Randomized methods for matrix factorizations

Eftychios Pnevmatikakis, CCM
November 1, 2019

$F_{\omega}(\alpha + m)!$ Conference

FLATIRON INSTITUTE
Center for Computational Mathematics

Basic Matrix Decompositions

Let $A \in \mathbb{C}^{m \times n}$, $(m \geq n$ wlog).

- SVD Decomposition:

$$A = U\Sigma V^*$$

  U, V are unitary matrices ($U^*U = I_m, V^*V = I_n$),
  $\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{\min(m,n)})$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min(m,n)} \geq 0$.

Basic Matrix Decompositions

Let $A \in \mathbb{C}^{m\times n}$, ($m \geq n$ wlog).

- SVD Decomposition:
$$A = U\Sigma V^*$$
U, V are unitary matrices ($U^*U = I_m, V^*V = I_n$),
$\Sigma = \mathrm{diag}(\sigma_1, \sigma_2, \ldots, \sigma_{\min(m,n)})$ with $\sigma_1 \geq \sigma_2 \geq \ldots \geq \sigma_{\min(m,n)} \geq 0$.

- QR decomposition:
$$A = QR$$
Q unitary, R upper triangular.

Low rank approximations I

Approximate $\underset{m\times n}{A} \approx \underset{m\times r}{B}\,\underset{r\times n}{C}$ with $r < m, n$, often $r \ll m, n$. Optimal solution given by the truncated SVD $U\Sigma_{(r)}V^*$:

$$\|A - U\Sigma_{(r)}V^*\| = \sigma_{r+1}$$

Spectral Norm: $\|A\| = \max_{\boldsymbol{x}\neq\boldsymbol{0}} \|A\boldsymbol{x}\|/\|\boldsymbol{x}\|$

$$\|A - U\Sigma_{(r)}V^*\|_F = \left(\sum_{i=r+1}^{\min\{m,n\}} \sigma_i\right)^{1/2}$$

Frobenius Norm: $\|A\|_F = \left(\sum A_{ij}^2\right)^{1/2}$

Low rank approximations II

- Find directions of maximal variance (PCA).
- Low dimensional embeddings (Marina's talk).
- High dimensional ill conditioned least squares.
- Nearest neighbors.
- Fast algorithms for PDE's (Manas' talk)
- Faster linear algebra: $A\boldsymbol{x}$ requires $\mathcal{O}(mn)$ calculations for general dense A, whereas $BC\boldsymbol{x}$ would require only $\mathcal{O}((m+n)r)$.

Low rank approximations III

General (deterministic) approaches are very accurate and well understood but in large dimensions complications arise:

- They generally require $\mathcal{O}(mnr)$ work (exceptions when $A\boldsymbol{x}$ can be evaluated rapidly with $o(mn)$ work).
- They require $\mathcal{O}(r)$ passes over the data which can be inefficient when A is too large to fit in memory.
- Harder to parallelize.

Randomness can help.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U \Sigma V^*$.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U \Sigma V^*$.

- Find unitary $m \times r$ matrix Q such that $A \approx QQ^* A$??? work.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U\Sigma V^*$.

- Find unitary $m \times r$ matrix Q such that $A \approx QQ^*A$??? work.
- Form $r \times n$ matrix $B = Q^*A$ $\mathcal{O}(mnr)$ work.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U\Sigma V^*$.

- Find unitary $m \times r$ matrix Q such that $A \approx QQ^*A$??? work.
- Form $r \times n$ matrix $B = Q^*A$ $\mathcal{O}(mnr)$ work.
- Compute SVD of B: $B = \hat{U}\Sigma V^*$ $\mathcal{O}(nr^2)$ work.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U\Sigma V^*$.

- Find unitary $m \times r$ matrix Q such that $A \approx QQ^*A$??? work.
- Form $r \times n$ matrix $B = Q^*A$ $\mathcal{O}(mnr)$ work.
- Compute SVD of B: $B = \hat{U}\Sigma V^*$ $\mathcal{O}(nr^2)$ work.
- Set $U = Q\hat{U}$ $\mathcal{O}(mr^2)$ work.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U\Sigma V^*$.

- Find unitary $m \times r$ matrix Q such that $A \approx QQ^*A$??? work.
- Form $r \times n$ matrix $B = Q^*A$ $\mathcal{O}(mnr)$ work.
- Compute SVD of B: $B = \hat{U}\Sigma V^*$ $\mathcal{O}(nr^2)$ work.
- Set $U = Q\hat{U}$ $\mathcal{O}(mr^2)$ work.

A general approach for low rank approximation

Given an $m \times n$ matrix A, compute an approximate rank-r SVD $A \approx U\Sigma V^*$.

- Find unitary $m \times r$ matrix Q such that $A \approx QQ^*A$??? work.
- Form $r \times n$ matrix $B = Q^*A$ $\mathcal{O}(mnr)$ work.
- Compute SVD of B: $B = \hat{U}\Sigma V^*$ $\mathcal{O}(nr^2)$ work.
- Set $U = Q\hat{U}$ $\mathcal{O}(mr^2)$ work.

Error depends only on the quality of the range approximation:

$$\|A - U\Sigma V^*\| = \|A - Q\hat{U}\Sigma V^*\| = \|A - QB\| = \|A - QQ^*A\|.$$

In practice, we form an approximate $l = r + s$ rank matrix ($s \sim 5 - 10$).

Randomized range finding

Given an $m \times n$ matrix A and a non-negative integer q, find an $m \times l$ orthonormal matrix Q whose range approximates the range of A.

1. Draw a suitable random $n \times l$ matrix Ω.

Randomized range finding

Given an $m \times n$ matrix A and a non-negative integer q, find an $m \times l$ orthonormal matrix Q whose range approximates the range of A.

1. Draw a suitable random $n \times l$ matrix Ω.
2. Form the matrix $Y = (AA^*)^q A\Omega$.

Randomized range finding

Given an $m \times n$ matrix A and a non-negative integer q, find an $m \times l$ orthonormal matrix Q whose range approximates the range of A.

1. Draw a suitable random $n \times l$ matrix Ω.
2. Form the matrix $Y = (AA^*)^q A\Omega$.
3. Compute the QR factorization of Y: $Y = QR$.

Randomized range finding

Given an $m \times n$ matrix A and a non-negative integer q, find an $m \times l$ orthonormal matrix Q whose range approximates the range of A.

1. Draw a suitable random $n \times l$ matrix Ω.
2. Form the matrix $Y = (AA^*)^q A\Omega$.
3. Compute the QR factorization of Y: $Y = QR$.

Randomized range finding

Given an $m \times n$ matrix A and a non-negative integer q, find an $m \times l$ orthonormal matrix Q whose range approximates the range of A.

1. Draw a suitable random $n \times l$ matrix Ω.
2. Form the matrix $Y = (AA^*)^q A\Omega$.
3. Compute the QR factorization of Y: $Y = QR$.

If $\mathrm{rank(A)} \leq l$ then $A = QQ^*A$ with probability 1. Suitable random matrices include:

- Gaussian random matrix. ($\mathcal{O}(mnl)$ for computing $A\Omega$ for general A).
- Subsampled random Fourier transform (SRFT) matrices ($\mathcal{O}(mn\log(l))$ for computing $A\Omega$ for general A).

Full algorithm

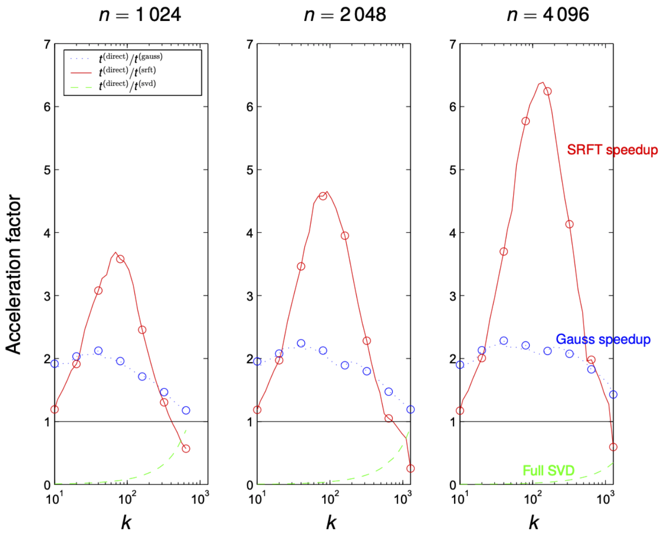
1. Draw a suitable random $n \times r$ matrix Ω — $\mathcal{O}(nr)$ work.
2. Form the matrix $Y = (AA^*)^q A\Omega$ — $(2q+1)T_{\text{mult}}$ work.
3. Compute the QR factorization of Y: $Y = QR$ — $\mathcal{O}(mr^2)$ work.
4. Form $r \times n$ matrix $B = Q^* A$ — $\mathcal{O}(mnr)$ work.
5. Compute SVD of B: $B = \hat{U}\Sigma V^*$ — $\mathcal{O}(nr^2)$ work.
6. Set $U = Q\hat{U}$ — $\mathcal{O}(mr^2)$ work.

Full algorithm

1. Draw a suitable random $n \times r$ matrix Ω — $\mathcal{O}(nr)$ work.
2. Form the matrix $Y = (AA^*)^q A\Omega$ — $(2q+1)\,T_{\mathrm{mult}}$ work.
3. Compute the QR factorization of Y: $Y = QR$ — $\mathcal{O}(mr^2)$ work.
4. Form $r \times n$ matrix $B = Q^* A$ — $\mathcal{O}(mnr)$ work.
5. Compute SVD of B: $B = \hat{U}\Sigma V^*$ — $\mathcal{O}(nr^2)$ work.
6. Set $U = Q\hat{U}$ — $\mathcal{O}(mr^2)$ work.

- Overall work may still be $\mathcal{O}(mnr)$ but can be faster.
- Requires $(2q+2)$ passes over the matrix A. It is possible to perform this task with just one pass.

A numerical example



Performance guarantees

- For Gaussian matrices and q power iterations:

$$\mathbb{E}\|A - U\Sigma V^*\| \leq \left[1 + \sqrt{\frac{r}{s-1}} + \frac{e\sqrt{r+s}}{s}\sqrt{\min\{m,n\} - r}\right]^{1/(2q+1)} \sigma_{r+1}$$

- Concentration of measure phenomena ensures error is close to expected value with very high probability.
- Power iteration becomes important for slowly decaying spectra.
- Can drop error to $\sim \sigma_{r+1}$ with $q \sim \log(\min\{m,n\})$. $(q^{1/\log(q)} = e)$
- Similar results for SRFT matrices.

Eigendecomposition of symmetric matrices

For A symmetric/Hermitian:

1. Find Q such that $A \approx QQ^*A$
2. Form $B = Q^*AQ$
3. Calculate $B = V\Lambda V^*$.
4. Set $U = QV$. Then $A \approx U\Lambda U^*$.

Eigendecomposition of symmetric matrices

For A symmetric/Hermitian:

1. Find Q such that $A \approx QQ^*A$
2. Form $B = Q^*AQ$
3. Calculate $B = V\Lambda V^*$.
4. Set $U = QV$. Then $A \approx U\Lambda U^*$.

For A symmetric PSD (Nyström method):

1. Find Q such that $A \approx QQ^*A$
2. Form $B_1 = AQ$ and $B = Q^*B_1$
3. Calculate Cholesky decomposition $B = C^*C$
4. Form $F = B_1C^{-1}$
5. Compute SVD $F = U\Sigma V^*$ and set $\Lambda = \Sigma^2$. Then $A \approx U\Lambda U^*$.

Application to high dimensional NMF

- Objective: Given $A \in \mathbb{R}^{m \times n}$ and target rank r, find non-negative matrices $B \in \mathbb{R}^{m \times r}$ $C \in \mathbb{R}^{r \times n}$ that solve

$$\min_{B,C \geq 0} \|A - BC\|_F$$

Application to high dimensional NMF

- Objective: Given $A \in \mathbb{R}^{m\times n}$ and target rank r, find non-negative matrices $B \in \mathbb{R}^{m\times r}$ $C \in \mathbb{R}^{r\times n}$ that solve

$$\min_{B,C\geq 0} \|A - BC\|_F$$

- Algorithms typically operate in an alternate fashion (Johannes' talk):

$$B_{k+1} = f(A, B_k, C_k)$$
$$C_{k+1} = g(A, B_{k+1}, C_k)$$

Application to high dimensional NMF

- Objective: Given $A \in \mathbb{R}^{m \times n}$ and target rank r, find non-negative matrices $B \in \mathbb{R}^{m \times r}$ $C \in \mathbb{R}^{r \times n}$ that solve

$$\min_{B,C \geq 0} \|A - BC\|_F$$

- Algorithms typically operate in an alternate fashion (Johannes' talk):

$$B_{k+1} = f(A, B_k, C_k)$$
$$C_{k+1} = g(A, B_{k+1}, C_k)$$

- Approximate $A \approx Q_l Q_l^* A$ and $A^* \approx Q_r Q_r^* A^*$ ($Q_l \in \mathbb{R}^{m \times (r+s)}, Q_r \in \mathbb{R}^{n \times (r+s)}$)

FLATIRON INSTITUTE
Center for Computational Mathematics

Application to high dimensional NMF

- Objective: Given $A \in \mathbb{R}^{m\times n}$ and target rank r, find non-negative matrices $B \in \mathbb{R}^{m\times r}$ $C \in \mathbb{R}^{r\times n}$ that solve

$$\min_{B,C\geq 0} \|A - BC\|_F$$

- Algorithms typically operate in an alternate fashion (Johannes' talk):

$$B_{k+1} = f(A, B_k, C_k)$$
$$C_{k+1} = g(A, B_{k+1}, C_k)$$

- Approximate $A \approx Q_l Q_l^* A$ and $A^* \approx Q_r Q_r^* A^*$ $(Q_l \in \mathbb{R}^{m\times(r+s)}, Q_r \in \mathbb{R}^{n\times(r+s)})$
- Proceed in with the same iterative scheme (but in lower dim space):

$$B_{k+1} = f(AQ_r, B_k, C_k Q_r)$$
$$C_{k+1} = g(Q_l^* A, Q_l^* B_{k+1}, C_k)$$

Application to calcium imaging dendritic data

Instead of operating with a $512^2 \times 5000$ matrix operate with two much smaller matrices $512^2 \times r$, $5000 \times r$ with $r \sim 100$. 10-100x speedup per iteration (similar convergence characteristics).

References

- Halko, N., Martinsson, P. G., and Tropp, J. A. (2011). Finding structure with randomness: Probabilistic algorithms for constructing approximate matrix decompositions. SIAM review, 53(2), 217-288. (material covered here)
- Woodruff, D. P. (2014). Sketching as a tool for numerical linear algebra. Foundations and Trends in Theoretical Computer Science, 10(1?2), 1-157.(Matrix sketching - not covered)
- Tepper, M., and Sapiro, G. (2016). Compressed nonnegative matrix factorization is fast and accurate. IEEE Transactions on Signal Processing, 64(9), 2269-2283. (NMF application)

One pass algorithms

It is possible to perform this task with a single pass (important when matrix does not fit in memory). Assume A is symmetric $n \times n$

1. Generate random matrix Ω
2. Compute $Y = A\Omega$
3. Find orthonormal Q such that $Y \approx QQ^*Y$.
4. Solve $Q^*Y = T(Q^*\Omega)$ with respect to T.
5. Perform eigenvalue decomposition: $T = \hat{U}D\hat{U}^*$.
6. Form $U = Q\hat{U}$
7. Then $A \approx UDU^*$.