Faster SVD via Accelerated Newton-Schulz Iteration

Askar Tsyganov*, Uliana Parkina*, Ekaterina Grishina,
Sergey Samsonov, Maxim Rakhuba

ICLR BlogPost Track

Rio de Janeiro 2026

Reminder

Definition 1 (Singular Value Decomposition (SVD))

Any matrix $M \in \mathbb{R}^{m \times n}$ with $m \geq n$ admits a singular value decomposition

$$M = U \Sigma V^{\top},$$

where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ have orthonormal columns, and Σ is a diagonal matrix whose diagonal entries are the non-negative singular values.

Reminder

Definition 1 (Singular Value Decomposition (SVD))

Any matrix $M \in \mathbb{R}^{m \times n}$ with $m \geq n$ admits a singular value decomposition

$$M = U \Sigma V^{\top},$$

where $U \in \mathbb{R}^{m \times m}$ and $V \in \mathbb{R}^{n \times n}$ have orthonormal columns, and Σ is a diagonal matrix whose diagonal entries are the non-negative singular values.

Definition 2 (Polar Decomposition)

Any matrix $M \in \mathbb{R}^{m \times n}$ with $m \geq n$ admits a (right) polar decomposition

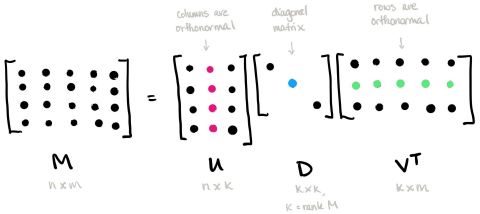
$$M = WH,$$

where $W \in \mathbb{R}^{m \times n}$ has orthonormal columns, and $H \in \mathbb{R}^{n \times n}$ is a PSD matrix.

Ways to Compute SVD on GPU

You have:

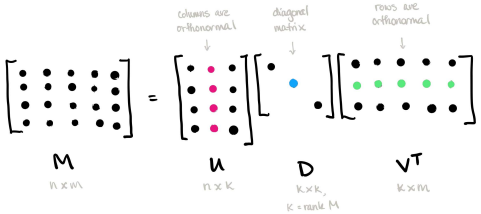
You want:



Ways to Compute SVD on GPU

You have:

You want:



Your ways:

- ◇ CUDA QR. cuSOLVER implementation of QR-based SVD.
- ◇ CUDA POLAR. cuSOLVER implementation of SVD via polar decomposition using the QDWH iteration.
- ◇ CUDA JACOBI. cuSOLVER implementation of Jacobi-based SVD.

SVD via Polar Decomposition

First, compute the polar factor W using iterative methods and decompose M as:

$$M = WH.$$

SVD via Polar Decomposition

First, compute the polar factor W using iterative methods and decompose M as:

$$M = WH.$$

Then find symmetric eigenvalue decomposition (EIGH) of H:

$$M = WV\Sigma V^\top.$$

SVD via Polar Decomposition

First, compute the polar factor W using iterative methods and decompose M as:

$$M = WH.$$

Then find symmetric eigenvalue decomposition (EIGH) of H:

$$M = WV\Sigma V^\top.$$

The resulting decomposition is obtained by setting $U = WV$:

$$M = U\Sigma V^\top.$$

SVD via Polar Decomposition

First, compute the polar factor W using iterative methods and decompose M as:

$$M = WH.$$

Then find symmetric eigenvalue decomposition (EIGH) of H:

$$M = WV\Sigma V^\top.$$

The resulting decomposition is obtained by setting $U = WV$:

$$M = U\Sigma V^\top.$$

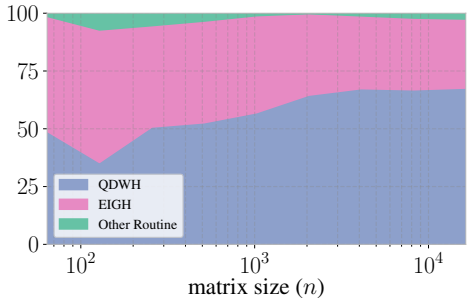


Figure: Comparison of different parts of the SVD computation by their share of the total runtime.

QR vs Matrix Multiplications

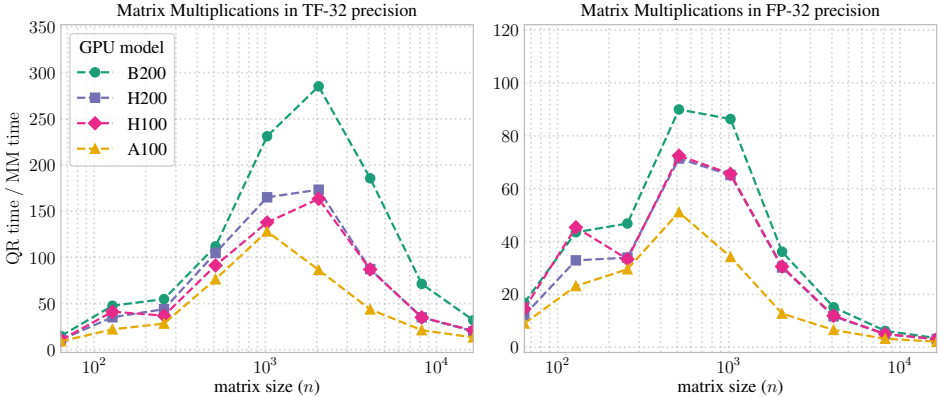


Figure: Comparison illustrating how the number of single matrix multiplications that can be executed within the runtime of a single QR decomposition (y-axis) varies with matrix size for a square matrix.

How to Compute Polar Decomposition using Matrix Multiplications?

We consider Newton-Schulz iteration, which solely relies on matrix multiplications:

$$X_{k+1} = \frac{3}{2}X_k - \frac{1}{2}X_k X_k^\top X_k, \qquad X_0 = M.$$

This iteration quadratically converges to a polar factor. Recent studies by Grishina et al. and Amsel et al. introduced accelerated versions of this iteration: CANS and Polar Express.

Experiments on Runtime

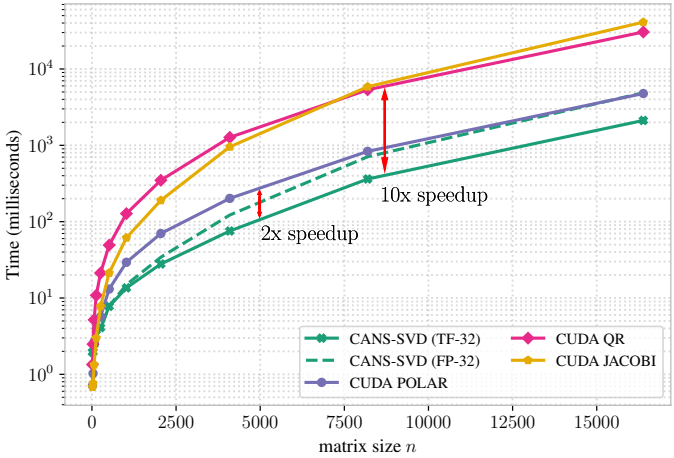


Figure: Algorithm comparison on a random square matrix on an NVIDIA B200 GPU.

Experiments on Accuracy

Table: Comparison of the accuracy of forward pass SVD algorithms on a random square matrix $M \in \mathbb{R}^{4096\times4096}$ for different condition numbers. The reported metric is the relative reconstruction error $\|M - U\Sigma V^\top\|_F / \|M\|_F$. Red indicates cases with significantly large errors.

Method	1.1	10	10^2	10^3	10^4	10^6
CANS SVD (fp32)	$4.4 \cdot 10^{-6}$	$4.9 \cdot 10^{-6}$	$4.2 \cdot 10^{-6}$	$4.3 \cdot 10^{-6}$	$4.4 \cdot 10^{-6}$	$5.3 \cdot 10^{-6}$
CANS SVD (tf32)	$7.5 \cdot 10^{-4}$	$9.7 \cdot 10^{-4}$	$8.2 \cdot 10^{-4}$	$8.5 \cdot 10^{-4}$	$8.7 \cdot 10^{-4}$	$1.2 \cdot 10^{-3}$
CUDA POLAR	$5.5 \cdot 10^{-6}$	$4.3 \cdot 10^{-6}$	$5.5 \cdot 10^{-6}$	$7.6 \cdot 10^{1}$	$7.6 \cdot 10^{1}$	$7.6 \cdot 10^{1}$
CUDA QR	$1.4 \cdot 10^{-5}$	$1.8 \cdot 10^{-5}$	$2.7 \cdot 10^{-5}$	$1.5 \cdot 10^{-5}$	$1.6 \cdot 10^{-5}$	$1.1 \cdot 10^{-5}$
CUDA JACOBI	$1.9 \cdot 10^{-3}$	$9.7 \cdot 10^{-4}$	$8.4 \cdot 10^{-4}$	$6.8 \cdot 10^{-4}$	$6.1 \cdot 10^{-4}$	$5.9 \cdot 10^{-4}$

Conclusion

◇ We proposed an efficient GPU-friendly SVD computation via the polar decomposition.

◇ Proposed algorithm allows performing computations in mixed precision, enabling additional speedup.