Universidade de Brasília
Departamento de Ciências Mecânicas
Programa de Pós-Graduação

Programa 5

Solução de Problemas Lineares - Estudo de caso

Disciplina: Métodos Numéricos
Professor: Dr. Rafael Gabler Gontijo

Aluno: Eng. Lucas Wanick — Mestrando em Engenharia Mecânica

7 de junho de 2025

1 Introdução

O cálculo das raízes de polinômios reais é um problema clássico da análise numérica, com aplicações em diversas áreas da engenharia. Entre os métodos tradicionais para esta tarefa, o **Método de Bairstow** destaca-se por sua capacidade de extrair raízes reais e complexas conjugadas a partir de um polinômio com coeficientes reais, sem necessidade de operações sobre números complexos durante o processo iterativo.

O método consiste em fatorar o polinômio original $P(x)$ em fatores quadráticos da forma:

$$x^2 + rx + s$$

A cada etapa, o algoritmo ajusta iterativamente os parâmetros (r, s) de modo que o fator extraído aproxime um par de raízes reais ou complexas conjugadas. A partir das raízes do fator quadrático, o polinômio é deflacionado, e o processo se repete até que todas as raízes sejam determinadas.

Apesar de sua eficiência, o Método de Bairstow é sensível às condições iniciais de (r, s), e pode apresentar dificuldades em casos de raízes múltiplas, raízes próximas ou polinômios degenerados. Esta sensibilidade motiva a utilização de uma abordagem de *busca em malha* (grid search), onde diversas combinações iniciais de (r, s) são testadas para mapear a convergência do método.

Curiosamente, o comportamento do Método de Bairstow sobre o espaço de chutes iniciais (r, s) revela padrões fractais — regiões com rápida convergência alternam com regiões de lenta convergência ou não convergência, produzindo figuras complexas e auto-similares. Estes **fractais de iteração** constituem uma ferramenta visual poderosa para estudar a dinâmica do método.

2 Descrição do Método

O Método de Bairstow parte de um polinômio de grau n com coeficientes reais:

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

Em cada etapa, busca-se dividir $P(x)$ por um fator quadrático proposto:

$$Q(x) = x^2 + rx + s$$

resultando em:

$$P(x) = Q(x) \cdot B(x) + R(x)$$

onde $B(x)$ é o quociente, e $R(x)$ é o resto, que deve tender a zero para que o fator $Q(x)$ represente um par de raízes válido.

O método utiliza as fórmulas de Bairstow, derivadas da diferenciação do esquema de Horner, para atualizar iterativamente (r, s) até que o resíduo seja menor que uma tolerância pré-definida.

Após a extração de um fator quadrático, o polinômio é deflacionado e o processo prossegue até que seu grau seja reduzido a zero.

3 Malha de Chutes e Relação com Fractais

Devido à sensibilidade do método às condições iniciais, implementou-se uma **malha de chutes** — uma discretização do plano (r, s) — onde para cada ponto (r, s) da malha é realizada uma tentativa completa de fatoração.

Em cada ponto, é registrado o número de iterações acumuladas para extrair todas as raízes. A malha resultante permite construir uma imagem fractal, onde a coloração representa o número de iterações.

Este mapa revela a topologia do espaço de chutes — regiões de estabilidade, de rápida convergência, de comportamento caótico e de não convergência — evidenciando a complexidade dinâmica do Método de Bairstow.

4 Estruturação do Código

O código foi estruturado em Python, com foco em modularidade e robustez. As principais escolhas e soluções foram:

- Controle do número de threads das bibliotecas `OpenBLAS`, `MKL` e `OpenMP` para evitar conflitos e sobrecarga em ambientes multicore;
- Função `safe_input()` para permitir interrupção segura do programa
- Detecção de polinômios degenerados (coeficientes nulos) com interrupção preventiva
- Estratégia de fallback linear quando o fator quadrático não converge;
- Eliminação de loops infinitos via controle refinado das iterações;
- Registro acumulado do número de iterações, respeitando o comportamento natural do método (cada fator quadrático ou linear contribui com até `maxit` iterações).

O código também implementa rotinas para geração de fractais em escala HSV, exportação de dados e interface interativa com o usuário.

5 Dificuldades Enfrentadas

Durante o desenvolvimento, as principais dificuldades enfrentadas foram:

- Loop infinito causado pelo reinício da contagem de iterações em casos de singularidade no Jacobiano;
- Sensibilidade extrema do método em polinômios com raízes múltiplas e/ou muito próximas;
- Necessidade de preservar o mapa de iterações mesmo quando o método não converge (para que o fractal seja informativo);
- Controle do custo computacional da malha, que cresce com a resolução e o grau do polinômio.

Todas essas questões foram superadas com ajustes na lógica de iteração, controle de deflação e validação da convergência.

6 Custo Computacional

O custo computacional do método cresce linearmente com o número de pontos da malha (r, s) e com o número de iterações realizadas por ponto.

$$C \propto N_r \times N_s \times \left(\sum_{\text{raízes}} \text{iter\_count} \right)$$

O uso de paralelismo (`joblib`) permite explorar múltiplos núcleos, mas o custo ainda é considerável para polinômios de alto grau e malhas finas. Tipicamente, malhas de 1000×1000 com polinômios de grau 5 ou 6 requerem minutos de execução.,

7 Exemplos e Resultados

Foram utilizados diversos polinômios de teste:

- Grau 3: $x^3 - 6x^2 + 11x - 6$;
- Grau 4: $x^4 - 10x^3 + 35x^2 - 50x + 24$;
- Grau 5: $x^5 - 5x^4 + 4x^3 + 22x^2 - 21x - 18$;
- Grau 6: $x^6 - 3x^4 + 3x^2 - 1$.

A seguir, apresentamos alguns dos fractais gerados:

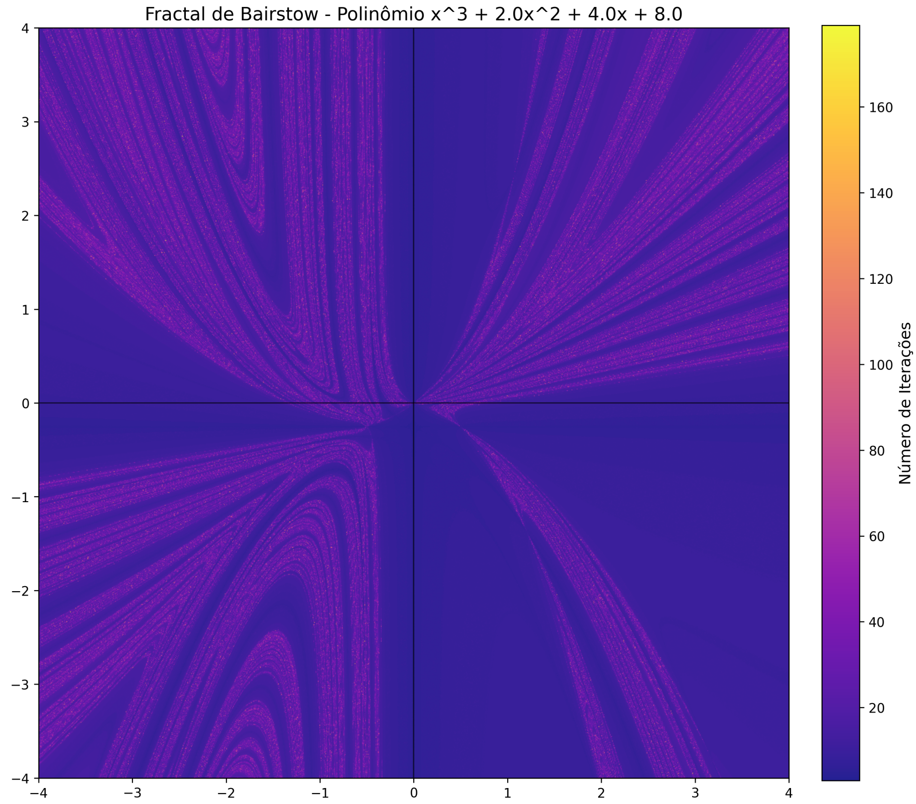

Figura 1: Fractal de iterações para $x^3 + 2x^2 + 4x + 8$.

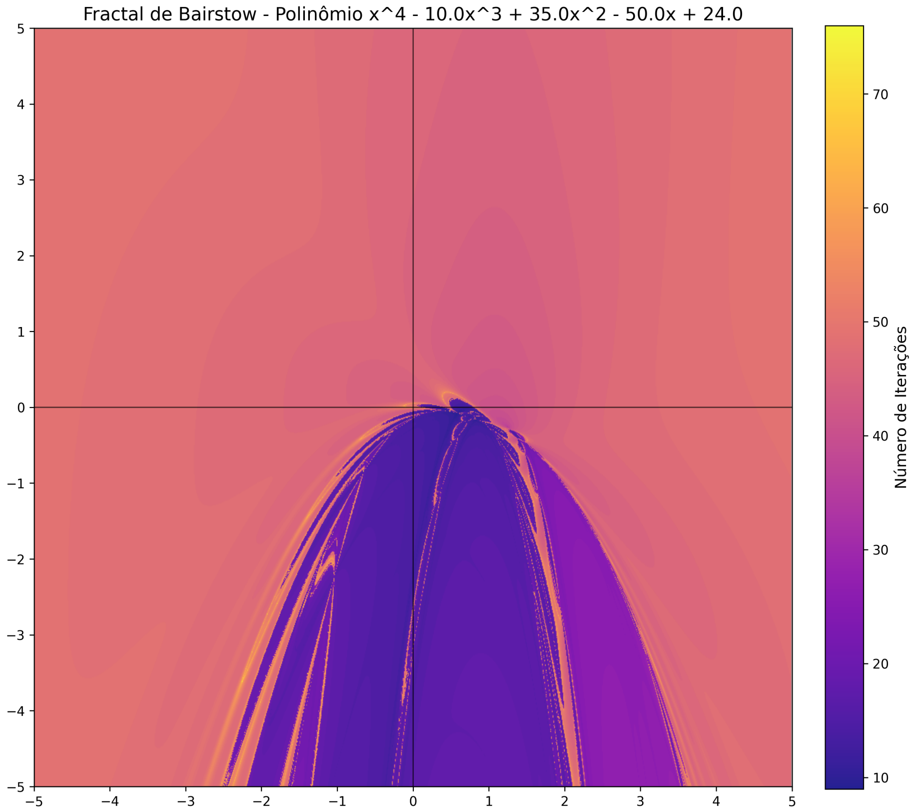

Figura 2: Fractal de iterações para $x^4 - 10x^3 + 35x^2 - 50x + 24$.

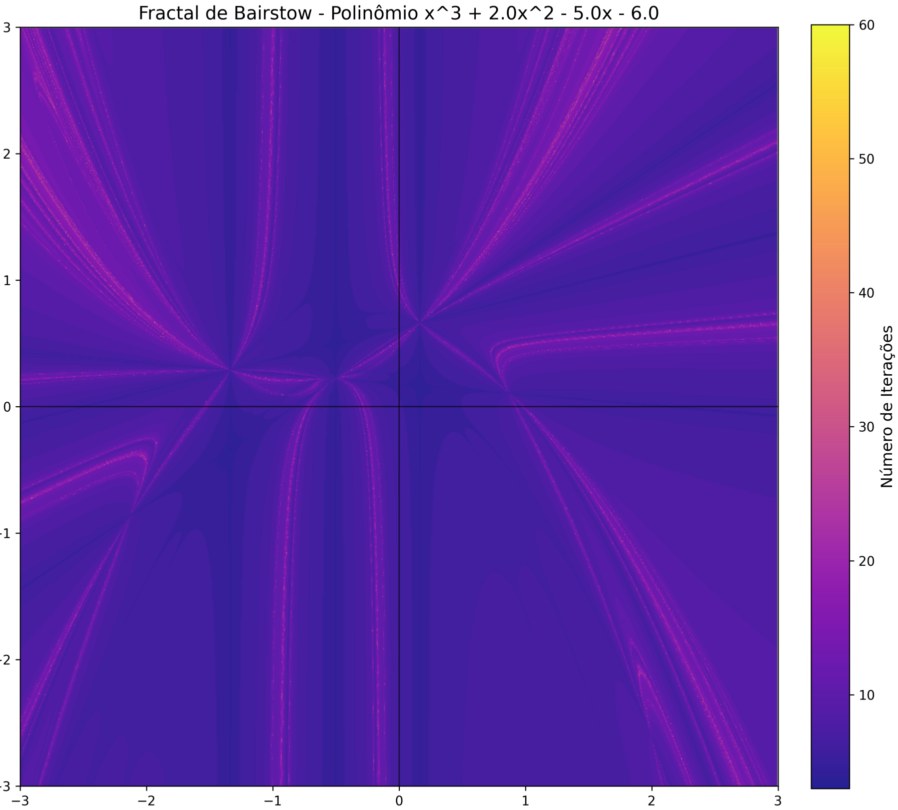

Figura 3: Fractal de iterações para $x^3 + 2x^2 - 5x - 6$.

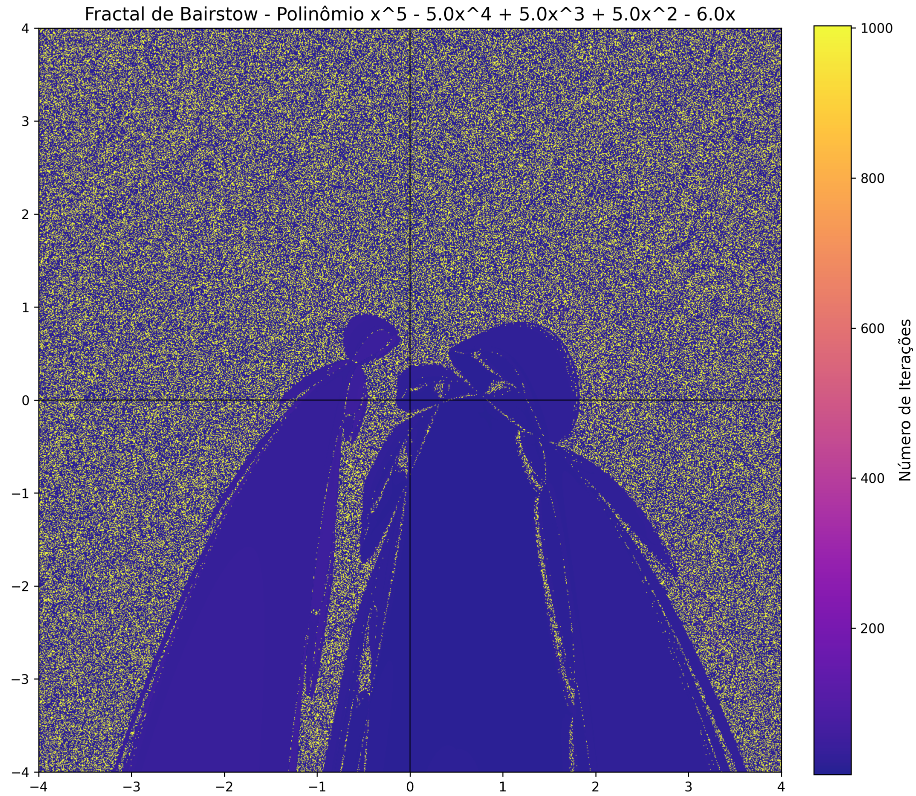

Figura 4: Fractal de iterações para $x^5 - 5x^4 + 5x^3 + 5x^2 - 6x$.

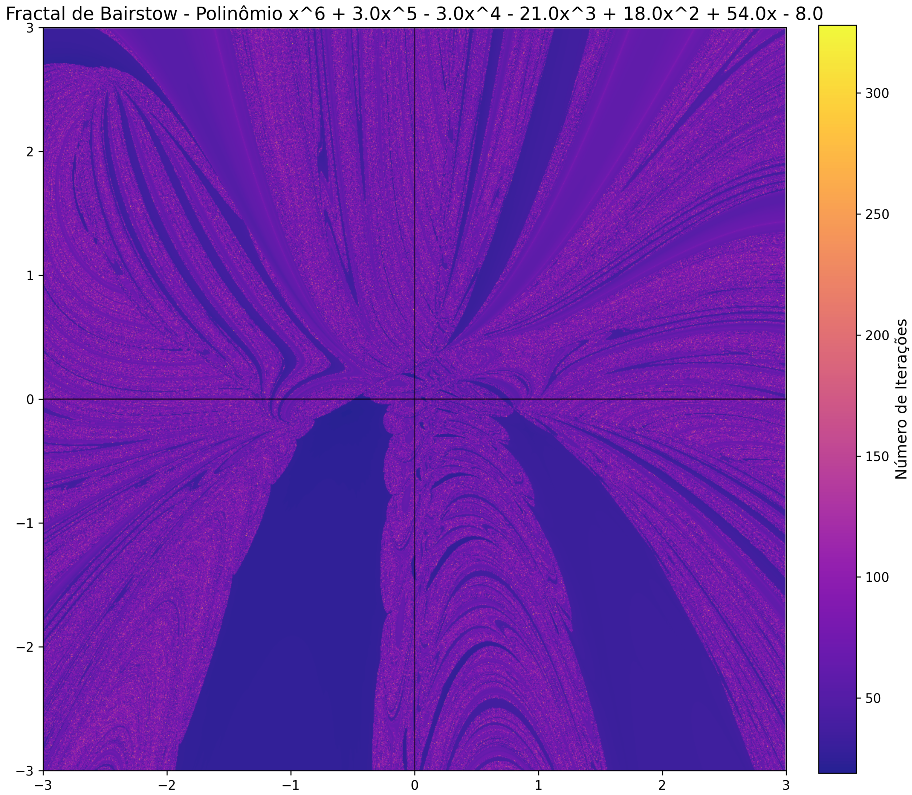

Figura 5: Fractal de iterações para $x^6 + 3x^5 - 3x^4 - 21x^3 + 18x^2 + 54x - 8$.

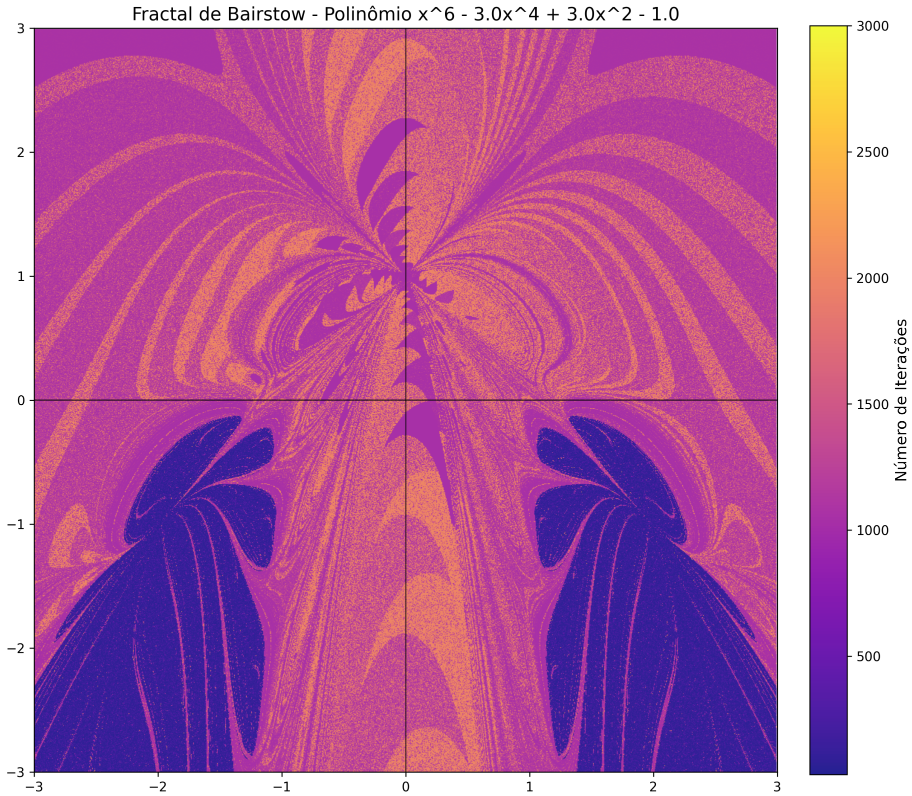

Figura 6: Fractal de iterações para $x^6 - 3x^4 + 3x^2 - 1$.

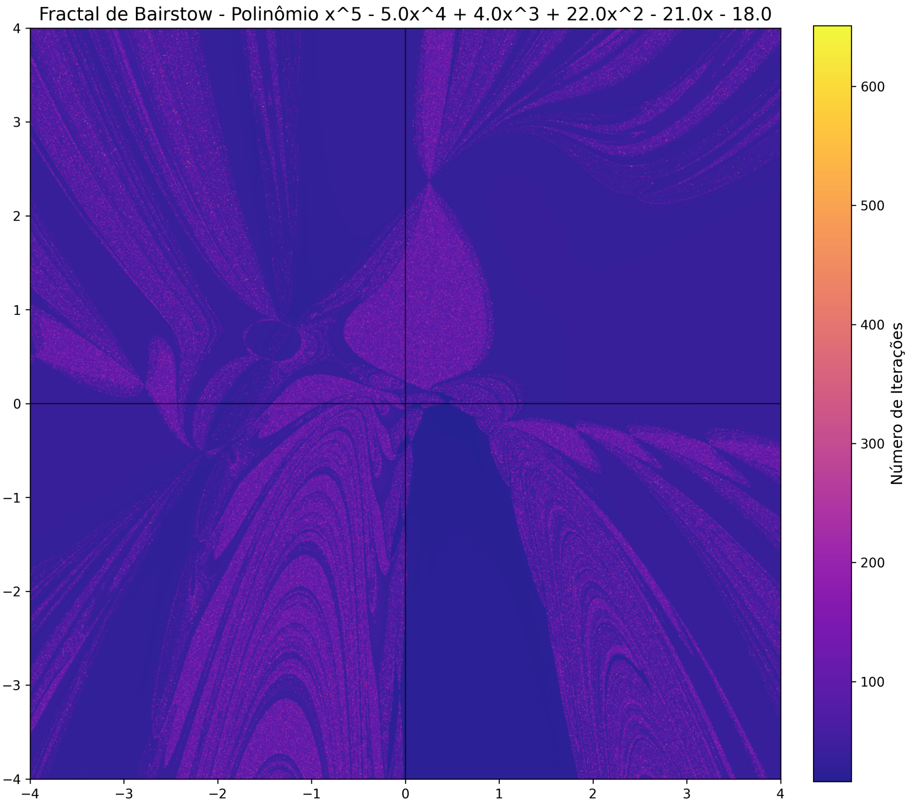

Figura 7: Fractal de iterações para $x^5 - 5x^4 + 4x^3 + 22x^2 - 21x - 18$.

Outros fractais podem ser gerados facilmente com diferentes configurações de polinômios e malhas.

8 Conclusão

O Programa 4 atendeu integralmente aos objetivos propostos, proporcionando:

- Uma implementação robusta e modular do Método de Bairstow;
- Ferramentas de visualização (fractais) que elucidam o comportamento dinâmico do método;
- Capacidade de exportação e análise dos dados de iteração;
- Controle preciso da execução, mesmo em casos difíceis.

O estudo dos fractais gerados revelou a alta complexidade e sensibilidade do Método de Bairstow, tornando esta ferramenta não apenas prática para cálculo de raízes, mas também rica para investigação didática e teórica.