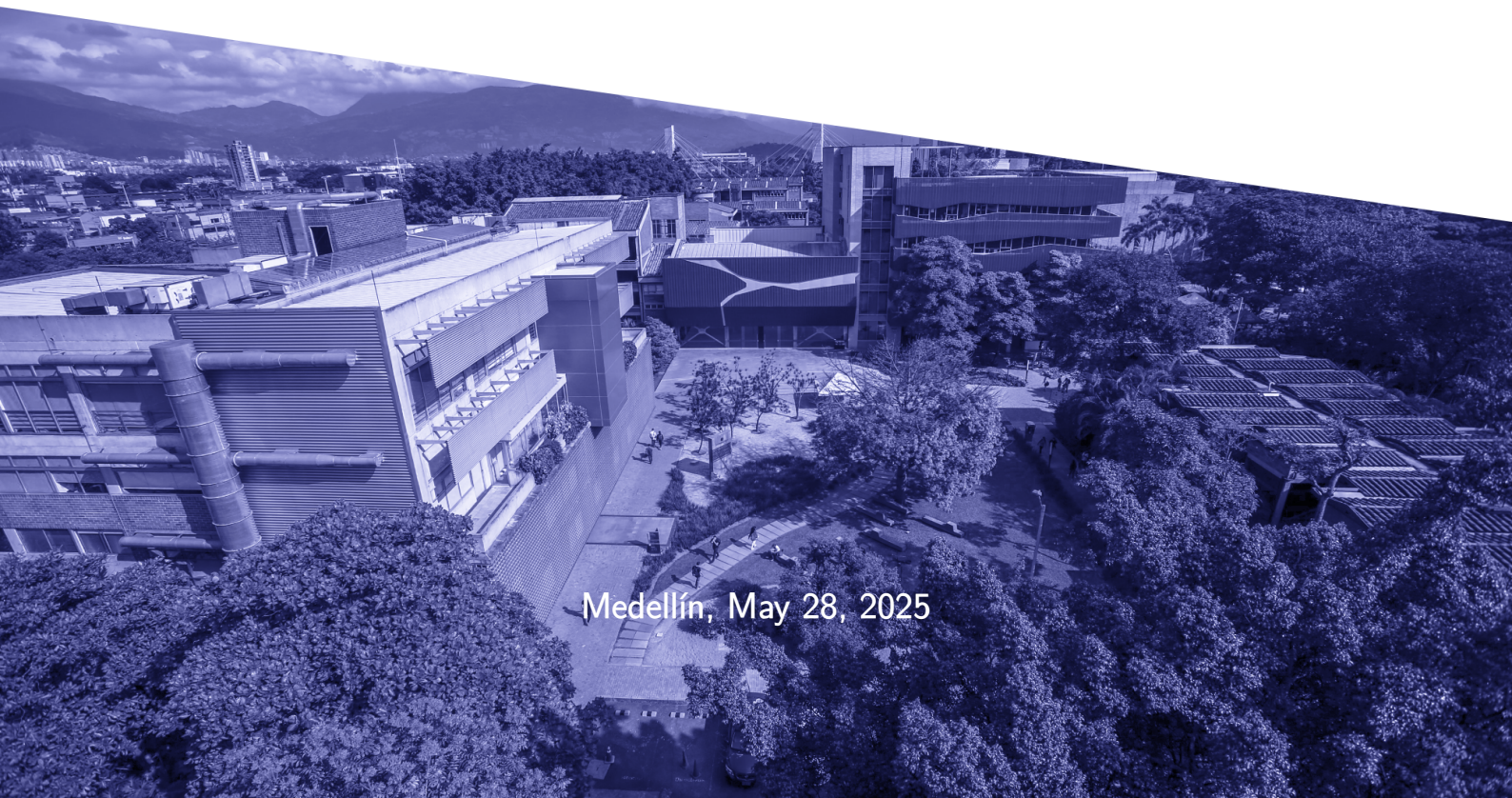
Proyecto Final

Análisis Numérico - C2561-ST0256-4463 (2025-1)

Group Miércoles 3:00 P.M.

Full Name	Student ID
Juan Manuel Young Hoyos	201810117010

Tutor: Julián Rendón Roldán



Medellín, May 28, 2025

Contents

1 Introducción **2**

2 Métodos Implementados **2**
2.1 Métodos de Intervalo 2
2.2 Métodos de Punto Fijo 2
2.3 Métodos Basados en Derivadas 2

3 Arquitectura del Sistema **3**
3.1 Tecnologías Utilizadas 3
3.2 Estructura de Componentes 3

4 Funcionalidades de la Interfaz **3**
4.1 Entrada de Datos 3
4.2 Visualización de Resultados 4

5 Sistema de Comparación **4**
5.1 Metodología de Análisis 4
5.2 Métricas Reportadas 4

6 Manejo de Casos Especiales **4**
6.1 Validación de Entrada 4
6.2 Manejo de Errores 5

7 Experiencia de Usuario **5**
7.1 Diseño de Interfaz 5
7.2 Accesibilidad 5

8 Casos de Uso y Ejemplos **5**
8.1 Función de Prueba Principal 5
8.2 Resultados Típicos 5

9 Conclusiones y Trabajo Futuro **6**
9.1 Logros del Proyecto 6
9.2 Extensiones Planificadas 6

Métodos Numéricos para Encontrar Raíces de Ecuaciones

1 | Introducción

En el presente capítulo se desarrolla una aplicación web interactiva denominada **Senku Solver** que implementa seis métodos numéricos fundamentales para la búsqueda de raíces de ecuaciones no lineales. La aplicación está construida utilizando tecnologías web modernas como React, TypeScript y Vite, proporcionando una interfaz de usuario intuitiva y responsiva que permite a los usuarios explorar, comparar y analizar el comportamiento de diferentes algoritmos numéricos.

2 | Métodos Implementados

La aplicación incluye la implementación completa de los siguientes métodos numéricos:

2.1 | Métodos de Intervalo

2.1.1 | Método de Bisección

Implementa el algoritmo clásico de bisección que garantiza la convergencia cuando la función cambia de signo en el intervalo dado. El método utiliza la fórmula:

$$x_m = \frac{x_i + x_s}{2}$$

donde se selecciona el subintervalo que mantiene el cambio de signo. La convergencia está garantizada con una tasa de convergencia lineal.

2.1.2 | Método de Regla Falsa (False Position)

Mejora el método de bisección utilizando interpolación lineal para aproximar la raíz. La fórmula utilizada es:

$$x_r = x_i - \frac{f(x_i)(x_s - x_i)}{f(x_s) - f(x_i)}$$

Este método generalmente converge más rápido que bisección manteniendo la robustez de los métodos de intervalo.

2.2 | Métodos de Punto Fijo

2.2.1 | Método de Punto Fijo

Transforma la ecuación $f(x) = 0$ en la forma $x = g(x)$ y utiliza la iteración:

$$x_{n+1} = g(x_n)$$

La convergencia depende de que $|g'(x)| < 1$ en el intervalo de interés. La aplicación permite al usuario especificar tanto $f(x)$ como $g(x)$ para máxima flexibilidad.

2.3 | Métodos Basados en Derivadas

2.3.1 | Método de Newton-Raphson

Implementa el algoritmo clásico de Newton que utiliza información de la derivada:

$$x_{n+1} = x_n - \frac{f(x_n)}{f'(x_n)}$$

Ofrece convergencia cuadrática cuando las condiciones iniciales son apropiadas, pero requiere el cálculo de la derivada.

2.3.2 | Método de la Secante

Aproxima la derivada utilizando dos puntos anteriores, eliminando la necesidad de calcular derivadas analíticamente:

$$x_{n+1} = x_n - \frac{f(x_n)(x_n - x_{n-1})}{f(x_n) - f(x_{n-1})}$$

Proporciona una convergencia superlineal (orden ≈ 1.618) sin requerir derivadas.

2.3.3 | Método de Raíces Múltiples

Utiliza una modificación del método de Newton para manejar raíces múltiples donde $f(x) = f'(x) = 0$:

$$x_{n+1} = x_n - \frac{f(x_n) \cdot f'(x_n)}{[f'(x_n)]^2 - f(x_n) \cdot f''(x_n)}$$

Este método restaura la convergencia cuadrática en presencia de raíces múltiples.

3 | Arquitectura del Sistema

3.1 | Tecnologías Utilizadas

La aplicación utiliza un stack tecnológico moderno:

- **Frontend**: React con TypeScript para type safety y mejor experiencia de desarrollo
- **Build Tool**: Vite para compilación rápida y hot reloading
- **UI Framework**: shadcn/ui con Tailwind CSS para interfaces modernas y responsivas
- **Routing**: TanStack Router para navegación tipo-segura
- **Icons**: Tabler Icons y Lucide React para iconografía consistente

3.2 | Estructura de Componentes

Cada método numérico sigue una arquitectura modular consistente:

- **Algoritmo**: Implementación pura en TypeScript sin dependencias de UI
- **Componente Principal**: Interfaz de usuario para configuración y resultados
- **Tabla de Iteraciones**: Componente especializado para mostrar datos tabulares
- **Ruta**: Configuración de navegación usando file-based routing

4 | Funcionalidades de la Interfaz

4.1 | Entrada de Datos

La aplicación proporciona una interfaz intuitiva para la entrada de datos:

- **Funciones Matemáticas**: Soporte completo para funciones algebraicas usando sintaxis JavaScript estándar
- **Operadores Soportados**: funciones trigonométricas (sin, cos, tan), exponencial (exp), logaritmo (log), raíz cuadrada, y operaciones aritméticas básicas
- **Validación en Tiempo Real**: Evaluación inmediata de expresiones para detectar errores
- **Ejemplos Contextuales**: Placeholders específicos para cada método con ejemplos relevantes

4.2 | Visualización de Resultados

Cada método presenta sus resultados de manera clara y organizada:

- **Estado de Convergencia**: Badges visuales indicando éxito o falla
- **Métricas Principales**: Raíz aproximada, error final, número de iteraciones
- **Tablas Detalladas**: Historial completo de iteraciones con formateo numérico apropiado
- **Mensajes Informativos**: Explicaciones contextuales del comportamiento del algoritmo

5 | Sistema de Comparación

5.1 | Metodología de Análisis

La aplicación incluye un módulo de comparación automática que ejecuta todos los métodos con los mismos parámetros y función objetivo. El análisis utiliza los siguientes criterios:

5.1.1 | Criterios de Evaluación

1. **Convergencia**: Prioridad máxima - el método debe converger a una solución válida
2. **Eficiencia Iterativa**: Entre métodos convergentes, se prefiere menor número de iteraciones
3. **Tiempo de Ejecución**: Medición precisa usando `performance.now()` para comparar velocidad
4. **Precisión**: Evaluación del error final alcanzado

5.1.2 | Proceso de Comparación

El sistema ejecuta automáticamente:

1. Configuración uniforme de parámetros (tolerancia, máximo de iteraciones)
2. Ejecución secuencial con medición temporal
3. Análisis estadístico de resultados
4. Identificación del método óptimo basado en criterios ponderados
5. Generación de informe exportable en formato CSV

5.2 | Métricas Reportadas

El informe de comparación incluye:

- **Tabla Comparativa**: Resultados lado a lado de todos los métodos
- **Identificación del Mejor Método**: Destacado visual del algoritmo óptimo
- **Estadísticas Agregadas**: Promedio de iteraciones, tiempo total, tasa de convergencia
- **Exportación de Datos**: Capacidad de descargar resultados para análisis posterior

6 | Manejo de Casos Especiales

6.1 | Validación de Entrada

El sistema implementa validaciones robustas:

- **Sintaxis de Funciones**: Verificación de expresiones matemáticas válidas
- **Condiciones de Convergencia**: Validación específica por método (ej. cambio de signo para bisección)
- **Parámetros Numéricos**: Verificación de rangos válidos para tolerancia e iteraciones

6.2 | Manejo de Errores

Cada algoritmo incluye manejo específico de casos problemáticos:

- **División por Cero**: Detección y manejo de derivadas nulas
- **No Convergencia**: Límites de iteración con mensajes informativos
- **Errores de Evaluación**: Captura de excepciones en evaluación de funciones

7 | Experiencia de Usuario

7.1 | Diseño de Interfaz

La aplicación utiliza principios de UX moderno:

- **Diseño Responsivo**: Adaptación automática a diferentes tamaños de pantalla
- **Tema Oscuro/Claro**: Soporte completo para preferencias de usuario
- **Navegación Intuitiva**: Sidebar organizado por categorías de métodos
- **Feedback Visual**: Indicadores de carga y estados de proceso

7.2 | Accesibilidad

Implementación de estándares de accesibilidad web:

- **Navegación por Teclado**: Soporte completo para usuarios sin mouse
- **Contraste Adecuado**: Cumplimiento de estándares WCAG
- **Etiquetas Semánticas**: HTML estructurado para lectores de pantalla

8 | Casos de Uso y Ejemplos

8.1 | Función de Prueba Principal

La aplicación utiliza por defecto la función:

$$f(x) = \log(\sin^2(x) + 1) - \frac{1}{2}$$

Esta función presenta características interesantes para el análisis:

- Comportamiento no lineal complejo
- Múltiples raíces en diferentes intervalos
- Permite evaluación comparativa significativa

8.2 | Resultados Típicos

En pruebas con la función por defecto, los métodos muestran el siguiente comportamiento relativo:

- **Newton-Raphson**: Convergencia más rápida cuando la derivada está disponible
- **Secante**: Buen balance entre velocidad y simplicidad de implementación
- **Bisección**: Mayor robustez pero convergencia más lenta
- **Regla Falsa**: Mejora sobre bisección manteniendo garantías de convergencia

9 | Conclusiones y Trabajo Futuro

9.1 | Logros del Proyecto

El desarrollo de Senku Solver representa una implementación exitosa de:

1. Sistema completo de métodos numéricos para búsqueda de raíces
2. Interfaz de usuario moderna e intuitiva
3. Herramientas de análisis y comparación automática
4. Arquitectura modular y extensible

9.2 | Extensiones Planificadas

El proyecto está diseñado para futuras extensiones:

- **Visualización Gráfica**: Integración de gráficas de funciones y convergencia
- **Métodos Adicionales**: Implementación de variantes y métodos especializados
- **Optimización WASM**: Migración de algoritmos críticos a WebAssembly para mayor rendimiento
- **API Backend**: Desarrollo de servicios para análisis más complejos

La implementación actual establece una base sólida para el desarrollo de una suite completa de herramientas de análisis numérico, cumpliendo con los objetivos académicos mientras proporciona una experiencia de usuario profesional y moderna.