Computer Project: Nonlinear System Solving

Zexi Fan, 2200010816

23rd April 2025

1 Problem Setting

We consider finding the zeros of the following three nonlinear equations (systems).

Extended Powell SIngular Function

- Number of equations: n, where n is a multiple of 4.
- Initial guess:
$$x^{(0)} = (\,3,\, -1,\, 0,\, 1, \ldots,\, 3,\, -1,\, 0,\, 1\,).$$
- True solution:
$$x^* = (0, 0, \ldots, 0).$$
- System definition: for each $i = 1, 2, \ldots, \frac{n}{4}$,
$$\begin{aligned}
f_{4i-3}(x) &= x_{4i-3} + 10\, x_{4i-2}, \\
f_{4i-2}(x) &= \sqrt{5}\,\bigl(x_{4i-1} - x_{4i}\bigr), \\
f_{4i-1}(x) &= \bigl(x_{4i-2} - 2\, x_{4i-1}\bigr)^2, \\
f_{4i}(x) &= \sqrt{10}\,\bigl(x_{4i-3} - x_{4i}\bigr)^2.
\end{aligned}$$

Trigonometric Function

- Number of equations: n.
- Initial guess:
$$x^{(0)} = \left(\tfrac{1}{n}, \tfrac{1}{n}, \ldots, \tfrac{1}{n}\right).$$
- True solution:
$$x^* = (0, 0, \ldots, 0).$$
- System definition: for each $i = 1, 2, \ldots, n$,
$$f_i(x) \;=\; n \;-\; \sum_{j=1}^{n} \Bigl[\cos(x_j) \;+\; i\,(1 - \cos(x_i)) \;-\; \sin(x_i)\Bigr].$$

Wood Function

Rewrite the system as the following:

- Number of equations: $n = 4$.
- Initial guess:
$$x^{(0)} = (-3,\ -1,\ -3,\ -1).$$
- True solution:
$$x^* = (1, 1, 1, 1).$$
- Equation:
$$\begin{cases} f_1(x) = (x_1 - 1)^2 \\ f_2(x) = 100(x_1 - x_2)^2 \\ f_3(x) = 90(x_3 - x_4)^2 + (x_3 - 1)^2 \\ f_4(x) = 10.1((1 - x_2)^2 + (1 - x_4)^2) + 19.8(1 - x_2)(1 - x_4) \end{cases}$$

2 Algorithm

2.1 Basic Principle

The quasi-Newton method constructs a sequence of matrices B_k that approximate the true Jacobian $J(x)$ in order to mimic the Newton update

$$x^{k+1} \;=\; x^k \;-\; J(x^k)^{-1}\, f(x^k).$$

Since computing $J(x^k)$ and its inverse directly is expensive, we replace them by an approximate Jacobian B_k and its inverse B_k^{-1}:

$$x^{k+1} \;=\; x^k \;-\; B_k^{-1}\, f(x^k).$$

We require B_k to satisfy the secant condition

$$f(x^k) - f(x^{k-1}) \;=\; B_k\big(x^k - x^{k-1}\big).$$

Define the step and function increment

$$s_k = x^{k+1} - x^k, \quad y_k = f(x^{k+1}) - f(x^k).$$

Then

$$B_{k+1} s_k = y_k,$$

and denoting $\Delta B_k = B_{k+1} - B_k$, we have

$$\Delta B_k\, s_k = y_k - B_k\, s_k.$$

A rank-one update that enforces this is

$$\Delta B_k = \frac{(y_k - B_k s_k)\,(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k},$$

so that

$$B_{k+1} = B_k \; + \; \frac{(y_k - B_k s_k)\,(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}.$$

The inverse can be updated efficiently via the Sherman–Morrison formula:

$$B_{k+1}^{-1} = B_k^{-1} + \frac{(s_k - B_k^{-1} y_k)\,(s_k - B_k^{-1} y_k)^T}{(s_k - B_k^{-1} y_k)^T y_k}.$$

2.2 Convergence Analysis

2.2.1 Local Convergence

If the initial guess x^0 is sufficiently close to the true solution x^*, and B_0 is a good approximation to $J(x^*)$, then the quasi-Newton iterates converge *superlinearly*.

2.2.2 Global Convergence

To ensure global convergence from arbitrary x^0, we incorporate a line search strategy. We choose a step length α (e.g. by backtracking) to satisfy a Wolfe-type condition such as

$$\|f(x^k + \alpha d^k)\| \;\le\; (1 - c\,\alpha)\,\|f(x^k)\|,$$

where $d^k = -B_k^{-1} f(x^k)$ and $c \in (0, 1)$. This guarantees a sufficient decrease in $\|f\|$ at each iteration.

2.3 Quasi-Newton Algorithm

1. **Initialization:**
   - Choose x^0.
   - Compute $B_0 = J(x^0)$ and set $B_0^{-1} = J(x^0)^{-1}$.
2. **For** $k = 0, 1, \ldots,$ **max_iter** $- 1$**:**
   (a) Evaluate $f(x^k)$.
   (b) Compute search direction:
   $$d^k = -\,B_k^{-1}\,f(x^k).$$
   (c) Perform line search to find α_k satisfying the chosen Wolfe condition.
   (d) Update iterate:
   $$x^{k+1} = x^k + \alpha_k\,d^k.$$
   (e) Set
   $$s_k = x^{k+1} - x^k, \quad y_k = f(x^{k+1}) - f(x^k).$$
   (f) Update B_k by the rank-one formula:
   $$B_{k+1} = B_k + \frac{(y_k - B_k s_k)\,(y_k - B_k s_k)^T}{(y_k - B_k s_k)^T s_k}.$$

(g) Update B_k^{-1} via Sherman–Morrison:

$$B_{k+1}^{-1} = B_k^{-1} + \frac{(s_k - B_k^{-1}y_k)\,(s_k - B_k^{-1}y_k)^T}{(s_k - B_k^{-1}y_k)^T y_k}.$$

2.4 Line Search Method

- Initialize $\alpha = 1$.
- Choose parameters $\rho \in (0,1)$ (e.g. 0.5) and $c \in (0,1)$ (e.g. 10^{-4}).
- **While** $\|f(x + \alpha d)\| > (1 - c\,\alpha)\,\|f(x)\|$:

$$\alpha \leftarrow \rho\,\alpha.$$

- Return the final α.

3 Experiment

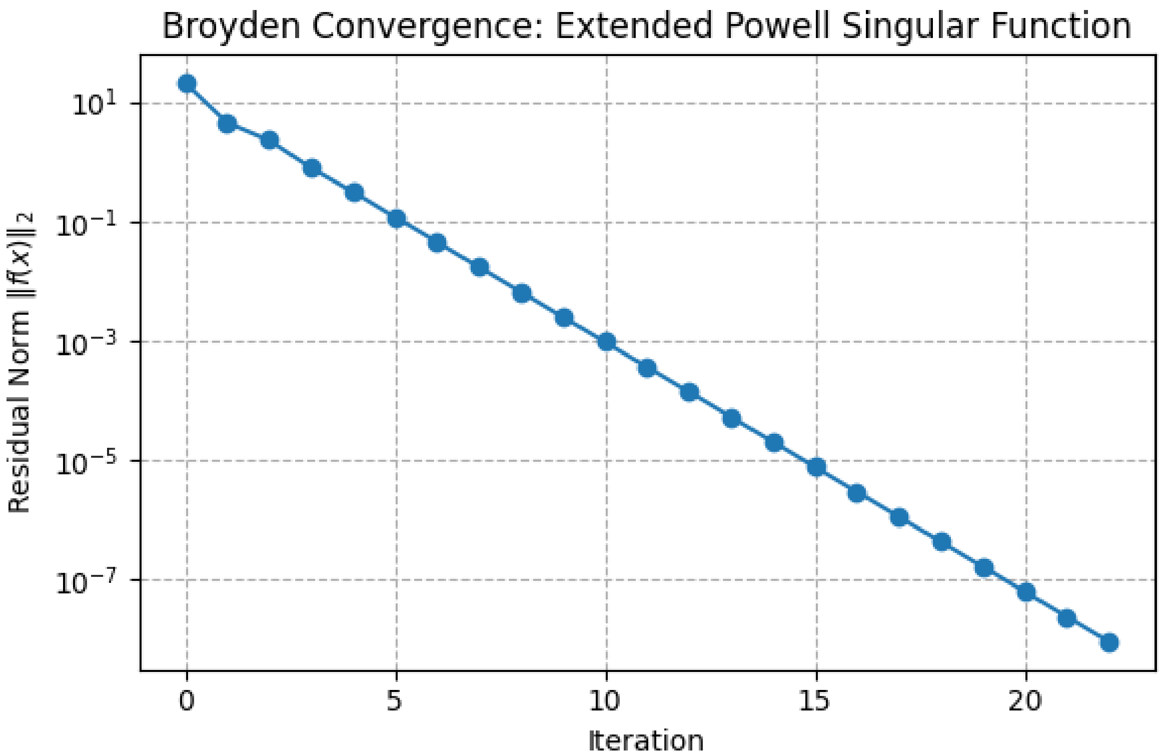

Figure 1: Error Order:PSF

The quasi-Newton method effectively approximates the true solution of PSF, $x^* = (0, 0, \dots, 0)$. As shown in the figure, the error converges rapidly with increasing iterations. Figure 1 illustrates the relationship between the number of iterations and the error on a logarithmic scale.

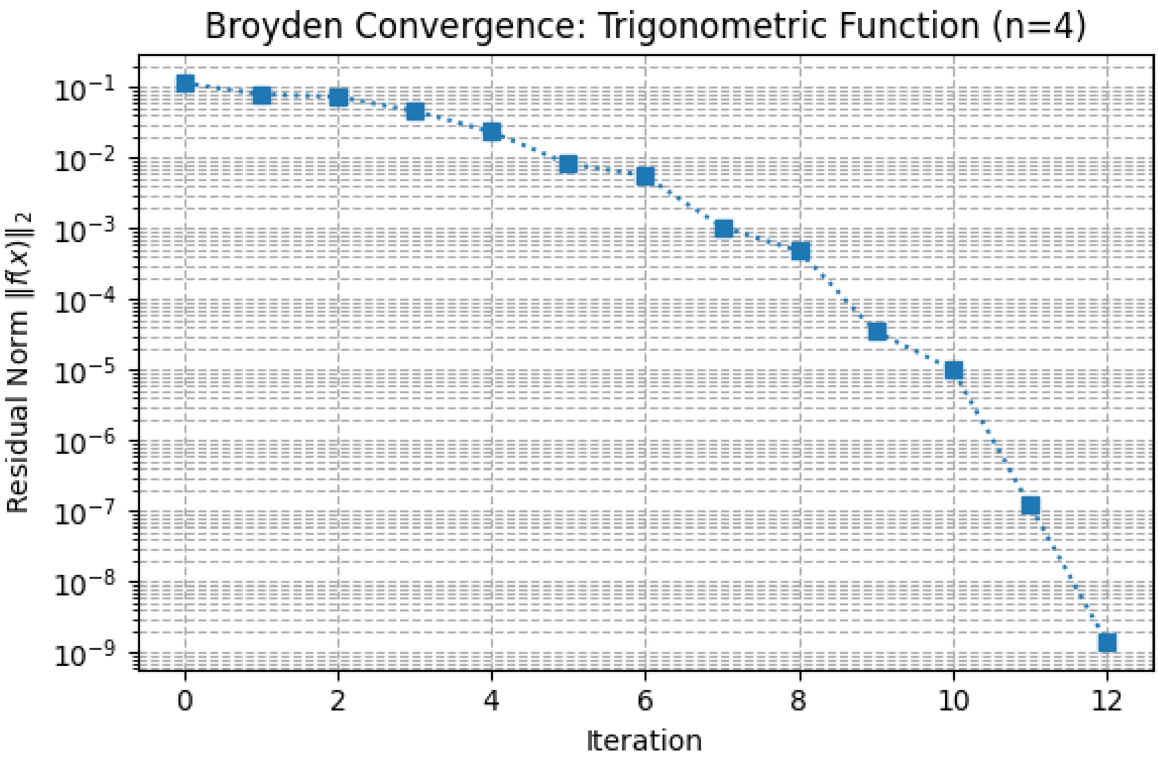

Figure 2: Error Order:TF

The quasi-Newton method also performs well in approximating the true solution of TF, $x^* = (0, 0, \ldots, 0)$. According to the figure, the error decreases rapidly as the number of iterations increases. Figure 2 presents the error versus iteration curve on a logarithmic scale.

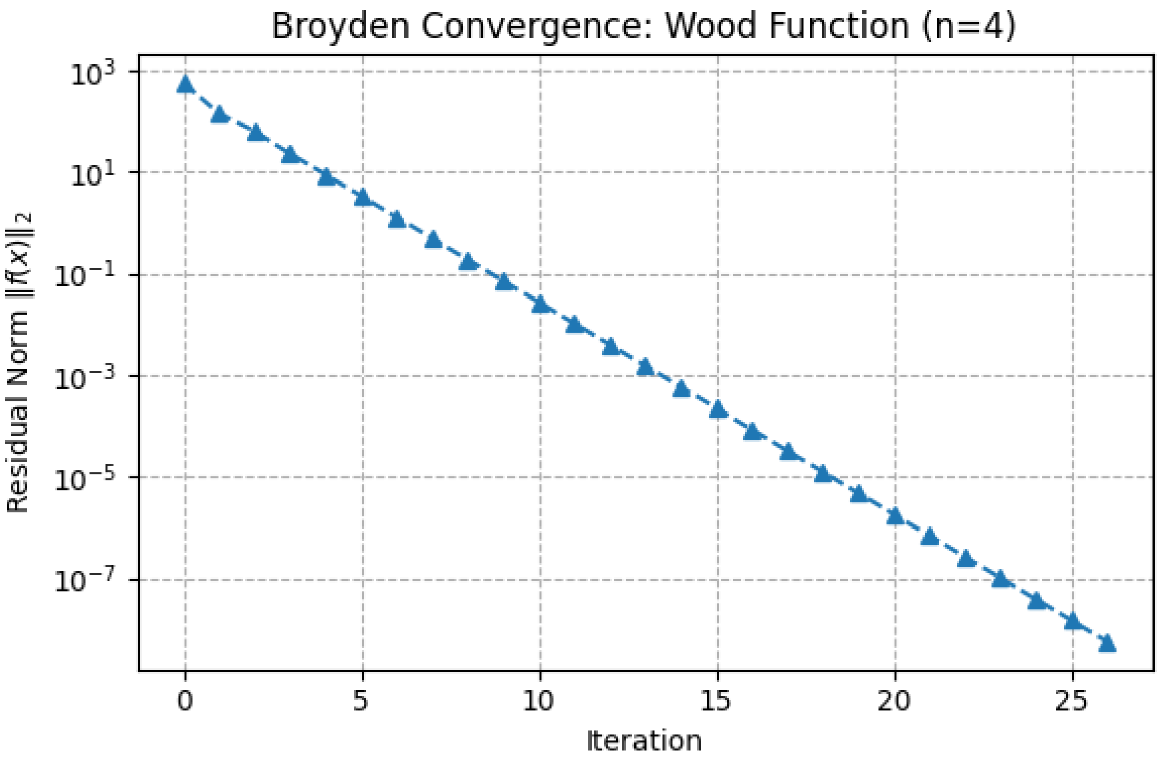

Figure 3: Error Order:WF

The quasi-Newton method effectively approximates the true solution of WF, $x^* = (1, 1, \ldots, 1)$. As shown in Figure 3, the error decreases rapidly with the number of iterations.

References