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

Senku Solver: Suite Completa de Métodos Numéricos

1 | Introducción

En el presente documento se desarrolla una aplicación web interactiva denominada **Senku Solver** que implementa una suite completa de métodos numéricos fundamentales, abarcando tanto la búsqueda de raíces de ecuaciones no lineales como la resolución de sistemas de ecuaciones lineales mediante métodos iterativos. 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 en ambas categorías.

2 | Capítulo 1: Métodos para Encontrar Raíces

La aplicación incluye la implementación completa de seis métodos numéricos para la búsqueda de raíces de ecuaciones no lineales:

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 | Capítulo 2: Métodos Iterativos para Sistemas Lineales

La aplicación implementa tres métodos iterativos fundamentales para resolver sistemas de ecuaciones lineales de la forma $Ax = b$, donde A es una matriz cuadrada de hasta 7×7, x es el vector solución y b es el vector de términos independientes.

3.1 | Fundamentos Teóricos

Todos los métodos iterativos implementados se basan en la descomposición de la matriz A en la forma:

$$A = D + L + U$$

donde:

- D es la matriz diagonal
- L es la matriz triangular inferior estricta
- U es la matriz triangular superior estricta

La convergencia de estos métodos está determinada por el **radio espectral** ρ de la matriz de iteración correspondiente. La condición necesaria y suficiente para convergencia es $\rho < 1$.

3.2 | Método de Jacobi

3.2.1 | Formulación

El método de Jacobi utiliza todos los valores de la iteración anterior para calcular la nueva aproximación:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1, j \neq i}^{n} a_{ij} x_j^{(k)} \right)$$

La matriz de iteración es:

$$T_J = D^{-1}(L + U)$$

3.2.2 | Implementación

La implementación incluye:

- Cálculo automático del radio espectral usando círculos de Gershgorin
- Predicción teórica de convergencia ($\rho < 1$)
- Validación de elementos diagonales no nulos
- Seguimiento detallado de iteraciones con errores absolutos y relativos

3.3 | Método de Gauss-Seidel

3.3.1 | Formulación

Gauss-Seidel mejora Jacobi utilizando los valores más recientes disponibles en cada iteración:

$$x_i^{(k+1)} = \frac{1}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)$$

La matriz de iteración es:

$$T_{GS} = (D + L)^{-1} U$$

3.3.2 | Ventajas Implementadas

- Convergencia generalmente más rápida que Jacobi
- Menor uso de memoria (actualización in-place)
- Análisis de dominancia diagonal automático
- Predicción mejorada de comportamiento de convergencia

3.4 | Método SOR (Successive Over-Relaxation)

3.4.1 | Formulación

SOR introduce un factor de relajación ω para acelerar la convergencia:

$$x_i^{(k+1)} = (1 - \omega) x_i^{(k)} + \frac{\omega}{a_{ii}} \left(b_i - \sum_{j=1}^{i-1} a_{ij} x_j^{(k+1)} - \sum_{j=i+1}^{n} a_{ij} x_j^{(k)} \right)$$

La matriz de iteración es:

$$T_{SOR} = (D - \omega L)^{-1} [(1 - \omega) D + \omega U]$$

3.4.2 | Optimización del Factor ω

La implementación incluye:

- **Estimación automática**: Cálculo de ω óptimo basado en propiedades de la matriz
- **Validación de rango**: $0 < \omega < 2$ para garantizar convergencia
- **Casos especiales**: $\omega = 1$ (Gauss-Seidel), $\omega < 1$ (sub-relajación), $\omega > 1$ (sobre-relajación)
- **Análisis de rendimiento**: Potencial aceleración hasta $2\times$ respecto a Gauss-Seidel

3.5 | Análisis de Convergencia

3.5.1 | Radio Espectral

Para cada método, la aplicación calcula el radio espectral utilizando aproximaciones basadas en:

- Círculos de Gershgorin para estimación inicial
- Análisis de dominancia diagonal estricta
- Predicción teórica de convergencia

3.5.2 | Condiciones de Convergencia

- **Dominancia diagonal estricta**: $|a_{ii}| > \sum_{j \neq i} |a_{ij}|$ garantiza convergencia
- **Matrices definidas positivas**: Convergencia asegurada para Gauss-Seidel y SOR
- **Validación automática**: Verificación de condiciones en tiempo real

4 | Sistema de Comparación Iterativa

4.1 | Metodología de Análisis Comparativo

La aplicación incluye un módulo especializado de comparación que ejecuta simultáneamente los tres métodos iterativos con los mismos parámetros y matriz objetivo.

4.1.1 | Criterios de Evaluación

1. **Convergencia**: Prioridad máxima - verificación de $\rho < 1$ y convergencia efectiva
2. **Eficiencia Iterativa**: Número de iteraciones requeridas para alcanzar la tolerancia
3. **Tiempo de Ejecución**: Medición precisa usando `performance.now()`
4. **Radio Espectral**: Análisis teórico de la velocidad de convergencia
5. **Precisión Final**: Error relativo alcanzado en la solución

4.1.2 | Proceso de Comparación Automática

1. **Configuración Uniforme**: Misma matriz, vector, tolerancia y límite de iteraciones
2. **Ejecución Paralela Conceptual**: Medición independiente de cada método
3. **Análisis Estadístico**: Cálculo de métricas comparativas
4. **Identificación del Método Óptimo**: Basado en criterios ponderados
5. **Generación de Informe**: Exportación completa en formato CSV

4.2 | Métricas del Informe Comparativo

4.2.1 | Tabla Comparativa Principal

- Estado de convergencia con badges visuales
- Radio espectral calculado para cada método
- Predicción teórica de convergencia ($\rho < 1$)
- Número de iteraciones y tiempo de ejecución
- Error final y solución completa
- Factor ω utilizado en SOR

4.2.2 | Estadísticas Agregadas

- Número de métodos convergentes
- Promedio de iteraciones entre métodos exitosos
- Mejor radio espectral obtenido
- Tiempo total de análisis

4.2.3 | Informe Automático de Análisis

El sistema genera recomendaciones automáticas basadas en:

- **Análisis de convergencia**: Verificación teórica vs. práctica
- **Rendimiento comparativo**: Identificación del método más eficiente
- **Recomendaciones específicas**: Sugerencias para optimización

5 | Arquitectura del Sistema

5.1 | Tecnologías Utilizadas

La aplicación utiliza un stack tecnológico moderno y robusto:

- **Frontend**: React 18 con TypeScript para type safety y mejor experiencia de desarrollo
- **Build Tool**: Vite para compilación rápida y hot reloading optimizado
- **UI Framework**: shadcn/ui con Tailwind CSS para interfaces modernas y responsivas
- **Routing**: TanStack Router para navegación tipo-segura y lazy loading
- **Icons**: Tabler Icons y Lucide React para iconografía consistente
- **Algoritmos**: Implementaciones puras en TypeScript sin dependencias externas

5.2 | Estructura Modular

Cada método numérico (tanto de raíces como iterativo) sigue una arquitectura consistente:

- **Algoritmo Core**: Implementación pura en `/src/utils/algorithms/`
- **Componente Principal**: Interfaz de usuario en `/src/features/methods/`
- **Tabla de Resultados**: Componente especializado para visualización tabular
- **Entrada de Datos**: Componentes reutilizables para configuración
- **Ruta**: Configuración de navegación usando file-based routing

5.3 | Componentes Compartidos

5.3.1 | MatrixInput

Componente unificado para entrada de sistemas lineales que incluye:

- Modo texto con sintaxis flexible (espacios o comas)
- Validación en tiempo real de matrices hasta 7×7
- Ejemplos precargados (2×2, 3×3, 4×4)
- Verificación de dominancia diagonal automática
- Estimación de parámetros óptimos (factor ω para SOR)

5.3.2 | Utilidades de Matriz

Funciones especializadas en `/src/utils/matrix-utils.ts`:

- Parsing y validación de entrada
- Verificación de dominancia diagonal
- Cálculo de radio espectral aproximado
- Estimación de factor ω óptimo
- Generación de ejemplos educativos

6 | Funcionalidades de la Interfaz

6.1 | Entrada de Datos Unificada

6.1.1 | Para Métodos de Raíces

- **Funciones Matemáticas**: Soporte completo para funciones algebraicas usando sintaxis JavaScript
- **Operadores Soportados**: funciones trigonométricas, exponencial, logaritmo, raíz cuadrada
- **Validación en Tiempo Real**: Evaluación inmediata para detectar errores
- **Ejemplos Contextuales**: Placeholders específicos con función por defecto

6.1.2 | Para Métodos Iterativos

- **Matriz y Vector**: Entrada flexible con validación automática
- **Guess Inicial**: Vector de aproximación inicial configurable
- **Parámetros de Convergencia**: Tolerancia y límite de iteraciones
- **Configuración SOR**: Factor ω automático o personalizado

6.2 | Visualización Avanzada de Resultados

6.2.1 | Métricas Principales

- **Estado de Convergencia**: Badges visuales con códigos de color
- **Radio Espectral**: Indicador visual de convergencia teórica
- **Eficiencia**: Iteraciones y tiempo de ejecución
- **Precisión**: Error final con notación científica

6.2.2 | Tablas Interactivas

- **Historial Completo**: Todas las iteraciones con formateo numérico
- **Exportación CSV**: Datos completos para análisis externo
- **Truncamiento Inteligente**: Primeras 50 iteraciones para rendimiento
- **Scroll Responsive**: Adaptación a diferentes tamaños de pantalla

7 | Sistemas de Comparación

7.1 | Comparación de Métodos de Raíces

Análisis automático que ejecuta todos los métodos de búsqueda de raíces con los mismos parámetros:

7.1.1 | Criterios de Evaluación

1. Convergencia exitosa a una raíz válida
2. Menor número de iteraciones entre métodos convergentes
3. Tiempo de ejecución optimizado
4. Precisión del resultado final

7.2 | Comparación de Métodos Iterativos

Sistema especializado para análisis de sistemas lineales:

7.2.1 | Análisis Teórico vs. Práctico

- Predicción de convergencia basada en radio espectral
- Verificación experimental de las predicciones teóricas
- Análisis de discrepancias entre teoría y práctica

7.2.2 | Identificación del Método Óptimo

Algoritmo que considera:

1. Convergencia exitosa (requisito obligatorio)
2. Menor número de iteraciones
3. Tiempo de ejecución más rápido
4. Radio espectral más favorable

8 | Casos de Uso y Validación

8.1 | Ejemplos por Defecto

8.1.1 | Función de Prueba para Raíces

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

Esta función permite evaluación comparativa efectiva debido a su comportamiento no lineal complejo.

8.1.2 | Sistema Lineal de Prueba

Matriz 3×3 diagonalmente dominante:

$$A = \begin{pmatrix} 4 & -1 & 0 \\ -1 & 4 & -1 \\ 0 & -1 & 4 \end{pmatrix}, \quad b = \begin{pmatrix} 3 \\ 7 \\ 6 \end{pmatrix}$$

Esta matriz garantiza convergencia para todos los métodos iterativos y permite demostrar las diferencias de rendimiento.

8.2 | Validación de Resultados

8.2.1 | Precisión Numérica

- Tolerancias configurables desde 10^{-1} hasta 10^{-15}
- Manejo de aritmética de punto flotante
- Detección de estancamiento numérico

8.2.2 | Casos Límite

- Matrices mal condicionadas
- Sistemas con convergencia lenta
- Matrices sin dominancia diagonal

9 | Manejo de Casos Especiales

9.1 | Validación Robusta

9.1.1 | Para Métodos de Raíces

- Sintaxis de funciones matemáticas
- Condiciones de convergencia específicas por método
- Parámetros numéricos en rangos válidos

9.1.2 | Para Métodos Iterativos

- Matrices cuadradas hasta 7×7
- Elementos diagonales no nulos
- Coherencia dimensional entre matriz y vectores
- Factor ω en rango válido $(0, 2)$ para SOR

9.2 | Manejo de Errores Específicos

9.2.1 | Algoritmos de Raíces

- División por cero en derivadas
- Ausencia de cambio de signo en métodos de intervalo
- Convergencia a puntos de inflexión

9.2.2 | Métodos Iterativos

- Matrices singulares o mal condicionadas
- Divergencia por radio espectral ≥ 1
- Stancamiento por tolerancia demasiado restrictiva

10 | Experiencia de Usuario

10.1 | Diseño de Interfaz Moderna

- **Navegación Organizada**: Sidebar categorizado por tipo de método
- **Diseño Responsivo**: Adaptación fluida a móviles, tablets y desktop
- **Tema Dual**: Soporte completo para modo oscuro y claro
- **Feedback Visual**: Estados de carga, progreso y resultados
- **Accessibility**: Navegación por teclado y etiquetas semánticas

10.2 | Flujo de Trabajo Optimizado

- **Valores por Defecto**: Ejemplos precargados para prueba inmediata
- **Validación Proactiva**: Verificación en tiempo real
- **Exportación Integrada**: Descarga de resultados en un clic
- **Comparación Opcional**: El usuario decide si ejecutar análisis comparativo

11 | Conclusiones y Trabajo Futuro

11.1 | Logros del Proyecto

El desarrollo de Senku Solver representa una implementación exitosa y completa de:

1. **Suite Dual**: Métodos de raíces (6 algoritmos) y métodos iterativos (3 algoritmos)
2. **Interfaz Unificada**: UX consistente entre diferentes categorías de métodos
3. **Análisis Comparativo Dual**: Sistemas especializados para cada tipo de problema
4. **Arquitectura Escalable**: Base sólida para futuras extensiones
5. **Calidad Académica**: Implementaciones fieles a la teoría numérica

11.2 | Innovaciones Técnicas

- **Radio Espectral Automático**: Cálculo y predicción de convergencia en tiempo real
- **Optimización SOR**: Estimación automática del factor ω óptimo
- **Validación Inteligente**: Verificación específica por método y contexto
- **Comparación Automática**: Análisis y recomendaciones sin intervención manual

11.3 | Extensiones Planificadas

11.3.1 | Mejoras Inmediatas

- **Visualización Gráfica**: Gráficas de convergencia y comportamiento iterativo
- **Métodos Adicionales**: Implementación de variantes especializadas
- **Exportación Avanzada**: Formatos adicionales (PDF, LaTeX)

11.3.2 | Desarrollo a Largo Plazo

- **Optimización WASM**: Migración de algoritmos críticos a WebAssembly
- **API Backend**: Servicios para análisis de matrices grandes
- **Capítulo 3**: Interpolación y aproximación numérica
- **Capítulo 4**: Diferenciación e integración numérica

11.4 | Impacto Educativo

Senku Solver establece un nuevo estándar para herramientas educativas de análisis numérico:

- **Aprendizaje Interactivo**: Experimentación directa con algoritmos
- **Análisis Comparativo**: Comprensión profunda de trade-offs algorítmicos
- **Validación Práctica**: Verificación de conceptos teóricos
- **Accesibilidad Universal**: Disponible en cualquier dispositivo con navegador web

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