Semana 2: Errores, Condición y Estabilidad en Métodos Numéricos

Análisis Numérico

1. Introducción: ¿Por qué múltiples métodos para un mismo problema?

En análisis numérico no existe un método único que sea óptimo en todos los casos. Un mismo problema puede abordarse mediante distintos métodos, y la elección depende del contexto: la precisión requerida, la estabilidad numérica, el costo computacional y las limitaciones del hardware.

En estas notas estudiaremos las principales fuentes de error que aparecen en los métodos numéricos y su relación con los conceptos de precisión, estabilidad numérica y problemas bien y mal planteados. En particular, distinguiremos **tres niveles** en los que pueden introducirse errores:

- El sistema de aritmética de punto flotante (hardware, inevitable).
- El algoritmo o método numérico utilizado (potencialmente evitable).
- El problema matemático que se desea resolver (posiblemente mal planteado).

Comprender la interacción entre estos tres niveles permite explicar por qué existen distintos métodos numéricos y cómo elegir el más adecuado en cada situación.

2. Errores introducidos por la aritmética de punto flotante (resumen)

En clase pasada analizamos cómo los números reales se representan en la computadora mediante aritmética de punto flotante de precisión finita. Debido a ello, la mayoría de los números reales no se representan exactamente y cada operación aritmética introduce un *error de redondeo*, acotado por la unidad de redondeo del sistema (machine epsilon).

Estos errores son **inevitables** y dependen del hardware. Aunque suelen ser pequeños en una sola operación, pueden acumularse o amplificarse dentro de un algoritmo, por lo que deben tenerse en cuenta al analizar el comportamiento numérico de un método.

3. Errores introducidos por los algoritmos numéricos

Además de los errores de redondeo, los algoritmos numéricos pueden introducir errores adicionales debido a su diseño. Entre los más importantes se encuentran:

- **cancelación numérica**
- **amplificación de errores**
- **uso de esquemas iterativos inestables**
- **formulaciones algebraicamente equivalentes pero numéricamente distintas**

A diferencia de los errores de punto flotante, estos errores **pueden reducirse o evitarse** mediante un diseño algorítmico adecuado. Esto motiva el estudio de la estabilidad numérica.

Un proceso numérico se dice **inestable** si errores pequeños cometidos en alguna etapa del cálculo se amplifican en etapas posteriores, comprometiendo la exactitud de los resultados.

3.1. Ejemplo 1: inestabilidad en una relación de recurrencia

Tarea pasada

3.2. Ejemplo 2: inestabilidad en una relación de recurrencia

Considere la relación:

$$y_0 = 1, \quad y_{n+1} = \frac{5}{3}y_n - \frac{2}{3}y_{n-1}, \quad n \geq 1 \tag{1}$$

Analíticamente, puede demostrarse por inducción que:

$$y_n = \left(\frac{2}{3}\right)^n \tag{2}$$

que converge a cero cuando $n \to \infty$. Sin embargo, al implementarse numéricamente, los errores de redondeo crecen exponencialmente, produciendo resultados incorrectos. Este es un ejemplo típico de **algoritmo inestable**.

4. Medidas de error, precisión y cifras significativas

Las medidas de error se utilizan para evaluar y cuantificar la precisión de los métodos numéricos. En general, se prefiere el error relativo:

$$E_{\text{rel}} = \frac{|\tilde{x} - x|}{|x|} \tag{3}$$

donde $\tilde{x}$ es el valor aproximado y x el valor exacto.

Decimos que $\tilde{x}$ aproxima a x con d **cifras significativas** si d es el mayor entero no negativo tal que:

$$E_{\text{rel}} \leq 5 \times 10^{-d} \tag{4}$$

5. Convergencia y rapidez de convergencia

Sea $\{x_n\}$ una sucesión generada por un método iterativo que converge a x^*. Definimos el error:

$$e_n = |x_n - x^*| \tag{5}$$

La convergencia no sólo indica si un algoritmo funciona, sino **qué tan rápido reduce el error** y qué tan sensible es a errores numéricos.

5.1. Orden o rapidez de convergencia

Decimos que la sucesión converge con **orden** p si existe una constante $C > 0$ y un entero N tal que:

$$e_{n+1} \leq C \cdot e_n^p, \text{ para todo } \quad n \geq N \tag{6}$$

- **Convergencia lineal** ($p = 1$, $C < 1$)**:** el error se reduce por un factor constante.
- **Convergencia cuadrática** ($p = 2$)**:** el error se reduce proporcionalmente al cuadrado del error anterior.
- **Convergencia superlineal** ($1 < p < 2$)**:** comportamiento intermedio.

5.2. Ejemplo

Sea

$$x_{n+1} = \tfrac{1}{2} x_n.$$

La rapidez de convergencia es lineal.

Este concepto será fundamental para analizar métodos como bisección, secante y Newton.

6. Condición de un problema numérico

La condición de un problema mide qué tan sensible es la solución ante pequeñas perturbaciones en los datos de entrada.

Un problema se dice:

- **bien condicionado** si pequeñas perturbaciones en los datos producen pequeños cambios en la solución;
- **mal condicionado** si pequeñas perturbaciones producen grandes cambios en la solución.

Para ciertos problemas puede definirse un **número de condición**, que cuantifica esta sensibilidad.

6.1. Ejemplo: evaluación de una función

Al evaluar una función $f(x)$, una perturbación Δx en x induce el error relativo aproximado:

$$\text{Error relativo en } f \approx \kappa(x) \times \text{Error relativo en } x \tag{7}$$

donde el **número de condición** está dado por:

$$\kappa(x) = \left| \frac{x \cdot f'(x)}{f(x)} \right| \tag{8}$$

Ejemplo numérico: Consideremos $f(x) = \sqrt{x}$ en $x = 100$:

- $f'(x) = \frac{1}{2\sqrt{x}} = \frac{1}{20}$
- $\kappa(100) = \left| \frac{100 \cdot (1/20)}{10} \right| = \left| \frac{5}{10} \right| = 0{,}5$

Esto significa que un error relativo del $1\,\%$ en x se traduce en un error relativo del $0.5\,\%$ en $f(x)$. El problema está **bien condicionado** ($\kappa \approx 1$).

En contraste, la función $f(x) = e^x$ en $x = 10$ tiene $\kappa(10) = 10$, lo que indica mayor sensibilidad.

Nota: El concepto de número de condición será fundamental cuando estudiemos **sistemas de ecuaciones lineales**. Allí veremos que una matriz puede tener un número de condición muy grande, lo que indica que el sistema es **mal condicionado** y requiere métodos especiales para su solución.

Ver otros ejemplos en el libro, sección 2.3. y 2.2 para ver recomendaciones de como efectuar evaluciones de funciones, que producen errorees menores.

7. Problemas bien y mal planteados

Siguiendo a **Hadamard**, un problema es **bien planteado** si cumple:

1. Existencia de solución
2. Unicidad de la solución
3. Dependencia continua de los datos

Si alguna de estas condiciones falla, el problema es **mal planteado**.

Ejemplo clásico: resolver $AX = b$.

La noción de problema bien planteado es **independiente del método numérico**, pero condiciona fuertemente su comportamiento. Los métodos numéricos funcionan de manera más confiable cuando el problema es bien planteado y bien condicionado.

8. Análisis numérico en la era del Machine Learning y AI

Los conceptos que hemos estudiado no son solo históricos: son **fundamentales en la investigación actual** de inteligencia artificial, aprendizaje automático y computación científica a gran escala.

8.1. Problemas modernos que requieren análisis numérico avanzado

- **Entrenamiento de redes neuronales profundas:** Los algoritmos de optimización (SGD, Adam, etc.) son métodos iterativos que deben balancear velocidad de convergencia vs. estabilidad numérica en dimensiones altísimas (millones o billones de parámetros).
- **Condición de matrices en deep learning:** El problema del *vanishing/exploding gradient* es fundamentalmente un problema de mal condicionamiento de las matrices jacobianas durante la retropropagación.
- **Precisión mixta y cuantización:** Los modelos modernos usan aritmética de punto flotante de 16 bits (FP16) o incluso 8 bits (INT8) para acelerar cómputos. Diseñar algoritmos estables bajo estas limitaciones extremas de precisión es un área activa de investigación.

8.2. ¿Por qué es importante esto para ustedes?

Los ingenieros y científicos que trabajan en AI/ML frecuentemente enfrentan:

- **Modelos que no convergen** (problema de estabilidad numérica)
- **Resultados que varían dramáticamente con pequeños cambios** (problema de condicionamiento)
- **Métodos que funcionan en papel pero fallan en implementación** (errores de punto flotante amplificados)
- **Trade-offs entre velocidad y precisión** (elección de métodos)

Comprender los fundamentos del análisis numérico les permite **diagnosticar estos problemas y elegir soluciones apropiadas**, en lugar de simplemente probar hiperparámetros al azar.

Alguna referencias: NeurIPS, ICML, ICLR buscando términos como 'optimization', 'numerical stability', 'preconditioning', 'mixed precision training'.

9. Conexión con los métodos numéricos del curso

Estos conceptos proporcionan el marco teórico para estudiar:

- **métodos de búsqueda de raíces** (bisección, secante, Newton)
- **métodos iterativos para sistemas de ecuaciones** (Jacobi, Gauss-Seidel)
- **métodos de cuadratura numérica**

La elección de un método depende de su **rapidez de convergencia**, **costo computacional**, **estabilidad**, y de la **condición del problema**.

10. Conclusión

El análisis numérico no es simplemente una colección de recetas para resolver problemas, sino un **marco conceptual** para:

- Comprender las limitaciones fundamentales de la computación
- Distinguir entre errores evitables e inevitables
- Diseñar algoritmos robustos y eficientes
- Enfrentar los desafíos computacionales de la AI moderna