot 2^{-1023} = 2^{-1024} \approx 5{,}556\times 10^{-309}$

Se pueden guardar 2^{52} mantisas diferentes y 2046 ($\approx 2^{11}$) exponentes, es decir que el sistema es capaz de representar un total de 2^{63} números positivos y un total de 2^{64} números (recuerde que se usan 64 bits). ∎

Redondeo

Para todos los números computacionales, las operaciones entre ellos pueden resultar en números que no pertenecen al sistema de números representables. En estos casos es necesario realizar un redondeo en el que se pide la condición natural

$$|x - rd(x)| = \min_{y\in A} |x-y| \quad \forall x \in D$$

donde $D = [x_{min}, x_{negmax}] \cup \{0\} \cup [x_{posmin}, x_{max}]$ y A es el conjunto finito de números representables.

Observe que la idea coincide con el redondeo justo hacia arriba y abajo al utilizar un sistema de base decimal.

Para el sistema **IEEE-754** esto se hace simplemente como

$$rd(x) = sign(x) \cdot \begin{cases} 0 \cdot m_1 \ldots m_{53} \cdot 2^e & , m_{54} = 0 \\ (0 \cdot m_1 \ldots m_{53} + 2^{-53}) \cdot 2^e & , m_{54} = 1 \end{cases} \tag{1.19}$$

Por supuesto que se introducirán errores de redondeo, las cuales se pueden acotar en términos absolutos como

$$|x - rd(x)| \leq \frac{1}{2} b^{-r} b^e \tag{1.20}$$

que no es muy útil dado su dependencia del exponente e, por lo que comúnmente se revisa el error relativo de redondeo dado por

$$\left\| \frac{x - rd(x)}{x} \right\| \leq \frac{1}{2} \frac{b^{-r} b^e}{|m| b^e} \leq \frac{1}{2} b^{-r+1} \tag{1.21}$$

para todo $x \in D, x \neq 0$.

Este error relativo está entonces acotado por la precisión de cómputo (machine precision) $eps := \frac{1}{2} b^{-r+1}$ para la cual vale siempre

$$rd(x) = x(1+\varepsilon) \text{ para algún } |\varepsilon| \leq eps \tag{1.22}$$

En el sistema **IEEE-754** esta cota es

$$eps = \frac{1}{2} 2^{-53+1} = 2^{-53} \approx 10^{-16} \tag{1.23}$$

Operaciones aritméticas básicas

Las operaciones en los números reales $* \in \{+, -, \cdot, /\}$ son reemplazadas por operaciones computacionales $\otimes \in \{\oplus, \ominus, \odot, \oslash\}$. Estas operaciones deben generar resultados dentro de la APF (los números computacionalmente representables). Algunas consecuencias es que la ley distributiva y asociativa de las operaciones $\oplus$ y $\odot$ se cumplen solamente de manera aproximada. En general para $x, y, z \in APF$

$$((x \oplus y) \oplus z) \neq x \oplus (y \oplus z) \tag{1.24}$$

$$(x \oplus y) \odot z \neq (x \odot z) \oplus (y \odot z) \tag{1.25}$$

Problemas Numéricos en este curso

Un problema numérico puede escribirse como el problema de encontrar una cantidad finita de valores $y_i \quad (i = 1, \ldots, n)$ mediante el uso de otra cantidad finita de valores conocidos $x_j \quad (j = 1, \ldots, m)$ a través de un funcional $y_i = f(x_1, \ldots, x_m)$.

En este curso nos ocuparemos principalmente de problemas en los que x_j y y_i son números reales. Problemas en $\mathbb{C}$ pueden ser tratados muy análogamente y quedan fuera de los contenidos de este curso.

Errores absoluto y relativo

Definición 1.2 — Errores absoluto y relativo. Mediante la existencia de errores en los datos conocidos (p.ej. a través de redondeo) $x_j + \Delta x_j$ se producen errores en los valores calculados $y_i + \Delta y_i$. El error absoluto es definido como $|\Delta y_i|$ mientras que para $y_i \neq 0$ el error relativo es definido como $\left\| \frac{\Delta y_i}{y_i} \right\|$

Por la naturaleza de la definición de los errores absoluto y relativo, los errores de interés en el análisis numérico son los de tipo relativo.

Veamos ahora en análisis diferencial al considerar pequeños errores en los datos , i.e. $|\Delta x_j| \ll |x_j|$. Suponiendo que para el cálculo de los valores y_i se tienen funcionales f_i tal que $y_i = f_i(x_1, \ldots, x_m)$ donde f_i es parcialmente diferenciable con respecto a $x_j \quad (\forall i, \forall j)$ entonces de acuerdo al Teorema de Taylor y usando $x = (x_1, \ldots, x_m)$

$$\Delta y_i = f_i(x + \Delta x) - f_i(x) = \sum_{j=1}^{m} \frac{\partial f_i}{\partial x_j}(x) \Delta x_j + \mathscr{R}_i^f(x, \Delta x) \tag{1.26}$$

$i = 1, \ldots, m$, donde $\mathscr{R}_i^f(x, \Delta x)$ denota el residual.

Además puede demostrarse que $\mathscr{R}_i^f(x, \Delta x)$ decae más rápido hacia 0 que el valor $|\Delta x| = \text{máx}_{j=1,\ldots,m} |\Delta x_j|$ dado que se está suponiendo que todos los Δx_i son pequeños en valor absoluto.

Esto se denota normalmente como $\mathscr{R}_i^f(x, \Delta x) = \mathscr{o}(|\Delta x|)$ usando la conocida Notación de Landau. La notación de Landau[2] para describir procesos asintóticos cuantitativos utiliza los símbolos $\mathscr{O}(\cdot)$, $\mathscr{o}(\cdot)$ como sigue:

Definición 1.3 — Orden de una función. Sean $g(t)$ y $h(t)$ funciones para $t \in R_+$, entonces la notación $g(t) = \mathscr{O}(h(t)), (t \to 0)$ significa que para valores pequeños de $t \in [0, t_0]$, existe $c \geq 0$ tal que se cumple $|g(t)| \leq c \cdot |h(t)|$. De modo similar, si $g(t) = \mathscr{o}(h(t)) \quad (t \to 0)$ para valores de $t \in [0, t_0]$ significa que existe una función $c(t)_{t \to 0} \to 0$ tal que $|g(t)| \leq c(t) \cdot |h(t)|$

■ **Ejemplo 1.9** Sea $g(t) \in C^2$ con expansión de Taylor

$$g(t + \Delta t) = g(t) + \Delta t g'(t) + \frac{\Delta t^2}{2} g''(\tau), \tau \in (t, t + \Delta t)$$

entonces

$$\frac{g(t + \Delta t) - g(t)}{\Delta t} = g'(t) + \mathscr{O}(\Delta t)$$

■

[2] Debido a Edmund Geary Hermann Landau (1877 – 1938). Profesor en Götingen de nacionalidad alemana. Obligado a pensionarse en 1934 por ser de origen judío. Trabajos en Teoría de números. Teoría de funciones complejas, entre otras.

Veamos ahora que pasa en una *APF* cuando los datos contienen pequeños errores y se realizan operaciones con ellos.

Definición 1.4 — Número de condición de la suma. La función de suma $f : \mathbb{R}^2 \to \mathbb{R}, y : f(x_1, x_2) = x_1 + x_2$ puede analizarse a tráves de su "número de condición", definido como

$$k_1 = \frac{\partial f}{\partial x_1}\frac{x_1}{f} = 1 \cdot \frac{x_1}{x_1+x_2} = \frac{1}{1+\frac{x_2}{x_1}}$$
$$k_2 = \frac{\partial f}{\partial x_2}\frac{x_2}{f} = \frac{1}{1+\frac{x_1}{x_2}}$$

Puede observarse fácilmente que los números k_1 y k_2 presentan problemas cuando $1 + \frac{x_i}{x_j} \approx 0$ por lo que la suma es problemática cuando $x_1 \approx -x_2$, es decir cuando ambos sumandos son aproximadamente iguales pero de diferente signo.

Otro problema conocido al sumar números es el siguiente:

Definición 1.5 — Cancelación catastrófica. Se llama "Cancelación catastrófica" a la pérdida de cifras en un número de punto flotante a través de realizar una resta de dos números del mismo signo. Esto es un problema cuando alguno de los números no es exactamente representable en la aritmética computacional.

■ **Ejemplo 1.10 — Cancelación catastrófica en aritmética decimal.** Considere la *APF* con base decimal y 4 dígitos significativos en contraste con la aritmetica exacta en $\mathbb{R}$

Aritmética en $\mathbb{R}$	APF ($b = 10, 4$ dígitos)
$x = 0{,}51232 \times 10^1$	$fl(x) = 0{,}5123 \times 10^1$
$y = 0{,}11230 \times 10^1$	$fl(y) = 0{,}1123 \times 10^1$
$(x-y) = 0{,}40002 \times 10^1$	$fl(z) = -0{,}4 \times 10^1$
$z = -0{,}4 \times 10^1$	
$(x-y)-z = 0{,}40002 \times 10^1 - 0{,}4 \times 10^1$	$fl(x-y) = 0{,}4 \times 10^1$
$= ,2 \times 10^-4$	$fl((x-y)-z) = 0{,}0 \times 10^1$

■

En el caso de la operación de multiplicación, el "número de condición" para el cálculo de $y = f(x_1, x_2) = x_1 \cdot x_2$ es

$$k_1 = \frac{\partial f}{\partial x_1}\frac{x_1}{f} = x_2 \cdot \frac{x_1}{x_1 \cdot x_2} = 1$$
$$k_2 = 1$$

por lo que al ser siempre bien condicionada, la multiplicación no representa problemas en función de qué valores tengan x_1, x_2.

Nota. Al realizar múltiples cálculos computacionales es inevitable la aparición de pequeños errores debido a redondeo o errores al evaluar funciones aritméticas básicas.

Además, estos errores se acumulan a lo largo del proceso de cálculo. El interés de Análisis Numérico radica en generar algoritmos en los que esta acumulación de errores no domine a la buena aproximación numérica.

Un ejemplo del ahorro en errores al usar menos operaciones puede verse en la evaluación de un polinomio cuadrático, es decir en el cálculo de valores $y = a_0 + a_1 x + a_2 x^2$.

(a) Evaluado directamente, este valor requiere 2 sumas y 3 multiplicaciones, es decir 5 operaciones de punto flotante.
(b) Evaluado en la forma equivalente $y = a_0 + x(a_1 + a_2 x)$ requiere 2 sumas y 2 multiplicaciones $\Rightarrow$ 4 operaciones dep f .

El mismo ejemplo visto para polinomios de grado 3 lleva en evaluación directa a 3 sumas y 6 multiplicaciones, es decir a 9 operaciones de punto flotante en el estilo de evaluación directa. Para grado 4 esto es 4 sumas y 10 multiplicaciones $\Rightarrow$ 14 operaciones de punto flotante. Usando la evaluación anidada $y = a_0 + x(a_1 + x(a_2 + \ldots + x(a_{n-1} + xa_n)\ldots))$ esto lleva en grado 3 y 4 respectivamente a

$$\begin{aligned} y &= a_0 + x(a_1 + x(a_2 + xa_3)) && \Rightarrow 3 \text{ sumas, } 3 \text{ multiplicaciones} \\ y &= a_0 + x(a_1 + x(a_2 + x(a_3 + xa_4))) && \Rightarrow 4 \text{ sumas, } 4 \text{ multiplicaciones} \end{aligned}$$

y en general, esta forma anidada requiere n sumas y n multiplicaciones para evaluar un polinomio de grado n, es decir $2n$ operaciones de punto flotante. A la evaluación polinomial en esta forma se le conoce como el **Algoritmo de Hörner**[3]

1.3 Ejercicios

Ejercicio 1.1. Software y programación (0 Puntos)
Practique los comandos básicos de MATLAB con el fin de poder introducir vectores, matrices, manipular elementos de los mismos, graficar información en el plano, etc. Existe una basta oferta de tutoriales y recursos en línea para esto. Si lo desea, realice lo equivalente en otro software o leguaje (R, java, python, etc.) ■

Ejercicio 1.2. Sistema IEEE-754 (25 Puntos)
Para el sistema IEEE-754 visto en clase, determine el valor de los siguientes números:
1. el penúltimo número representable, y
2. el segundo número positivo más pequeño representable.

¿Qué consecuencias buenas o malas habrá en la diferencia de espaciamiento entre los números del sistema? ■

[3]Debido a William George Hörner (1786 – 1837). Matemático irlandés. Otras fuentes atribuyen este algoritmo a fuentes mucho más antiguas como Zhu Shije en China del siglo XIII.

Ejercicio 1.3. Leyes asociativa y distributiva (35 Puntos)
Muestre un ejemplo en el que las leyes asociativa y distributiva no se cumplen, acorde a las expresiones (1.24) y (1.25).
Encuentre además las cotas para los números de punto flotante y tales que se cumpla $x \oplus y = x$. ■

Ejercicio 1.4. Generando una APF (20 Puntos)
Invente su propio sistema de números de punto flotante utilizando base binaria y espacio de 2 Bytes (16 bits). Defina una longitud de mantisa y encuentre los números positivos de mayor y menor tamaño para este sistema. ¿Cuántos números distintos puede representar su sistema? ■

Ejercicio 1.5. Sistema decimal (20 Puntos)
Escriba ahora un sistema decimal ($b = 10$) con una mantisa de 4 cifras y un exponente de 3 cifras. Esto aunado al signo en la representación da un total de 8 'trozos' de información a ser guardados para los elementos de la aritmética. De esta manera, la construcción es más intuitiva que en los sistemas binarios. ¿Por qué cree entonces que no se usa el sistema decimal en los sistemas de cómputo? ■

Ejercicio 1.6. Número de condición (20 Puntos)
Use nuevamente el número de condición $k_i = \frac{\partial f}{\partial x_i}\frac{x_i}{f}$ para mostrar que la división x_1/x_2 está bien condicionada para $i = 1, 2$. ■

Ejercicio 1.7. Precisión de una computadora (20 Puntos)
El nivel de precisión de una computadora está definido como el número

$$\text{eps} = \min_{x \in D,\, x \neq 0} \left| \frac{\text{rd}(x) - x}{x} \right|.$$

en donde $D = [x_{min}, x_{negmax}] \cup \{0\} \cup [x_{posmin}, x_{max}]$ y $\text{rd}(\cdot)$ denota la función de redondeo. ¿Cómo cree que podría determinarse esta constante experimentalmente? Defina un algoritmo para ello, impleméntelo y determine así el número eps de su computadora. ■

Ejercicio 1.8. Aproximación de Taylor (30 Puntos)
Considere la aproximación a la función exponencial dada por la suma de Taylor

$$T_n = \sum_{k=0}^{n} \frac{x^k}{k!}.$$

Escriba un pequeño programa que calcule esta suma utilizando una cantidad diferente de términos en la aproximación para $n \in \{1,2,3,\ldots,20\}$ y considere los cálculos para valores de $x \in \{-10,-1,1,10\}$.

(a) Compare en una tabla la calidad de los resultados con el resultado de la función 'exp(x)' de MATLAB para los 20 niveles de aproximación.

(b) Explique los malos resultados para valores negativos de x y modifique su algoritmo de cálculo para que la calidad de los resultados no dependa del signo de x. Repita la comparación del inciso (a) con el algoritmo modificado.

■

Ejercicio 1.9. Evaluación de polinomios (30 Puntos)

Considere un polinomio expresado en las dos formas equivalentes

$$p(x) = a_0 + xa_1 + a_2x^2 + a_3x^3 + \cdots + a_nx^n$$

$$p(x) = a_0 + x(a_1 + x(a_2 + \cdots + x(a_{n-1} + xa_n)))$$

1. Suponga que los coeficientes a_i estan dados y determine el número de operaciones de punto flotante para evaluar ambas representaciones del polinomio en un punto x.
2. Defina 16 números enteros para ser usados como coeficientes de un polinomio de grado 15. Implemente ambas representaciones de la evaluación y realice un número grande de evaluaciones para graficar su polinomio para $x \in [-2,2]$. Observe y comente sobre la cantidad de evaluaciones necesarias para que las diferencias se hagan notar en los tiempos de cálculo al usar una u otra representación.
   Algunas funciones que pueden ser útiles: `linspace`, `tic`, `toc`, `plot`, `for`, `function`.

■

2. Interpolación y Splines

Un problema fundamental en la práctica es la representación numérica de objetos matemáticos simples como las funciones reales, o bien de evaluaciones puntuales de éstas. Es evidente que muchos de los modelos que se utilizan para representar fenómenos reales se basan en funciones que, en su forma más simple, corresponden a funciones en los números reales. En este sentido, hay dos puntos importantes a considerar:

(a) Una función $f(x)$ solo puede conocerse en un número finito de puntos $x_0, \cdots, x_n$ y deberá ser reconstruida usando solamente esta información. Un ejemplo de reconstrucción es la representación gráfica de la función, efectuada comúnmente a traves de conocer algunos puntos de la función y conectarlos con una línea para lograr el símil gráfico de tener una función graficada.

(b) Una función $f(x)$ deberá ser representada numéricamente de modo que la evaluación en un valor dado x sea fácil de calcular en cualquier momento. Esto no es nada sencillo si la función es complicada, contiene múltiples funciones trigonométricas, etc. pues entre más variaciones existan en la función de interés, será mas difícil calcular las evaluaciones en un punto dado.

En ambos casos se tendrá una dependencia funcional $y = f(x)$ con una cantidad de grados de libertad que puede llegar a ser muy grande e incluso ser infinita. Para esto, las funciones son comúnmente aproximadas a través de familias o clases de funciones mejor conocidas, como pueden ser:

- Polinomios $p(x) = a_0 + a_1 x + \cdots + a_n x^n$
- Funciones racionales $r(x) = \dfrac{a_0 + a_1 x + \cdots + a_n x^n}{b_0 + b_1 x + \cdots + b_m x^m}$

- Polinomios trigonométricos $t(x) = \frac{1}{2}a_0 + \sum_{k=1}^{n} [a_k \cos(kx) + b_k \sin(bx)]$
- Sumas exponenciales $e(x) = \sum_{k=1}^{n} a_k \exp(b_k x)$

Por razones obvias, habrá una diferencia al aproximar una función utilizando estas clases de funciones. Además existe una diferencia terminológica cuando la aproximación consiste en mera "cercanía" o cuando se tienen puntos en los que la función original y la función proveniente de las clases de funciones conocidas comparten valores en ciertos puntos del dominio.

Definición 2.1 — Interpolación y Aproximación. Sea P una clase de funciones como las antes mencionadas. La asociación de un elemento $g \in P$ a la función f a través de una relación puntual en un número determinado de puntos

$$g(x_i) = y_i = f(x_i) \quad i = 1, \ldots, n$$

se denomina Interpolación. Si la relación de cercanía entre g y f esta dada en un sentido de "mejor" representación como pudiera ser:

$$\max_{a \leq x \leq b} |f(x) - g(x)| \text{ es mínimo para } g \in P,$$

o bien

$$\left(\int_a^b |f(x) - g(x)| dx \right)^{\frac{1}{2}} \text{ es mínimo para } g \in P$$

entonces se habla de Aproximación. Es evidente que la interpolación es un tipo de aproximación en la que la función de representación g satisface que

$$\max_{i=1,\ldots,n} |f(x_i) - g(x_i)| \text{ es mínimo para } g \in P.$$

2.1 Interpolación Polinomial

En esta sección consideraremos a P_{n-1} como el espacio vectorial de los polinomios reales de grado menor o igual a $n-1$

$$P_{n-1} = \left\{ p(x) = a_1 + a_2 x + \ldots + a_n x^{n-1} | a_i \in R, i = 1, \ldots, n \right\} \tag{2.1}$$

para considerar el problema de interpolación de Lagrange[1] como sigue:

[1] Debido a Joseph Louis de Lagrange (1736 – 1813). Matemático francés que fue director de la Mathematische Klasse der Berliner Akademie y después se convirtió en Profesor en París. Sus cotribuciones incluyen obras fundamentales en Cálculo Variacional, Mecánica, Mecánica Celeste, entre otras áreas.

Definición 2.2 — Problema de interpolación de Lagrange. El problema de interpolación de Lagrange consiste en determinar un polinomio $p \in P_{n-1}$ tal que dados n pares de puntos $(x_1, y_1), \ldots, (x_n, y_n)$ con $y_i = f(x_i)$ para una función $f : \mathbb{R} \to \mathbb{R}$ se cumpla que $p(x_i) = y_i$ para todo $i = 1, \ldots, n$.

Teorema 2.1 — Solución del problema de interpolación de Lagrange. El problema de interpolación de Lagrange tiene solución y su solución es única.

Demostración (draft)

- Unicidad: Considere 2 soluciones al problema de interpolación p_1, p_2 y construya $p = p_1 - p_2$. El nuevo polinomio satisface que $p(x_i) = 0, \quad i = 1, \ldots, n$, es decir que tiene n raíces, además de que $p \in P_{n-1}$. Por consecuencia, deberá tenerse que $p(x) = 0 \quad \forall x, p = \overline{0} \in P_{n-1}$.
- Existencia Considere el polinomio genérico $p \in P_{n-1}$ expresado en la base de monomios $\{1, x, x^2, \ldots, x^{n-1}\}$ y construya el sistema lineal correspondiente a evaluar la ecuación $a_1 + a_2 x + \ldots + a_n x^{n-1} = b$ para cada par (x_i, y_i) a interpolar. Si los x_i's son distintos se obtendrá un sistema lineal con soluciones únicas.

Como es bien sabido, pueden construirse diferentes bases para el espacio vectorial de los polinomios. De acuerdo a la construcción tomada por Lagrange para la interpolación polinomial, la base de los polinomios a tomar es la formada por los polinomios en P_{n-1} de la forma

$$L_i^{(n)}(x) = \prod_{j=1, j\neq i}^{n} \frac{x - x_j}{x_i - x_j} \tag{2.2}$$

que cuentan con la propiedad de que $L_i^{(n)}(x) = 1$ en el caso de que $x = x_i$ y son iguales a cero en el resto de los nodos. De hecho, esta propiedad es fundamental para demostrar que el conjunto $\left\{ L_i^{(n)}, \quad i = 1, \ldots, n \right\}$ es una base de P_{n-1}.

Definición 2.3 — Interpolante de Lagrange. Dados los n pares $(x_1, y_1), \ldots, (x_n, y_n)$ el polinomio de la forma

$$p(x) := \sum_{i=1}^{n} y_i L_i^{(n)}(x) \quad \in P_{n-1} \tag{2.3}$$

cumple con la propiedad de que $p(x_i) = y_i$ y es llamado Interpolante de Lagrange del conjunto de (x_i, y_i)'s.

Este es un buen ejemplo para fabricar un programa en el que diferentes tareas se le asignen a diferentes funciones. Además es una tarea que comúnmente se requiere realizar cuando se tienen datos reales provenientes de mediciones en algunos puntos pero se requiere conocer más valores de los que se tiene.

■ **Ejemplo 2.1 — Funciones para interpolar puntos.** Supongamos que se desea construir un polinomio interpolante de Lagrange para un conjunto de datos (X_i, Y_i) con

$i \in \{1,\ldots,n\}$. Y además evaluarlo para un conjunto de puntos diferentes guardados en un vector x de longitud N, por ejemplo $N = 1000$. Este problema puede resolverse definiendo la siguiente estructura o algoritmo:

Función Principal

```
%-- Dados los datos X, Y , x,  realizar un ciclo
N = length(x);
 for s=1:N
   y(s)=Eval_pLagrange(X,Y,x(s))
 end

%-- Post-processing , por ejemplo
plot(x,y, X,Y, '*')
```

Función auxiliar para evaluar un punto en $p(x)$

```
function y= Eval_pLagrange(X, Y, evalx)
   n=length(X);
   p=0;
   for i=1:n
      p=p+ Y(i)*oneLagrange_pol(X,i,evalx);
   end
end
```

Función auxiliar para evaluar un término de la sumatori en $p(x)$

```
function L=oneLagrange_pol(X,k,evalx)
   L=1;
   n=length(X)
   for i=1:n
      if(i~=k)
         L=L*((evalx-X(i))/(X(k)-X(i)))
      end
   end
end
```

■

Los interpolantes de Lagrange tienen la ventaja de ser intuitivos en cuanto a que por construcción $L_i^{(n)}(x)$ es igual a 1 en x_i y es igual a 0 en los demás x_j's. Sin embargo presentan el inconveniente de que cada elemento de la base depende de todos los puntos x_j, por lo que al intentar incluir un nuevo punto x_j en la construcción se tendrían que recalcular todos los términos $L_i^{(n)}(x)$.

Una alternativa a este problema es considerar la base de polinomios como lo hizo Newton [2] definida como $\{N_i, \quad i = 1,\ldots,n\}$ que se construye como sigue:

$$N_1(x) = 1, \tag{2.4}$$

$$N_i(x) = \prod_{j=1}^{i-1}(x-x_j), \quad i \in \{2,\ldots,n\}, \tag{2.5}$$

y que puede ser usada para definir un polinomio

$$p(x) = \sum_{i=1}^{n} a_i N_i(x). \tag{2.6}$$

Con este Ansatz para la forma del polinomio $p(x)$ puede encontrarse por recursión sucesiva que

$$y_1 = p(x_1) = a_1 \tag{2.7}$$

$$y_2 = p(x_2) = a_1 + a_2(x_2 - x_1) \tag{2.8}$$

$$y_3 = p(x_3) = a_1 + a_2(x_3 - x_1) + a_3(x_3 - x_1)(x_3 - x_2) \tag{2.9}$$

$$\vdots$$

$$y_n = p(x_n) = a_1 + a_2(x_n - x_1) + \cdots + a_n(x_n - x_1)\cdots(x_n - x_{n-1}) \tag{2.10}$$

que puede ser visto como un sistema de ecuaciones lineales. Estas ecuaciones pueden ser modificadas fácilmente para formar un sistema de ecuaciones lineales como:

$$M_n \bar{a} = \bar{y} \tag{2.11}$$

con $\bar{a}$ y $\bar{y}$ los vectores con los componentes a_i, y_i y la matriz del sistema

$$M_n = \begin{pmatrix} 1 & 0 & 0 & \cdots & 0 \\ 1 & (x_2 - x_1) & 0 & \cdots & 0 \\ 1 & (x_3 - x_1) & (x_3 - x_1)(x_3 - x_2) & \cdots & 0 \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & (x_n - x_1) & (x_n - x_1)(x_n - x_2) & \cdots & (x_n - x_1)\cdots(x_n - x_{n-1}) \end{pmatrix} \tag{2.12}$$

de tamaño $n \times n$.

En caso de que un nuevo punto (x_{n+1}, y_{n+1}) deba ser considerado, bastaría con construir la matriz M_{n+1} como

$$M_{n+1} = \left[\begin{array}{ccc|c} & M_n & & 0_{nx1} \\ \hline 1 & (x_{n+1} - x_1) & \cdots & \prod_{j=1}^{n}(x_{n+1} - x_j) \end{array}\right] \tag{2.13}$$

[2] Isaac Newton $(1643 - 1727)$. Físico y Matemático inglés. Profesor en Cambridge que es ampliamente reconocido por haber realizado desarrollos fundamentales en Mecánica Clásica y Cálculo Diferencial.

y considerar ahora el sistema de ecuaciones

$$M_{n+1}\begin{pmatrix} a_1 \\ \vdots \\ a_{n+1} \end{pmatrix} = \begin{pmatrix} y_1 \\ \vdots \\ y_{n+1} \end{pmatrix} \tag{2.14}$$

Además el sistema es de forma triangular inferior, lo que hace que resolver el problema de encontrar los coeficientes a_i sea simple de resolver numéricamente.

En un capítulo posterior de este curso nos ocuparemos de ver algoritmos para resolver problemas con este tipo de matrices. Por ahora basta con observar que el valor de a_1 está ya resuelto, esto hace que a_2 sea muy fácil de calcular y este proceso puede continuarse hasta calcular todos los coeficientes a_i. En Matlab, puede usarse por ahora la resolución por medio del comando '\' (backslash) y en R puede usarse la función *'solve'*. Una vez conocidos los coeficientes a_i es fácil evaluar el polinomio de acuerdo a las ecuaciones (2.5) y (2.6) como

$$p(x) = a_1 + a_2(x-x_1) + a_3(x-x_1)(x-x_2) + \cdots + a_n(x-x_1)\cdots(x-x_n) \tag{2.15}$$

Hasta ahora hemos asumido que el conjunto de puntos a aproximar se verá bien reflejado si lo hacemos a través de un polinomio. Sin embargo, los pares (x_i, y_i) a aproximar pueden provenir de una función arbitraria con gran cantidad de oscilaciones o con tendencias cambiantes. De esta reflexión resulta la pregunta inmediata sobre qué tan bien aproximará nuestro polinomio de interpolación (Lagrange o Newton) a la función original. A este respecto existe el siguiente teorema importante:

Teorema 2.2 — Calidad de la aproximación polinomial. Sea $f \in C^n[a,b]$ y p el polinomio de grado menor o igual a $n-1$ que interpola a los n pares de la forma $(x_i, f(x_i)), i \in \{1,\ldots,n\}, x_i \in [a,b]$. Entonces para todo $x \in [a,b]$ existe un $\varphi = \varphi(x) \in (a,b)$ tal que

$$f(x) - p(x) = \frac{1}{n!} f^{(n)}(\varphi(x)) \prod_{j=1}^{n} (x - x_j) \tag{2.16}$$

(Sin demostración en este curso)

Note que, a diferencia de una cota de error para una aproximación de Taylor alrededor de un punto $x*$, esta cota de error entre $p(x)$ y $f(x)$ incluye información acerca de todo el conjunto de puntos $x_1, \ldots, x_n$. Además es interesante ver que este resultado es válido para todas las funciones continuas, independientemente de que tengan una forma que provenga de un polinomio, una función trigonométrica o cualquier otra función que cumpla la condición de continuidad.

Veamos algunos ejemplos de la información que puede ser obtenida a través de este teorema:

■ **Ejemplo 2.2 — Calidad de la interpolación de la función $\cos(x)$.** Considere la interpolación polinomial a la función $f(x) = \cos(x)$ a través de un polinomio de grado 6 que interpola utilizando 7 puntos en el intervalo $[0,1]$. De acuerdo al teorema 2.2 existe un $\varphi(x)$ tal que

$$\cos(x) - p(x) = \frac{1}{7!}\operatorname{sen}(\varphi(x))\prod_{j=1}^{6}(x - x_j) \tag{2.17}$$

y sabemos que

$$|\operatorname{sen}(\varphi(x))| \leq 1 \tag{2.18}$$

y como $x \in [0,1]$ entonces se tiene que

$$|x - x_j| \leq 1 \Rightarrow \left|\prod_{j=1}^{6}(x - x_j)\right| \leq 1 \tag{2.19}$$

$$\Rightarrow |\cos(x) - p(x)| \leq \frac{1}{5040} \approx 0{,}198412x10^{-3} \tag{2.20}$$

■

■ **Ejemplo 2.3 — Calidad de la interpolacion de un polinomio.** Si $f \in P_m$ con $m \leq n-1 \Rightarrow f^{(n)} \equiv 0$, lo que indica según el Teorema 2.2 que $f(x) - p(x) = 0$. Es decir que cualquier función polinomial de grado $m \leq n-1$ será exactamente interpolada. Si $m \geq n \Rightarrow f^{(n)} \neq 0$ y el polinomio p solo será cercano a f. ■

■ **Ejemplo 2.4 — Un caso de mala calidad de la interpolacion.** Desafortunadamente, la dependencia de la cota en la derivada puede hacer también que las cosas no funcionen bien cuando la derivada no está bien acotada.
Considere la función $f(x) = (1+x^2)^{-1}$, cuyas derivadas son

$$f^{(n)}: \quad n = 1 \qquad -1(1+x^2)^{-2}(2x) \tag{2.21}$$

$$n = 2 \qquad 2\cdot 1(1+x^2)^{-3}(4x^2) + \cdots \tag{2.22}$$

$$n = 3 \qquad -3\cdot 2\cdot 1(1+x^2)^{-4}(8x^3) + \cdots \tag{2.23}$$

$$\vdots \qquad\qquad \vdots$$

$$n \qquad (-1)^n n!(1+x^2)^{-(n+1)}2^n x^n + \cdots \tag{2.24}$$

por lo que los valores de la función de derivada orden n pueden crecer enormemente. Este crecimiento puede explicarse como:

$$\left|f^{(n)}\right| \approx \mathscr{O}(2^n n!(1+x^2)^{-n-n}x) = 2^n n!\mathscr{O}(|x|^{-n-2}), \tag{2.25}$$

por lo que

$$\left|\frac{1}{n!}f^{(n)}\right| \approx 2^n\mathscr{O}(|x|^{-n-2}), \tag{2.26}$$

y el factor $2^n n!$ puede crecer fácilmente, independientemente de los valores que tome la variable x. ∎

En este último ejemplo pareciera quedar a la vista que usar mayor cantidad de puntos provocará una aproximación más pobre debido únicamente a la presencia de la derivada en la cota del teorema anterior. Esto podría resolverse restringiendo el polinomio de interpolación a través de sus derivadas, o intentando aproximar a la función original a trozos. Estos son precisamente las estrategias que abordaremos en las siguientes secciones de este capítulo.

2.2 Interpolación de Hermite

La interpolación de Hermite[3] utiliza no sólo las evaluaciones de la función original, sino también los valores de sus derivados, por ejemplo

$$\begin{array}{ccc} x_1 & x_2 & \cdots x_n \\ f(x_1) & f(x_2) & \cdots f(x_n) \\ f'(x_1) & f'(x_2) & \cdots f'(x_n) \end{array}$$

Al incluir los valores de las derivadas, se evitan comportamientos en que solo el punto es coincidente con el valor del interpolante y puede esperarse una mejor aproximación.

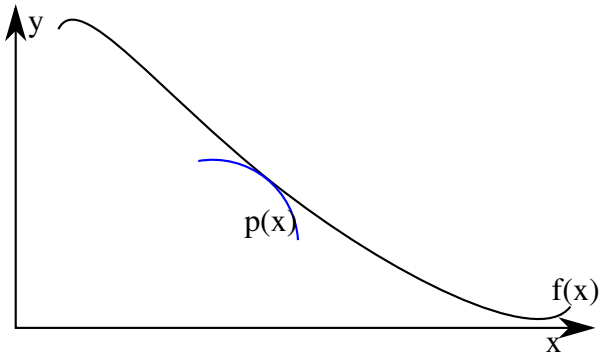


Figura 2.1: Aproximación por interpolación de Hermite

Como es de esperarse, el número de ecuaciones aumentará al incluir las derivadas para determinar el polinomio interpolante. Similar a lo que mostramos para la interpolación de Lagrange en el teorema 2.1, aquí es posible demostrar el siguiente:

Teorema 2.3 Dados n puntos distintos en $[a,b]$, $x_1, \cdots, x_n$ y dada una función $f \in$

[3]Charles Hermite $(1822-1901)$. Matemático frances, Prof. en la École Polytechnique y la Sorbonne en París. Contribuyó en teoría de números y de las funciones elípticas.

$C^1([a,b])$ existe un único polinomio p de grado mínimo tal que

$$\left.\begin{array}{l} p(x_i) = f(x_i) \\ p'(x_i) = f'(x_i) \end{array}\right\} \text{ para } i = 1,\ldots,n \tag{2.27}$$

(Sin demostración en este curso)

El polinomio p que satisface las condiciones de la interpolación de Hermite dado por la ecuación (2.27) es llamado "polinomio osculante" debido a su propiedad de tangencialidad a la función en los puntos $(x_i, f(x_i))$. De esta misma ecuación (2.27) se puede ver que el interpolante debe satisfacer $2n$ condiciones por lo que el grado del polinomio será a lo máximo $2n-1$. En la práctica muchas veces no contamos con $f'(x)$ sino solo con puntos discretos $(x_i, f(x_i))$. Para aproximar f' se puede usar el teorema del valor medio, que asegura la existencia de $\varphi \in (x_i, x_{i+1})$ tal que

$$f'(\varphi) = \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i}$$

siempre y cuando $f \in C^1([x_i, x_{i+1}])$, lo cual esta dado en nuestro caso.

Con esta idea en mente, se puede aproximar a una derivada en un punto a través de diferencias finitas utilizando alguna de las siguientes aproximaciones:

- Diferencias hacia atrás:

$$f'(x_i) \approx \frac{f(x_i) - f(x_{i-1})}{x_i - x_{i-1}} \tag{2.28}$$

- Diferencias hacia adelante:

$$f'(x_i) \approx \frac{f(x_{i+1}) - f(x_i)}{x_{i+1} - x_i} \tag{2.29}$$

- Diferencias centrales:

$$f'(x_i) \approx \frac{f(x_{i+1}) - f(x_{i-1})}{x_{i+1} - x_{i-1}} \tag{2.30}$$

Veamos como encontrar el polinomio de Hermite usando la aproximación por diferencias hacia adelante:

Buscamos un polinomio de la forma

$$p(x) = a_1 + a_2 x + \ldots + a_{n+1}x^n + a_{n+2}x^{n+1} + \ldots a_{2n}x^{2n-1} \tag{2.31}$$

con derivada

$$p'(x) = \quad a_2 + \ldots + n a_{n+1}x^{n-1} + (n+1)a_{n+2}x^n + \ldots (2n-1)a_{2n}x^{2n-2} \tag{2.32}$$

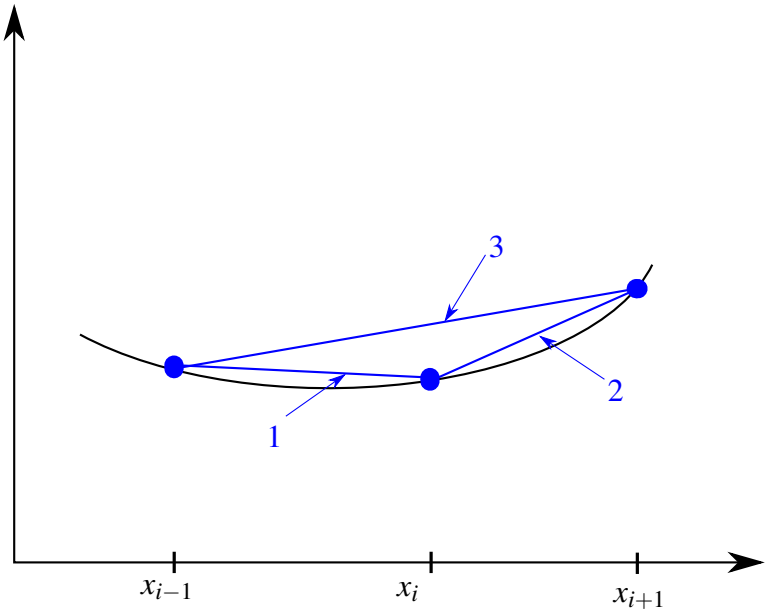


Figura 2.2: Distintas aproximaciones para la derivada: Diferencias hacia atrás (1), Diferencias hacia adelante (2) y Diferencias centrales (3).

La ecuación (2.31) con el polinomio nos da un sistema lineal de ecuaciones

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{2n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{2n-1} \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{2n} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \end{pmatrix} \tag{2.33}$$

y entonces lo que tenemos es un sistema lineal definido para $2n$ incógnitas y n ecuaciones. Las otras $n+1$ ecuaciones las obtendremos de la ecuación de la derivada. Para ello usaremos la aproximación por diferencias hacia adelante de acuerdo a la ecuación (2.28) para todos los x_i con $i = 1,\ldots,n-1$ y para x_n usaremos la aproximación por diferencias finitas hacia atrás de acuerdo con la ecuación (2.29). De esta manera, el sistema de x_1 hasta x_{n-1} quedaría como

$$\begin{bmatrix} 0 & 1 & 2x_1 & \cdots & (2n-1)x_1^{2n-2} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 1 & 2x_{n-1} & \cdots & (2n-1)x_{n-1}^{2n-2} \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{2n} \end{pmatrix} = \begin{pmatrix} \frac{f(x_2)-f(x_1)}{x_2-x_1} \\ \vdots \\ \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}} \end{pmatrix} \tag{2.34}$$

y la última ecuación para la derivada en x_n sería tomada la aproximación hacia atrás de

acuerdo a la ecuación (2.29) como

$$\begin{bmatrix} 0 & 1 & 2x_n & \cdots & (2n-1)x_n^{2n-2} \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{2n} \end{pmatrix} = \frac{f(x_n) - f(x_{n-1})}{x_n - x_{n-1}} \tag{2.35}$$

El sistema a resolver puede formarse por bloques con las ecuaciones (2.36), (2.34) y (2.35) y tendrá la forma

$$\begin{bmatrix} 1 & x_1 & x_1^2 & \cdots & x_1^{2n-1} \\ \vdots & \vdots & \vdots & & \vdots \\ 1 & x_n & x_n^2 & \cdots & x_n^{2n-1} \\ 0 & 1 & 2x_1 & \cdots & (2n-1)x_1^{2n-2} \\ \vdots & \vdots & \vdots & & \vdots \\ 0 & 1 & 2x_{n-1} & \cdots & (2n-1)x_{n-1}^{2n-2} \\ 0 & 1 & 2x_n & \cdots & (2n-1)x_n^{2n-2} \end{bmatrix} \begin{pmatrix} a_1 \\ a_2 \\ a_3 \\ \vdots \\ a_{2n} \end{pmatrix} = \begin{pmatrix} f(x_1) \\ \vdots \\ f(x_n) \\ \frac{f(x_2)-f(x_1)}{x_2-x_1} \\ \vdots \\ \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}} \\ \frac{f(x_n)-f(x_{n-1})}{x_n-x_{n-1}} \end{pmatrix} \tag{2.36}$$

o bien

$$M\bar{a} = \bar{b} \tag{2.37}$$

con $M \in \mathbb{R}^{2n\times 2n}$, $\bar{a} = (a_1, \ldots, a_{2n})^T \in \mathbb{R}^{2n}$, $\bar{b} \in \mathbb{R}^{2n}$.

La construcción de M y $\bar{b}$ es muy sencilla de realizar en Matlab, R o cualquier lenguaje de programación. Con esta interpolación se espera que las problemáticas de polinomios que "exploten" en áreas donde no hay datos, sean disminuidos o eliminados. Sin embargo notemos dos posibles problemas:

- En un problema práctico, pudiera ser que la cantidad de datos que se tiene sea muy grande, haciendo que un sistema de tamaño $2n \times 2n$ deba ser mantenido en memoria computacional y quizá sobrepasando el límite de datos a ser manejado con sencillez. Esto se extiende a la capacidad y confiabilidad al resolver el sistema $M\bar{a} = \bar{b}$.
- La construcción de la matriz M implica cálculos de grandes potencias de x (hasta x^{2n-1}). En caso de que n sea medianamente grande y $x \gg 1$, estos valores pueden crecer e incluso llegar al área de overflow (obteniendo valores de $\pm Inf$ en nuestra aritmética de punto flotante). Del mismo modo si $x \ll 1$, puede llegarse fácilmente al underflow.

Como ya hemos visto y experimentado, utilizar polinomios para interpolar puntos se convierte en un problema con grandes oscilaciones en el resultado. Esto es conocido como el fenómeno de Runge[4], quien encontró esta problemática y la asoció a que, a

[4]Carl David Tolmé Runge $(1856-1927)$. Matemático alemán, estudiante de Weierstrass que trabajó con Felix Klein desde su Lehrstuhl en Göttingen.

pesar de que aunque la teoría dada por el teorema de Weierstrass [5] asegura que toda función continua en $[a,b]$ tiene un conjunto de polinomios en $P_1, P_2, \ldots$ que converge uniformemente a f, este teorema no da ninguna estrategia para encontrar dichos polinomios en la práctica.

Una alternativa es no intentar resolver el problema de manera global en el intervalo $[x_1, x_n]$, sino resolver pequeños problemas en subintervalos de este dominio, con la esperanza de que usando cada vez solo unos pocos puntos el grado polinomial se mantenga bajo.

2.3 Splines

Desde la década de los 1920's estaba claro que la interpolación polinomial contenía el problemático fenómeno de Runge. Sin embargo no fue sino hasta los trabajos de Schoenberg[6] en la década de los 1940's cuando se propuso una técnica que evitara este tipo de problema.

La idea básica es unir a la sucesión de puntos

$$a = x_1 < x_2 < \cdots < x_n = b \tag{2.38}$$

evaluados en la función $f : [a,b] \to \mathbb{R}$ de manera que en cada subintervalo $[x_i, x_{i+1}]$ se tenga un polinomio capaz de describir una trayectoria suave y cuyo grado polinomial sea de preferencia pequeño.

Esto puede verse que será resuelto con polinomios de un orden mayor o igual que tres, por lo que en la práctica son los de orden cúbico los que más se utilizan. Por esta razón nos concentraremos en este tipo de splines.

Definición 2.4 — Splines Cúbicos. Considere $f : [a,b] \to \mathbb{R}$ y un conjunto de nodos $a = x_1 < x_2 < \cdots < x_n = b$. Un spline cúbico para f es una función $S : [a,b] \to \mathbb{R}$ tal que se cumple:

(i) $S|_{[x_j, x_{j+1}]} = S_j$ donde $S_j \in \mathbb{P}_3([x_j, x_{j+1}] \quad \forall j \in \{1,2,\ldots,n-1\}$
(ii) $S(x_j) = f(x_j) \quad \forall j \in \{1,\ldots,n\}$
(iii) $S_{j+1}(x_{j+1}) = S_j(x_{j+1}) \quad \forall j \in \{1,\ldots,n-2\}$
(iv) $S'_{j+1}(x_{j+1}) = S'_j(x_{j+1}) \quad \forall j \in \{1,\ldots,n-2\}$
(v) $S''_{j+1}(x_{j+1}) = S''_j(x_{j+1}) \quad \forall j \in \{1,\ldots,n-2\}$
(vi) Una de las siguientes condiciones de frontera se satisfacen
  (a) $S''(x_1) = S''(x_n) = 0$ (frontera libre)

[5] Karl Weierstrass (1815 – 1897). Matemático alemán llamado el "padre del análisis moderno". Formalizó conceptos del cálculo y el análisis como la definición de función continua. Teorema de Borel-Weierstrass.

[6] Matemático de origen Rumano. Estudió en Berlin y Göttingen con Edmund Landau y desde 1930 trabajo en los EEUU llegando a ser Profesor de la University of Pennsylvania. En 1943-1945 fue liberado de su puesto para unirse al grupo de científicos trabajando en el Aberdeen Proving Ground (el sitio de investigación más antiguo de la armada estadounidense) donde realizó su mayor contribución científica: los splines.

(b) $S'(x_1) = f'(x_1)$, $S'(x_n) = f'(x_n)$ (frontera sujeta)
(c) Una mezcla de (a) y (b), por ejemplo $S''(x_1) = 0$, $S'(x_n) = f'(x_n)$

Con el fin de simplificar los desarrollos de construcción del spline cúbico denotaremos a los polinomios S_j con coeficientes a_j, b_j, c_j, d_j de la forma

$$S_j(x) = a_j + b_j(x - x_j) + c_j(x - x_j)^2 + d_j(x - x_j)^3. \tag{2.39}$$

Este polinomio corresponde a la función S en $[x_j, x_{j+1}]$ y la completa determinación del spline consiste en encontrar los coeficientes para los trozos $S_1, S_2, \ldots, S_{n-1}$. El polinomio tiene las primeras dos derivadas dadas por

$$\begin{aligned} S'_j(x) &= b_j + 2c_j(x - x_j) + 3d_j(x - x_j)^2, & (2.40)\\ S''_j(x) &= 2c_j + 6d_j(x - x_j). & (2.41) \end{aligned}$$

Ahora, considerando la condición de interpolación (ii) de la definición 2.4 debe cumplirse que $S_j(x_j) = f(x_j)$ por lo que al evaluar el polinomio de la ecuación (2.39) en el punto x_j tendremos que

$$\Rightarrow\ a_j = f(x_j). \tag{2.42}$$

Usando el hecho que S_j y S_{j+1} deben coincidir en el punto x_{j+1} según la condición (iii) de la definición 2.4, se tiene que

$$\begin{aligned} S_{j+1}(x_{j+1}) &= a_{j+1}, \\ &= a_j + b_j(x_{j+1} - x_j) + c_j(x_{j+1} - x_j)^2 + d_j(x_{j+1} - x_j)^3. \end{aligned} \tag{2.43}$$

En estos desarrollos estamos considerando un conjunto de puntos x_j que no necesariamente estan igualmente espaciados entre ellos. Si denotamos los espaciamientos entre x_j y x_{j+1} como h_j tendríamos que la ecuación anterior se convierte en

$$a_{j+1} = a_j + b_j h_j + c_j h_j^2 + d_j h_j^3. \tag{2.44}$$

Usando ahora la condición (iv) en la definición 2.4 respecto a la primera derivada tenemos

$$b_{j+1} = b_j + 2c_j(x_{j+1} - x_j) + 3d_j(x_{j+1} - x_j)^2, \tag{2.45}$$

o bien

$$b_{j+1} = b_j + 2c_j h_j + 3d_j h_j^2. \tag{2.46}$$

Análogamente, usando la condición (v) de la definición 2.4 correspondiente a la segunda derivada tendremos

$$2c_{j+1} = 2c_j + 6d_j(x_{j+1} - x_j), \tag{2.47}$$

o bien

$$2c_{j+1} = 2c_j + 6d_j h_j, \tag{2.48}$$

que al despejar d_j se convierte en

$$d_j = \frac{c_{j+1} - c_j}{3h_j}. \tag{2.49}$$

Reemplacemos ahora este valor de d_j en (2.49) en la ecuación (2.44) para obtener

$$0 = (a_{j+1} - a_j) - b_j h_j - c_j h_j^2 - \frac{h_j^2}{3}(c_{j+1} - c_j), \tag{2.50}$$

y observemos que esto nos permite ahora despejar el valor de b_j como

$$b_j h_j = (a_{j+1} - a_j) - \frac{h_j^2}{3}(c_{j+1} + 2c_j), \tag{2.51}$$

$$\Rightarrow\ b_j = \frac{1}{h_j}(a_{j+1} - a_j) - \frac{h_j}{3}(c_{j+1} + 2c_j), \tag{2.52}$$

y como el índice j es cualquiera de los índices posibles, podemos considerar esta misma expresión para un índice anterior $j-1$ de manera análoga como

$$b_{j-1} = \frac{1}{h_{j-1}}(a_j - a_{j-1}) - \frac{h_{j-1}}{3}(c_j + 2c_{j-1}). \tag{2.53}$$

Reemplacemos ahora el valor de d_j que obtuvimos en (2.49) en la ecuación (2.46)

$$\begin{aligned} b_{j+1} \quad &= b_j + 2c_j h_j + 3h_j^2 \frac{c_{j+1} - c_j}{3h_j} \\ &= b_j + 2c_j h_j + h_j(c_{j+1} - c_j) \\ &= b_j + h_j(c_j + c_{j+1}) \end{aligned} \tag{2.54}$$

Análogamente, esta ecuación puede escribirse para los índices j y $(j-1)$ como

$$b_{j-1} + h_{j-1}(c_{j-1} + c_j) = b_j \tag{2.55}$$

Reemplazando ahora la ecuación (2.53) en el lado izquierdo de esta ecuación y la ecuación (2.52) en el lado derecho de la misma se obtiene

$$\frac{1}{h_{j-1}}(a_j - a_{j-1}) - \frac{h_{j-1}}{3}(c_j + 2c_{j-1}) + h_{j-1}(c_{j-1} + c_j) = \frac{1}{h_j}(a_{j+1} - a_j) - \frac{h_j}{3}(c_{j+1} + 2c_j) \tag{2.56}$$

Reordenando para tener coeficientes c del lado izquierdo y coeficientes a del lado derecho, así como multiplicando todos los términos por 3 se llega a

$$h_j(c_{j+1}+2c_j)-h_{j-1}(c_j+2c_{j-1})+3h_{j-1}(c_{j-1}+c_j)=\frac{3}{h_j}(a_{j+1}-a_j)-\frac{3}{h_{j-1}}(a_j-a_{j-1}) \tag{2.57}$$

Ahora que el lado derecho de la ecuación (2.57) solo contiene coeficientes a_j's y el izquierdo solo c_j's podemos reescribir el lado izquierdo como

$$\begin{aligned} c_{j-1}(-2h_{j-1}+3h_{j-1})+c_j(2h_j-h_{j-1}+3h_{j-1})+c_{j+1}(h_j) \\ =h_{j-1}c_{j-1}+2(h_j+h_{j-1})c_j+h_jc_{j+1} \\ =\begin{bmatrix} h_{j-1} & 2(h_j+h_{j-1}) & h_j \end{bmatrix}\begin{bmatrix} c_{j-1} \\ c_j \\ c_{j+1} \end{bmatrix}, \end{aligned} \tag{2.58}$$

por lo que si ahora definimos las nuevas variables auxiliares

$$t_j=\frac{3}{h_j}(a_{j+1}-a_j)-\frac{3}{h_{j-1}}(a_j-a_{j-1}), \tag{2.59}$$

podemos usar la formulación en (2.58) para reescribir la ecuación (2.57) como

$$h_{j-1}c_{j-1}+2(h_j+h_{j-1})c_j+h_jc_{j+1}=t_j. \tag{2.60}$$

Esta ecuación puede reescribirse para todos los puntos interiores de nuestro problema de encontrar los polinomios cúbicos, es decir, que podemos reescribirla para $j\in\{2,\ldots,n-2\}$, resultando en el sistema lineal de ecuaciones

$$\begin{bmatrix} h_1 & 2(h_1+h_2) & h_2 & 0 & \cdots & 0 \\ 0 & h_2 & 2(h_2+h_3) & h_3 & \cdots & 0 \\ 0 & 0 & h_3 & 2(h_3+h_4) & \cdots & 0 \\ 0 & 0 & 0 & h_4 & \cdots & 0 \\ \vdots & \vdots & & & & \\ 0 & 0 & 0 & 0 & \cdots & h_{n-1} \end{bmatrix}\begin{bmatrix} c_1 \\ c_2 \\ c_3 \\ c_4 \\ \vdots \\ c_{n-1} \end{bmatrix}=\begin{bmatrix} t_2 \\ t_3 \\ t_4 \\ t_5 \\ \vdots \\ t_{n-1} \end{bmatrix} \tag{2.61}$$

La matriz del sistema es de tamaño $(n-2)\times n$ y es tridiagonal por lo que se requieren dos ecuaciones más para poder cerrar el sistema. Estas ecuaciones faltantes corresponden a lo que se espera de los puntos frontera x_1 y x_n con respecto a la condición (vi) de la definición 2.4.

Para el caso de la condición de frontera libre en x_1 se tendría la condición de que $S_1''(x_1)=0$, y basta revisar la forma de la segunda derivada en la ecuación (2.41) para ver que el primer trozo del Spline debería cumplir que

$$0=2c_1+6d_1(x_1-x_1), \tag{2.62}$$

por lo que se tiene que $c_1 = 0$, o equivalentemente

$$\begin{bmatrix} 1 & 0 & \cdots & 0 \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} = 0. \tag{2.63}$$

De manera similar, la condición de frontera libre en x_n corresponde a tener que la ecuación (2.41) en el último trozo del spline se convierte en la condición $c_{n-1} = 0$, o equivalentemente

$$\begin{bmatrix} 0 & 0 & \cdots & 1 \end{bmatrix} \begin{bmatrix} c_1 \\ \vdots \\ c_{n-1} \end{bmatrix} = 0, \tag{2.64}$$

y el sistema completo de ecuaciones puede formarse haciendo uso del sistema en (2.61) y complementándolo con las ecuaciones (2.63) y (2.64) para formar un sistema para las $n-1$ variables c_j.

Con esto, el sistema completo para el caso de fronteras libres es de la forma $Mc = F$ con

$$M = \begin{bmatrix} 1 & 0 & 0 & 0 & \cdots & 0 & 0 & 0 \\ h_1 & 2(h_1+h_2) & h_2 & 0 & \cdots & 0 & 0 & 0 \\ & h_2 & 2(h_2+h_3) & h_3 & \cdots & 0 & 0 & 0 \\ & & \ddots & \ddots & \ddots & & & \\ & & & & & & & \\ & & & & & h_{n-2} & 2(h_{n-2}+h_{n-1}) & h_{n-1} \\ 0 & 0 & 0 & 0 & \cdots & 0 & 0 & 1 \end{bmatrix}_{n\times n} \tag{2.65}$$

$$\begin{aligned} c &= \begin{bmatrix} c_1 & c_2 & \cdots & c_n \end{bmatrix}^T \in R^n \\ F &= \begin{bmatrix} 0 & t_2 & t_3 & \cdots & t_{n-1} & 0 \end{bmatrix}^T \in R^n \end{aligned} \tag{2.66}$$

Otra posibilidad en la frontera de x_1 es el caso de frontera fija o sujeta, en la que se asume que conocemos la primera derivada como un valor fijo $S_1'(x_1)$. Usando la ecuación (2.40) esto queda como

$$S_1'(x_1) = b_1 + 2c_1(x_1 - x_1) + 3d_j(x_1 - x_1)^2, \tag{2.67}$$

es decir que $b_1 = S_1'(x_1)$ y dado que ya sabemos que la ecuación (2.42) define claramente a los a_j's, podemos usar la ecuación (2.53) como

$$b_1 = \frac{1}{h_1}(a_2 - a_1) - \frac{h_1}{3}(2c_1 + c_2) \tag{2.68}$$

o bien

$$2h_1c_1 + h_1c_2 = \frac{3}{h_1}(a_2 - a_1) - 3S_1'(x_1) \tag{2.69}$$

De manera análoga, si se tiene una condición de frontera sujeta en x_n quiere decir que conocemos el valor de $S_{n-1}'(x_n)$, es decir

$$S_{n-1}'(x_n) = b_{n-1} + 2c_{n-1}h_{n-1} + 3d_{n-1}h_{n-1}^2 \tag{2.70}$$

y haciendo uso de (2.49) y (2.53) esto puede reescribirse como

$$-3S_{n-1}'(x_n) = -\frac{3}{h_{n-1}}(a_n - a_{n-1}) + h_{n-1}(2c_{n-1} + c_n) - 6c_{n-1}h_{n-1} - 3h_{n-1}^2\frac{c_n - c_{n-1}}{h_{n-1}}. \tag{2.71}$$

Reordenando términos se tiene que

$$\frac{3}{h_{n-1}}(a_n - a_{n-1}) - 3S'(x_n) = (2h_{n-1} - 6h_{n-1} + 3h_{n-1})c_{n-1} + (h_{n-1} - 3h_{n-1})c_n, \tag{2.72}$$

o bien

$$h_{n-1}c_{n-1} + 2h_{n-1}c_n = 3S'(x_n) - \frac{3}{h_{n-1}}(a_n - a_{n-1}). \tag{2.73}$$

Estas ecuaciones (2.69) y (2.73) servirán para ampliar el sistema de ecuaciones y tener una matriz de tamaño $n \times n$ que, usando fronteras fijas puede formarse muy similar al sistema de las ecuaciones (2.65) y (2.74) pero intercambiando el primer y último renglón de M por

$$\begin{aligned} M_1 &= \begin{bmatrix} 2h_1 & h_1 & 0 & \cdots & 0 \end{bmatrix} \\ M_n &= \begin{bmatrix} 0 & \cdots & 0 & h_{n-1} & 2h_{n-1} \end{bmatrix} \end{aligned} \tag{2.74}$$

de acuerdo a las ecuaciones (2.69) y (2.73) respectivamente. Los cambios correspondientes al lado derecho F también deben hacerse de la forma

$$\begin{aligned} F_1 &= -3f'(x_1) + \frac{3}{h_1}(a_2 - a_1) \\ F_2 &= 3f'(x_n) - \frac{3}{h_{n-1}}(a_n - a_{n-1}) \end{aligned} \tag{2.75}$$

En caso de que las condiciones de frontera sean distintas en x_1 y x_n, solamente el primer o último renglón del sistema deberán ser modificados.

Con esto, sabemos como evaluar los coeficientes c_j's de los trozos polinomiales S_j. Sin embargo, el algoritmo completo debe calcular todos los coeficientes. Usando todos los desarrollos anteriores puede escribirse un algoritmo completo para encontrar el spline cúbico que los interpola de la forma descrita a continuación.

Dado los puntos $x_1, \ldots, x_n$ y sus respectivos valores $f(x_1), \ldots, f(x_n)$, realice os siguientes pasos:

- Defina $a_j = f(x_j)$, $j = 1, \ldots, n-1$
- Obtenga $c_1, \ldots, c_n$ resolviendo el sistema lineal definido por la matriz y lado derecho en las ecuaciones (2.65), (2.66) . En caso de condiciones de frontera fija, modifique este sistema de acuerdo a las condiciones en (2.74) y (2.75).
- Calcule los coeficientes b_j de acuerdo a la ecuación (2.53) como

$$b_j = \frac{1}{h_j}(a_{j+1} - a_j) - \frac{h_j}{3}(c_{j+1} + 2c_j)$$

- Calcule los coeficientes d_j de acuerdo a la ecuación (2.49) como

$$d_j = \frac{1}{3h_j}(c_{j+1} - c_j)$$

Una vez qe el spline cúbico es conocido, puede implementarse un algoritmo simple para la evaluación del mismo a través de la forma original definido en la condición (i) de la definición 2.4.

2.4 Ejercicios

En estos ejercicios, se intenta realizar observaciones sobre los métodos de interpolación. Para ello, deberán considerarse los tiempos de cálculo y la complejidad de implementación inherentes a los métodos, sin considerar los tiempos referentes al pre-procesamiento y post-procesamiento de datos. Ejemplos de estos tiempos son la selección de datos a interpolar o los comandos utilizados para realizar gráficos ilustrativos del problema. Para esto, es recomendable que las funciones generales de los métodos tengan argumentos de salida, entre los cuales se encuentre el tiempo usado para realizar los cálculos.

Ejercicio 2.1. Interpolación de Lagrange (35 Puntos)
Defina una función (en MATLAB, R, etc.) para interpolar por el método de Lagrange y úsela para aproximar a la función $f : [0,8] \to \mathbb{R}$, $f(x) = \text{sen}(x)$ utilizando n puntos igualmente espaciados. Repita este ejercicio para $n = 2, 4, 7, 11, 16$ y haga observaciones sobre la calidad en las aproximaciones. ■

Ejercicio 2.2. Interpolación de Newton y función de Runge (35 Puntos)
Considere la función de Runge

$$f(x) = \frac{1}{1 + 25x^2}, \qquad x \in [-1, 1].$$

Use una implementación de la interpolación de Newton (en MATLAB, R, etc.) para aproximar a esta función en $[-1, 1]$ utilizando 5, 9, 13, 17 y 21 puntos igualmente espaciados. Observe como el grado polinomial afecta la calidad de la aproximación.

Realice una aproximación similar usando el programa de interpolación de Lagrange del ejercicio anterior. ¿Son los resultados mejores/peores/equivalentes? ■

Ejercicio 2.3. Comparando interpolaciones (30 Puntos)
Defina dos siluetas simples que puedan representarse como un conjunto de 2 a 5 funciones en $\mathbb{R}^2$, use al menos una silueta con solo 3 puntos y otra con al menos 10 puntos. Implemente un programa que utilice interpolación de Lagrange y de Newton y compare los resultados de ambas aproximaciones en términos de 'calidad visual' (subjetivo), eficiencia en tiempo de calculo y complejidad de la implementación. ■

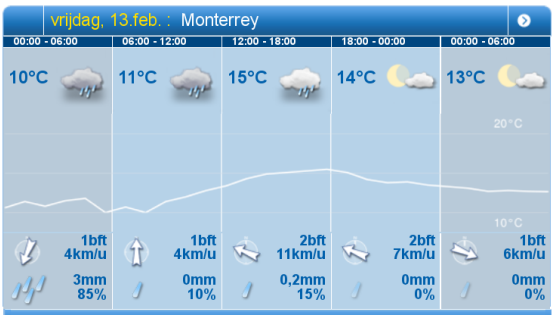


Figura 2.3: Temperatura pronosticada para Monterrey el día 13.02.2015

Los modelos de predicción climática se basan en la inclusión de información sobre corrientes de presión, archivos históricos, estado actual de la temperatura, humedad, y una serie amplia de factores con influencia climática. La compleja relación de todos estos factores resulta en modelos basados en una gran cantidad de datos que no es posible actualizar de manera constante.
Por esta razón, en ocasiones se toman predicciones en periodos largos de tiempo (del orden de 12 o 24 horas) para predecir el estado del tiempo y con base en estos valores se calcula indirectamente una predicción para cada hora del día, basada principalmente en una interpolación de los valores predichos para el periodo largo de tiempo.

Ejercicio 2.4. Interpolando Temperaturas (25 Puntos)
Asuma que las predicciones diarias de temperatura son conocidas para dos horas representativas (por ejemplo al amanecer y después del mediodía) en un conjunto de varios días. Diseñe un pseudo-código por escrito en el que describa cómo pueden ser calculadas las predicciones en intervalos de 1 hora, incluyendo la información sobre los datos de entrada y salida de cada parte del pseudo-código.

En su diseño no es necesario que describa el pseudo-código para calcular la interpolación. Asuma que existe la función

```
[evaly,execTime] = Interpolate(dataX,dataY,evalx)
```

que acepta los datos a interpolar y un vector de puntos a evaluar, y retorna un vector (*evaly*) de valores evaluados y la cantidad de segundos utilizada para calcular la interpolación (*execTime*). ∎

Ejercicio 2.5. Interpolación de Hermite Generalizada (25 Puntos)

La generalización del método de interpolación de Hermite puede ser detreminada a través de requerir que el polinomio interpolante p cumpla con las condiciones:

$$\left.\begin{array}{r} p(x_i) = f(x_i) \\ p'(x_i) = f'(x_i) \\ p''(x_i) = f''(x_i) \\ \vdots \\ p^{(m)}(x_i) = f^{(m)}(x_i) \end{array}\right\} \text{ para } i = 1,\ldots,n \tag{2.76}$$

Revise el método de solución de la Sección 2.2 y transforme el problema generalizado de Hermite en un problema de resolver un sistema lineal de ecuaciones. ¿qué grado deberá tener el polinomio considerado? ¿cómo se puede atacar la problematica de no conocer las derivadas? ∎

Ejercicio 2.6. Interpolación de Hermite (50 Puntos)

Implemente rutinas en MATLAB para una función de interpolación utilizando el método de Hermite. Utilice la aproximación a las derivadas por diferencias hacia adelante $\frac{dy}{dx} \approx \frac{y_{i+1}-y_i}{x_{i+1}-x_i}$ para todos los puntos x_i, $i = 1,\ldots,n-1$.

Consulte las predicciones diarias de temperatura en un servicio de Internet para los siguientes días y a dos horas representativas. Pruebe su código de inerpolación de Hermite realizando las predicciones de la temperatura en intervalos de 1 hora. Construya los resultados y represéntelos de forma gráfica en el espacio tiempo-temperatura. Use comandos de MATLAB (`xtick,grid on,xlabel,ylabel,plot,...`) para lograr un gráfico que muestre los bloques de 24 horas, y que incluya la descripción y unidades en los ejes. Además, use las opciones que ofrece la función `plot` para graficar como asteriscos los datos originales tomados de Internet. ∎

Ejercicio 2.7. Implementación de splines cúbicos (40 Puntos)

Implemente un código para el cálculo del spline cúbico pasando por una lista de n puntos de la forma (X,Y) usando condiciones de frontera libre en ambos lados. Use preferentemente versiones vectoriales para las evaluaciones de las operaciones que así lo permitan. ∎

Ejercicio 2.8. Splines cúbicos con frontera fija (15 Puntos)

Modifique el código del ejercicio anterior para generar un programa que permita el

uso de condiciones de frontera fija. ■

Ejercicio 2.9. Spline para la fución de Runge (15 Puntos)
Use alguno de los códigos implementados para e cálculo de splines y construya una aproximación a la función de Runge

$$f(x) = \frac{1}{1+25x^2}, \qquad x \in [-1,1].$$

Realice esta aproximación usando puntos igualmente espaciados y comente sobre la cantidad de puntos que, según su criterio, son suficientes para lograr buenas aproximaciones. ■

Ejercicio 2.10. Utilizando los Splines cúbicos (15 Puntos)
Diseñe alguna silueta utilizando splines cúbicos. Trate de combinar ambas versiones para las condiciones de frontera. ■

3. Integración Numérica

Los usos prácticos de las integrales (cálculo de áreas, longitudes de curvas, volúmenes de sólidos, cálculos de FEM, etc) hacen que la necesidad de resolver integrales de manera numérica sea una tarea imprescindible. En muchos casos, es imposible integrar funciones de manera analítica dada su complejidad o incluso debido a que no existen funciones primitivas básicas que puedan ser utilizadas para ello.

En este capítulo nos concentraremos en resolver el problema de aproximación numérica a la integral de una función real, es decir $f : [a,b] \subset \mathbb{R} \to \mathbb{R}$. Las consideraciones para problemas en mayores dimensiones (por ejemplo en $\mathbb{R}^2$, $\mathbb{R}^3$) pueden generalizarse fácilmente de los contenidos presentados en este capítulo.

3.1 Reglas básicas de integración numérica

Empezaremos por considerar el problema de integrar una función $f : \mathbb{R} \to \mathbb{R}$, más precisamente $f : [a,b] \to \mathbb{R}$ donde $a,b \in \mathbb{R}$, $a < b$. Por otro lado, es suficiente con resolver la integración para una función definida en el $[0,1]$, pues el paso entre una función $\tilde{f} : [a,b] \to \mathbb{R}$ a otra de la forma $f : [0,1] \to \mathbb{R}$ puede realizarse fácilmente con un reescalamiento

$$x = \frac{1}{b-a}(\tilde{x} - a). \tag{3.1}$$

Desde los primeros conocimientos de integración se habla de que la integral $\int_0^1 f(x)dx$ corresponde al valor del área debajo de la curva $f(x)$ y se presenta la idea de que el área puede ser aproximada por el área de una gráfica de barras por debajo de la gráfica de

f. Incluso hay una conexión estrecha con la Integral de Riemann que se enseña en los cursos de cálculo integral. Como veremos a continuación, las estrategias de integración numérica más básicas retoman este concepto, usando luego algunas generalizaciones para obtener buenas aproximaciones al valor exacto de la integral.

Lo más simple que se puede hacer es tomar algún valor de entre todos los valores de $f(x)$ para algún $x \in [0,1]$ y tomar el área bajo la barra de esa altura tal como se muestra en la Figura 3.1, donde se tomó un punto $x_0 = 0{,}5$ y la barra de altura $f(x_0)$ y ancho 1 para aproximar al área bajo la curva de f usando el área de este rectángulo.

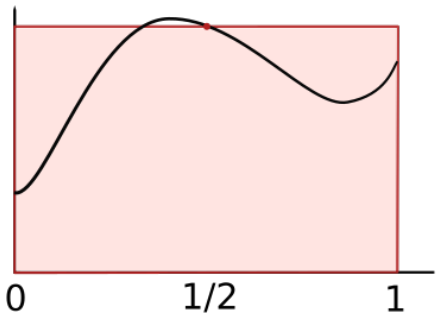


Figura 3.1: Área bajo la función f aproximada por el área del rectángulo de ancho 1 y altura $f(x_0)$ para algún $x_0 \in [0,1]$.

Claramente, el tomar un $x_0 \in [0,1]$ y usar el área bajo la línea con altura $f(x_0)$ dará un valor aproximado $\int_a^b f(x)dx = 1 \cdot f(x_0) = f(x_0)$ y esto será una aproximación muy sujeta a grandes errores para una amplia diversidad de funciones. Las mejoras posibles a esta estrategia son:

- Considerar el área bajo una curva que represente a una función más compleja que una constante.
- Considerar más de una barra de altura constante (histograma).

La segunda opción la analizaremos más adelante. Para la primera opción resulta natural tomar algo más complejo que una función constante, es decir una aproximación polinomial de grado mayor que cero (la constante).

Usando una aproximación en $\mathbb{P}_1$, basta con tomar $x_1 = a$, $x_2 = b$ y construir la línea que pasa por los puntos $(a, f(a))$, $(b, f(b))$. De esta manera, la aproximación ilustrada en la Figura 3.2 estará dada como

$$\int_0^1 f(x)dx = (f(1) + f(2))\frac{1}{2}. \tag{3.2}$$

Por otro lado conocemos la representación de este elemento de $\mathbb{P}_1$ en forma de Lagrange como

$$p_1(x) = \sum_{j=1}^{2} f(x_j)\ell_j(x), \tag{3.3}$$

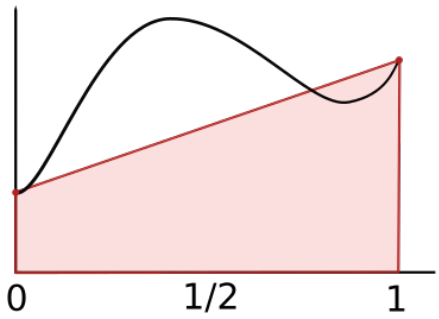


Figura 3.2: Aproximación usando un polinomio lineal a través del área de un trapecio.

con los polinomios

$$\ell_1(x) = \prod_{i=1,i\neq 1}^{2} \frac{x-x_i}{x_1-x_i} = \frac{x-x_2}{x_1-x_2} = \frac{x-1}{0-1} = 1-x, \tag{3.4}$$

$$\ell_2(x) = \prod_{i=1,i\neq 2}^{2} \frac{x-x_i}{x_2-x_i} = \frac{x-x_1}{x_2-x_1} = \frac{x-0}{1-0} = x. \tag{3.5}$$

Así que tomando el polinomio $p(x)$ como aproximación se tiene que

$$\begin{aligned}
\int_0^1 f(x)dx &\approx \int_0^1 p_1(x)dx, \\
&= \int_0^1 \sum_{j=1}^{2} f(x_j)\ell_j(x)dx, \\
&= f(0)\int_0^1 xdx + f(1)\int_0^1 (1-x)dx, \\
&= f(0)\left[\frac{x^2}{2}\right]_0^1 + f(1)\left[x-\frac{x^2}{2}\right]_0^1, \\
&= f(0)\frac{1}{2} + f(1)\frac{1}{2}, \\
&= \frac{1}{2}\left(f(0)+f(1)\right).
\end{aligned} \tag{3.6}$$

A esta fórmula de integración usando un polinomio de primer grado se le conoce como la regla del trapecio.

De manera similar, podemos usar una construcción similar para obtener una aproximación utilizando un polinomio cuadrático ($p \in \mathbb{P}_2$) con la base de Lagrange usando los nodos $\{0,\ 1/2,\ 1\}$. Esta base puede verse fácilmente que está formada por los

polinomios

$$\ell_0^{(2)}(x) = \frac{x-1/2}{0-1/2}\cdot\frac{x-1}{0-1} = (-2x+1)(1-x) = 2x^2-3x+1 \tag{3.7}$$

$$\ell_1^{(2)}(x) = \frac{x-0}{1/2-0}\cdot\frac{x-1}{1/2-1} = (2x)(-2x+2) = -4x^2+4x \tag{3.8}$$

$$\ell_2^{(2)}(x) = \frac{x-0}{1-0}\cdot\frac{x-1/2}{1-1/2} = (x)(2x-1) = 2x^2-x \tag{3.9}$$

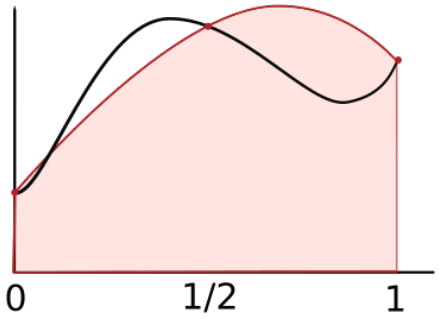


Figura 3.3: Aproximación mediante la regla de Simpson

Y la aproximación a la integral de $f(x)$ haciendo uso del mejor polinomio en $\mathbb{P}_2$ que aproxima a esta función es ta como se ilustra en la Figura 3.3:

$$\begin{aligned}
\int_0^1 f(x)dx &\approx \int_0^1 p_2(x) \\
&= f(0)\int_0^1 (2x^2-3x+1)dx + f\left(\frac{1}{2}\right)\int_0^1 (-4x^2+4x)dx + \cdots \\
&\quad \cdots + f(1)\int_0^1 (2x^2-x)dx \\
&= f(0)\left[\frac{2}{3}x^3-\frac{3}{2}x^2+x\right]_0^1 + f\left(\frac{1}{2}\right)\left[-\frac{4}{3}x^3+2x^2\right]_0^1 + \cdots \\
&\quad \cdots + f(1)\left[\frac{2}{3}x^3-\frac{x^2}{2}\right]_0^1 \\
&= f(0)\frac{1}{6} + f\left(\frac{1}{2}\right)\frac{2}{3} + f(1)\frac{1}{6} \\
&= \frac{1}{6}\left[f(0)+4f\left(\frac{1}{2}\right)+f(1)\right]
\end{aligned} \tag{3.10}$$

Esta aproximación es conocida como la Regla de Simpson [1] para la integración y corresponde a integrar la función cuadrática que pasa por los puntos $(0, f(0))$, $\left(\frac{1}{2}, f\left(\frac{1}{2}\right)\right)$, $(1, f(1))$.

[1] Thomas Simpson (1710 – 1761). Matemático Británico a quien se le atribuye este desarrollo. Sin embargo, Johames Kepler ya lo había usado 100 años antes que Simpson.

De manera similar a las ecuaciones (3.6) y (3.10), el proceso de tomar un polinomio en $\mathbb{P}_n$ para algún $n > 2$ puede realizarse para obtener una fórmula similar. Para el caso $n = 3$ existe la regla conocida como Regla de Simpson de $3/8$ y para $n = 4$ se conoce como Regla de Milne. La siguiente definición contiene un resumen de las reglas de integración para polinomios de orden 1 hasta 4.

Definición 3.1 — Reglas de integración para $0 \leq x \leq 1$**.** La integral en el intervalo unitario $\int_0^1 f(x)dx$ puede ser aproximada utilizando las reglas de integración para polinomios. Estas reglas pueden resumirse como

Grado	Regla	Nodos / Fórmula
$\mathbb{P}_0$	Punto medio	$\{\frac{1}{2}\}$
		$f\left(\frac{1}{2}\right)$
$\mathbb{P}_1$	Trapecio	$\{0,1\}$
		$\frac{1}{2}(f(0)+f(1))$
$\mathbb{P}_2$	Simpson	$\{0,\frac{1}{2},1\}$
		$\frac{1}{6}(f(0)+4f\left(\frac{1}{2}\right)+f(1))$
$\mathbb{P}_3$	Simpson $\frac{3}{8}$	$\{0,\frac{1}{3},\frac{2}{3},1\}$
		$\frac{1}{8}(f(0)+3f\left(\frac{1}{3}\right)+3f\left(\frac{2}{3}\right)+f(1))$
$\mathbb{P}_4$	Milne	$\{0,\frac{1}{4},\frac{1}{2},\frac{3}{4},1\}$
		$\frac{1}{90}\left(7f(0)+32f\left(\frac{1}{4}\right)+12f\left(\frac{1}{2}\right)+32f\left(\frac{3}{4}\right)+7f(1)\right)$

En caso de que la integral a calcular se requiera en un intervalo más general $[a,b]$, basta con considerar el reescalamiento de los nodos para obtener las versiones modificadas de las reglas contenidas en la siguiente definición:

Definición 3.2 — Reglas de integración para $a \leq x \leq b$**.** La integral en el intervalo unitario $\int_0^1 f(x)dx$ puede ser aproximada utilizando las reglas de integración para polinomios. Estas reglas pueden resumirse como
Regla del punto medio

$$(b-a)f\left(\frac{a+b}{2}\right) \tag{3.11}$$

Regla del trapecio

$$\left(\frac{b-a}{2}\right)[f(a)+f(b)] \tag{3.12}$$

Regla de Simpson

$$\left(\frac{b-a}{6}\right)\left[f(a)+4f\left(\frac{a+b}{2}\right)+f(b)\right] \tag{3.13}$$

Regla de Simpson de 3/8

$$+3f\left(a+2\frac{b-a}{3}\right)+f(b)\Big] \tag{3.14}$$

Regla de Milne

$$\left(\frac{b-a}{90}\right)\left[7f(a)+32f\left(a+\frac{b-a}{4}\right)+12f\left(a+\frac{b-a}{2}\right)+32f\left(a+3\frac{b-a}{4}\right)+7f(b)\right] \tag{3.15}$$

De manera más general, en muchas ocasiones se presentan estas reglas de integración utilizando la fórmula que las engloba y generaliza y que es conocida como la fórmula de Newton-Cotes[2] dada por

$$\int_a^b f(x)dx \approx (b-a)\sum_{j=0}^{n}\alpha_j^{(n)} f\left(a+j\frac{b-a}{n}\right) \tag{3.16}$$

y los valores de los $\alpha_j^{(n)}$ son precisamente los que se presentan en las ecuaciones (3.11)–(3.15) y que pueden generalizarse para $n > 4$.

Como es natural, las fórmulas se complican cuando n crece y, además para $n \geq 8$ aparecen signos positivos y negativos en los factores $\alpha_j^{(n)}$, lo que hace más probable la aparición de cancelaciones numéricas no deseadas. Es por esto que en la práctica se utilizan valores de n que comúnmente son las de las ecuaciones (3.11)-(3.15) o quizá la correspondiente a $n = 5$.

Para funciones más complejas, puede utilizarse la segunda idea que presentamos al inicio del capítulo y que corresponde a dividir el intervalo en varios trozos y realizar varias integrales. Veamos como podría hacerse esto usando la regla del punto medio:

Primero que nada, partiremos el intervalo $[a,b]$ en m subintervalos entre los nodos $a = x_1 < \ldots < x_{m+1} = b$ de longitud $h = \frac{b-a}{m}$ y usaremos la propoedad de las integrales que asegura que

$$\int_a^b f(x)dx = \sum_{j=1}^{m}\int_{a+(j-1)h}^{a+jh} f(x)dx. \tag{3.17}$$

[2]Roger Cotes (1682 – 1716). Matemático ingles que colaboró con Isaac Newton y es particularmente reconocido como uno de los principales revisores de la afamada obra Principia Mathematica de Isaac Newton.

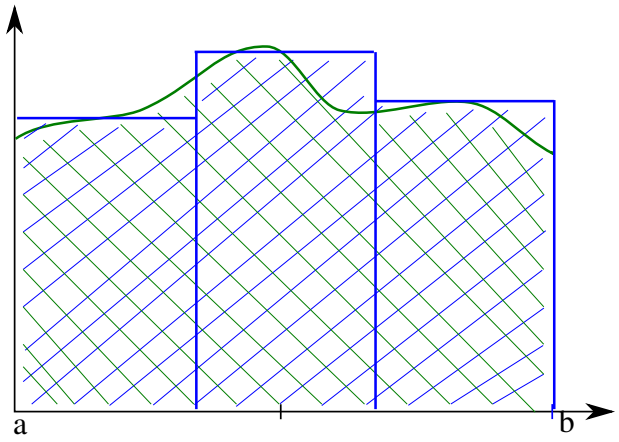


Figura 3.4: Integración aproximada mediante la regla de punto medio compuesta

Con esto no es difícil ver que se obtendrá una aproximación dada por la siguiente Regla del punto medio compuesta

$$\int_a^b f(x)dx \approx h \sum_{j=1}^{m} f\left((a+jh) - \frac{h}{2}\right). \tag{3.18}$$

Esta aproximación corresponde a tomar el histograma con m barras y alturas iguales a la evaluación de f en el punto medio de cada barra. La ilustración en la Figura 3.4 muestra un ejemplo de la aproximación que esta regla calcula utilizando tres particiones del intervalo ($m = 3$).

De manera análoga puede construirse la Regla del Trapecio Compuesta

$$\int_a^b f(x)dx \approx h \left[\frac{1}{2}f(a) + \sum_{j=1}^{m-1} f(a+jh) + \frac{1}{2}f(b)\right], \tag{3.19}$$

o también la regla usando polinomios cuadráticos presentada en la ecuación (3.10) que genera la Regla de Simpson compuesta

$$\int_a^b f(x)dx \approx \frac{h}{3}\left[f(x_1) + 4\sum_{j=1}^{\frac{m}{2}} f(x_{2j}) + 2\sum_{j=1}^{\frac{m}{2}-1} f(x_{2j+1}) + f(x_{m+1})\right] \tag{3.20}$$

donde se ha modificado un poco el significado de la variable m, denotando nuevamente $x_1 = a$, $x_m = b$, $x_i = a + jh$ pero ahora la aproximación se realiza usando un polinomio cuadrático en cada intervalo de tamaño $2h$.

3.2 Cuadraturas de Gauss y errores de integración

Realizar integración numérica debe estar sujeto a cierto nivel de comprobación de que los resultados son buenos. Consideremos la integral $\int_0^1 f(x)dx$ y una cierta fórmula de integración en los nodos $0 = x_1 < x_2 < \cdots < x_n = 1$ con pesos $w_1, w_2, \cdots, w_n \in \mathbb{R}$ de la forma

$$Q(f) = \sum_{i=1}^{n} w_i f(x_i). \tag{3.21}$$

Estamos ahora interesados en la cantidad de error que se comete al calcular esta aproximación, para lo cual usaremos la siguiente definición:

Definición 3.3 — Orden de Integración. Dada una función en $[0,1]$ y una aproximación de $\int_0^1 f(x)dx$ a través de una cuadratura como la de la ecuación (3.21), el error de integración se define como

$$E(f) = \int_0^1 f(x) - Q(f) \equiv \int_0^1 f(x)dx - \sum_{i=1}^{n} w_i f(x_i). \tag{3.22}$$

Con esto, el orden de integración de la regla de cuadratura Q se define como el número m tal que se cumple

$$\begin{aligned} &E(p) = 0 \qquad \forall p \in \mathbb{P}_{m-1}, \\ &E(p) \neq 0 \quad \text{para algún } p \in \mathbb{P}_m. \end{aligned} \tag{3.23}$$

Esto significa que para una cuadratura de orden m, todas las funciones polinomiales de grado menor a m pueden ser aproximados de manera exacta (salvo errores menores de redondeo).

■ **Ejemplo 3.1 — La regla del trapecio.** $Q(f)$ está definida por los pesos $w_1 = w_2 = \frac{1}{2}$ y los nodos $\{0,1\}$. Veamos cual es el error de integración para esta fórmula de $Q(f)$:

Sea $p \in \mathbb{P}_0$ (constante), entonces

$$\int_0^1 p(x)dx = c\,[x]\Big|_0^1 = c, \tag{3.24}$$

$$Q(p) = \sum_{i=1}^{2} w_i p(x_i) = \frac{1}{2}p(0) + \frac{1}{2}p(1) = c, \tag{3.25}$$

y por lo tanto $E(p) = 0$ para polinomios de grado cero. Continuando con un polinomio $p \in \mathbb{P}_1$ se tiene que $p(x) = ax + b$ y entonces

$$\int_0^1 p(x)dx = a\left[\frac{x^2}{2}\right]_0^1 + b\,[x]_0^1 = \frac{1}{2}a + b, \tag{3.26}$$

$$Q(p) = \frac{1}{2}p(0) + \frac{1}{2}p(1) = \frac{1}{2}b + \frac{1}{2}(a+b) = \frac{1}{2}a + b, \tag{3.27}$$

y como ests dos expresiones son iguales, entonces $E(p) = 0$ también para polinomios de primer grado.

Consideremos ahora un polinomio $p \in \mathbb{P}_2$, $p(x) = ax^2 + bx + c$, entonces tenemos para el cálculo de error de integración que

$$\int_0^1 p(x)dx = \left[\frac{ax^3}{3} + \frac{bx^2}{2} + c\right]\Big|_0^1 = \frac{a}{3} + \frac{b}{2} + c, \tag{3.28}$$

$$Q(p) = \frac{1}{2}p(0) + \frac{1}{2}p(1) = \frac{1}{2}c + \frac{1}{2}(a+b+c) = \frac{1}{2}a + \frac{b}{2} + c, \tag{3.29}$$

$$\Rightarrow E(p) = -\frac{1}{6}.$$

Como conclusión, la regla del trapecio tiene orden 2. ∎

No es difícil ver que se puede utilizar la linealidad de la integración para descomponer este proceso en uno más simple en el que solo se verifique para los polinomios $x^j \in \mathbb{P}_j$, $j \in \mathbb{N}_0$.

$$E(1) = \int_0^1 1dx - \left(\frac{1}{2}(1) + \frac{1}{2}(1)\right) = 0 \tag{3.30}$$

$$E(x) = \frac{1}{2}\left[x^2\right]\Big|_0^1 - \left(\frac{1}{2}(0) + \frac{1}{2}(1)\right) = \frac{1}{2} - \frac{1}{2} = 0 \tag{3.31}$$

$$E(x^2) = \left[\frac{x^3}{3}\right]\Big|_0^1 - \left(\frac{1}{2}(0) + \frac{1}{2}(1)\right) = \frac{1}{3} - \frac{1}{2} = -\frac{1}{6}$$

Es decir, que se tiene orden 2 para la regla del trapecio.

∎ **Ejemplo 3.2 — Regla de Simpson.** Es de esperarse que sea mejor que la regla del trapecio, por eso iniciamos con un polinomio de orden 2, para ver si la aproximación nos lleva a un nivel de error cero

$$E(x^2) = \int_0^1 x^2dx - \left[\frac{1}{6}(0)^2 + \frac{4}{6}\left(\frac{1}{2}\right)^2 + \frac{1}{6}(1)^2\right] = \frac{1}{3} - \left[\frac{1}{6} + \frac{1}{6}\right] = 0 \tag{3.32}$$

es decir que el orden es al menos 3, veamos el siguiente grado polinomial

$$E(x^3) = \int_0^1 x^3dx - \left[\frac{1}{6}(0)^3 + \frac{4}{6}\left(\frac{1}{2}\right)^3 + \frac{1}{6}(1)^3\right] = \frac{1}{4} - \left[\frac{1}{12} + \frac{1}{6}\right] = \frac{1}{4} - \frac{3}{12} = 0 \tag{3.33}$$

entonces el orden es incluso mayor a 3 dado que para polinomios de tercer grado el resultado del error sigue siendo igual a cero. Considerando ahora el siguiente polinomio de grado 4 tenemos que

$$E(x^4) = \int_0^1 x^4dx - \left[\frac{1}{6}(0)^4 + \frac{4}{6}\left(\frac{1}{2}\right)^4 + \frac{1}{6}(1)^4\right] = \frac{1}{5} - \left[\frac{1}{24} + \frac{1}{6}\right]$$

$$= \frac{1}{5} - \frac{5}{24} \neq 0 \tag{3.34}$$

por lo que podemos concluir que la regla de Simpson tiene orden 4. ∎

Observe el interesante resultado para la regla de Simpson en contraste con la regla de Simpson $\frac{3}{8}$, que utiliza en su construcción 4 nodos (uno más) pero que solo tiene orden 4. Este resultado sorprende un poco pues al agregar un nodo y usar 3 nodos (Simpson) se obtienen mejoras de 2 órdenes de aproximación, pero al agregar otro nodo y llegar a 4 (Simpson $\frac{3}{8}$) no hay ningún avance en la calidad de aproximación. Una explicación a esto podría ser que las aproximaciones tienen la limitante de nodos igualmente espaciados.

De acuerdo a los desarrollos de Gauss[3], basta ver que la cuadratura del punto medio puede derivarse de manera inversa, primero pidiendo que aproxime bien hasta orden 2 y después viendo si existe un peso w_1 y nodo x_1 tal que los polinomios en $\mathbb{P}_0$ y $\mathbb{P}_1$ son exactamente aproximados. Esto es

$$\int_0^1 x^0 dx = 1 \stackrel{!}{=} w_1 x_1^0, \tag{3.35}$$

$$\int_0^1 x^1 dx = \frac{1}{2} \stackrel{!}{=} w_1 x_1^1. \tag{3.36}$$

De la primera ecuación resulta que $w_1 = 1$ pues $x_1^0 = 1$. De la segunda ecuación resulta entonces $\frac{1}{2} = x_1$. Lo que es precisamente la regla del punto medio.

Similarmente, tenemos una integración con dos nodos (Trapecio) pero quizá podemos encontrar otra cuyo orden de aproximación sea mayor a la obtenida hasta ahora que es 2. Se necesita encontrar los valores de w_1,w_2 como pesos y de x_1, x_2 como nodos y esperamos que se obtenga un orden al menos igual a 3 para así mejorar el método conocido. Entonces debemos resolver el siguiente problema

$$\int_0^1 x^0 dx = 1 \stackrel{!}{=} w_1 x_1^0 + w_2 x_2^0, \tag{3.37}$$

$$\int_0^1 x^1 dx = \frac{1}{2} \stackrel{!}{=} w_1 x_1^1 + w_2 x_2^1, \tag{3.38}$$

$$\int_0^1 x^2 dx = \frac{1}{3} \stackrel{!}{=} w_1 x_1^2 + w_2 x_2^2, \tag{3.39}$$

$$\int_0^1 x^3 dx = \frac{1}{4} \stackrel{!}{=} w_1 x_1^3 + w_2 x_2^3, \tag{3.40}$$

el cual tiene 4 ecuaciones y 4 incógnitas. No es sencillo de resolver pero después de

[3] Atribuidos a Karl Friedrich Gauss (1777-1855). Matemático Alemán conocido como el "príncipe de las matemáticas" y reconocido por muchos como el matemático más prolífico después de los precursores de la antiguedad griega.

diversos desarrollos algebráicos puede verse que la solución está dada por

$$w_1 = \tfrac{1}{2}, \tag{3.41}$$
$$w_2 = \tfrac{1}{2}, \tag{3.42}$$
$$x_1 = \tfrac{1}{2}\left(1 - \tfrac{1}{\sqrt{3}}\right), \tag{3.43}$$
$$x_2 = \tfrac{1}{2}\left(1 + \tfrac{1}{\sqrt{3}}\right). \tag{3.44}$$

Con estos valores puede verse también que

$$\begin{aligned}\int_0^1 x^4 dx = \frac{1}{5} \neq w_1 x_1^4 + w_2 x_2^4 &= \frac{1}{2}\frac{28 - 4\sqrt{3}}{16 \cdot 9} + \frac{1}{2}\frac{28 + 4\sqrt{3}}{16 \cdot 9} \\ &= \frac{56}{32 \cdot 9} = \frac{7}{36}\end{aligned}$$

por lo que el orden de la cuadratura (3.41)–(3.44) es exactamente 4.

En general puede demostrarse que usando cuadraturas de Gauss puede obtenerse el mejor orden del error, siendo éste igual al doble del número de nodos usados para la integración. Los desarrollos generales hacen uso de la base de polinomios de Legendre[4] normalizados para obtener los nodos y lo pesos adecuados para la integración.

La siguiente tabla compara los métodos de integración, incluyendo la cuadratura de Gauβ con 3 puntos y con los nodos/pesos dados por

$$w_1 = \frac{5}{18} \qquad x_1 = \frac{1}{2}\left(1 - \sqrt{\frac{3}{5}}\right), \tag{3.45}$$
$$w_2 = \frac{8}{18} \qquad x_2 = \frac{1}{2}, \tag{3.46}$$
$$w_3 = \frac{5}{18} \qquad x_3 = \frac{1}{2}\left(1 + \sqrt{\frac{3}{5}}\right). \tag{3.47}$$

# Nodos	Nombre	Orden
1	Punto Medio / Gauβ-1	2
2	Trapecio	2
2	Gauβ-2	4
3	Simpson	4
3	Gauβ-3	6
4	Simpson $\frac{3}{8}$	4
4	Gauβ- 4	8

[4] Adrien- Marie Legendre (1752 – 1833) . Matemático francés conocido por sus contribuciones en funciones elípticas y álgebra realizados principalmente en París.

Dado que la teoría detrás de las cuadraturas de Gauβ es desarrollada para mayor simplicidad en el intervalo $[-1,1]$, los pesos w_i y nodos x_i son comunmente definidos en este intervalo. Para 1, 2 y 3 nodos los valores son

$$n=1 \qquad w_1=2 \qquad x_1=0 \tag{3.48}$$

$$\begin{aligned} n=2 \qquad w_1&=1 \qquad x_1=-\sqrt{\frac{1}{3}} \\ w_2&=1 \qquad x_2=+\sqrt{\frac{1}{3}} \end{aligned} \tag{3.49}$$

$$\begin{aligned} n=3 \qquad w_1&=\tfrac{5}{9} \qquad x_1=-\sqrt{\frac{3}{5}} \\ w_2&=\tfrac{8}{9} \qquad x_2=0 \\ w_3&=\tfrac{5}{9} \qquad x_3=+\sqrt{\frac{3}{5}} \end{aligned} \tag{3.50}$$

En la práctica, debe transformarse una integral $\int_a^b f(x)dx$ mediante el cambio de variable

$$x = \frac{b-a}{2}z+\frac{a+b}{2} \tag{3.51}$$

$$dx = \frac{b-a}{2}dz \tag{3.52}$$

que transforma $[a,b]$ en $[-1,1]$ y luego realizar la integral

$$\begin{aligned} \int_a^b f(x)dx &= \frac{b-a}{2}\int_{-1}^{1} f\left(\frac{b-a}{2}z+\frac{a+b}{2}\right)dz \\ &\approx \frac{b-a}{2}\sum_{i=1}^{n} w_i f\left(\frac{b-a}{2}z_i+\frac{a+b}{2}\right) \end{aligned}$$

Donde los z_i son los nodos derivados para las cuadraturas de Gauβ correspondientes.

3.3 Ejercicios

Ejercicio 3.1. Regla de Simpson 3/8 (10 Puntos)
Realice los desarrollos para encontrar la fórmula de la Regla de Simpson 3/8 para aproximaciones a la integral con polinomios en $\mathbb{P}_3$. ∎

Ejercicio 3.2. Regla de Milne (10 Puntos)
Realice los desarrollos para encontrar la fórmula de la Regla de Milne para aproxi-

maciones a la integral con polinomios en $\mathbb{P}_4$. ■

Ejercicio 3.3. Implementación de aproximaciones de orden 0 y 1 (15 Puntos)
Implemente un programa que aproxime $\int_a^b f(x)dx$ utitlizando las aproximaciones polinomiales en $\mathbb{P}_0$, y $\mathbb{P}_1$ a través de las reglas del punto medio y del trapecio. ■

Ejercicio 3.4. Implementación de aproximaciones de orden 2, 3 y 4 (20 Puntos)
Similar al programa del ejercicio anterior, implemente un programa que aproxime $\int_a^b f(x)dx$ utitlizando las aproximaciones polinomiales en $\mathbb{P}_2$, $\mathbb{P}_3$ y $\mathbb{P}_4$ a través de las reglas de Simpson, de Simpson 3/8 y de Milne. ■

Ejercicio 3.5. Cálculo de las integrales numéricas (25 Puntos)
Use el programa del ejercicio anterior para calcular las siguientes integrales:

(a) $\int_1^5 (3x-4)dx$

(b) $\int_{-2}^1 (x^2-2x+3)dx$

(c) $\int_7^9 (x^3-5x^2+2x)dx$

(d) $\int_0^2 (2x^4-x^2+7)dx$

(e) $\int_0^3 \sin(\pi x)dx$

(f) $\int_0^{15} 6(x+1)e^{-3(x+1)^2}dx$

(g) $\int_0^1 [x+\sin(6x)/8]\,dx$

Calcule las integrales exactas analíticamente y compare los errores relativos entre el resultado obtenido computacionalmente y el resultado exacto obtenido de la fórmula analítica. Para la función en el último inciso, grafique la función junto con las aproximaciones en $\mathbb{P}_1$, $\mathbb{P}_2$, $\mathbb{P}_3$ y $\mathbb{P}_4$. ■

Ejercicio 3.6. Reglas compuestas (20 Puntos)
Implemente las reglas compuestas del trapecio y de Simpson y evalúe las integrales correspondientes a los ejercicios 21(f) y 21(g) utilizando $2,3,4\ldots$ subintervalos. Analice los resultados. ■

Ejercicio 3.7. Cuadratura de Gauss (25 Puntos)

Implemente la integración numérica por medio de cuadraturas de Gauss y compare los resultados de integrar con tres nodos usando puntos igualmente espaciados (Regla de Simpson) y puntos espaciados según los cálculos de Gauss.

Considere las integrales de los incisos 21(c), 21(d) y 21(f) y compare los errores relativos[a] en el cálculo de cada una de estas integrales. ■

[a] Igual al valor absoluto de la resta entre el valor exacto y el aproximado, dividido por el valor exacto.