Ruta de Aprendizaje en Matemáticas Puras y Aplicadas

David E Luna M

5 de septiembre de 2025

Índice general

Capítulo 1

Fase 1: Estructura General (Esqueleto)

1.1. Fundamentos Históricos

Las matemáticas han acompañado a la humanidad desde la prehistoria, cuando la necesidad de contar, medir y organizar el mundo impulsó los primeros sistemas numéricos.

1.1.1. Matemáticas prehistóricas y antiguas

Los primeros vestigios matemáticos aparecen en huesos tallados con marcas de conteo, como el hueso de Ishango (África, hace más de 20.000 años). Posteriormente, civilizaciones como la egipcia y la mesopotámica desarrollaron sistemas de numeración para la agricultura, la construcción y el comercio.

1.1.2. La matemática griega

Los griegos, en particular Euclides, sentaron las bases de la matemática deductiva, estableciendo axiomas y demostraciones. Esto marcó el paso de la matemática práctica a la matemática formal.

1.1.3. Evolución hacia la matemática moderna

Durante la Edad Media y el Renacimiento, el álgebra y la notación matemática se expandieron. Con Newton y Leibniz, surge el cálculo diferencial e integral, piedra angular de la ciencia moderna.

1.2. Concepto de Estructura en Matemáticas

Una **estructura matemática** es un conjunto acompañado de reglas u operaciones que permiten manipular sus elementos.

Ejemplo simple:

- Conjunto de números naturales $\mathbb{N}$.
- Operación suma $+$.

- Propiedad: $a + b \in \mathbb{N}$ para todo $a, b \in \mathbb{N}$.

Esto constituye la base de lo que más adelante llamaremos *sistemas algebraicos*.

1.3. Aplicación a la Informática

En la informática, estas estructuras se reflejan en:

- **Variables** $\rightarrow$ Representan elementos de un conjunto.
- **Operadores** $\rightarrow$ Implementan funciones matemáticas.
- **Estructuras de datos** $\rightarrow$ Conjuntos con reglas (listas, pilas, árboles, grafos).

Así, la fase 1 establece el *esqueleto* conceptual: entender que toda la programación descansa sobre estructuras formales que tienen su raíz en la matemática.

Capítulo 2

Fase 2: Aritmética y Álgebra Básica

2.1. Aritmética Fundamental

2.1.1. Números Naturales, Enteros, Racionales y Reales

- **Naturales** $\mathbb{N} = \{0, 1, 2, 3, \dots\}$.
- **Enteros** $\mathbb{Z} = \{\dots, -2, -1, 0, 1, 2, \dots\}$.
- **Racionales** $\mathbb{Q} = \left\{\frac{p}{q} \mid p \in \mathbb{Z}, q \neq 0\right\}$.
- **Reales** $\mathbb{R}$ incluyen números racionales e irracionales.

Ejemplo: $2 \in \mathbb{N},\ -5 \in \mathbb{Z},\ \frac{3}{4} \in \mathbb{Q},\ \pi \in \mathbb{R}$.

2.1.2. Operaciones Básicas

- Suma, resta, multiplicación y división.
- Jerarquía de operaciones: paréntesis, potencias/raíces, multiplicación/división, suma/resta.

Ejemplo: $2 + 3 \cdot 4 = 2 + 12 = 14$.

2.2. Bases Numéricas

El sistema decimal (base 10) no es el único:

- **Binario** (base 2): usado en computación.
- **Octal** (base 8).
- **Hexadecimal** (base 16).

Ejemplo: $(1011)_2 = 1 \cdot 2^3 + 0 \cdot 2^2 + 1 \cdot 2^1 + 1 \cdot 2^0 = 11_{10}$.

2.3. Álgebra Básica

2.3.1. Expresiones Algebraicas

Son combinaciones de números, letras (variables) y operaciones.
Ejemplo: $3x + 2y - 7$.

2.3.2. Ecuaciones

Una ecuación es una igualdad con incógnitas.
Ejemplo: Resolver $2x + 3 = 7$.
Solución: $2x = 4 \Rightarrow x = 2$.

2.3.3. Propiedades de la Igualdad

- Si $a = b$, entonces $a + c = b + c$.
- Si $a = b$, entonces $a \cdot c = b \cdot c$ (si $c \neq 0$).

2.4. Aplicación a la Informática

- En programación, las variables representan incógnitas (x, y).
- Los algoritmos de conversión de bases numéricas son fundamentales en compiladores y sistemas.
- La aritmética modular ($a \bmod n$) es la base de la criptografía y los autómatas finitos.

Ejemplo en pseudocódigo (congruencias modulares):

```
funcion es_par(n):
    si n mod 2 == 0:
        retornar Verdadero
    si no:
        retornar Falso
```

Capítulo 3

Fase 3: Geometría y Trigonometría

3.1. Geometría Euclidiana

3.1.1. Elementos básicos

- **Punto**: posición en el espacio, sin dimensiones.
- **Recta**: sucesión infinita de puntos en una dirección.
- **Plano**: superficie infinita en dos dimensiones.

3.1.2. Postulados de Euclides

Euclides propuso cinco postulados que forman la base de la geometría plana. Ejemplo: por dos puntos distintos pasa una única recta.

3.1.3. Figuras básicas

- Triángulos: equilátero, isósceles, escaleno.
- Cuadriláteros: cuadrados, rectángulos, paralelogramos.
- Círculos: radios, diámetros, cuerdas, arcos.

Ejemplo: El área de un círculo es $A = \pi r^2$.

3.2. Geometría Analítica

3.2.1. Sistema de coordenadas

Introducido por Descartes. Representa puntos con pares ordenados (x, y).
Ejemplo: Distancia entre dos puntos $A(x_1, y_1)$ y $B(x_2, y_2)$:

$$d(A, B) = \sqrt{(x_2 - x_1)^2 + (y_2 - y_1)^2}.$$

3.2.2. Rectas y pendientes

Ecuación de la recta: $y = mx + b$, donde m es la pendiente y b la intersección con el eje y.

3.3. Trigonometría

3.3.1. Razones trigonométricas

En un triángulo rectángulo con ángulo θ:

$$\sin\theta = \frac{\text{cateto opuesto}}{\text{hipotenusa}}, \quad \cos\theta = \frac{\text{cateto adyacente}}{\text{hipotenusa}}, \quad \tan\theta = \frac{\text{cateto opuesto}}{\text{cateto adyacente}}.$$

3.3.2. Identidades fundamentales

$$\sin^2\theta + \cos^2\theta = 1$$
$$1 + \tan^2\theta = \sec^2\theta$$

3.3.3. Funciones trigonométricas

- Son periódicas.
- Se usan para modelar fenómenos oscilatorios (ondas, señales).

3.4. Aplicaciones a la Informática

- **Gráficos por computadora**: geometría analítica para dibujar objetos en 2D/3D.
- **Motores de videojuegos**: trigonometría para cálculos de ángulos, rotaciones y trayectorias.
- **Procesamiento de señales**: las funciones trigonométricas modelan ondas de audio y video.

Ejemplo en pseudocódigo (coordenadas polares a cartesianas):

```
funcion polar_a_cartesiano(r, theta):
    x = r * cos(theta)
    y = r * sin(theta)
    retornar (x, y)
```

Capítulo 4

Fase 4: Álgebra Intermedia

4.1. Polinomios

4.1.1. Definición

Un polinomio en una variable es una expresión de la forma:

$$P(x) = a_n x^n + a_{n-1} x^{n-1} + \cdots + a_1 x + a_0$$

donde $a_i \in \mathbb{R}$ y $n \in \mathbb{N}$.

4.1.2. Operaciones con polinomios

- **Suma**: $(2x^2 + 3x) + (x^2 - 5) = 3x^2 + 3x - 5$.
- **Multiplicación**: $(x + 2)(x - 3) = x^2 - x - 6$.
- **División**: Algoritmo de división de polinomios.

4.1.3. Factorización

- Sacar factor común: $6x^2 + 9x = 3x(2x + 3)$.
- Diferencia de cuadrados: $a^2 - b^2 = (a - b)(a + b)$.
- Trinomio cuadrado perfecto: $a^2 \pm 2ab + b^2 = (a \pm b)^2$.

4.2. Ecuaciones e Inecuaciones

4.2.1. Ecuaciones cuadráticas

La forma general es $ax^2 + bx + c = 0$. La fórmula general:

$$x = \frac{-b \pm \sqrt{b^2 - 4ac}}{2a}$$

Ejemplo: $x^2 - 5x + 6 = 0 \Rightarrow x = 2, 3$.

4.2.2. Sistemas de ecuaciones lineales

- Método de sustitución.
- Método de igualación.
- Método de reducción.
- Método matricial (Regla de Cramer).

Ejemplo:

$$\begin{cases} x + y = 5 \\ 2x - y = 1 \end{cases} \quad \Rightarrow x = 2, y = 3$$

4.2.3. Inecuaciones

Ejemplo: Resolver $2x + 3 > 7$.

$$2x > 4 \Rightarrow x > 2$$

4.3. Funciones Polinómicas

Una función polinómica es $f(x) = P(x)$, donde $P(x)$ es un polinomio. Ejemplo: $f(x) = x^2 - 4x + 3$ tiene raíces en $x = 1, 3$.

4.4. Aplicaciones a la Informática

- **Algoritmos**: las ecuaciones lineales aparecen en problemas de optimización y programación lineal.
- **Gráficos**: polinomios modelan trayectorias y curvas en simulaciones.
- **Computación simbólica**: los compiladores usan álgebra para simplificación de expresiones.

Ejemplo en pseudocódigo (evaluar polinomio con el método de Horner):

```
funcion horner(P, x):
    # P es una lista de coeficientes [a_n, a_{n-1}, ..., a_0]
    resultado = 0
    para coef in P:
        resultado = resultado * x + coef
    retornar resultado
```

Capítulo 5

Fase 5: Funciones y Pre-Cálculo

5.1. Concepto de Función

5.1.1. Definición

Una función f es una relación entre dos conjuntos A y B, tal que a cada $x \in A$ le corresponde un único $y \in B$. Se denota:

$$f : A \to B, \quad y = f(x)$$

5.1.2. Dominio y rango

- **Dominio**: conjunto de entrada permitido.
- **Rango**: conjunto de valores que puede tomar $f(x)$.

Ejemplo: $f(x) = \sqrt{x}$, su dominio es $[0, \infty)$ y su rango también $[0, \infty)$.

5.2. Funciones Importantes

5.2.1. Funciones polinómicas

Ejemplo: $f(x) = x^3 - 2x^2 + 1$.

5.2.2. Funciones racionales

Ejemplo: $f(x) = \frac{1}{x-1}$, dominio: $\mathbb{R} \setminus \{1\}$.

5.2.3. Funciones exponenciales

$$f(x) = a^x, \quad a > 0, a \neq 1$$

Ejemplo: $f(x) = 2^x$.

5.2.4. Funciones logarítmicas

$$f(x) = \log_a(x), \quad x > 0$$

Propiedades:

$$\log_a(xy) = \log_a(x) + \log_a(y), \quad \log_a\left(\frac{x}{y}\right) = \log_a(x) - \log_a(y)$$

5.2.5. Funciones trigonométricas

$$\sin(x), \cos(x), \tan(x)$$

Identidades básicas:

$$\sin^2(x) + \cos^2(x) = 1$$

Capítulo 6

Fase 6: Cálculo Diferencial e Integral I

6.1. Límites y Continuidad

6.1.1. Definición de límite

$$\lim_{x \to a} f(x) = L \quad \Leftrightarrow \quad \forall \epsilon > 0, \exists \delta > 0 : 0 < |x - a| < \delta \Rightarrow |f(x) - L| < \epsilon$$

Ejemplo:

$$\lim_{x \to 2} (3x + 1) = 7$$

6.1.2. Límites laterales y al infinito

$$\lim_{x \to \infty} \frac{1}{x} = 0$$

6.1.3. Continuidad

Una función es continua en a si:

$$\lim_{x \to a} f(x) = f(a)$$

6.2. Derivadas

6.2.1. Definición

La derivada de f en a:

$$f'(a) = \lim_{h \to 0} \frac{f(a + h) - f(a)}{h}$$

6.2.2. Reglas de derivación

- $(f + g)' = f' + g'$
- $(fg)' = f'g + fg'$

- $(f/g)' = \frac{f'g - fg'}{g^2}$
- $(x^n)' = nx^{n-1}$

Ejemplo: Si $f(x) = x^2$, entonces $f'(x) = 2x$.

6.2.3. Derivadas trigonométricas

$$(\sin x)' = \cos x, \quad (\cos x)' = -\sin x$$

6.3. Aplicaciones de la Derivada

- Optimización: hallar máximos y mínimos.
- Velocidad y aceleración en física.
- Análisis de algoritmos: comportamiento asintótico.

Ejemplo: Maximizar $A(x) = x(10 - x)$, área de un rectángulo con perímetro fijo.

$$A'(x) = 10 - 2x = 0 \Rightarrow x = 5$$

6.4. Integrales

6.4.1. Integral indefinida

$$\int f(x)\,dx = F(x) + C \quad \text{tal que } F'(x) = f(x)$$

Ejemplo:

$$\int 2x\,dx = x^2 + C$$

6.4.2. Integral definida

$$\int_a^b f(x)\,dx = \lim_{n\to\infty} \sum_{i=1}^{n} f(x_i)\Delta x$$

Ejemplo:

$$\int_0^1 x^2 dx = \left[\frac{x^3}{3}\right]_0^1 = \frac{1}{3}$$

6.4.3. Teorema Fundamental del Cálculo

Si $F'(x) = f(x)$, entonces:

$$\int_a^b f(x)\,dx = F(b) - F(a)$$

6.5. Métodos de Integración

- Sustitución: $u = g(x)$.
- Integración por partes: $\int u\,dv = uv - \int v\,du$.

Ejemplo por partes:

$$\int xe^x dx = xe^x - \int e^x dx = (x-1)e^x + C$$

6.6. Aplicaciones de la Integral

- Áreas bajo curvas.
- Volúmenes de sólidos de revolución.
- Trabajo en física.
- Modelos de acumulación en informática (e.g., costo de cómputo).

6.7. Ejemplo en pseudocódigo

Método del trapecio para aproximar integrales:

```
funcion trapecio(f, a, b, n):
    h = (b - a) / n
    suma = (f(a) + f(b)) / 2
    para i desde 1 hasta n-1:
        x = a + i * h
        suma = suma + f(x)
    retornar h * suma
```

Capítulo 7

Fase 7: Álgebra Lineal y Geometría Analítica

7.1. Vectores en el plano y en el espacio

7.1.1. Definición

Un vector es un elemento caracterizado por magnitud, dirección y sentido. En coordenadas cartesianas:

$$\vec{v} = (v_1, v_2, \ldots, v_n) \in \mathbb{R}^n$$

7.1.2. Operaciones básicas

$$\vec{u} + \vec{v} = (u_1 + v_1, \ldots, u_n + v_n)$$
$$c\vec{v} = (cv_1, \ldots, cv_n), \quad c \in \mathbb{R}$$

7.1.3. Producto escalar

$$\vec{u} \cdot \vec{v} = \sum_{i=1}^{n} u_i v_i$$
$$\cos\theta = \frac{\vec{u} \cdot \vec{v}}{\|\vec{u}\|\|\vec{v}\|}$$

7.1.4. Producto vectorial (en $\mathbb{R}^3$)

$$\vec{u} \times \vec{v} = \begin{vmatrix} \hat{i} & \hat{j} & \hat{k} \\ u_1 & u_2 & u_3 \\ v_1 & v_2 & v_3 \end{vmatrix}$$

7.2. Rectas y planos

7.2.1. Ecuación de la recta en $\mathbb{R}^2$

$$y = mx + b$$

o en forma vectorial:

$$\vec{r}(t) = \vec{r}_0 + t\vec{v}$$

7.2.2. Ecuación del plano en $\mathbb{R}^3$

$$ax + by + cz + d = 0$$

7.3. Matrices

7.3.1. Definición

Una matriz A de tamaño $m \times n$ es:

$$A = \begin{bmatrix} a_{11} & a_{12} & \dots & a_{1n} \\ a_{21} & a_{22} & \dots & a_{2n} \\ \vdots & \vdots & \ddots & \vdots \\ a_{m1} & a_{m2} & \dots & a_{mn} \end{bmatrix}$$

7.3.2. Operaciones

- Suma: $(A + B)_{ij} = a_{ij} + b_{ij}$
- Producto: $(AB)_{ij} = \sum_{k=1}^{n} a_{ik} b_{kj}$
- Transpuesta: $A^T = (a_{ji})$

7.3.3. Determinantes

Para 2×2:

$$\det \begin{bmatrix} a & b \\ c & d \end{bmatrix} = ad - bc$$

Para 3×3:

$$\det A = a(ei - fh) - b(di - fg) + c(dh - eg)$$

7.4. Sistemas de ecuaciones lineales

7.4.1. Forma matricial

$$AX = B$$

$$\begin{bmatrix} a_{11} & a_{12} \\ a_{21} & a_{22} \end{bmatrix} \begin{bmatrix} x_1 \\ x_2 \end{bmatrix} = \begin{bmatrix} b_1 \\ b_2 \end{bmatrix}$$

7.4.2. Métodos de resolución

- Método de Gauss (eliminación).
- Regla de Cramer (si $\det A \neq 0$).

7.5. Espacios vectoriales

7.5.1. Definición

Un espacio vectorial V sobre un campo $\mathbb{K}$ cumple:

$$\forall \vec{u}, \vec{v} \in V, \quad \vec{u} + \vec{v} \in V$$

$$\forall c \in \mathbb{K}, \vec{v} \in V, \quad c\vec{v} \in V$$

7.5.2. Bases y dimensión

Un conjunto de vectores $\{v_1, v_2, \ldots, v_n\}$ es base si:

- Son linealmente independientes.
- Generan todo V.

La cantidad de vectores en la base = dimensión de V.

7.6. Aplicaciones en informática

- **Gráficos 2D/3D**: Transformaciones mediante matrices.
- **Machine Learning**: Representación de datos como vectores y tensores.
- **Sistemas**: Resolución de ecuaciones lineales para simulaciones.
- **Criptografía**: Espacios vectoriales finitos.

7.7. Ejemplo en pseudocódigo

Resolución de un sistema $Ax = b$ con eliminación de Gauss:

```
funcion gauss(A, b):
    n = longitud(b)
    para k desde 0 hasta n-1:
        # pivote
        para i desde k+1 hasta n-1:
            factor = A[i][k] / A[k][k]
            para j desde k hasta n-1:
                A[i][j] = A[i][j] - factor * A[k][j]
            b[i] = b[i] - factor * b[k]
    # sustitución hacia atrás
```

```
x = vector(n)
para i desde n-1 hasta 0:
    suma = b[i]
    para j desde i+1 hasta n-1:
        suma = suma - A[i][j]*x[j]
    x[i] = suma / A[i][i]
retornar x
```

Capítulo 8

Fase 8: Cálculo Multivariable y Ecuaciones Diferenciales Ordinarias

8.1. Funciones de varias variables

8.1.1. Definición

Una función de n variables es:

$$f : \mathbb{R}^n \to \mathbb{R}, \quad f(x_1, x_2, \ldots, x_n)$$

8.1.2. Límites y continuidad

Se define el límite como:

$$\lim_{(x,y)\to(x_0,y_0)} f(x,y) = L$$

si para todo $\epsilon > 0$ existe $\delta > 0$ tal que

$$\sqrt{(x-x_0)^2 + (y-y_0)^2} < \delta \implies |f(x,y) - L| < \epsilon$$

8.2. Derivadas parciales

8.2.1. Definición

$$\frac{\partial f}{\partial x}(x,y) = \lim_{h\to 0} \frac{f(x+h,y) - f(x,y)}{h}$$

8.2.2. Gradiente

$$\nabla f(x,y,z) = \left(\frac{\partial f}{\partial x}, \frac{\partial f}{\partial y}, \frac{\partial f}{\partial z}\right)$$

8.2.3. Plano tangente

El plano tangente en (x_0, y_0, z_0) es:

$$z - z_0 = f_x(x_0,y_0)(x-x_0) + f_y(x_0,y_0)(y-y_0)$$

8.3. Integrales múltiples

8.3.1. Dobles integrales

$$\iint_D f(x,y)\,dA$$

8.3.2. Cambio de coordenadas

En polares:

$$\iint_D f(r\cos\theta, r\sin\theta)\,r\,dr\,d\theta$$

8.3.3. Triples integrales

$$\iiint_V f(x,y,z)\,dV$$

8.4. Campos vectoriales y teoremas fundamentales

8.4.1. Divergencia y rotacional

$$\nabla\cdot\vec{F} = \frac{\partial F_1}{\partial x} + \frac{\partial F_2}{\partial y} + \frac{\partial F_3}{\partial z}$$

$$\nabla\times\vec{F} = \begin{vmatrix} \hat{\imath} & \hat{\jmath} & \hat{k} \\ \partial_x & \partial_y & \partial_z \\ F_1 & F_2 & F_3 \end{vmatrix}$$

8.4.2. Teoremas

- Green (circulación y flujo en 2D).
- Gauss (divergencia en 3D).
- Stokes (rotacional en superficies).

8.5. Ecuaciones diferenciales ordinarias (EDO)

8.5.1. Definición

Una EDO es una ecuación que involucra derivadas de una función de una variable:

$$\frac{dy}{dx} = f(x,y)$$

8.5.2. Clasificación

- Orden: máximo de derivadas involucradas.
- Lineal / no lineal.
- Homogénea / no homogénea.

8.5.3. Ejemplos

1. $y' = ky \quad \Rightarrow \quad y = Ce^{kx}$
2. $y'' + \omega^2 y = 0 \quad \Rightarrow \quad y = A\cos(\omega x) + B\sin(\omega x)$

8.6. Métodos de resolución

8.6.1. Variables separables

$$\frac{dy}{dx} = g(x)h(y) \quad \Rightarrow \quad \frac{dy}{h(y)} = g(x)dx$$

8.6.2. Factor integrante (ecuaciones lineales de primer orden)

$$y' + P(x)y = Q(x)$$

Multiplicamos por:

$$\mu(x) = e^{\int P(x)dx}$$

8.7. Aplicaciones en informática

- **Simulaciones físicas**: Movimiento de partículas, trayectorias.
- **Modelado poblacional**: Crecimiento logístico, ecuaciones de Lotka-Volterra.
- **Gráficos**: Campos vectoriales para fluidos.
- **Machine learning**: Optimización con gradientes.

8.8. Ejemplo en pseudocódigo

Método de Euler para aproximar soluciones de EDO:

```
funcion euler(f, x0, y0, h, n):
    x = x0
    y = y0
    para i desde 1 hasta n:
        y = y + h * f(x, y)
        x = x + h
    retornar y
```

Capítulo 9

Fase 9: Matemática Discreta y Teoría de Grafos

9.1. Lógica matemática

9.1.1. Proposiciones

Una proposición es una afirmación que puede ser verdadera o falsa. Ejemplo: p: "2 es par", q: "5 es primo".

9.1.2. Conectivos lógicos

$$\neg p, \quad p \wedge q, \quad p \vee q, \quad p \to q, \quad p \leftrightarrow q$$

9.1.3. Tablas de verdad

Ejemplo: Implicación $p \to q$

p	q	$p \to q$
V	V	V
V	F	F
F	V	V
F	F	V

9.2. Teoría de conjuntos

9.2.1. Operaciones

$$A \cup B, \quad A \cap B, \quad A - B, \quad A^c$$

9.2.2. Producto cartesiano

$$A \times B = \{(a, b) : a \in A, b \in B\}$$

9.3. Relaciones y funciones

9.3.1. Relaciones

Una relación R sobre un conjunto A es un subconjunto $R \subseteq A \times A$. Propiedades: reflexiva, simétrica, transitiva.

9.3.2. Funciones

Una función $f : A \to B$ asigna a cada $a \in A$ un único $b \in B$. Tipos: inyectiva, sobreyectiva, biyectiva.

9.4. Inducción matemática

9.4.1. Principio de inducción

Para demostrar $P(n)$:

1. Caso base: verificar $P(1)$.
2. Paso inductivo: suponer $P(k)$ y demostrar $P(k+1)$.

9.4.2. Ejemplo

Demostrar:

$$1 + 2 + \cdots + n = \frac{n(n+1)}{2}$$

9.5. Combinatoria

9.5.1. Principios de conteo

- Regla de la suma.
- Regla del producto.

9.5.2. Permutaciones

$$P(n) = n!$$

9.5.3. Combinaciones

$$C(n,k) = \binom{n}{k} = \frac{n!}{k!(n-k)!}$$

9.6. Teoría de grafos

9.6.1. Definición

Un grafo $G = (V, E)$ consiste en un conjunto de vértices V y aristas E.

9.6.2. Tipos de grafos

- No dirigidos y dirigidos.
- Ponderados.
- Conexos, completos, bipartitos.

9.6.3. Representación

- Matriz de adyacencia.
- Lista de adyacencia.

9.6.4. Caminos y ciclos

- Camino Euleriano: recorre cada arista una vez.
- Ciclo Hamiltoniano: visita cada vértice una vez.

9.7. Algoritmos fundamentales en grafos

9.7.1. Búsquedas

- BFS (Breadth-First Search).
- DFS (Depth-First Search).

9.7.2. Caminos mínimos

- Algoritmo de Dijkstra.
- Algoritmo de Floyd-Warshall.

9.7.3. Árboles generadores mínimos

- Algoritmo de Prim.
- Algoritmo de Kruskal.

9.8. Aplicaciones en informática

- Redes (Internet, grafos sociales).
- Compiladores (análisis sintáctico).
- Criptografía (teoría de números y grafos).
- Inteligencia artificial (búsquedas en grafos).

9.9. Ejemplo en pseudocódigo

Búsqueda en anchura (BFS):

```
funcion BFS(grafo, nodo_inicio):
    crear cola vacia
    marcar nodo_inicio como visitado
    encolar nodo_inicio
    mientras la cola no esté vacía:
        nodo = desencolar
        procesar nodo
        para cada vecino de nodo:
            si vecino no está visitado:
                marcar vecino como visitado
                encolar vecino
```

Capítulo 10

Fase 10: Probabilidad, Estadística y Combinatoria Avanzada

10.1. Fundamentos de probabilidad

10.1.1. Espacios de probabilidad

Un espacio de probabilidad se define como la terna $(\Omega, \mathcal{F}, P)$ donde:

- Ω: espacio muestral.
- $\mathcal{F}$: σ-álgebra de eventos.
- P: medida de probabilidad.

10.1.2. Axiomas de Kolmogórov

1. $P(A) \geq 0$ para todo evento A.
2. $P(\Omega) = 1$.
3. Si $A \cap B = \emptyset$, entonces $P(A \cup B) = P(A) + P(B)$.

10.2. Probabilidad condicional e independencia

$$P(A|B) = \frac{P(A \cap B)}{P(B)}, \quad P(A \cap B) = P(A)P(B) \text{ si independientes}$$

10.2.1. Teorema de Bayes

$$P(A|B) = \frac{P(B|A)P(A)}{P(B)}$$

10.3. Variables aleatorias

10.3.1. Definición

Una variable aleatoria X es una función $X : \Omega \to \mathbb{R}$.

10.3.2. Esperanza y varianza

$$E[X] = \sum_x xP(X = x) \quad \text{o} \quad E[X] = \int_{-\infty}^{\infty} xf(x)dx$$
$$Var(X) = E[X^2] - (E[X])^2$$

10.3.3. Covarianza y correlación

$$Cov(X, Y) = E[(X - E[X])(Y - E[Y])], \quad \rho(X, Y) = \frac{Cov(X, Y)}{\sigma_X \sigma_Y}$$

10.4. Distribuciones de probabilidad

10.4.1. Discretas

- Bernoulli: $P(X = 1) = p$, $P(X = 0) = 1 - p$.
- Binomial: $P(X = k) = \binom{n}{k} p^k (1 - p)^{n-k}$.
- Poisson: $P(X = k) = \frac{\lambda^k e^{-\lambda}}{k!}$.

10.4.2. Continuas

- Uniforme: $f(x) = \frac{1}{b-a}$ en $[a, b]$.
- Normal: $f(x) = \frac{1}{\sqrt{2\pi\sigma^2}} e^{-(x-\mu)^2/(2\sigma^2)}$.
- Exponencial: $f(x) = \lambda e^{-\lambda x}$.

10.5. Teoremas fundamentales

10.5.1. Ley de los grandes números

$$\overline{X}_n = \frac{1}{n} \sum_{i=1}^{n} X_i \xrightarrow{p} \mu$$

10.5.2. Teorema central del límite

Si X_i son independientes e idénticamente distribuidas con media μ y varianza σ^2, entonces:

$$\frac{\overline{X}_n - \mu}{\sigma/\sqrt{n}} \xrightarrow{d} N(0, 1)$$

10.6. Estadística inferencial

10.6.1. Estimadores

Propiedades: insesgado, consistente, eficiente.

10.6.2. Intervalos de confianza

$$IC = \hat{\theta} \pm z_{\alpha/2} \frac{\sigma}{\sqrt{n}}$$

10.6.3. Pruebas de hipótesis

- Hipótesis nula H_0, alternativa H_1.
- Niveles de significancia α.
- Ejemplo: prueba Z, prueba χ^2.

10.7. Combinatoria avanzada

10.7.1. Principio de inclusión-exclusión

$$|A \cup B| = |A| + |B| - |A \cap B|$$

10.7.2. Números combinatorios

$$\binom{n}{k} = \frac{n!}{k!(n-k)!}$$

10.7.3. Números de Stirling

$$S(n, k) = \text{número de formas de particionar un conjunto de } n \text{ elementos en } k \text{ subconjuntos no vacíos}$$

10.7.4. Números de Bell

$$B_n = \sum_{k=0}^{n} S(n, k)$$

10.8. Procesos estocásticos (introducción)

- Cadenas de Markov.
- Procesos de Poisson.
- Caminatas aleatorias.

10.9. Aplicaciones

- Machine learning (regresión, clasificación).
- Criptografía probabilística.
- Algoritmos aleatorizados.
- Teoría de la información.

10.10. Ejemplo práctico

10.10.1. Regla de Bayes aplicada

Un test médico tiene 99 % de sensibilidad y 95 % de especificidad. La prevalencia de la enfermedad es 1 %. Si el test da positivo, ¿cuál es la probabilidad real de estar enfermo?

$$P(E|+) = \frac{0{,}99 \times 0{,}01}{0{,}99 \times 0{,}01 + 0{,}05 \times 0{,}99} \approx 0{,}167$$

Aunque el test es muy preciso, la probabilidad real es solo del 16,7 % debido a la baja prevalencia.

Capítulo 11

Fase 11: Optimización, Métodos Numéricos y Complejidad Computacional

11.1. Introducción

En esta fase, el objetivo es profundizar en los métodos matemáticos y computacionales que permiten resolver problemas complejos de optimización y cálculo numérico, fundamentales en matemáticas aplicadas, ciencias de la computación, economía, ingeniería y física. Se estudiarán tanto los enfoques teóricos como los algoritmos prácticos, siempre con énfasis en su precisión, estabilidad y complejidad computacional.

11.2. Optimización

11.2.1. Optimización sin restricciones

- Condiciones de primer y segundo orden.
- Métodos de descenso más pronunciado.
- Método de Newton y cuasi-Newton.

11.2.2. Optimización con restricciones

- Método de multiplicadores de Lagrange.
- Programación cuadrática.
- Programación lineal: el método simplex.
- Programación entera.

11.2.3. Optimización convexa

- Definición de convexidad.
- Dualidad en optimización.

- Aplicaciones en machine learning y teoría de juegos.

11.3. Métodos Numéricos

11.3.1. Resolución de ecuaciones no lineales

- Método de bisección.
- Método de Newton-Raphson.
- Método de la secante.

11.3.2. Sistemas de ecuaciones lineales

- Eliminación de Gauss.
- Factorización LU.
- Métodos iterativos: Jacobi, Gauss-Seidel.

11.3.3. Interpolación y aproximación

- Interpolación de Lagrange.
- Interpolación de Newton.
- Splines cúbicos.

11.3.4. Derivación e integración numérica

- Fórmulas de diferencias finitas.
- Regla del trapecio.
- Regla de Simpson.

11.3.5. Ecuaciones diferenciales

- Método de Euler.
- Métodos de Runge-Kutta.
- Métodos implícitos.

11.4. Complejidad Computacional

11.4.1. Clases de complejidad

- Clase P.
- Clase NP.
- NP-completitud.

11.4.2. Análisis de algoritmos

- Notación asintótica: O, Ω, Θ.
- Complejidad temporal y espacial.
- Estabilidad numérica.

11.4.3. Problemas intratables

- Problemas NP-duros.
- Algoritmos aproximados.
- Algoritmos aleatorizados.

11.5. Ejemplo Práctico

Ejemplo: Supongamos que queremos minimizar la función:

$$f(x, y) = x^2 + y^2 - 4x - 6y + 13$$

Solución:

1. Calculamos las derivadas parciales:

$$\frac{\partial f}{\partial x} = 2x - 4, \quad \frac{\partial f}{\partial y} = 2y - 6$$

2. Igualamos a cero:

$$\begin{aligned} 2x - 4 = 0 \quad &\Rightarrow \quad x = 2 \\ 2y - 6 = 0 \quad &\Rightarrow \quad y = 3 \end{aligned}$$

3. El punto crítico es $(2, 3)$.

4. Evaluamos la función en ese punto:

$$f(2, 3) = 2^2 + 3^2 - 4(2) - 6(3) + 13 = 4 + 9 - 8 - 18 + 13 = 0$$

5. Por lo tanto, el mínimo se alcanza en $(2, 3)$ y el valor mínimo es 0.

11.6. Bibliografía Recomendada

- S. Boyd, L. Vandenberghe. *Convex Optimization.*
- Jorge Nocedal, Stephen J. Wright. *Numerical Optimization.*
- Richard L. Burden, J. Douglas Faires. *Numerical Analysis.*
- Christos Papadimitriou. *Computational Complexity.*

Capítulo 12

Fase 12: Teoría de la Información, Criptografía y Seguridad Matemática

12.1. Introducción

En esta fase se estudian los fundamentos de la transmisión y protección de la información. La teoría de la información proporciona las bases matemáticas para entender la compresión y la transmisión eficiente de datos, mientras que la criptografía aplica herramientas algebraicas y combinatorias para garantizar la confidencialidad, integridad y autenticidad de la información.

12.2. Teoría de la Información

12.2.1. Conceptos fundamentales

- Información como reducción de incertidumbre.
- Entropía de Shannon.
- Redundancia y eficiencia.

12.2.2. Medidas de información

- Entropía $H(X)$ de una variable aleatoria.
- Información mutua $I(X;Y)$.
- Entropía condicional $H(X|Y)$.

12.2.3. Teoremas principales

- Teorema de codificación de Shannon.
- Límite de compresión sin pérdida.
- Capacidad de un canal.

12.2.4. Aplicaciones

- Compresión de datos (Huffman, Lempel-Ziv).
- Teoría de códigos: códigos de detección y corrección de errores.

12.3. Criptografía Clásica

12.3.1. Sistemas de sustitución y transposición

- Cifrado César.
- Cifrado de Vigenère.
- Cifrado por transposición.

12.3.2. Análisis de seguridad

- Ataques por fuerza bruta.
- Criptoanálisis clásico: análisis de frecuencia.

12.4. Criptografía Moderna

12.4.1. Criptografía simétrica

- Algoritmos de bloque (AES, DES).
- Algoritmos de flujo (RC4).

12.4.2. Criptografía asimétrica

- RSA (aritmética modular, factorización).
- Diffie-Hellman (logaritmo discreto).
- Criptografía basada en curvas elípticas (ECC).

12.4.3. Funciones hash y autenticación

- Funciones hash criptográficas (SHA-2, SHA-3).
- Firmas digitales.
- Protocolos de autenticación.

12.5. Seguridad Matemática y Complejidad

12.5.1. Problemas matemáticos difíciles

- Factorización de enteros grandes.
- Logaritmo discreto.
- Isogenias de curvas elípticas.

12.5.2. Criptografía poscuántica

- Sistemas basados en retículas (lattices).
- Códigos correctores de errores como base criptográfica.
- Funciones hash resistentes a ataques cuánticos.

12.6. Ejemplo Práctico

Ejemplo: Calcular la entropía de una variable aleatoria X que toma valores {A, B, C} con probabilidades $p(A) = 0{,}5$, $p(B) = 0{,}25$, $p(C) = 0{,}25$.
Solución:

$$H(X) = -\sum p(x)\log_2 p(x) = -\Big(0{,}5\log_2 0{,}5 + 0{,}25\log_2 0{,}25 + 0{,}25\log_2 0{,}25\Big)$$

$$H(X) = -\Big(0{,}5\cdot(-1) + 0{,}25\cdot(-2) + 0{,}25\cdot(-2)\Big) = 1{,}5 \text{ bits}$$

12.7. Bibliografía Recomendada

- Claude E. Shannon. *A Mathematical Theory of Communication.*
- Thomas M. Cover, Joy A. Thomas. *Elements of Information Theory.*
- Alfred J. Menezes, Paul C. van Oorschot, Scott A. Vanstone. *Handbook of Applied Cryptography.*
- Jonathan Katz, Yehuda Lindell. *Introduction to Modern Cryptography.*

Capítulo 13

Fase 13: Matemática Computacional y Teoría de la Computación

13.1. Objetivos

Esta fase busca introducir los fundamentos matemáticos del cómputo, cubriendo desde los autómatas finitos hasta las máquinas de Turing, las clases de complejidad y la teoría de lenguajes formales. Se trata de los pilares teóricos que sustentan compiladores, verificación formal y la informática teórica.

13.2. Contenidos Principales

13.2.1. Lenguajes formales

- Definición de alfabeto y cadena.
- Operaciones sobre lenguajes: unión, concatenación, estrella de Kleene.
- Gramáticas formales (Jerarquía de Chomsky):
  - Tipo 0: Gramáticas irrestrictas.
  - Tipo 1: Gramáticas sensibles al contexto.
  - Tipo 2: Gramáticas libres de contexto (CFG).
  - Tipo 3: Gramáticas regulares.

13.2.2. Autómatas

- Autómatas finitos deterministas (DFA).
- Autómatas finitos no deterministas (NFA).
- Equivalencia DFA/NFA.
- Expresiones regulares y su relación con autómatas.
- Autómatas con pila (PDA) y lenguajes libres de contexto.
- Autómatas lineales acotados.

13.2.3. Máquinas de Turing

- Definición formal de máquina de Turing.
- Variantes: cinta múltiple, no determinista, universal.
- Tesis de Church–Turing.
- Lenguajes recursivos y recursivamente enumerables.

13.2.4. Decidibilidad

- Problemas decidibles e indecidibles.
- Problema de la parada.
- Reducciones entre problemas.
- Ejemplos clásicos de indecidibilidad (Post Correspondence Problem).

13.2.5. Complejidad computacional

- Medición de recursos: tiempo y espacio.
- Clases de complejidad: P, NP, co-NP, PSPACE, EXPTIME.
- Problemas NP-completos (SAT, 3-SAT).
- Hipótesis P vs NP.

13.2.6. Aplicaciones

- Construcción de compiladores.
- Verificación y model checking.
- Seguridad informática y criptografía.
- Optimización de algoritmos y análisis de rendimiento.

13.3. Ejemplos Prácticos

- Construir un DFA que reconozca cadenas binarias con número par de ceros.
- Diseñar una gramática libre de contexto para expresiones aritméticas.
- Simular una máquina de Turing que acepte $\{a^n b^n \mid n \geq 0\}$.
- Demostrar la indecidibilidad del problema de la parada por contradicción.
- Clasificar un algoritmo de ordenamiento simple en términos de P (ej.: mergesort $\in$ O(n log n) $\subseteq$ P).

13.4. Libros Recomendados

- Hopcroft, Motwani y Ullman: *Introduction to Automata Theory, Languages, and Computation.*
- Sipser: *Introduction to the Theory of Computation.*
- Papadimitriou: *Computational Complexity.*

13.5. Recursos Gratuitos

- MIT OpenCourseWare: 6.045J *Automata, Computability, and Complexity.*
- Stanford: cursos y notas sobre teoría de autómatas.
- YouTube: conferencias y seminarios (Sipser, Ullman, Papadimitriou).
- Herramientas prácticas: JFLAP (simulación de autómatas y máquinas de Turing).

13.6. Anexo Histórico

- **Lógica Clásica (siglo IV a.C.)**: Aristóteles y el inicio del razonamiento formal.
- **George Boole (siglo XIX)**: álgebra booleana y lógica simbólica.
- **David Hilbert / Kurt Gödel (siglo XX)**: programa de Hilbert y teoremas de incompletitud.
- **Alonzo Church (1930s)**: cálculo lambda y definiciones formales de computabilidad.
- **Alan Turing (1936)**: máquina de Turing y formalización del algoritmo.
- **Noam Chomsky (1956)**: jerarquía de gramáticas aplicable a lenguajes formales.
- **Décadas recientes**: complejidad computacional, verificación formal y aplicaciones en síntesis de programas.

Mapa conceptual: jerarquía y relaciones (Jerarquía de Chomsky → Autómatas → Turing → Complejidad)

[>=stealth, node distance=28mm, auto, semithick] box/.style = draw, rounded corners, fill=white, drop shadow, minimum width=50mm, minimum height=8mm, align=center, arrow/.style = ->, thick
[box] (chomsky) Jerarquía de Chomsky
(Tipo 3 → Tipo 2 → Tipo 1 → Tipo 0); [box, below=of chomsky] (automata) Autómatas
(DFA / NFA / PDA / LBA); [box, below=of automata] (turing) Máquinas de Turing
(Determinista / No determinista / Universal); [box, right=of automata] (regex) Expresiones Regulares

(equivalentes a Tipo 3); [box, left=of automata] (grammars) Gramáticas (CFG, Gramáticas sens. al contexto); [box, below=of turing] (complexity) Complejidad (P, NP, PSPACE, EXPTIME);
[arrow] (chomsky) – (automata); [arrow] (automata) – (turing); [arrow] (automata) – (regex); [arrow] (grammars) – (automata); [arrow] (turing) – (complexity);
[below=2mm of chomsky, font=] Clasificación de lenguajes según restricciones de producción; [below=2mm of turing, font=] Modelo que captura la noción de algoritmo; [below=2mm of complexity, font=] Medición de recursos: tiempo y espacio;

13.7. Observaciones finales

Dominar esta fase te posiciona para diseñar lexers/lexers, construir parsers, verificar propiedades de programas y estudiar problemas profundos como P vs NP. Se recomienda: (1) implementar autómatas y conversores NFA→DFA en código (Rust/Python), (2) usar JFLAP para simular, (3) leer Sipser y Hopcroft & Ullman en paralelo con ejercicios prácticos.

Capítulo 14

Fase 14: Compiladores y Lenguajes de Programación

14.1. Objetivos

Consolidar los conocimientos de teoría de autómatas y lenguajes formales aplicados a la construcción de compiladores, enlazando los conceptos de gramáticas, autómatas y máquinas de Turing con herramientas prácticas como parsers, analizadores léxicos y generadores de código.

14.2. Contenidos Principales

14.2.1. Arquitectura de un compilador

- Etapas clásicas:
  1. Análisis léxico (scanner).
  2. Análisis sintáctico (parser).
  3. Análisis semántico.
  4. Optimización intermedia.
  5. Generación de código.
  6. Optimización de código.
- Relación con la teoría: DFA/NFA en análisis léxico, CFG en parsing, atributos semánticos en análisis semántico.

14.2.2. Análisis Léxico

- Expresiones regulares y autómatas como base.
- Herramientas clásicas: `lex`, `flex`.
- Implementación manual de un lexer con un DFA.

14.2.3. Análisis Sintáctico

- Algoritmos LL(1), LR(0), SLR(1), LALR(1).
- Herramientas: `yacc`, `bison`.
- Construcción manual de un parser recursivo descendente.

14.2.4. Análisis Semántico

- Árbol de sintaxis abstracta (AST).
- Tablas de símbolos.
- Chequeo de tipos y atributos semánticos.

14.2.5. Optimización y Generación de Código

- Representaciones intermedias (IR).
- Optimizaciones locales y globales.
- Traducción a lenguaje ensamblador o bytecode.
- Registro de asignaciones y gestión de memoria.

14.2.6. Lenguajes de Programación

- Paradigmas: imperativo, funcional, orientado a objetos, lógico.
- Diseño de lenguajes: sintaxis, semántica, pragmática.
- Lenguajes experimentales y lenguajes de dominio específico (DSLs).

14.3. Ejemplos Prácticos

- Implementar un lexer en Python o Rust que reconozca números, identificadores y operadores básicos.
- Construir un parser LL(1) para expresiones aritméticas con paréntesis y jerarquía de operadores.
- Generar un AST y recorrerlo para evaluar expresiones.
- Diseñar un mini-lenguaje con asignación y expresiones aritméticas.

14.4. Libros Recomendados

- Aho, Lam, Sethi, Ullman: *Compilers: Principles, Techniques, and Tools* (el Dragón).
- Appel: *Modern Compiler Implementation in ML/Java/C.*
- Grune, Bal, Jacobs: *Modern Compiler Design.*

14.5. Recursos Gratuitos

- **Nand2Tetris**: construcción de una computadora y compilador desde cero.
- **Crafting Interpreters** (Robert Nystrom, gratuito online).
- MIT OCW: 6.035 *Computer Language Engineering*.
- Herramientas prácticas: `ANTLR`, `Bison`, `Flex`.

14.6. Anexo Histórico

- **1950s**: primeros compiladores (Fortran, Lisp).
- **1960s**: desarrollo de teoría formal aplicada a compilación.
- **1970s**: nacimiento de las herramientas `yacc` y `lex`.
- **1980s-2000s**: optimización avanzada, compiladores JIT.
- **Actualidad**: compiladores para lenguajes seguros (ej. Rust), optimización para arquitecturas modernas, compilación cruzada y DSLs.

14.7. Observaciones finales

Esta fase conecta la teoría de lenguajes y autómatas con aplicaciones prácticas. Dominar compiladores permite entender cómo un lenguaje de alto nivel se transforma en instrucciones ejecutables, y abre el camino hacia la creación de nuevos lenguajes o la implementación de intérpretes especializados.

Capítulo 15

Fase 15: Matemática Avanzada y Fronteras de Investigación

15.1. Objetivos

Culminar el recorrido con una visión de las matemáticas y la computación en la frontera del conocimiento. En esta fase se busca:

- Integrar los conocimientos de las fases previas en un marco unificado.
- Introducir ramas modernas de la matemática aplicada y pura.
- Exponer los problemas abiertos que siguen vigentes en matemáticas y ciencias de la computación.
- Brindar un punto de partida para investigación independiente o académica formal.

15.2. Contenidos Principales

15.2.1. Matemática avanzada

- Teoría de categorías y fundamentos de la matemática.
- Topología algebraica y homología.
- Geometría diferencial y variedades.
- Teoría de números avanzada y criptografía poscuántica.
- Álgebra homológica y teoría de representaciones.

15.2.2. Computación avanzada

- Teoría de complejidad más allá de NP (EXPTIME, PSPACE, BPP, clases cuánticas como BQP).
- Computación cuántica y algoritmos cuánticos (Shor, Grover).
- Sistemas distribuidos y verificación formal.

- Autómatas celulares y sistemas dinámicos.
- Inteligencia artificial desde perspectiva formal (máquinas de aprendizaje, autómatas probabilísticos, cadenas de Markov ocultas).

15.2.3. Problemas abiertos

- Problema P vs NP.
- Hipótesis de Riemann.
- Conjetura de Goldbach.
- Problema de la parada en variantes extendidas.
- Problemas abiertos en criptografía poscuántica.

15.3. Ejemplos Prácticos

- Diseñar un autómata probabilístico para modelar cadenas de Markov ocultas (HMM).
- Implementar un algoritmo cuántico simulado (ej. algoritmo de Grover).
- Analizar un problema de optimización compleja y discutir su clase de complejidad.
- Explorar un caso práctico de criptografía poscuántica.

15.4. Libros Recomendados

- Saunders Mac Lane: *Categories for the Working Mathematician.*
- Michael Sipser: *Introduction to the Theory of Computation* (capítulos avanzados).
- Nielsen y Chuang: *Quantum Computation and Quantum Information.*
- Terence Tao: *Analysis* y *Structure and Randomness.*

15.5. Recursos Gratuitos

- Clay Mathematics Institute: listado oficial de problemas del milenio.
- MIT OpenCourseWare: cursos de matemáticas avanzadas y computación cuántica.
- ArXiv.org: repositorio de investigación en matemáticas y computación.
- nLab: enciclopedia colaborativa de teoría de categorías y matemáticas modernas.

15.6. Anexo Histórico

- **Siglo XX**: formalización de la matemática moderna (Bourbaki, Gödel, Turing).
- **Segunda mitad del siglo XX**: auge de la teoría de categorías y la computación.
- **Siglo XXI**: desarrollo de la criptografía poscuántica, inteligencia artificial formal y matemáticas computacionales avanzadas.
- **Presente**: problemas abiertos marcan la agenda de investigación y las fronteras de la matemática.

15.7. Observaciones finales

Esta fase no se estudia de manera lineal ni cerrada: es un punto de inicio hacia la investigación, donde la curiosidad, la lectura de artículos y la participación en comunidades académicas son esenciales. El estudiante debe asumir que este camino es infinito, pero cada paso abre nuevas perspectivas y aplicaciones.