Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

IMPLEMENTASI ALGORITMA SHERMAN-MORRISON UNTUK PENYELESAIAN SISTEM PERSAMAAN LINIER MENGGUNAKAN MATLAB

Nurul Hidayah[1,*)], Bella Arisha[2)]

[1,2)]Program Studi Pendidikan Matematika, Fakultas Keguruan dan Ilmu Pendidikan, Universitas Jambi
[*]email: nurulhidayah@unja.ac.id

Abstrak:
Artikel ini membahas implementasi algoritma Sherman-Morrison untuk menyelesaikan sistem persamaan linier $Ax = b$, di mana A merupakan matriks nonsingular berukuran n×n sehingga sistem memiliki solusi tunggal. Penelitian ini dilatarbelakangi oleh kebutuhan pendekatan penyelesaian yang adaptif dan efisien bagi sistem linier yang mengalami perubahan lokal secara berulang. Algoritma Sherman-Morrison merupakan bentuk khusus dari rumus Sherman–Morrison–Woodbury yang berlaku untuk modifikasi rank-satu, sehingga memungkinkan pembaruan invers matriks tanpa perhitungan ulang secara menyeluruh. Kajian dilakukan melalui pembuktian teoritis rumus tersebut dan implementasi komputasional menggunakan MATLAB. Implementasi disusun sebagai fungsi khusus yang dikembangkan secara modular untuk mengeksekusi langkah-langkah algoritma. Pengujian dilakukan terhadap empat jenis matriks: acak penuh, simetris positif definit, jarang *(sparse*), dan hampir singular. Evaluasi mencakup akurasi solusi, error residual, serta waktu eksekusi, kemudian dibandingkan dengan metode standar (A\b). Hasil implementasi menunjukkan bahwa algoritma memberikan solusi yang akurat dan efisien pada sistem dengan skala kecil serta matriks sparse, namun cenderung tidak stabil bila diterapkan pada matriks yang hampir singular. Dengan demikian, algoritma ini layak digunakan untuk kasus pembaruan rank-satu dengan kondisi yang baik, serta relevan untuk penerapan dalam konteks pembelajaran dan komputasi numerik.

Kata Kunci: MATLAB; Sherman-Morrison; Sistem Persamaan Linier

Abstract: *This article discusses the implementation of the Sherman-Morrison algorithm for solving the linear system Ax = b, where A is a nonsingular matrix of size n×n, ensuring a unique solution. The study is motivated by the need for adaptive and efficient approaches to linear systems that undergo frequent localized structural changes. The Sherman-Morrison algorithm is a special case of the Sherman-Morrison-Woodbury formula, applicable to rank-one modifications, enabling matrix inverse updates without full recomputation. The study involves a theoretical proof of the formula and a computational implementation using MATLAB. The implementation is structured as a custom*

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

function developed modularly to execute each step of the algorithm. The algorithm is tested on four types of matrices: fully random, symmetric positive definite, sparse, and nearly singular. The evaluation includes solution accuracy, residual error, and execution time, compared to the standard method (A\b). The results show that the algorithm yields accurate and efficient solutions for small-scale systems and sparse matrices, but becomes unstable when applied to nearly singular matrices. Therefore, this algorithm is well-suited for well-conditioned rank-one update cases and is relevant for applications in numerical computation and educational contexts.

Keywords: *MATLAB; Sherman-Morrison; System of Linear Equations*

PENDAHULUAN

Sistem persamaan linier merupakan permasalahan dasar dalam aljabar linier dan banyak digunakan dalam berbagai bidang, seperti teknik, fisika, dan ekonomi. Umumnya sistem ini dinyatakan dalam bentuk matriks sebagai $A\boldsymbol{x} = \boldsymbol{b}$, dengan A sebagai matriks koefisien, $\boldsymbol{x}$ sebagai vektor peubah, dan $\boldsymbol{b}$ sebagai vektor konstanta. Bila jumlah peubah sama dengan jumlah persamaan, maka A berbentuk matriks bujur sangkar berukuran $n \times n$, dan vektor $\boldsymbol{x}$ serta $\boldsymbol{b}$ masing-masing berukuran $n \times 1$. Tujuan utama dari sistem ini adalah mencari solusi dari persamaan tersebut. Apabila A bersifat nonsingular, maka sistem akan memiliki solusi tunggal. Penyelesaian sistem semacam ini dapat dilakukan melalui berbagai metode, antara lain substitusi, eliminasi Gauss, metode campuran, hingga eliminasi Gauss-Jordan berbasis operasi baris elementer (Anton & Rorres, 2014).

Dalam berbagai aplikasi, sistem linier sering kali harus diselesaikan secara berulang dengan sedikit perubahan pada nilai-nilai dalam matriks koefisiennya. Dalam kasus seperti ini, menghitung ulang invers matriks secara keseluruhan menjadi tidak efisien, khususnya untuk sistem berdimensi besar. Oleh karena itu, diperlukan pendekatan yang memungkinkan pembaruan solusi tanpa harus menyelesaikan seluruh sistem dari awal. Salah satu solusi adalah menggunakan algoritma Sherman-Morrison, yang merupakan bentuk khusus dari rumus Sherman-Morrison-Woodbury (SMW) dan berlaku untuk modifikasi rank-satu (Hager, 1989; Sherman & Morrison, 1950). Secara matematis, SMW berbentuk:

$$(A + UCV)^{-1} = A^{-1} - A^{-1}U(C^{-1} + VA^{-1}U)^{-1}VA^{-1} \quad (1)$$

dengan $A \in M_{n\times n}(\mathbb{R})$ nonsingular, $U \in M_{n\times m}(\mathbb{R})$, $C \in M_{m\times m}(\mathbb{R})$ nonsingular, dan $V \in M_{m\times n}(\mathbb{R})$.

Kontribusi awal dari Hager memperlihatkan efisiensi pembaruan invers dengan SMW, yang kemudian dikembangkan lebih lanjut oleh Egidi & Maponi (2006) melalui


Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

implementasi dalam pemrograman FORTRAN dan disempurnakan oleh Maponi (2007). Rumus ini memungkinkan invers diperbarui secara efisien dan telah diterapkan dalam berbagai bidang seperti dinamika struktur (Song et al., 2015), sistem stokastik (Y. Wu et al., 2020), sistem fasa (Smejkal & Mikyška, 2021), operator ruang Hilbert (Arias et al., 2015), sistem Kalman ensembel (Nino Ruiz et al., 2015), sistem integral-diferensial linier (F. Wu, 2016), dan seleksi satelit adaptif pada GNSS berbasis kriteria GDOP (Shi et al., 2023). Selain itu, invers Moore–Penrose dapat direduksi menjadi bentuk SMW (Xu, 2017), invers khusus dari matriks kompleks (Stanimirović et al., 2016), dan bentuk khusus invers seperti Drazin dan *core inverse* (Li et al., 2022). Implementasi praktis SMW terbukti efektif dalam teknik simulasi, seperti simulasi elastisitas 2D (Chai et al., 2021), sistem Toeplitz (Aoulad & Tajani, 2023), serta aplikasi elektromagnetik sudut lebar (X. Chen et al., 2023). Pendekatan ini juga menunjukkan efisiensi tinggi pada sistem berdimensi menengah dibanding metode klasik Kronecker (J. Chen et al., 2021), serta dalam simulasi adaptif elastisitas 2D yang melibatkan perubahan dimensi sistem (Hao & Simoncini, 2021).

Namun, sejauh ini masih minim kajian yang secara eksplisit mengimplementasikan algoritma Sherman-Morrison dalam konteks pengembangan algoritma numerik secara modular berbasis MATLAB, khususnya untuk penyelesaian sistem persamaan linier. Hal ini penting karena sebagian besar pembelajaran dan studi literatur hanya berfokus pada teori atau implementasi menggunakan bahasa pemrograman seperti FORTRAN, sedangkan MATLAB menawarkan keunggulan dalam komputasi matriks, visualisasi, dan kemudahan modularisasi algoritma (Cahyono, 2016; Irwan, 2017; Keviczky et al., 2019; Sahid, 2005; Tomozei et al., 2023; Zulhendri, 2016).

Berdasarkan latar belakang tersebut, artikel ini bertujuan untuk mengimplementasikan algoritma Sherman-Morrison dalam menyelesaikan sistem persamaan linier, menganalisis efisiensinya secara komputasional, dan mengevaluasi akurasi hasilnya melalui simulasi berbasis MATLAB.

METODE PENELITIAN

Penelitian ini menggunakan pendekatan analitik dan komputasional. Secara analitik, penulis melakukan kajian literatur untuk memahami dasar matematis algoritma Sherman-Morrison, khususnya dalam konteks pembaruan invers matriks akibat perubahan rank-satu. Literatur utama yang menjadi rujukan mencakup hasil-hasil dari Sherman & Morrison (1950), Hager (1989), serta pengembangan dan implementasi berikutnya oleh Egidi & Maponi (2006) dan Maponi (2007).


Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

Implementasi penelitian ini dilakukan menggunakan MATLAB versi R2013a pada perangkat laptop Dell Latitude E7440 dengan prosesor prosesor Intel(R) Core(TM) i5-4310U. Proses pengembangan algoritma dimulai dengan merancang algoritma Sherman-Morrison dalam bentuk fungsi tersendiri (*custom function*), yang menerima masukan berupa matriks dan vektor, serta mengembalikan vektor solusi. Implementasi ini tidak menggunakan fungsi bawaan MATLAB untuk penyelesaian langsung, tetapi seluruh proses dikonstruksi secara modular untuk mengeksekusi pembaruan invers secara bertahap.

1. Logika dan Struktur Algoritma

   Berikut adalah algoritma Sherman-Morrison dalam bentuk pseudocode yang menyajikan proses penyelesaian sistem $A\boldsymbol{x} = \boldsymbol{b}$:

   Input : Matriks $A \in M_{n\times n}(\mathbb{R})$ dan vektor $\boldsymbol{b} \in \mathbb{R}^n$

   Output : Solusi x dari sistem $A\boldsymbol{x} = \boldsymbol{b}$

   1) Identifikasi matriks koefisien $A \in M_{n\times n}(\mathbb{R})$;
   2) Identifikasi vektor kolom $\boldsymbol{b} \in \mathbb{R}^n$;
   3) Definisikan $A_0 \leftarrow diag(A)$;
   4) Hitung matriks $C \leftarrow A - A_0$;
   5) Untuk $i = 1$ sampai n lakukan:
      $\boldsymbol{u}_i \leftarrow$ kolom ke-i dari C
      $\boldsymbol{v}_i \leftarrow$ basis standar $\boldsymbol{e}_i \in \mathbb{R}^n$ (vektor dengan 1 di posisi ke-i, lainnya 0)
   6) Hitung solusi awal: $\boldsymbol{x}_0 \leftarrow {A_0}^{-1} * \boldsymbol{b}$;
   7) Jika $n > 0$ maka:
      Untuk $i = 1$sampai n lakukan: $\boldsymbol{y}_{0,i} \leftarrow {A_0}^{-1} * \boldsymbol{u}_i$;
   8) Untuk $s = 1$ sampai $n - 1$ lakukan:
      $\boldsymbol{x}_s \leftarrow \boldsymbol{x}_{s-1} - [\boldsymbol{y}_{s-1,s} * ({\boldsymbol{v}_s}^T * \boldsymbol{x}_{s-1})\,] / [\,1 + {\boldsymbol{v}_s}^T * \boldsymbol{y}_{s-1,s}]$;
      Untuk $i = s + 1$ sampai n lakukan:
      $\boldsymbol{y}_{s,i} \leftarrow \boldsymbol{y}_{s-1,i} - [\,\boldsymbol{y}_{s-1,s} * ({\boldsymbol{v}_s}^T * \boldsymbol{y}_{s-1,i})\,] / [\,1 + {\boldsymbol{v}_s}^T * \boldsymbol{y}_{s-1,s}]$;
   9) Hitung solusi akhir:
      $\boldsymbol{x}_n \leftarrow \boldsymbol{x}_{n-1} - [\boldsymbol{y}_{n-1,n} * ({\boldsymbol{v}_n}^T * \boldsymbol{x}_{n-1,n})\,] / [\,1 + {\boldsymbol{v}_n}^T * \boldsymbol{y}_{n-1,n}\,]$;
   10) Keluarkan hasil: $x \leftarrow x_{\mathrm{n}}$.

   Struktur kode lengkap disajikan dalam bagian hasil, dan dirancang agar mudah diuji serta dikembangkan.

2. Variasi Struktur Pengujian

   Penelitian ini menguji algoritma Sherman-Morrison pada empat jenis sistem linier sebagai berikut:

Tabel 1. Variasi Struktur Pengujian

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

No	Ukuran Matriks	Jenis Matriks	Tujuan Uji
1	4×4	Acak penuh	Uji awal dan verifikasi
2	10×10	Simetris positif definit	Uji kestabilan numerik
3	100×100	Jarang (*Sparse*)	Uji efisiendi pada struktur sparsitas
4	100×100	Hampir singular	Uji ketahanan terhadap kondisi mendekati singular

3. Desain Pengujian dan Validasi Hasil

Pengujian dilakukan untuk menilai akurasi numerik dan efisiensi waktu eksekusi algoritma. Validasi hasil dilakukan dengan cara sebagai berikut:

1) Membandingkan solusi dari algoritma dengan hasil penyelesaian standar MATLAB (A\b);
2) Mengukur selisih terhadap solusi menggunakan $\|x_{SM} - x_{MATLAB}\|$;
3) Menghitung error residual $\|A\boldsymbol{x} - \boldsymbol{b}\|$ dengan toleransi $\varepsilon < 10^{-6}$;
4) Mengukur waktu eksekusi algoritma sebagai indikator efisiensi komputasi.

Replikasi uji dilakukan sebanyak 10 kali percobaan untuk setiap variasi dalam meminimalkan pengaruh fluktuasi performa sistem dan menghasilkan rata-rata waktu yang lebih baik.

HASIL DAN PEMBAHASAN

Hasil

Dalam kasus khusus ketika $A \in M_{n\times n}(\mathbb{R})$ adalah matriks nonsingular, $\boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$, dan $C = [1]$, maka persamaan (1) menjadi bentuk yang dikenal sebagai rumus Sherman-Morrison. Hasil ini dirumuskan sebagai berikut.

Teorema 1. (Egidi & Maponi, 2006) Diberikan $A \in M_{n\times n}(\mathbb{R}), \boldsymbol{u}, \boldsymbol{v} \in \mathbb{R}^n$ sedemikian sehingga $\det(A) \neq 0$ dan $\det(A + \boldsymbol{u}\boldsymbol{v}^T) \neq 0$ maka

$$(A + \boldsymbol{u}\boldsymbol{v}^T)^{-1} = A^{-1} - \frac{A^{-1}\boldsymbol{u}\boldsymbol{v}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} \tag{2}$$

Bukti. Diketahui $A \in M_{n\times n}(\mathbb{R})$ dan $\det(A) \neq 0$, artinya A^{-1} ada. Kemudian diketahui $det(A + \boldsymbol{u}\boldsymbol{v}^T) \neq 0$, artinya $(A + \boldsymbol{u}\boldsymbol{v}^T)^{-1}$ ada. Misalkan $X = A + \boldsymbol{u}\boldsymbol{v}^T$ dan $Y = (A + \boldsymbol{u}\boldsymbol{v}^T)^{-1}$. Jika X invers dari Y dan Y invers dari X maka $XY = I$ dan $YX = I$. Perhatikan bahwa:

$$\begin{aligned} XY &= (A + \boldsymbol{u}\boldsymbol{v}^T)(A + \boldsymbol{u}\boldsymbol{v}^T)^{-1} \\ &= (A + \boldsymbol{u}\boldsymbol{v}^T)\left(A^{-1} - \frac{A^{-1}\boldsymbol{u}\boldsymbol{v}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right) \end{aligned}$$

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

$$= AA^{-1} + \boldsymbol{uv}^T A^{-1} - \frac{AA^{-1}\boldsymbol{uv}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} - \frac{\boldsymbol{uv}^T A^{-1}\boldsymbol{uv}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}$$
$$= I + \boldsymbol{uv}^T A^{-1} - \frac{\boldsymbol{uv}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} - \frac{\boldsymbol{uv}^T A^{-1}\boldsymbol{uv}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}$$
$$= I + \boldsymbol{uv}^T A^{-1} - \left(\frac{\boldsymbol{u}(\boldsymbol{v}^T A^{-1} + \boldsymbol{v}^T A^{-1}\boldsymbol{uv}^T A^{-1})}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right)$$
$$= I + \boldsymbol{uv}^T A^{-1} - \left(\frac{\boldsymbol{u}(1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u})\boldsymbol{v}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right)$$
$$= I + \boldsymbol{uv}^T A^{-1} - \boldsymbol{uv}^T A^{-1} = I$$

Selanjutnya, perhatikan bahwa:

$$YX = (A + \boldsymbol{uv}^T)^{-1}(A + \boldsymbol{uv}^T)$$
$$= \left(A^{-1} - \frac{A^{-1}\boldsymbol{uv}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right)(A + \boldsymbol{uv}^T)$$
$$= A^{-1}A + A^{-1}\boldsymbol{uv}^T - \frac{A^{-1}\boldsymbol{uv}^T A^{-1}A}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} - \frac{A^{-1}\boldsymbol{uv}^T A^{-1}\boldsymbol{uv}^T}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}$$
$$= I + A^{-1}\boldsymbol{uv}^T - \frac{A^{-1}\boldsymbol{uv}^T}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} - \frac{A^{-1}\boldsymbol{uv}^T A^{-1}\boldsymbol{uv}^T}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}$$
$$= I + A^{-1}\boldsymbol{uv}^T - \left(\frac{A^{-1}\boldsymbol{u}(\boldsymbol{v}^T + \boldsymbol{v}^T A^{-1}\boldsymbol{uv}^T)}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right)$$
$$= I + A^{-1}\boldsymbol{uv}^T - \left(\frac{A^{-1}\boldsymbol{u}(1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u})\boldsymbol{v}^T}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right)$$
$$= I + A^{-1}\boldsymbol{uv}^T - A^{-1}\boldsymbol{uv}^T = I$$

Oleh karena $XY = I$ dan $YX = I$, maka persamaan (2) terbukti.

Akibat 1. (Egidi & Maponi, 2006) Diberikan $A \in M_{n\times n}(\mathbb{R}), \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{b} \in \mathbb{R}^n$. Misalkan $\det(A) \neq 0$ dan $\det(A + \boldsymbol{uv}^T) \neq 0$, $\boldsymbol{x} = \boldsymbol{x_n} \in \mathbb{R}^n$ adalah solusi dari $A\boldsymbol{x} = \boldsymbol{b}$, dan $\boldsymbol{y} = \boldsymbol{y_n} \in \mathbb{R}^n$ adalah solusi dari $A\boldsymbol{y} = \boldsymbol{u}$. Maka solusi dari

$$(A + \boldsymbol{uv}^T)\boldsymbol{x} = \boldsymbol{b} \tag{3}$$

diberikan oleh

$$\boldsymbol{x} = \boldsymbol{x_n} - \frac{\boldsymbol{y_n}\boldsymbol{v}^T\boldsymbol{x_n}}{1 + \boldsymbol{v}^T\boldsymbol{y_n}} \tag{4}$$

Bukti. Diketahui $A \in M_{n\times n}(\mathbb{R}), \boldsymbol{u}, \boldsymbol{v}, \boldsymbol{b} \in \mathbb{R}^n$. Kemudian, diketahui $\det(A) \neq 0$ yang berarti A^{-1} ada dan $det(A + \boldsymbol{uv}^T) \neq 0$ yang berarti $(A + \boldsymbol{uv}^T)^{-1}$ ada. Misalkan $\boldsymbol{x} = \boldsymbol{x_n} \in \mathbb{R}^n$ adalah solusi dari $A\boldsymbol{x} = \boldsymbol{b}$ dan $\boldsymbol{y} = \boldsymbol{y_n} \in \mathbb{R}^n$ adalah solusi dari $A\boldsymbol{y} = \boldsymbol{u}$. Akan dibuktikan bahwa persamaan (4) adalah solusi dari (3). Perhatikan bahwa:

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

$$A\boldsymbol{x} = \boldsymbol{b} \Rightarrow \boldsymbol{x} = A^{-1}\boldsymbol{b} \text{ atau } \boldsymbol{x_n} = A^{-1}\boldsymbol{b}$$

dan

$$A\boldsymbol{y} = \boldsymbol{u} \Rightarrow \boldsymbol{y} = A^{-1}\boldsymbol{u} \text{ atau } \boldsymbol{y_n} = A^{-1}\boldsymbol{u}.$$

Sehingga,

$$(A + \boldsymbol{uv}^T)\boldsymbol{x} = \boldsymbol{b} \Rightarrow \boldsymbol{x} = (A + \boldsymbol{uv}^T)^{-1}\boldsymbol{b}.$$

Perhatikan bahwa:

$$\begin{aligned} \boldsymbol{x} &= (A + \boldsymbol{uv}^T)^{-1}\boldsymbol{b} \\ &= \left(A^{-1} - \frac{A^{-1}\boldsymbol{uv}^T A^{-1}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}}\right)\boldsymbol{b} \\ &= A^{-1}\boldsymbol{b} - \frac{A^{-1}\boldsymbol{uv}^T A^{-1}\boldsymbol{b}}{1 + \boldsymbol{v}^T A^{-1}\boldsymbol{u}} \end{aligned} \quad (5)$$

Substitusi $\boldsymbol{x_n} = A^{-1}\boldsymbol{b}$ dan $\boldsymbol{y_n} = A^{-1}\boldsymbol{u}$ ke persamaan (5) sehingga menjadi

$$\boldsymbol{x} = \boldsymbol{x_n} - \frac{\boldsymbol{y_n}\boldsymbol{v}^T\boldsymbol{x_n}}{1 + \boldsymbol{v}^T\boldsymbol{y_n}}.$$

Matriks koefisien A direpresentasikan sebagai penjumlahan matriks diagonal A_0 dan sejumlah perturbasi rank-satu:

$$A = A_0 + \boldsymbol{u}_1\boldsymbol{v}_1{}^{\boldsymbol{T}} + \boldsymbol{u}_2\boldsymbol{v}_2{}^{\boldsymbol{T}} + \cdots + \boldsymbol{u}_n\boldsymbol{v}_n{}^{\boldsymbol{T}}$$

dengan

$$A_0 = \begin{pmatrix} a_{11} & 0 & 0 & \cdots & 0 \\ 0 & a_{22} & 0 & \cdots & 0 \\ 0 & 0 & a_{33} & \vdots & 0 \\ \vdots & \vdots & \cdots & \ddots & \vdots \\ 0 & 0 & 0 & \cdots & a_{nn} \end{pmatrix}, \boldsymbol{u}_1\boldsymbol{v}_1{}^{\boldsymbol{T}} = \begin{pmatrix} 0 \\ a_{21} \\ a_{31} \\ \vdots \\ a_{n1} \end{pmatrix}\begin{pmatrix} 1 \\ 0 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T_{n\times 1}, \boldsymbol{u}_2\boldsymbol{v}_2{}^{\boldsymbol{T}} =$$

$$\begin{pmatrix} a_{12} \\ 0 \\ a_{32} \\ \vdots \\ a_{n2} \end{pmatrix}\begin{pmatrix} 0 \\ 1 \\ 0 \\ \vdots \\ 0 \end{pmatrix}^T_{n\times 1}, \text{ dan } \boldsymbol{u}_n\boldsymbol{v}_n{}^{\boldsymbol{T}} = \begin{pmatrix} a_{1n} \\ a_{2n} \\ a_{3n} \\ \vdots \\ 0 \end{pmatrix}\begin{pmatrix} 0 \\ 0 \\ 0 \\ \vdots \\ 1 \end{pmatrix}^T_{n\times 1} \text{ atau}$$

$$A = A_0 + \sum_{i=1}^{n} \boldsymbol{u}_i\boldsymbol{v}_i{}^{\boldsymbol{T}} \quad (6)$$

Representasi (6) diperluas menjadi bentuk rekursif sebagai berikut:

$$A_s = A_{s-1} + \boldsymbol{u}_s\boldsymbol{v}_s{}^{\boldsymbol{T}}, \text{ untuk } s = 1,2,\ldots,n \quad (7)$$

dengan A_s adalah matriks nonsingular.

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

Misalkan diberikan sistem persamaan linier $A_s\boldsymbol{x} = \boldsymbol{b}$ dan $A_s\boldsymbol{y} = \boldsymbol{u_i}$ untuk $s = 1,2,..,n$ dengan iterasi $i = s+1, s+2, \dots, n$. Maka $\boldsymbol{x} = \boldsymbol{x_s} \in \mathbb{R}^n$ adalah solusi dari $A_s\boldsymbol{x} = \boldsymbol{b}$ dan $\boldsymbol{y} = \boldsymbol{y_{s,i}} \in \mathbb{R}^n$ adalah solusi dari $A_s\boldsymbol{y} = \boldsymbol{u_i}$ dimana $i = s+1, s+2, \dots, n$. Jika $s = n$ dengan n adalah ukuran matriks koefisien bujur sangkar A maka $\boldsymbol{x_n}$ merupakan solusi dari $A\boldsymbol{x} = \boldsymbol{b}$, sehingga diperoleh

$$\begin{aligned}
\boldsymbol{x}_s &= {A_s}^{-1}\boldsymbol{b} = (A_{s-1} + \boldsymbol{u}_s\boldsymbol{v}_s)^{-1}\boldsymbol{b} \\
&= \left({A_{s-1}}^{-1} - \frac{{A_{s-1}}^{-1}\boldsymbol{u}_s\boldsymbol{v}_s{}^T{A_{s-1}}^{-1}}{1 + \boldsymbol{v}_s{}^T{A_{s-1}}^{-1}\boldsymbol{u}_s}\right)\boldsymbol{b} \\
&= {A_{s-1}}^{-1}\boldsymbol{b} - \frac{{A_{s-1}}^{-1}\boldsymbol{u}_s\boldsymbol{v}_s{}^T{A_{s-1}}^{-1}\boldsymbol{b}}{1 + \boldsymbol{v}_s{}^T{A_{s-1}}^{-1}\boldsymbol{u}_s} \\
&= \boldsymbol{x}_{s-1} - \frac{\boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\boldsymbol{x}_{s-1}}{1 + \boldsymbol{v}_s{}^T\boldsymbol{y}_{s-1,s}}
\end{aligned} \tag{8}$$

dan

$$\begin{aligned}
\boldsymbol{y}_{s,i} &= {A_s}^{-1}\boldsymbol{u}_i = (A_{s-1} + \boldsymbol{u}_s\boldsymbol{v}_s)^{-1}\boldsymbol{u}_i \\
&= \left({A_{s-1}}^{-1} - \frac{{A_{s-1}}^{-1}\boldsymbol{u}_s\boldsymbol{v}_s{}^T{A_{s-1}}^{-1}}{1 + \boldsymbol{v}_s{}^T{A_{s-1}}^{-1}\boldsymbol{u}_s}\right)\boldsymbol{u}_i \\
&= {A_{s-1}}^{-1}\boldsymbol{u}_i - \frac{{A_{s-1}}^{-1}\boldsymbol{u}_s\boldsymbol{v}_s{}^T{A_{s-1}}^{-1}\boldsymbol{u}_i}{1 + \boldsymbol{v}_s{}^T{A_{s-1}}^{-1}\boldsymbol{u}_s} \\
&= \boldsymbol{y}_{s-1,i} - \frac{\boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\boldsymbol{y}_{s-1,i}}{1 + \boldsymbol{v}_s{}^T\boldsymbol{y}_{s-1,s}}
\end{aligned} \tag{9}$$

Berdasarkan persamaan (8) dan (9) maka diperoleh algoritma Sherman-Morrison untuk menyelesaikan sistem persamaan linier $A\boldsymbol{x} = \boldsymbol{b}$.

Diberikan matriks nonsingular $A, A_0 \in M_{n\times n}(\mathbb{R}), \boldsymbol{u}_i, \boldsymbol{v}_i \in \mathbb{R}^n$ untuk $i = 1,2,\dots,n$, sehingga persamaan (6) berlaku, maka dapat dihitung solusi $\boldsymbol{x} = \boldsymbol{x}_n \in \mathbb{R}^n$ dari sistem $A\boldsymbol{x} = \boldsymbol{b}$ dengan mengikuti langkah-langkah berikut:

(1) mengidentifikasi matriks koefisien $A_{n\times n}$;
(2) mengidentifikasi vektor kolom $\boldsymbol{b}$;
(3) mendefinisikan matriks A_0 yaitu matriks diagonal dari A yang dapat dibalik;
(4) menghitung matriks $A - A_0$;
(5) menentukan vektor $\boldsymbol{u}_i$ yang merupakan entri pada kolom ke-i dari matriks $A - A_0$ untuk $i = 1,2,\dots,n$;
(6) menentukan vektor $\boldsymbol{v}_i$ yang merupakan elemen ke-i dari basis standar $\mathbb{R}^n$ untuk $i = 1,2,\dots,n$;
(7) menghitung $\boldsymbol{x} = \boldsymbol{x}_0$ sebagai solusi dari $A_0\boldsymbol{x} = \boldsymbol{b}$;


Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

(8) jika $n > 0$ maka untuk iterasi $i = 1,2,\dots,n$ dapat dihitung $\boldsymbol{y} = \boldsymbol{y}_{0,i}$ sebagai solusi dari $A_0\boldsymbol{y} = \boldsymbol{u}_i$;

(9) selanjutnya melakukan perhitungan

$$\boldsymbol{x}_s = \boldsymbol{x}_{s-1} - \frac{\boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\boldsymbol{x}_{s-1}}{1 + \boldsymbol{v}_s{}^T\boldsymbol{y}_{s-1,s}}$$

untuk iterasi $s = 1,2,\dots,n-1$ sebagai solusi dari $A_s\boldsymbol{x} = \boldsymbol{b}$, dan

$$\boldsymbol{y}_{s,i} = \boldsymbol{y}_{s-1,i} - \frac{\boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\boldsymbol{y}_{s-1,i}}{1 + \boldsymbol{v}_s{}^T\boldsymbol{y}_{s-1,s}}$$

untuk iterasi $i = s+1, s+2, \dots, n$ sebagai solusi dari $A_s\boldsymbol{y} = \boldsymbol{u}_i$;

(10) menghitung

$$\boldsymbol{x}_n = \boldsymbol{x}_{n-1} - \frac{\boldsymbol{y}_{n-1,n}\boldsymbol{v}_n{}^T\boldsymbol{x}_{n-1,n}}{1 + \boldsymbol{v}_n{}^T\boldsymbol{y}_{n-1,n}}$$

sebagai solusi dari sistem $A\boldsymbol{x} = \boldsymbol{b}$.

Setelah diperoleh algoritma tersebut, langkah selanjutnya adalah memvalidasi keabsahan dan keakuratan algoritma secara matematis untuk memastikan bahwa solusi yang dihasilkan benar-benar menyelesaikan sistem persamaan linier yang diberikan.

Teorema 2. (Maponi, 2007) Diberikan sistem persamaan linier $A\boldsymbol{x} = \boldsymbol{b}$. Misalkan $A \in M_{n\times n}(\mathbb{R})$ adalah matriks yang entri pada diagonal utamanya tidak sama dengan nol. Misalkan $A_s \in M_{n\times n}(\mathbb{R})$ didefinisikan sebagai

$$A_s = A_{s-1} + \boldsymbol{u}_s\boldsymbol{v}_s{}^T \tag{10}$$

untuk $s = 1,2,\dots,n$ dengan $A_0 \in M_{n\times n}(\mathbb{R})$ merupakan matriks diagonal dari A yang dapat dibalik, vektor $\boldsymbol{u}_s$ merupakan entri pada kolom ke-s dari matriks $A - A_0$, dan $\boldsymbol{v}_s = \boldsymbol{e}_s$ merupakan elemen ke-s dari basis standar $\mathbb{R}^n$. Jika A_s nonsingular maka vektor $\boldsymbol{x} = \boldsymbol{x}_n \in \mathbb{R}^n$ dikomputasikan dengan algoritma Sherman-Morrison adalah solusi dari $A\boldsymbol{x} = \boldsymbol{b}$

Bukti. Diketahui A_s adalah matriks nonsingular, artinya $A_s{}^{-1}$ ada. Perhatikan bahwa:

$$\begin{aligned}\det(A_s) &= \det(A_{s-1} + \boldsymbol{u}_s\boldsymbol{v}_s{}^T)\\ &= \det(A_{s-1})\det(1 + A_{s-1}{}^{-1}\boldsymbol{u}_s\boldsymbol{v}_s{}^T)\\ &= \det(A_{s-1})\det\left(1 + \boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\right)\\ &= \det(A_{s-1})\left(1 + \boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\right)\end{aligned}$$

Oleh karena $A_s{}^{-1}$ ada, maka $\det(A_s) \neq 0$, sehingga diperoleh $\left(1 + \boldsymbol{y}_{s-1,s}\boldsymbol{v}_s{}^T\right) \neq 0$. Dengan demikian, algoritma tersebut dapat berlaku.

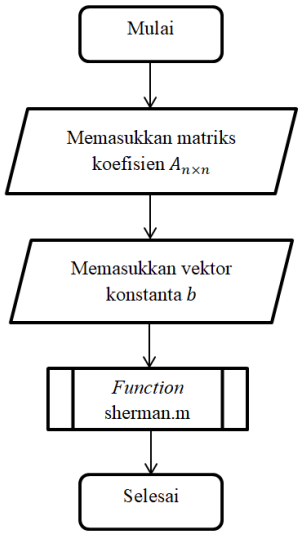

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

Implementasi algoritma Sherman-Morrison dilakukan pada empat jenis sistem persamaan linier yang berbeda, yaitu matriks acak penuh berukuran 4×4, matriks simetris positif definit berukuran 10×10, matriks jarang (*sparse*) berukuran 100×100, dan matriks hampir singular berukuran 100×100. Pengujian dilakukan dengan membandingkan hasil algoritma Sherman-Morrison dengan metode penyelesaian langsung MATLAB $A\backslash b$. Parameter yang diuji meliputi error residual, selisih solusi antara kedua metode, serta waktu eksekusi. Hasil tersebut disajikan dalam bentuk tabel berikut.

Tabel 2. Hasil Pengujian Sherman-Morrison dan $A\backslash b$

Ukuran Matriks	Jenis Matriks	Error Residual		Selisih Solusi	Waktu Eksekusi	
		SM	$A\backslash b$		SM	$A\backslash b$
4×4	Acak penuh	5.75e-15	1.21e-15	1.99e-14	0.00039	0.00006
10×10	Simetris positif definit	4.67e-16	3.54e-16	2.54e-17	0.00098	0.00008
100×100	Jarang (*Sparse*)	9.09e-13	2.73e-15	1.26e-12	0.05244	0.06330
100×100	Hampir singular	3.91e-03	5.12e-05	2.38e+06	0.07380	0.00033

Untuk memberikan pemahaman visual terhadap alur proses algoritma dan tahapan evaluasi yang dilakukan, disertakan dua diagram alir berikut.

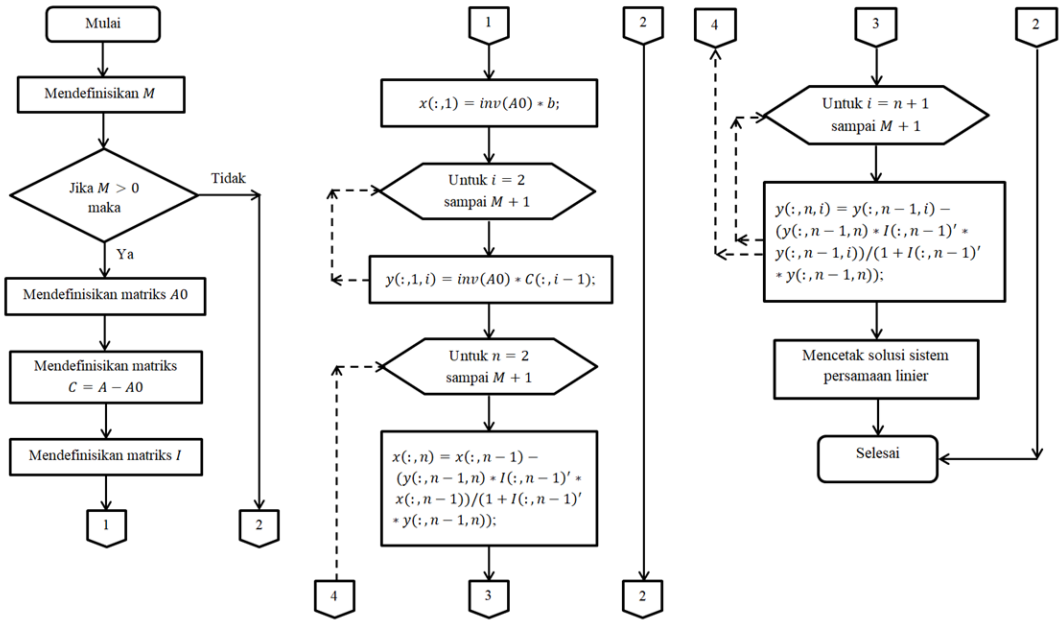


Gambar 1. Diagram alir algoritma Sherman-Morrison untuk penyelesaian SPL

Gambar 1 menunjukkan alur utama program yang dimulai dari proses memasukkan matriks koefisien berukuran $n \times n$ dan vektor konstanta b, kemudian memanggil

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

fungsi sherman.m untuk menyelesaikan sistem persamaan linier, dan diakhiri dengan menampilkan hasil. Sementara Gambar 2 berikut merinci langkah-langkah dalam fungsi sherman.m.



Gambar 2. Diagram alir struktur fungsi sherman.m dalam implementasi MATLAB

Proses diawali dengan mendefinisikan ukuran matriks atau M dan memeriksa apakah $M > 0$. Jika ya, maka matriks diagonal A_0, matriks selisih C , dan matriks identitas I disiapkan. Selanjutnya, solusi awal dihitung dan dilanjutkan dengan proses iteratif yang terdiri dari dua perulangan, yaitu *loop* pertama memperbarui vektor solusi x, dan *loop* kedua memperbarui vektor bantuan y. Iterasi ini mengikuti formula Sherman-Morrison yang bersifat rekursif, hingga seluruh elemen solusi diperoleh. Diagram ini tidak hanya menggambarkan alur logika, tetapi juga memperjelas struktur keputusan dan ketergantungan antar iterasi dalam algoritma.

Untuk memperjelas proses implementasi dan struktur algoritma yang digunakan, berikut ini disertakan kode program fungsi sherman.m yang digunakan dalam pengujian:

```
function sol = shermman(A, b)
    A = full(A);                    % Konversi ke matriks penuh jika
                                      input sparse
    A0 = diag(diag(A));             % Matriks diagonal
    M = size(A,1);                  % Ukuran
    C = A - A0;                     % Komponen non-diagonal
```

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

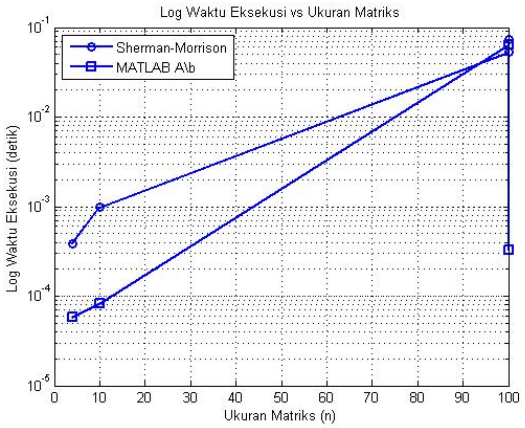
```
    I = eye(M);                    % Matriks identitas

    % Periksa apakah A0 tidak singular
    if any(diag(A0) == 0)
        error('A0 memiliki elemen diagonal nol. Tidak dapat membalik
A0.');
    end
    x = zeros(M, M+1);
    y = zeros(M, M+1, M+1);
    x(:,1) = A0 \ b;
    for i = 2:M+1
        y(:,1,i) = A0 \ C(:,i-1);
    end

    % Iterasi Sherman-Morrison
    for n = 2:M+1
        denom = 1 + I(:,n-1)' * y(:,n-1,n);
        x(:,n) = x(:,n-1) - (y(:,n-1,n) * (I(:,n-1)' * x(:,n-1))) /
denom;
        for i = n+1:M+1
            y(:,n,i) = y(:,n-1,i) - (y(:,n-1,n) * (I(:,n-1)' * y(:,n-
1,i))) / denom;
        end
    end
    sol = x(:,M+1);
end
```

Pembahasan

Berdasarkan Tabel 2, diperoleh bahwa algoritma Sherman-Morrison memberikan hasil yang sangat baik pada sistem linier dengan kondisi numerik stabil. Untuk kasus matriks acak penuh berukuran 4×4 dan matriks simetris positif definit 10×10, errorr residual yang diperoleh berada pada kisaran 10^{-15} hingga 10^{-16}, dan selisih solusi antar metode mendekati nol. Ini menunjukkan akurasi tinggi dan kestabilan algoritma pada sistem berdimensi kecil. Hasil ini sejalan dengan temuan Egidi dan Maponi (2006), yang menunjukkan bahwa pendekatan berbasis pembaruan invers rank-satu dapat memberikan solusi akurat untuk sistem kecil-menengah. Namun, ketika diterapkan pada matriks jarang (*sparse*) 100×100, error residual meningkat menjadi sekitar 10^{-13}. Meskipun begitu, selisih solusi tetap dalam batas toleransi numerik yaitu sekitar 10^{-12} yang berarti bahwa hasilnya masih dapat diterima secara komputasional. Studi oleh Aoulad dan Tajani (2023) dalam konteks sistem sparse Toeplitz juga menunjukkan bahwa algoritma SMW memiliki hasil yang stabil jika diterapkan pada struktur sparse tertentu, meskipun efisiensi waktunya masih bergantung pada implementasi.

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

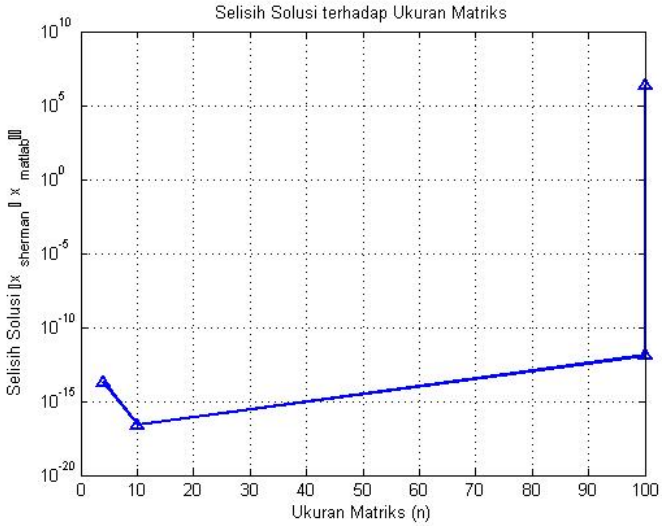
Hasil berbeda ditunjukkan pada kasus matriks. Dalam kasus ini, error residual membesar menjadi 3.91×10^{-3} dan selisih solusi mencapai lebih dari dua juta. Nilai ini sangat besar dibandingkan kasus sebelumnya, menunjukkan bahwa algoritma Sherman-Morrison tidak mampu menjaga kestabilan numerik saat diterapkan pada sistem dengan matriks yang hampir tak berbalik (*ill-conditioned*). Hal ini sesuai dengan temuan Li et al. (2022) yang menunjukkan bahwa modifikasi rank-satu seperti SMW tidak efektif untuk matriks dengan nilai determinan mendekati nol tanpa regularisasi tambahan. Sebaliknya, metode $A\backslash b$ tetap memberikan error residual kecil dan hasil yang stabil.



Gambar 3. Log Waktu Eksekusi terhadap Ukuran Matriks

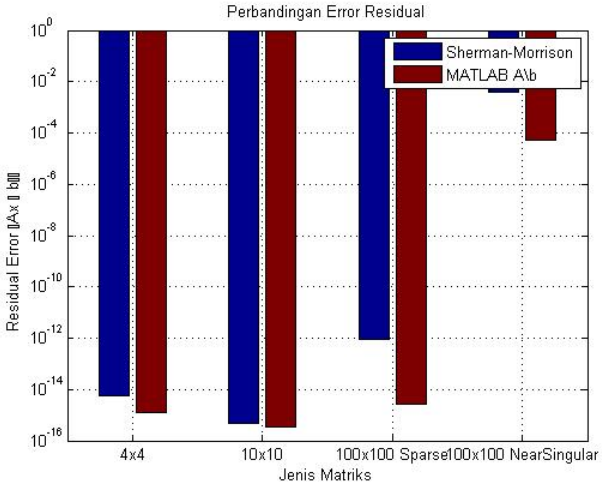
Gambar 3 menunjukkan perbandingan waktu eksekusi antara algoritma Sherman-Morrison dan MATLAB $A\backslash b$. Dalam skala logaritmik, terlihat bahwa waktu eksekusi meningkat signifikan seiring bertambahnya ukuran matriks. Sherman-Morrison menunjukkan pertambahan waktu yang lebih tinggi, khususnya pada matriks jarang (*sparse*) dan matriks hampir singular.

Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index



Gambar 4. Selisih Solusi terhadap Ukuran Matriks

Gambar 4 menampilkan selisih solusi terhadap ukuran matriks. Pada ukuran kecil, selisih sangat kecil dan mendekati nol. Namun, terjadi lonjakan tajam pada jenis matriks hampir singular 100×100, yang mencerminkan kegagalan akurasi akibat kondisi numerik yang buruk.



Gambar 5. Perbandingan Error Residual

Gambar 5 menunjukkan perbandingan error residual antara metode Sherman-Morrison dan $A\backslash b$. Untuk semua jenis matriks kecuali matriks hampir singular, kedua metode menunjukkan hasil yang cukup mendekati. Pada matriks hampir singular terjadi peningkatan drastis yang menandakan ketidakstabilan solusi.


Leibniz: Jurnal Matematika
Juli 2025
Vol. 5 No. 02 Hal. 150-166
e-ISSN: 2775-2356
https://ejurnal.unisap.ac.id/leibniz/index

Hasil-hasil ini menunjukkan bahwa algoritma Sherman-Morrison sangat layak diterapkan pada sistem kecil dan *sparse*, namun penggunaannya harus dihindari pada sistem dengan matriks yang mendekati singular. Secara keseluruhan, tujuan penelitian untuk mengimplementasikan algoritma Sherman-Morrison dalam MATLAB secara modular, serta mengevaluasi akurasi dan efisiensinya, telah tercapai. Penelitian ini juga mengisi kekosongan kajian implementatif terhadap algoritma Sherman-Morrison dalam konteks pembelajaran dan komputasi numerik berbasis MATLAB.

SIMPULAN

Berdasarkan hasil dan analisis yang telah dilakukan, algoritma Sherman-Morrison cukup efektif digunakan dalam penyelesaian sistem persamaan linier dengan modifikasi rank-satu, terutama pada sistem berdimensi kecil dan *sparse*. Ketepatan solusi yang dihasilkan cukup tinggi, sebagaimana ditunjukkan oleh nilai error residual dan selisih solusi yang sangat kecil bila dibandingkan dengan metode $A\backslash b$. Efisiensi komputasi juga tercermin pada sistem kecil meskipun tidak melampaui kinerja $A\backslash b$ secara waktu eksekusi. Sebaliknya, ketidakstabilan numerik teridentifikasi ketika algoritma ini diterapkan pada sistem dengan matriks hampir singular, sebagaimana ditunjukkan oleh error residual dan selisih solusi yang sangat besar. Oleh karena itu, pemanfaatannya hanya sesuai untuk sistem dengan pembaruan lokal dan matriks nonsingular yang terjaga kondisinya. Dengan demikian, efisiensi komputasional, akurasi hasil, serta implementasi berbasis MATLAB melalui struktur modular telah tercapai secara menyeluruh melalui pengujian dan evaluasi.

DAFTAR PUSTAKA

Anton, H., & Rorres, C. (2014). *Elementary Linear Algebra: Applications version* (11th ed.). Wiley.

Aoulad, O. F., & Tajani, C. (2023). A new algorithm for solving Toeplitz linear systems. *Mathematical Modeling and Computing*, 10(3), 807-815.

Arias, M. L., Corach, G., & Maestripieri, A. (2015). Range additivity, shorted operator and the Sherman-Morrison-Woodbury formula. *Linear Algebra and Its Applications*, 467, 86-99.

Cahyono, B. (2016). PENGGUNAAN SOFTWARE MATRIX LABORATORY (MATLAB) DALAM PEMBELAJARAN ALJABAR LINIER. *Phenomenon : Jurnal Pendidikan MIPA*, 3(1), 45-62.

Chai, P., Zhang, J., He, R., Lin, W., & Shu, X. (2021). Application of Sherman-Morrison formula in adaptive analysis by BEM. *Engineering Analysis with Boundary Elements*, 128, 244-256.

Chen, J., Liu, Y., Zhang, W., Liu, Y., Wang, D., & Sun, L. (2021). Reformulation of frequency based substructuring method considering elastic joints according to

sherman-morrison-woodbury formula. *Tehnicki Vjesnik*, 28(6), 1983-1988.

Chen, X., Zhang, L., Gu, C., & Li, Z. (2023). Wideband Sherman-Morrison-Woodbury Formula-Based Algorithm for Electromagnetic Scattering Problems. *IEEE Transactions on Antennas and Propagation*, 71(6), 1-2.

Egidi, N., & Maponi, P. (2006). A Sherman-Morrison approach to the solution of linear systems. *Journal of Computational and Applied Mathematics*, 189, 703–718.

Hager, W. W. (1989). Updating the Inverse of a Matrix. *Journal Society for Industrial and Applied Mathematics*, 31(2), 221–239.

Hao, Y., & Simoncini, V. (2021). The Sherman–Morrison–Woodbury formula for generalized linear matrix equations and applications. *Numerical Linear Algebra with Applications*, 28(5), 1-25.

Irwan, M. (2017). Pengantar Matlab Untuk Sistem Persamaan Linear. *Jurnal Sains Matematika Dan Statistika*, 6(1), 30.

Keviczky, L., Bars, R., Hetthéssy, J., & Bányász, C. (2019). Introduction to MATLAB. In *Advanced Textbooks in Control and Signal Processing*, 1-27.

Li, T., Mosić, D., & Chen, J. (2022). The Sherman-Morrison-Woodbury Formula for the Generalized Inverses. *Filomat*, 36(15), 5307-5313.

Maponi, P. (2007). The solution of linear systems by using the Sherman-Morrison formula. *Linear Algebra and Its Applications*, 420, 276–294.

Nino Ruiz, E. D., Sandu, A., & Anderson, J. (2015). An efficient implementation of the ensemble Kalman filter based on an iterative Sherman–Morrison formula. *Statistics and Computing*, 25(3), 561-577.

Sahid. (2005). *Pengantar Komputasi Numerik dengan MATLAB*. Andi Offset: Yogyakarta.

Sherman, J., & Morrison, W. J. (1950). Adjusment of an Inverse Matrix Cor responding to a Change in One Element of a Given Matrix. *The Annals of Mathematical Statistics*, 21(1), 124–127.

Shi, J., Li, K., Chai, L., Liang, L., Tian, C., & Xu, K. (2023). Fast satellite selection algorithm for GNSS multi-system based on Sherman–Morrison formula. *GPS Solutions*, 27(1), 1-9.

Smejkal, T., & Mikyška, J. (2021). Efficient solution of linear systems arising in the linearization of the VTN-phase stability problem using the Sherman-Morrison iterations. *Fluid Phase Equilibria*, 527, 112832.

Song, Q., Liu, Z., Wan, Y., Ju, G., & Shi, J. (2015). Application of Sherman-Morrison-Woodbury formulas in instantaneous dynamic of peripheral milling for thin-walled component. *International Journal of Mechanical Sciences*, 96-97, 79-90.

Stanimirović, P. S., Katsikis, V. N., & Pappas, D. (2016). Computing {2,4} and {2,3}-inverses by using the Sherman-Morrison formula. *Applied Mathematics and Computation*, 273, 584-603.

Tomozei, I. ., Axinte, T., & Paraschiv, L. (2023). A brief introduction to the Matlab program. *Technium: Romanian Journal of Applied Sciences and Technology*, 9,

54-59.

Wu, F. (2016). Sherman-Morrison-Woodbury Formula for Linear Integrodifferential Equations. *Mathematical Problems in Engineering*, 2016(1), 1-6.

Wu, Y., Zhu, R., Cao, Z., Liu, Y., & Jiang, D. (2020). Model updating using frequency response functions based on sherman-morrison formula. *Applied Sciences (Switzerland)*, 10(14), 4985.

Xu, X. (2017). Generalization of the Sherman–Morrison–Woodbury formula involving the Schur complement. *Applied Mathematics and Computation*, 309, 183-191.

Zulhendri. (2016). Pembelajaran Aljabar Linear Berbantuan Matlab Program Studi Pendidikan Matematika. *Jurnal Cendekia: Jurnal Pendidikan Matematika*, *00*(2), 55-64.