Structure exploiting Lanczos method for Hadamard product of low-rank matrices

June 6, 2021

Report Type:	Mini Project
Student:	Zilan Cheng
SCIPER:	321854
Instructor:	Prof. Daniel Kressner
Teaching Assistant:	Alice Cortinovis
Date:	June 6, 2021

Structure exploiting Lanczos method for Hadamard product of low-rank matrices

Zilan Cheng

Answer (a)

We will use some properties and definitions of matrix products.
(1) We prove $(AC) * (BD) = (A \odot^T B)(C \odot D)$.
Proof: For the unity of symbol, we assume $A \in \mathbb{R}^{m\times k_A}$, $B \in \mathbb{R}^{m\times k_B}$,$C \in \mathbb{R}^{k_A\times n}$,$D \in \mathbb{R}^{k_B\times n}$.
According to the definition of Transpose Khatri-Rao product, we rewrite $A \odot^T B$ as follows:

$$A \odot^T B = (A^T \odot B^T)^T = \begin{pmatrix} a_1^T \otimes b_1^T \\ a_2^T \otimes b_2^T \\ . \\ . \\ . \\ a_m^T \otimes a_m^T \end{pmatrix}$$

where a_i and b_i denote the ith rows of A and B, respectively.
According to the definition of Khatri-Rao product, we rewrite $C \odot D$ as follows:

$$C \odot D = \begin{pmatrix} c_1 \otimes d_1 & c_2 \otimes d_2 & ... & c_n \otimes d_n \end{pmatrix}$$

where c_j and d_j denote the jth columns of C and D, respectively.
Then we have:

$$\begin{aligned}
(A \odot^T B)(C \odot D) &= \begin{pmatrix} a_1^T \otimes b_1^T \\ a_2^T \otimes b_2^T \\ . \\ . \\ . \\ a_m^T \otimes a_m^T \end{pmatrix} \begin{pmatrix} c_1 \otimes d_1 & c_2 \otimes d_2 & ... & c_n \otimes d_n \end{pmatrix} \\
&= \begin{pmatrix} a_1^T c_1 \otimes b_1^T d_1 & a_1^T c_2 \otimes b_1^T d_2 & ... & a_1^T c_n \otimes b_1^T d_n \\ a_2^T c_1 \otimes b_2^T d_1 & a_2^T c_2 \otimes b_2^T d_2 & ... & a_2^T c_n \otimes b_2^T d_n \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ a_m^T c_1 \otimes b_m^T d_1 & a_m^T c_2 \otimes b_m^T d_2 & ... & a_m^T c_n \otimes b_m^T d_n \end{pmatrix} \\
&= \begin{pmatrix} (ac)_{11}(bd)_{11} & (ac)_{12}(bd)_{12} & ... & (ac)_{1n}(bd)_{1n} \\ (ac)_{21}(bd)_{21} & (ac)_{22}(bd)_{22} & ... & (ac)_{2n}(bd)_{2n} \\ . & . & & . \\ . & . & & . \\ . & . & & . \\ (ac)_{m1}(bd)_{m1} & (ac)_{m2}(bd)_{m2} & ... & (ac)_{mn}(bd)_{mn} \end{pmatrix} \\
&= (AC) * (BD)
\end{aligned}$$

where $(ac)_{pq}$ and $(bd)_{pq}$ denote the entry locate in the pth row and qth column of AC and BD, respectively. The third equation is obtained because there is no difference between Kronecker product and scalar product of 1×1 matrices.
(2) We directly use the property (4) in [2, Section 2.1]:

$$(A \otimes B)(C \odot D) = (AC) \odot (BD)$$

Then it is easy to prove that Hadamard product of A and B admits the following representation:

$$\begin{aligned} A * B &= (U_A \Sigma_A V_A^T) * (U_B \Sigma_B V_B^T) \\ &= (U_A^T \odot^T U_B^T)(\Sigma_A V_A^T \odot \Sigma_B V_B^T) \\ &= (U_A^T \odot U_B^T)^T (\Sigma_A \otimes \Sigma_B)(V_A^T \odot V_B^T) \end{aligned}$$

Answer (b)

According to the property (5),(6), and (7) in [2, Section 2.1], we rewrite $(A * B)x$ as follows:

$$\begin{aligned} (A * B)x &= [(U_A^T \odot U_B^T)^T (\Sigma_A \otimes \Sigma_B)(V_A^T \odot V_B^T)]x \\ &= [(U_A^T \odot U_B^T)^T (\Sigma_A \otimes \Sigma_B)][(V_A^T \odot V_B^T)x] \\ &= [(U_A^T \odot U_B^T)^T (\Sigma_A \otimes \Sigma_B)] vec(V_B^T diag(x) V_A) \\ &= (U_A^T \odot U_B^T)^T [(\Sigma_A \otimes \Sigma_B) vec(V_B^T diag(x) V_A)] \\ &= (U_A^T \odot U_B^T)^T vec(\Sigma_B V_B^T diag(x) V_A \Sigma_A) \\ &= diag(U_B \Sigma_B V_B^T diag(x) V_A \Sigma_A U_A^T) \end{aligned}$$

Answer (c)

Similarly as implementing the matrix-vector multiplication $(A * B)x$, $(A * B)^T x$ can be rewritten as follows:

$$\begin{aligned} (A * B)^T x &= [(U_A^T \odot U_B^T)(\Sigma_A \otimes \Sigma_B)(V_A^T \odot V_B^T)^T]x \\ &= diag(V_B \Sigma_B U_B^T diag(x) U_A \Sigma_A V_A^T) \end{aligned}$$

We now can define the function $(A * B)x$, the function$(A * B)^T x$ and finally the function $[(A * B)(A * B)^T]x = (A * B)[(A * B)^T x]$ in matlab.

Then we can define the Lanczos algorithm as follows:

Algorithm 1: Lanczos algorithm

Input: Random vector: x, tolerance: tol=10^{-8}, max iterations: maxIter=30

Set $q = v/||v||$, $Q = [q]$

for k=1,2,...,maxit **do**

$r = (A * B)(A * B)^T q$ using the fast matrix-vector multiplication

$\alpha_k = q^T r$

$r = r - \alpha_k q$

Reorthogonalize r versus the columns of Q

Set $\beta_k = ||r||$ and compute error ω_k

if $\omega_k < tol$ **then** quit

$q = r/\beta_k$

$Q = [Q\ q]$

end

Output: Q, $T = tridiag((\alpha_1, ..., \alpha_k), (\beta_1, ..., \beta_{k-1}))$

After applying Lanczos, we obtain a tridiagonal matrix T, $(A*B)(A*B)^T = QTQ^T$. We compute the full spectral decomposition of T and obtain $T = P\Lambda P^T$.Then

$$\begin{aligned}(A * B)(A * B)^T &= Q(P\Lambda P^T)Q^T \\ &= (QP)\Lambda(QP)^T\end{aligned}$$

If we denote the SVD of $A * B$ as $A * B = U\Sigma V^T$, then $U = QP$,$\Sigma = \sqrt{\Lambda}$. We use the following equation to compute V.

$$(A * B)^T u_i = (V\Sigma U^T)u_i = \sigma_i v_i$$

where σ_i is the i^{th} singular value, u_i is the i^{th} left singular vector, and v_i is the i^{th} right singular vector.

Answer (d)

We obtain that the ranks of truncated matrices A,B and C are 5,7, and 7.
The rank of C by representation of (1) is 10. It is computed according to:

$$C = A * B = (U_A^T \odot U_B^T)^T(\Sigma_A \otimes \Sigma_B)(V_A^T \odot V_B^T)$$

where we truncate the diagonal matrix $\Sigma_A \otimes \Sigma_B$'s diagonal entries lower than $\sigma < 10^{-4}$. To specify, $rank((U_A^T \odot U_B^T)^T) = 19$, $rank(\Sigma_A \otimes \Sigma_B) = 29$, $rank(V_A^T \odot V_B^T) = 10$.

Answer (e)

Now we are able to plot times needed (**Figure 1**) and errors (**Figure 2**) for both direct 'truncated SVD' and 'lanczos + fast mvMult'.

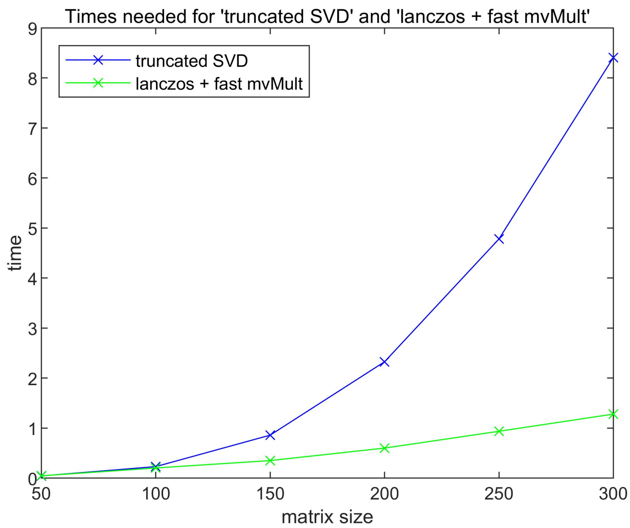

Figure 1: Times needed for 'truncated SVD' and 'lanczos + fast mvMult'

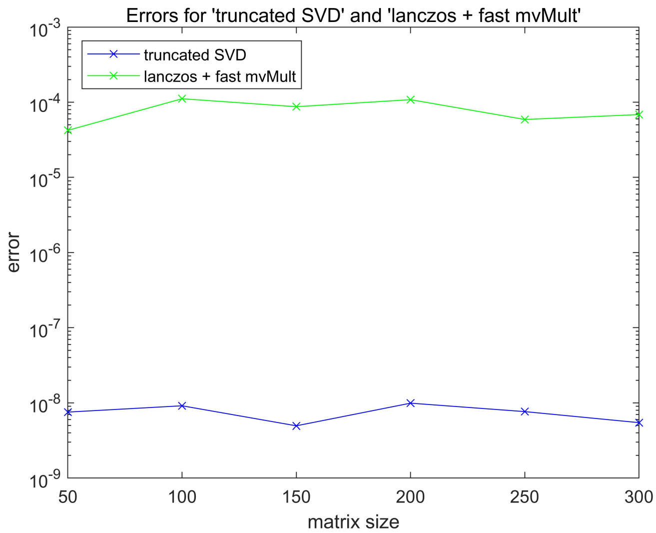

Figure 2: Errors for 'truncated SVD' and 'lanczos + fast mvMult'

We observe:
(1) As matrix size grows, 'lanczos + fast mvMult' performs much better than the usual truncated SVD in time.
(2) The usual truncated SVD performs better in errors. In fact, SVD always gives the best approximation. However, the approximation given by 'lanczos + fast mvMult' can also be accepted in not-very-high tolerance(e.g.10^{-4}).

References

[1] Horst D. Simon, Hongyuan Zha. Low-Rank Matrix Approximation Using the Lanczos Bidiagonalization Process with Applications. SIAM Journal on Scientific Computing, 21(6), 2257–2274.

[2] Daniel Kressner, Lana Periša. Recompression of Hadamard Products of Tensors in Tucker Format. SIAM Journal on Scientific Computing, 39(5), A1879–A1902.