Jurnal Ilmiah Matematika dan Pendidikan Matematika (JMP)
Vol. 16 No. 1, Juni 2024, hal. 57-72
ISSN (Cetak) : 2085-1456; ISSN (*Online*) : 2550-0422

KOMPRESI GAMBAR MENGGUNAKAN SINGULAR VALUE DECOMPOSITION (SVD) DENGAN PYTHON

Deddy Rahmadi*
Program Studi Matematika, Universitas Islam Negeri Sunan Kalijaga
deddy.rahmadi@uin-suka.ac.id

Muhammad Hafizh Naufal
Program Studi Matematika, Universitas Islam Negeri Sunan Kalijaga

Alvian Nur Rohman
Program Studi Matematika, Universitas Islam Negeri Sunan Kalijaga

Fajar Dwi Setyaji
Program Studi Matematika, Universitas Islam Negeri Sunan Kalijaga

ABSTRACT. *Image Compression Using Singular Value Decomposition (SVD) with Python is a powerful technique for dimension reduction and data compression. SVD decomposes a matrix into three separate matrices: U, Σ, and V^T. U contains the left singular vectors, Σ is a diagonal matrix containing the singular values, and V^T represents the right singular vectors. One of the main applications of SVD is image compression. By choosing the right number of singular values, we can reconstruct images with reduced storage and transmission requirements. In this study, we provide results performing SVD on rgb and grayscale images, and also analyze the approximate image compression results with Q_{comp}, MSE,* $\log(\sigma_k)$*, and cumulative sum.*

Keywords*: Image Compression, Python, SVD.*

ABSTRAK. Kompresi Gambar Menggunakan Singular Value Decomposition (SVD) dengan Python adalah teknik yang kuat untuk mereduksi dimensi dan kompresi data. Singular Value Decomposition melakukan dekomposisi matriks menjadi tiga matriks terpisah: U, Σ, dan V^T. U berisi vektor singular kiri, Σ adalah matriks diagonal yang berisi nilai singular, dan V^T mewakili vektor singular kanan. Salah satu aplikasi utama dari SVD adalah kompresi gambar. Dengan memilih jumlah nilai singular yang tepat, kita dapat merekonstruksi gambar dengan persyaratan penyimpanan dan transmisi yang berkurang. Dalam penelitian ini, kita memberikan hasil yang melakukan SVD pada gambar RGB dan *grayscale*, dan juga menganalisis hasil kompresi gambar perkiraan dengan Q_{comp}, MSE, $\log(\sigma_k)$, dan cumulative sum.

Kata Kunci: Kompresi Gambar, Python, Singular Value Decomposition.

*Penulis Korespondensi
Info Artikel : dikirim 12 Juni 2024; direvisi 22 Juni 2024; diterima 24 Juni 2024.

1. PENDAHULUAN

Kompresi gambar adalah proses mengurangi ukuran *file* gambar tanpa mengorbankan kualitas visual secara signifikan. Berdasarkan Phandany dkk. [7] salah satu metode yang digunakan untuk kompresi gambar adalah menggunakan aljabar linear, khususnya teknik yang dikenal sebagai *Singular Value Decomposition* (SVD).

Menurut Leon [5], SVD adalah teknik dalam aljabar linear yang digunakan untuk memecah suatu matriks menjadi tiga komponen dasar: matriks orthogonal, matriks diagonal, dan matriks orthogonal lainnya yang transpos. SVD dalam kompresi gambar dapat digunakan untuk mengurangi redundansi informasi dalam gambar. Dalam artikel ini, kita akan gunakan *Python* untuk menerapkan kompresi gambar menggunakan SVD karena *Python* menawarkan kemudahan penggunaan, kecepatan, fleksibilitas, dan berbagai pustaka yang membuatnya ideal untuk melakukan SVD dalam berbagai aplikasi.

Python adalah salah satu bahasa pemrograman yang sangat populer, dengan berbagai *library* dan alat yang tersedia untuk pengolahan gambar. Dengan menggunakan *library* seperti NumPy dan OpenCV, berdasarkan (Brunton & Kutz, 2022) kita dapat menerapkan algoritma SVD untuk kompresi gambar.

Penelitian terdahulu telah menunjukkan bahwa penggunaan SVD dalam kompresi gambar dapat menghasilkan pengurangan ukuran file yang signifikan dengan sedikit atau tanpa penurunan kualitas visual. Menurut Andrews dan Patterson, SVD telah berhasil diterapkan dalam berbagai aplikasi pengolahan citra, termasuk pengurangan kebisingan dan restorasi citra. Penelitian mereka menunjukkan bahwa dengan mempertahankan hanya sebagian kecil dari nilai singular, dimungkinkan untuk merekonstruksi gambar yang memiliki kualitas mendekati gambar aslinya. Selain itu, Smith dan Zhang pada tahun 2021 dalam studi mereka menemukan bahwa metode SVD dapat mengungguli metode kompresi tradisional seperti JPEG dalam hal ketajaman gambar dan ketahanan terhadap artefak kompresi, terutama pada tingkat kompresi yang lebih tinggi. Mereka juga menyoroti keunggulan SVD dalam mempertahankan struktur spasial dan detail penting dalam gambar, yang sangat penting untuk aplikasi medis dan

ilmiah di mana ketepatan visual adalah krusial. Lebih lanjut, *Image Processing* pada citra satelit juga telah diterapkan oleh Rahmadi dkk. [9] menggunakan metode *Discrete Cosine Transform*. Penelitian-penelitian ini menggarisbawahi potensi besar SVD sebagai alat yang efektif untuk kompresi gambar, yang mendorong eksplorasi lebih lanjut dalam penggunaan teknik ini dalam berbagai konteks praktis. Lebih lanjut, penelitian ini menggunakan SVD, rasio *quality compression*, dan *Mean Square Error* (MSE) dalam metodenya.

2. HASIL DAN PEMBAHASAN

2.1 Singular Value Decomposition dan Teorema Aproksimasi

Tentang *singular value decomposition* telah dikemukakan oleh Leon [5].

Teorema 2.1. Diberikan $A \in R^{m\times n}$ dengan $m \geq n$. Terdapat matriks ortogonal $U \in R^{m\times m}$ dan $V \in R^{n\times n}$ dan sebuah matriks diagonal $\Sigma = diag(\sigma_1, \dots, \sigma_n) \in R^{m\times n}$ sedemikian sehingga $A = U\Sigma V^T$.

Sebuah teorema tentang aproksimasi matriks dijelaskan oleh Eckart & Young [3] yang menyatakan:

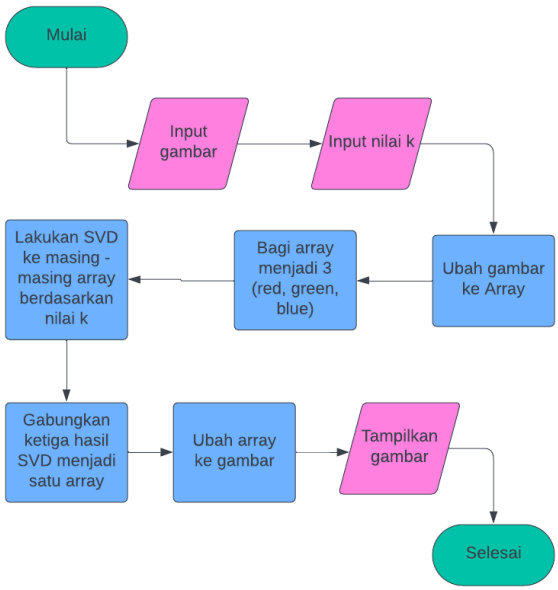
Teorema 2.2. Jika terdapat $r < n$, dimana r adalah kolom utama (*leading columns*) dari U dan V dinotasikan $\tilde{U}$ dan $\tilde{V}$, $r \times r$ adalah sub-blok utama (*leading sub-block*) dari Σ dinotasikan $\tilde{\Sigma}$ sedemikian sehingga $A \approx \tilde{U}\tilde{\Sigma}\tilde{V}^T$. Jika $r = n$ maka $A = \tilde{U}\tilde{\Sigma}\tilde{V}^T$.

2.2 Kompresi Gambar Menggunakan Python

Berdasarkan Teorema 2.2, Brunton & Kutz [1] juga mengemukakan bahwa dapat dilakukan reduksi terhadap suatu matriks menggunakan SVD, sehingga berikut dilakukan kompresi gambar berbentuk matriks yang berisi data warna berukuran 1920×2880. Menggunakan SVD, kita dapat mengatur tingkat kompresi gambar dengan memilih nilai k. Nilai k adalah rank matriks A setelah direduksi.

Gambar terdiri pixel-pixel data. Pixel dapat direpresentasikan menjadi array tiga dimensi (tinggi, lebar, warna). Dalam SVD digunakan matriks, maka gambar dipecah menjadi tiga array warna dua dimensi, yaitu *red* (merah), *green* (hijau), dan *blue* (biru). Setiap array berisi intensitas warna bersesuaian. Nilai intensitas warna antara 0 – 255.

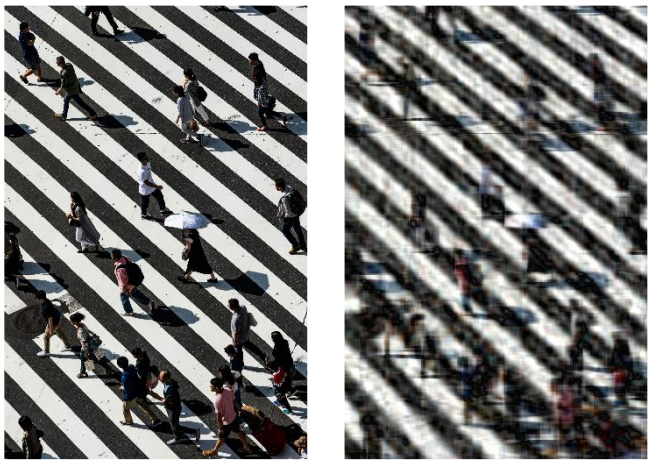
Komputasi dilakukan menggunakan *google collaboratory* dengan library numpy.linalg dengan flowchart sebagai berikut:



Gambar 1. Flowchart Kompresi Gambar

Dilakukan input 5 gambar berbeda yaitu *Cat*, *People*, *Tree*, *Building*, dan *Landscape* dengan 10 nilai *k* berbeda yaitu 2, 5, 20, 30, 50, 100, 200, 400, 800, dan 1000. Berikut disajikan perbandingan kualitas gambar asli dengan gambar terkompres.

Gambar 2. Perbandingan *Cat* Original (Sebelah Kiri) dan dengan $k = 5$ (Sebelah Kanan)



Gambar 3. Perbandingan *People* Original (Sebelah Kiri) dan dengan $k = 20$ (Sebelah Kanan)

Gambar 4. Perbandingan *Tree* Original (Sebelah Kiri) dan dengan $k = 50$ (Sebelah Kanan)

Gambar 5. Perbandingan *Landscape* Original (Sebelah Kiri) dan dengan $k = 200$ (Sebelah Kanan)

Gambar 6. Perbandingan *Building* Original (Sebelah Kiri) dan dengan $k = 200$ (Sebelah Kanan)

Dapat diperoleh bahwa semakin besar nilai k kualitas gambar akan semakin baik, akan tetapi perlu diperhatikan juga ukuran file gambar tersebut. Berikut disajikan tabel ukuran file setiap gambar dalam satuan Kilobyte (KB) untuk melihat perbandingan ukuran gambar dengan k berbeda.

Tabel 1. Ukuran gambar (KiloByte)

k	*Cat*	*People*	*Tree*	*Building*	*Landscape*
2	180	283	261	250	126
5	222	357	346	344	159
20	307	485	527	403	221
30	331	528	597	426	243
50	362	595	695	451	275
100	408	698	844	487	320
200	459	809	981	523	362
400	509	913	1108	558	383
800	521	929	1121	594	398
1000	541	945	1148	600	421
Original	894	1324	1643	1048	614

Berdasarkan tabel diatas, ukuran file gambar akan semakin besar jika nilai k yang dipilih semakin besar. Sehingga untuk meminimalkan ukuran gambar, dibentuk tabel Q_{comp} (rasio *quality compression*) untuk mencari nilai Q_{comp} optimal yang dapat menghasilkan hasil maksimal. Nilai Q_{comp} digunakan untuk membantu membandingkan gambar dengan nilai k berbeda. Semakin kecil nilai Q_{comp}, kualitas gambar semakin baik dan ukuran file semakin besar. Berikut disajikan tabel rasio antara ukuran asli dibagi ukuran kompres. Lebih lanjut, menentukan nilai rasio kualitas kompresi dapat dilakukan dengan menghitung selisih ukuran file sebelum dan sesudah kompresi, kemudian dibagi dengan ukuran file sebelum kompresi dan dikalikan 100%. Nilai rasio kualitas kompresi yang lebih tinggi menunjukkan tingkat kompresi yang lebih besar, sehingga ukuran file menjadi lebih kecil. Namun, hal ini juga dapat menyebabkan penurunan kualitas gambar atau video.

Menentukan nilai rasio kualitas kompresi yang optimal tergantung pada beberapa faktor, seperti:

1. Jenis data: Rasio optimal untuk gambar dan video berbeda. Biasanya, gambar dapat dikompresi dengan rasio yang lebih tinggi tanpa kehilangan kualitas yang signifikan dibandingkan video.
2. Tingkat detail yang diinginkan: Jika Anda membutuhkan detail yang tinggi, Anda harus menggunakan rasio kompresi yang lebih rendah. Sebaliknya, jika Anda ingin menghemat ruang penyimpanan, Anda dapat menggunakan rasio kompresi yang lebih tinggi, meskipun dengan sedikit penurunan kualitas.
3. Kegunaan file: Jika file hanya untuk dilihat, rasio kompresi yang lebih tinggi mungkin dapat diterima. Namun, jika file akan diedit atau dicetak, Anda harus menggunakan rasio kompresi yang lebih rendah untuk menjaga kualitas.

Tabel 2. Q_{comp}

k	*Cat*	*People*	*Tree*	*Building*	*Landscape*
2	4,966667	4,678445	6,295019	4,192	4,8730159
5	4,027027	3,708683	4,748555	3,0465116	3,8616352
20	2,912052	2,729897	3,117647	2,6004963	2,7782805
30	2,700906	2,507576	2,752094	2,4600939	2,526749
50	2,469613	2,22521	2,364029	2,3237251	2,2327273
100	2,191176	1,896848	1,946682	2,1519507	1,91875
200	1,947712	1,636588	1,674822	2,0038241	1,6961326
400	1,756385	1,450164	1,482852	1,8781362	1,6031332
800	1,715931	1,425188	1,465656	1,7643098	1,5427136
1000	1,652495	1,401058	1,431185	1,7466667	1,4584323

Diperoleh bahwa jika nilai Q_{comp} semakin mendekati 1, maka kualitas semakin mendekati aslinya dan ukuran gambar semakin besar.

2.3 Kompresi Gambar pada *Grayscale*

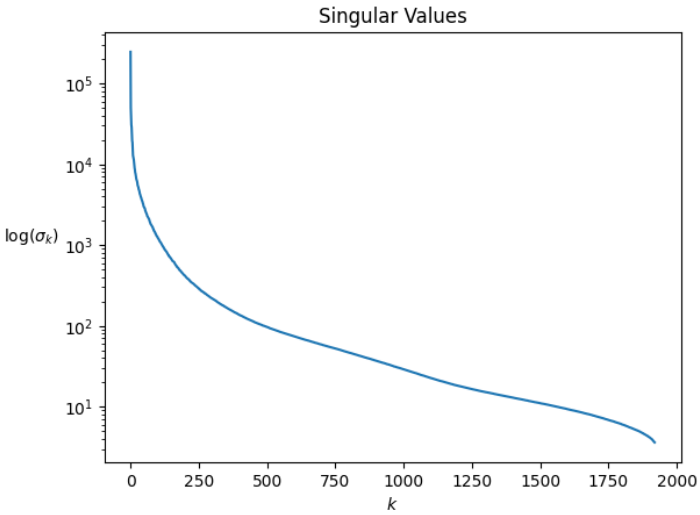
Gambar *grayscale* adalah jenis gambar yang hanya menggunakan warna abu-abu, hitam, dan putih. Array yang terbentuk lebih kecil dibandingkan dengan gambar berwarna. Karena pada *grayscale*, intensitas warna yang digunakan hanya satu yaitu abu-abu. Sehingga cara yang digunakan dalam mengkompresi gambar sama seperti sebelumnya. Berikut hasil kompresi gambar:

Gambar 7. *Face* asli

Gambar 8. Kompresi gambar Face dengan $k = 5, 20, 50$ (Mulai dari Kiri, Tengah, dan Kanan)

Gambar 9. Kompresi gambar Face dengan $k = 100, 200, 400$ (Mulai dari Kiri, Tengah, dan Kanan)

Perhatikan bahwa berdasarkan hasil kompresi gambar diatas pada nilai $k = 5, 20, 50, 100,$ terdapat perbedaan kualitas gambar yang signifikan. Sedangkan pada nilai $k = 200, 400$, kualitas gambar tidak terlihat perbedaannya. Berdasarkan (Brunton & Kutz, 2022) di gambar grayscale, untuk mendapatkan nilai k optimal dapat dianalisis melalui dua grafik berikut.

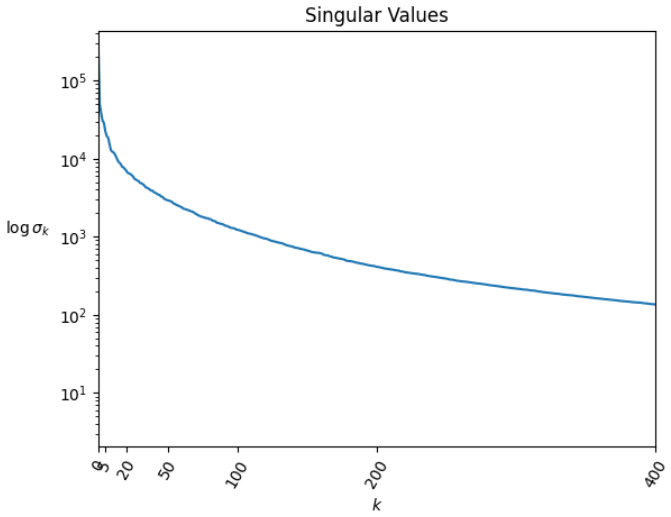


Gambar 10. Grafik nilai singular σ_k

Grafik nilai singular adalah grafik nilai k atas $\log \sigma_k$,

$$f(k) = \log \sigma_k \, .$$

Dapat dilihat bahwa semakin kecil nilai k, $\log \sigma_k$ semakin besar, sehingga semakin besar nilai k semakin kecil peningkatan kualitas gambar yang diperoleh.



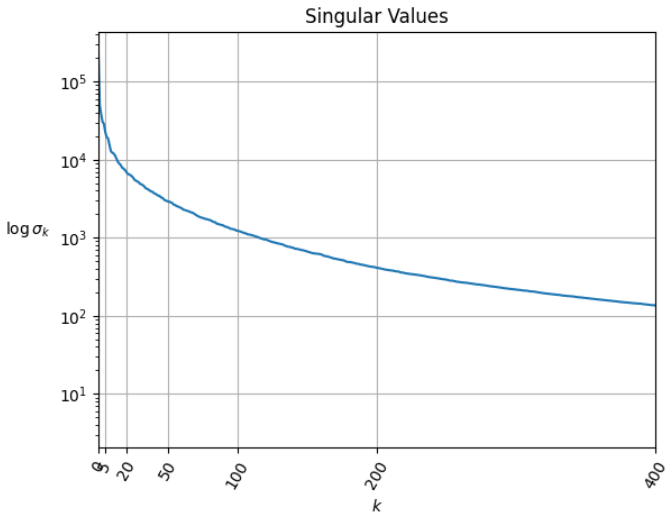
Gambar 11. Grafik nilai singular σ_k untuk $k = [0{,}400]$

Perhatikan bahwa pada beberapa nilai k awal, perubahan nilai $\log \sigma_k$ cukup signifikan. Dapat diperoleh bahwa pada Gambar 11 bahwa perubahan nilai $\log \sigma_k$ semakin kecil pada $k > 50$. Dapat dilihat juga berdasarkan kualitas kompresi

pada Gambar 8 dan Gambar 9 pada nilai $k = 100$ masih terdapat bercak putih yang berarti kualitas gambar masih kurang. Kemudian pada $k = 200$ kualitas gambar sudah cukup baik. Sehingga nilai k yang optimal didapat yaitu $100 < k \leq 200$. Jika nilai $k \leq 100$, kualitas gambar kurang baik. Sedangkan jika $k > 200$ ukuran file terlalu besar.

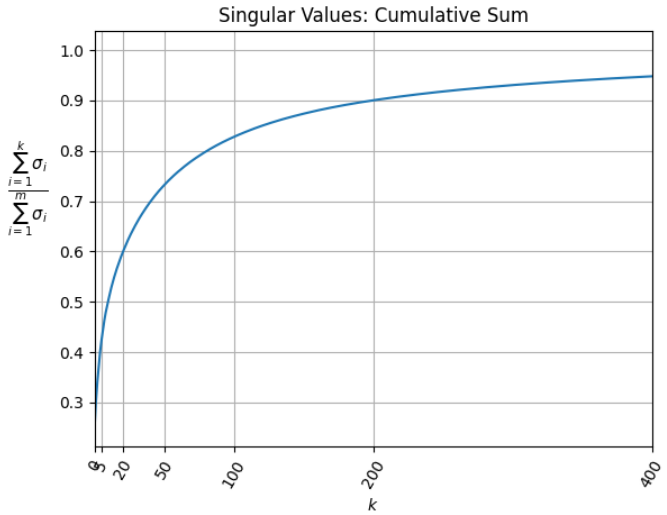
Berikutnya adalah grafik energi kumulatif adalah nilai k atas jumlah nilai matriks singular dibagi jumlah kumulatif nilai matriks singular dengan rumus

$$f(k) = \frac{\sum_{i=1}^{k} \sigma_i}{\sum_{i=1}^{m} \sigma_i}.$$



Gambar 12. Grafik energi kumulatif

Semakin besar nilai k, semakin besar nilai $f(k)$. Berdasarkan grafik, jumlah nilai singular σ_k akan semakin mendekati aslinya seiring meningkatnya nilai k. Dapat dilihat bahwa jumlah kumulatif nilai singular cukup mengalami perubahan yang signifikan pada beberapa nilai k awal. Kemudian penambahan nilai singular akan semakin sedikit pada k setelahnya. Agar semakin jelas, disajikan grafik berikut untuk nilai k dari 0 sampai 400.



Gambar 13. Grafik energi kumulatif untuk $k = [0{,}400]$

Dapat dilihat bahwa perubahan nilai singular pada $k < 50$ cukup signifikan. Sedangkan pada $k > 100$ cukup sedikit perubahannya. Jika dibandingkan dengan sampel hasil kompresi pada Gambar 8 dan Gambar 9, dapat diperoleh bahwa nilai k optimal adalah $100 < k \leq 200$. Selanjutnya disajikan tabel ukuran file, Q_{comp}, dan MSE (*Mean Square Error*):

Tabel 3. Ukuran file, Q_{comp}, dan MSE pada gambar *Face*

k	ukuran	Q_{comp}	MSE
2	186	2,854839	11902,5
5	215	2,469767	9985,6
20	236	2,25	8702,5
30	275	1,930909	6553,6
50	301	1,76412	5290
100	319	1,664577	4494,4
200	331	1,60423	4000
400	392	1,354592	1932,1
800	409	1,298289	1488,4
1000	433	1,226328	960,4
original	531	1	0

Diperoleh hasil yang sama ukuran file gambar akan semakin besar jika nilai k yang dipilih semakin besar. Jika nilai Q_{comp} semakin mendekati 1, maka

kualitas semakin mendekati aslinya dan ukuran gambar semakin besar, karena rasio kompresi yang mendekati 1 menghasilkan gambar dengan kualitas yang hampir sama dengan gambar asli, namun ukuran filenya akan jauh lebih besar.

Mean Square Error (MSE) menghitung selisih antara ukuran kompresi dengan ukuran original dari data, kemudian mengkuadratkan selisih tersebut agar tidak ada selisih yang bernilai negatif, lalu selisih kuadrat dijumlahkan dan diambil rata-rata dari semua sampel data Selvam & Selvam [11]. Atau dapat dirumuskan sebagai berikut $MSE = \frac{1}{n}\sum_{i=1}^{n}(original - compressed_i)^2$. Semakin kecil nilai MSE maka gambar semakin mendekati ukuran aslinya.

3. KESIMPULAN DAN SARAN

Kita peroleh bahwa dalam melakukan kompresi gambar menggunakan SVD di python. Semakin besar k (jumlah nilai eigen), maka semakin bagus kualitas gambar dan semakin besar ukuran gambar. Meskipun demikian, terdapat nilai k yang optimal sehingga kualitas gambar maksimal dan ukuran gambar relatif lebih kecil. Mohan, B. C [6] mengatakan bahwa algoritma SVD memiliki kompleksitas komputasi yang tinggi sehingga memiliki kelemahan yang lambat dalam melakukan komputasi pada kompresi citra digital.

Menurut Phandany dkk. [7] selain SVD, kompresi gambar dapat dilakukan menggunakan algoritma DCT, DWT, BTC, AMBTC. Selain kompresi gambar, SVD dapat digunakan sebagai kompresi model bahasa besar (LLM) Wang dkk. [12], pencitraan tomografi terkomputasi (CT) [10], algoritma struktur kompleks untuk matriks quaternion [2] dan terus berkembang

DAFTAR PUSTAKA

[1] Brunton, S. L. dan Kutz, J. N, *Data-driven Science and Engineering: Machine Learning, Dynamical Systems, and Control*, Cambridge University Press, 2022.

[2] Chang, J. H. dan Ding, J. J. (2003), *Quaternion Matrix Singular Value Decomposition and Its Applications for Color Image Processing*,

Proceedings 2003 International Conference on Image Processing (Cat. No. 03CH37429), IEEE, **1** (2003), 1-805.

[3] Eckart, C. dan Young, G, *The Approximation of One Matrix by Another of Lower Rank*, Psychometrika, **1**(3) (1936), 211-218.

[4] Kahu, S. dan Rahate, R, *Image Compression Using Singular Value Decomposition*, International Journal of Advancements in Research & Technology, **2**(8) (2013), 244-248.

[5] Leon, S. J., *Linear algebra with applications*, Pearson, 2015.

[6] Mohan, B. C., Swamy, K. V., dan Kumar, S. S., *A Comparative performance evaluation of SVD and Schur Decompositions for Image Watermarking*, *IJCA* Proceedings on International Conference on VLSI, Communications and Instrumentation (ICVCI), **14** (2011), 25-29).

[7] Phandany, J. L., Sambul, A. M., dan Lumenta, A. S., *Comparative Study of Digital Image Optimal Compression Algorithm Using Python*, **11** (2022), 23-34.

[8] Prasantha, H. S., Shashidhara, H. L., dan Murthy, K. B., *Image Compression Using SVD*, International conference on computational intelligence and multimedia applications (ICCIMA 2007), IEEE, **3** (2007), 143-145.

[9] Rahmadi, D. dan Rachmawati, S., *Landsat Satellite Image Quality Improvement Using Discrete Cosine Transform Method*, Engineering Headway, **6** (2024), 167-171.

[10] Rao, G. S., Muddada, L. P. R., dan Rao, B. P., *Comparative Analysis of SVD and Progressive SPIHT Techniques for Compression of MRI and CT Images*, Proceedings of International Conference on Sustainable Computing in Science, Technology and Management (SUSCOM), Amity University Rajasthan, Jaipur-India, 2019.

[11] Selvam, S. K., dan Selvam, S., *Image Compression Techniques Using Linear Algebra with SVD Algorithm*, Asian Journal of Engineering and Applied Technology, N.M.S.S.V.N. College. India, 2021.

[12] Wang, X., Zheng, Y., Wan, Z., dan Zhang, M., *SVD-LLM: Truncation-aware singular value decomposition for large language model compression*, arXiv preprint arXiv:2403.07378, 2024.