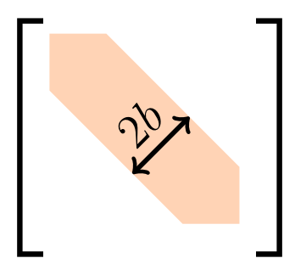
Skoltech
Skolkovo Institute of Science and Technology

# Matrix Decompositions Cheat Sheet

## Numerical linear algebra course at Skoltech by Ivan Oseledets

Poster is prepared by TAs Maxim Rakhuba and Alexandr Katrutsa

Based on the idea by Skoltech students, NLA 2016

| Name | Definition | ∃ | ! | Algorithms | Use cases |
|---|---|---|---|---|---|
| SVD<br>(Singular Value Decomposition) | $A = \underset{U}{\begin{bmatrix}\;\end{bmatrix}_{m\times m}} \underset{\Sigma}{\begin{bmatrix}\sigma_1 & & \\ & \ddots & \\ & & \sigma_r\end{bmatrix}_{m\times n}} \underset{V^*}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}}$<br>- $r = \mathrm{rank}(A)$<br>- $U, V$ — unitary<br>- $\sigma_1 \geq \ldots \geq \sigma_r > 0$ are nonzero *singular values*<br>- columns of $U, V$ are *singular vectors*<br>**Note:** SVD can be also defined with $U \in \mathbb{C}^{m\times p}$, $\Sigma \in \mathbb{R}^{p\times p}$ and $V \in \mathbb{C}^{n\times p}$, $p = \min\{n.m\}$ | ✓ | - Singular values are unique<br>- If all $\sigma_i$ are different, $U$ and $V$ are unique up to unitary diagonal $D$: $U\Sigma V^* = (UD)\Sigma(VD)^*$<br>- If some $\sigma_i$ coincide, then $U$ and $V$ are not unique | - SVD via spectral decomposition of $AA^*$ and $A^*A$ – stability issues<br>- Stable algorithm, $\mathcal{O}(mn^2)$ flops ($m > n$):<br>1. Bidiagonalize $A$ by Householder reflections $A = U_1 B V_1^* = U_1 \begin{bmatrix}\;\end{bmatrix} V_1^*$<br>2. Find SVD of $B = U_2 \Sigma V_2^*$ by spectral decomposition of $T$ (2 options):<br>a) $T = B^*B$, don't form $T$ explicitly!<br>b) $T = \begin{bmatrix} & B^* \\ B & \end{bmatrix}$, permute $T$ to tridiagonal<br>3. $U = U_1 U_2, \quad V = V_1 V_2$ | - Data compression, as Eckart-Young theorem states that truncated SVD $A_k = \underset{U_k}{\begin{bmatrix}\;\end{bmatrix}_{m\times k}} \underset{\Sigma_k}{\begin{bmatrix}\sigma_1 & & \\ & \ddots & \\ & & \sigma_k\end{bmatrix}_{k\times k}} \underset{V_k^*}{\begin{bmatrix}\;\end{bmatrix}_{k\times n}}$ yields best rank-$k$ approximation to $A$ in $\lVert \cdot \rVert_{2,F}$<br>- Calculation of pseudoinverse $A^+$, e.g. in solving over/underdetermined, singular, or ill-posed linear systems<br>- Feature extraction in machine learning<br>**Note:** SVD is also called principal component analysis (PCA) |
| Skeleton<br>(also known as Rank decomposition) | $\mathcal{X} = \underset{U}{\begin{bmatrix}\;\end{bmatrix}_{m\times r}} \underset{V^\top}{\begin{bmatrix}\;\end{bmatrix}_{r\times n}}$<br>or using matrix entries<br>$\underset{A}{\begin{bmatrix}\;\end{bmatrix}_{m\times n}} = \underset{\widehat{C}}{\begin{bmatrix}\;\end{bmatrix}_{m\times r}} \underset{\widehat{A}^{-1}}{\begin{bmatrix}\;\end{bmatrix}^{-1}_{r\times r}} \underset{\widehat{R}}{\begin{bmatrix}\;\end{bmatrix}_{r\times n}}$<br>- $r = \mathrm{rank}(A)$<br>- $C$ and $\widehat{C}$ are full column rank<br>- $R$ and $\widehat{R}$ are full row rank | ✓ | Not unique:<br>- in $A = CR$ version $\forall S$: $\det(S) \neq 0$: $CR = CSS^{-1}R = \widetilde{C}\widetilde{R}$<br>- in $A = \widehat{C}\widehat{A}^{-1}\widehat{R}$ version any $r$ linearly independent columns and rows can be chosen | Assuming $m > n$:<br>- truncated SVD, $\mathcal{O}(mn^2)$ flops, $C = U_r\Sigma_r, R = V_r^*$<br>- RRQR: $\mathcal{O}(mnr)$ flops<br>- Cross approximation: $\mathcal{O}((n+m)r^2)$ flops. It is based on greedy maximization of $\lvert\det(\widehat{A})\rvert$. Might fail on some $A$.<br>- Optimization methods (ALS, ...) for $\lVert A - CR \rVert \to \min\limits_{C,R}$, sometimes with additional constraints, e.g.<br>– nonnegativity of $C$ and $R$ elements<br>– small norms of $C$ and $R$ | - Model reduction, data compression, and speedup of computations in numerical analysis: given rank-$r$ matrix with $r \ll n, m$ one needs to store $\mathcal{O}((n+m)r) \ll nm$ elements<br>- Feature extraction in machine learning, where it is also known as matrix factorization<br>- All applications where SVD applies, since Skeleton decomposition can be transformed into truncated SVD form |
| Schur | $A = \underset{U}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}} \underset{T}{\begin{bmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_n\end{bmatrix}_{n\times n}} \underset{U^*}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}}$<br>- $U$ is unitary<br>- $\lambda_1, \ldots, \lambda_n$ are *eigenvalues*<br>- columns of $U$ are *Schur vectors* | ✓ | - Not unique in terms of both $U$ and $T$: permutation of $\lambda_1, \ldots, \lambda_n$ in $T$ will change both $U$ and off-diagonal part of $T$ | - QR algorithm, $\mathcal{O}(n^4)$ flops: $A_k = Q_k R_k,\ A_{k+1} = R_k Q_k$<br>- "Smart" QR algorithm, $\mathcal{O}(n^3)$ flops:<br>1. Reduce $A$ to upper Hessenberg form $\widetilde{A} = Q^*AQ = \begin{bmatrix}\;\end{bmatrix}$<br>**Note:** then each iteration of QR algorithm will cost $\mathcal{O}(n^2)$<br>2. Run QR algorithm for $\widetilde{A}$ with shifting strategy to speed-up convergence | - Computation of matrix spectrum<br>- Computation of matrix functions (Schur-Parlett algorithm)<br>- Solving matrix equations (e.g. Sylvester equation) |
| Spectral | $A = \underset{S}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}} \underset{\Lambda}{\begin{bmatrix}\lambda_1 & & \\ & \ddots & \\ & & \lambda_n\end{bmatrix}_{n\times n}} \underset{S^{-1}}{\begin{bmatrix}\;\end{bmatrix}^{-1}_{n\times n}}$<br>- $\lambda_1, \ldots, \lambda_n$ are *eigenvalues*<br>- columns of $S$ are *eigenvectors* | - ∃ iff $\forall \lambda_i$ its geometric multiplicity equals algebraic multiplicity<br>- ∃ and $S$ – unitary iff $A$ is normal: $AA^* = A^*A$, e.g. Hermitian | - If all $\lambda_i$ are different, then unique up to permutation and scaling of eigenvectors<br>- If some $\lambda_i$ coincide, $S$ is not unique | - If $A = A^*$, Jacobi method: $\mathcal{O}(n^3)$<br>- If $AA^* = A^*A$, QR algorithm: $\mathcal{O}(n^3)$<br>- If $AA^* \neq A^*A$, $\mathcal{O}(n^3)$ flops:<br>1. Find Schur form $A = UTU^*$ via QR algorithm<br>2. Given $T$ find its eigenvectors $V$<br>3. $S = UV$, $\Lambda = \mathrm{diag}(T)$ | - Full spectral decomposition is rarely used unless all eigenvectors are needed<br>- If one needs only spectrum, Schur decomposition is the method of choice<br>- If matrix has no spectral decomposition, Schur decomposition is preferable for numerics compared to Jordan form |
| QR | $A = \underset{Q \text{ is left unitary}}{\begin{bmatrix}\;\end{bmatrix}_{m\times n}} \underset{R}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}} \quad m \geq n$<br>$A = \underset{Q \text{ is unitary}}{\begin{bmatrix}\;\end{bmatrix}_{m\times m}} \underset{R}{\begin{bmatrix}\;\end{bmatrix}_{m\times n}} \quad m < n$ | ✓ | - Unique if all diagonal elements of $R$ are set to be positive | Assuming $m > n$:<br>- Gram-Schmidt (GS) process: $2mn^2$ flops; not stable<br>- modified Gram-Schmidt (MGS) process: $2mn^2$ flops; stable<br>- via Householder reflections: $2mn^2 - (2/3)n^3$ flops; best for dense matrices, sequential computer architectures; stable<br>- via Givens rotations: $3mn^2 - n^3$ flops; best for sparse matrices, parallel computer architectures; stable | - Computation of orthogonal basis in a linear space<br>- Solving least squares problem ($m > n$): $\lVert Ax - b \rVert_2 \to \min\limits_x \ \Rightarrow\ x = R^{-1}Q^*b$<br>- Solving linear systems<br>**Note:** more stable, but has larger constant than LU<br>- Don't confuse QR decomposition and QR algorithm! |
| RRQR<br>(Rank Revealing QR) | $AP = \underset{Q \text{ is unitary}}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}} \underset{R}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}}$, $r$, $n-r$<br>- $P$ is permutation matrix<br>- $r = \mathrm{rank}(A)$ | ✓ | - Not unique since any $r$ linearly independent columns can be selected | - Basic algorithm: *Householder QR with column pivoting*. On $k$-th iteration:<br>1. Find column of largest norm in $R_k$`[:,k:n]`<br>2. Permute this column and the $k$-th column<br>3. Zero subcolumn of the $k$-th column by Householder reflection $\to R_{k+1}$<br>Complexity: $\mathcal{O}(nmr)$ flops | - Solving rank deficient least squares problem<br>- Finding subset of linearly independent columns<br>- Computation of matrix approximation of a given rank |
| LU | $A = \underset{L}{\begin{bmatrix}1 & & \\ & \ddots & \\ & & 1\end{bmatrix}_{n\times n}} \underset{U}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}}$ | Let $\det(A) \neq 0$<br>- LU ∃ iff all leading minors $\neq 0$ | - Unique if $\det(A) \neq 0$ | - Different versions of Gaussian elimination, $\mathcal{O}(n^3)$ flops. In LU for stability use permutation of rows or columns (LUP)<br>- $\mathcal{O}(n^3)$ can be decreased for *sparse matrices* by appropriate permutations, e.g.<br>– minimum degree ordering<br>– Cuthill–McKee algorithm<br>- Banded matrix with bandwidth $b$  can be decomposed using $\mathcal{O}(nb^2)$ flops | LU, LDL, Cholesky are used for<br>- solving linear systems. Given $A = LU$, complexity of solving $Ax = b$ is $\mathcal{O}(n^2)$:<br>1. Forward substitution: $Ly = b$<br>2. Backward substitution: $Ux = y$<br>- matrix inversion<br>- computation of determinant<br>Cholesky is also used for<br>- computing QR decomposition |
| LDL | $A = \underset{L}{\begin{bmatrix}1 & & \\ & \ddots & \\ & & 1\end{bmatrix}_{n\times n}} \underset{D}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}} \underset{L^*}{\begin{bmatrix}1 & & \\ & \ddots & \\ & & 1\end{bmatrix}_{n\times n}}$ | Let $\det(A) \neq 0$<br>- LDL ∃ iff $A = A^*$ and all leading minors $\neq 0$ |  |  |  |
| Cholesky | $A = \underset{L}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}} \underset{L^*}{\begin{bmatrix}\;\end{bmatrix}_{n\times n}}$ | - Cholesky ∃ iff $A = A^*$ and $A \succeq 0$ | - Unique if $A \succ 0$ |  |  |

## References

(1) G. H. Golub and C. F. Van Loan, *Matrix computations*, JHU Press, 4th ed., 2013.

(2) L. N. Trefethen and D. Bau III, *Numerical linear algebra*, vol. 50, SIAM, 1997.

(3) E. E. Tyrtyshnikov, *A brief introduction to numerical analysis*, Springer Science & Business Media, 2012.

## Contact information

Course materials: https://github.com/oseledets/nla2016

Email: i.oseledets@skoltech.ru

Our research group website: oseledets.github.io

LaTeX TikZposter