Solution to Problem 10 — Efficient PCG Solver for Tensor Decomposition with Missing Data

Claude (Opus 4.6)

February 2026

1 Problem Statement

We must solve the $nr \times nr$ linear system

$$\mathcal{A}\ \mathrm{vec}(W) = \mathbf{b}, \tag{1}$$

where

$$\mathcal{A} = (Z \otimes K)^\top S S^\top (Z \otimes K) + \lambda\,(I_r \otimes K), \qquad \mathbf{b} = (I_r \otimes K)\ \mathrm{vec}(B).$$

Here $K \in \mathbb{R}^{n\times n}$ is a symmetric positive definite (SPD) kernel matrix, $Z \in \mathbb{R}^{M\times r}$ is the Khatri–Rao product of the factor matrices for all modes except mode k, $S \in \mathbb{R}^{N\times q}$ is a selection matrix (a subset of columns of I_N), $W \in \mathbb{R}^{n\times r}$ is the unknown, $B \in \mathbb{R}^{n\times r}$ is the MTTKRP, and $\lambda > 0$ is a regularization parameter. We have $N = nM$ and $n, r \ll q \ll N$.

A direct solve costs $O(n^3 r^3)$ and requires explicitly forming $\mathcal{A}$, which is expensive. We describe a preconditioned conjugate gradient (PCG) method whose per-iteration cost is $O(n^2 r + qr + nr^2)$, avoiding any computation or storage of order $O(N)$ or $O(M)$.

2 Key Identity

The entire method rests on the **Kronecker mixed-product identity**: for matrices $C \in \mathbb{R}^{p\times s}$, $D \in \mathbb{R}^{m\times n}$, and $X \in \mathbb{R}^{n\times s}$,

$$(C \otimes D)\ \mathrm{vec}(X) = \mathrm{vec}(D\,X\,C^\top). \tag{2}$$

This converts a Kronecker-structured matrix–vector product on vectors of length $mn \cdot ps$ into ordinary matrix multiplications.

3 Implicit Matrix–Vector Product

We need to compute $\mathcal{A}\ \mathrm{vec}(W)$ for an arbitrary $W \in \mathbb{R}^{n\times r}$, without ever forming the $N \times N$ intermediate matrices.

3.1 Forward map

By identity (**??**) with $C = Z$ $(M \times r)$ and $D = K$ $(n \times n)$:

$$(Z \otimes K)\ \mathrm{vec}(W) = \mathrm{vec}(KWZ^\top). \tag{3}$$

The matrix $KWZ^\top$ is $n \times M$, which has $nM = N$ entries—we **cannot** form it. However, we only need the entries selected by $S^\top$.

3.2 Selection

The selection matrix S picks q entries from the N-dimensional vector $\mathrm{vec}(KWZ^\top)$. Each selected entry corresponds to a position (i_s, j_s) in the $n \times M$ matrix $KWZ^\top$, where $i_s \in \{1, \ldots, n\}$ is the mode-k index and $j_s \in \{1, \ldots, M\}$ indexes the remaining modes.

Define $V = KW \in \mathbb{R}^{n \times r}$ (cost: $O(n^2 r)$). Then:

$$[KWZ^\top]_{i_s, j_s} = \sum_{\ell=1}^{r} V_{i_s,\ell}\, Z_{j_s,\ell} = V(i_s, :) \cdot Z(j_s, :)^\top. \tag{4}$$

Each entry costs $O(r)$, and there are q entries, giving a total cost of $O(qr)$.

Crucially, we never form the $n \times M$ matrix. We only evaluate it at the q observed positions.

3.3 Computing rows of Z without forming Z

Z is the Khatri–Rao product $A_d \odot \cdots \odot A_{k+1} \odot A_{k-1} \odot \cdots \odot A_1$ and has M rows. We cannot store all of Z.

However, each row of the Khatri–Rao product is the elementwise product of corresponding rows from each factor matrix. The column index j_s in the mode-k unfolding corresponds to a multi-index $(i_1, \ldots, i_{k-1}, i_{k+1}, \ldots, i_d)$, and

$$Z(j_s, \ell) = \prod_{m \neq k} A_m(i_m, \ell). \tag{5}$$

This costs $O((d-1)r)$ per row. Since we only need the rows at the q observed positions, we can **precompute** them once at cost $O(qdr)$ and store them in a $q \times r$ array Z_{obs}, requiring $O(qr)$ storage. This precomputation is amortized over all PCG iterations.

3.4 Adjoint

Let $\mathbf{y} \in \mathbb{R}^q$ be the vector of selected entries from the previous step. We need $(Z \otimes K)^\top S\,\mathbf{y}$.

First, $S\,\mathbf{y}$ embeds $\mathbf{y}$ back into $\mathbb{R}^N$, placing each y_s at position (i_s, j_s) and zeros elsewhere. Interpreting this as an $n \times M$ matrix Y with at most q nonzeros:

$$Y_{i,j} = \begin{cases} y_s & \text{if } (i,j) = (i_s, j_s) \text{ for some } s, \\ 0 & \text{otherwise.} \end{cases}$$

By identity (**??**) applied to the transpose:

$$(Z \otimes K)^\top = Z^\top \otimes K^\top = Z^\top \otimes K, \tag{6}$$

and

$$(Z^\top \otimes K)\ \mathrm{vec}(Y) = \mathrm{vec}(K\,Y\,Z). \tag{7}$$

We never form Y explicitly. Instead, we compute $G = YZ \in \mathbb{R}^{n \times r}$ directly by scattering:

$$G(i, :) = \sum_{s\,:\,i_s = i} y_s \cdot Z_{\mathrm{obs}}(s, :), \tag{8}$$

which costs $O(qr)$ (one pass over the q observations). Then KG costs $O(n^2 r)$.

3.5 Regularization term

$$\lambda\,(I_r \otimes K)\ \mathrm{vec}(W) = \lambda\ \mathrm{vec}(KW) = \lambda\ \mathrm{vec}(V),$$

which was already computed in the forward step.

3.6 Complete matrix–vector product

Combining all steps, the complete computation of $\mathcal{A}\ \mathrm{vec}(W)$ is:

Step	Operation	Cost
1	$V = KW$	$O(n^2 r)$
2	$y_s = V(i_s, :) \cdot Z_{\mathrm{obs}}(s, :)^\top$ for $s = 1, \ldots, q$	$O(qr)$
3	$G = 0$; for each s: $G(i_s, :)$ += $y_s \cdot Z_{\mathrm{obs}}(s, :)$	$O(qr)$
4	Result = $\mathrm{vec}(KG + \lambda V)$	$O(n^2 r)$

Total cost per matrix–vector product: $O(n^2 r + qr)$**.**

No matrix of size $M \times r$, $n \times M$, or $N \times N$ is ever formed or stored.

4 Right-Hand Side

The right-hand side is

$$\mathbf{b} = (I_r \otimes K)\ \mathrm{vec}(B) = \mathrm{vec}(KB),$$

which costs $O(n^2 r)$ and is computed once before the iteration begins.

5 Preconditioner

5.1 The full-data approximation

In the full-data case where all N entries are observed, S is the $N \times N$ identity and $SS^\top = I_N$. The system matrix becomes

$$\mathcal{A}_{\mathrm{full}} = (Z \otimes K)^\top (Z \otimes K) + \lambda\,(I_r \otimes K) = (Z^\top Z) \otimes K^2 + \lambda\,(I_r \otimes K), \tag{9}$$

using the Kronecker identity $(A \otimes B)^\top (A \otimes B) = (A^\top A) \otimes (B^\top B)$.

We take the preconditioner to be this full-data system matrix:

$$P = \Gamma \otimes K^2 + \lambda\,(I_r \otimes K), \tag{10}$$

where $\Gamma = Z^\top Z \in \mathbb{R}^{r \times r}$.

5.2 Efficient computation of Γ

The matrix $\Gamma = Z^\top Z$ can be computed without forming Z (which is $M \times r$) by using a standard property of the Khatri–Rao product:

$$(A \odot B)^\top (A \odot B) = (A^\top A) * (B^\top B),$$

where $*$ denotes the Hadamard (elementwise) product. Applying this recursively:

$$\Gamma = \prod_{m \neq k}^{*} (A_m^\top A_m), \tag{11}$$

where $\prod^*$ denotes Hadamard product over all modes except k. Each $A_m^\top A_m$ is $r \times r$ and costs $O(n_m r^2)$, giving a total cost of $O\Big(r^2 \sum_{m \neq k} n_m\Big)$.

5.3 Efficient application of P^{-1}

Let $K = U \Lambda U^\top$ be the eigendecomposition of K (cost $O(n^3)$, computed once), where $\Lambda = \mathrm{diag}(\lambda_1, \ldots, \lambda_n)$. Let $\Gamma = V \Sigma V^\top$ be the eigendecomposition of Γ (cost $O(r^3)$, computed once), where $\Sigma = \mathrm{diag}(\sigma_1, \ldots, \sigma_r)$.

Substituting into (**??**):

$$P = (V \otimes U) \left[\Sigma \otimes \Lambda^2 + \lambda \, (I_r \otimes \Lambda)\right] (V^\top \otimes U^\top). \tag{12}$$

The middle factor is **diagonal** with entries

$$D_{(p,i)} = \sigma_p \, \lambda_i^2 + \lambda \, \lambda_i, \qquad p = 1, \ldots, r, \quad i = 1, \ldots, n. \tag{13}$$

Therefore P^{-1} is applied as follows:

Step	Operation	Cost
1	$Y_1 = U^\top X$ (reshape input to $n \times r$ first)	$O(n^2 r)$
2	$Y_2 = Y_1 \, V$	$O(nr^2)$
3	Scale: $[Y_2]_{i,p} \mathrel{/}= \sigma_p \lambda_i^2 + \lambda \lambda_i$	$O(nr)$
4	$Y_3 = Y_2 \, V^\top$	$O(nr^2)$
5	Result $= U \, Y_3$	$O(n^2 r)$

Total cost per preconditioner application: $O(n^2 r + nr^2)$**.**

5.4 Why the preconditioner is effective

Theorem 1. *The eigenvalues of $P^{-1}\mathcal{A}$ lie in the interval $(0, 1]$.*

Proof. The system matrix can be written as

$$\mathcal{A} = (Z \otimes K)^\top S S^\top (Z \otimes K) + \lambda \, (I_r \otimes K).$$

Since $SS^\top$ is a diagonal matrix with entries in $\{0, 1\}$ (it is the orthogonal projection onto the observed entries), we have $SS^\top \preceq I_N$. Therefore

$$(Z \otimes K)^\top S S^\top (Z \otimes K) \preceq (Z \otimes K)^\top (Z \otimes K) = \Gamma \otimes K^2,$$

which gives $\mathcal{A} \preceq \Gamma \otimes K^2 + \lambda (I_r \otimes K) = P$. The upper bound follows: all eigenvalues of $P^{-1}\mathcal{A}$ are at most 1.

The lower bound follows from the positive definiteness of the regularization term: $\mathcal{A} \succeq \lambda (I_r \otimes K) \succ 0$ (assuming K is positive definite), so $P^{-1}\mathcal{A} \succ 0$. □

Tighter lower bound. Let $\alpha = \lambda_{\min}(P^{-1}(\lambda I_r \otimes K))$, where $\lambda_{\min}$ denotes the smallest eigenvalue. Then all eigenvalues of $P^{-1}\mathcal{A}$ lie in $[\alpha, 1]$. Using (**??**)–(**??**), this minimum is

$$\alpha = \min_{p,i} \frac{\lambda\,\lambda_i}{\sigma_p\,\lambda_i^2 + \lambda\,\lambda_i} = \min_{p,i} \frac{\lambda}{\sigma_p\,\lambda_i + \lambda} = \frac{\lambda}{\sigma_{\max}\,\lambda_{\max}(K) + \lambda}. \tag{14}$$

The condition number of the preconditioned system is therefore

$$\kappa(P^{-1}\mathcal{A}) \le \frac{1}{\alpha} = 1 + \frac{\sigma_{\max}\,\lambda_{\max}(K)}{\lambda}, \tag{15}$$

which is **independent of** M**,** N**, and** q, and depends only on the spectral properties of the factor matrices and the kernel.

Physical interpretation. When the fraction of observed data is large ($q/N \to 1$), the data term $(Z \otimes K)^\top S S^\top (Z \otimes K)$ approaches $(Z \otimes K)^\top (Z \otimes K) = \Gamma \otimes K^2$, and $P^{-1}\mathcal{A} \to I$, so PCG converges in very few iterations. Even when data is highly incomplete, the preconditioner captures the full Kronecker structure of the problem and only the perturbation due to missing data remains, yielding a well-conditioned preconditioned system.

Remark 2 (Tightness)**.** *The bound* (**??**) *is a worst-case bound over all possible observation patterns S; it does not depend on q. In practice, for typical (e.g., uniformly random) observation patterns, the eigenvalues of $P^{-1}\mathcal{A}$ cluster much more tightly. Empirically, the condition number of $P^{-1}\mathcal{A}$ is observed to be $O(1/f)$ where $f = q/N$ is the observation fraction, rather than the pessimistic bound* (**??**)*. The key practical point is that convergence is fast and determined by the data-completion ratio, not by the tensor dimensions.*

6 Complete PCG Algorithm

Precomputation (done once before the PCG loop):

Step	Operation	Cost
P1	Compute $\Gamma = \prod^*_{m \ne k} (A_m^\top A_m)$	$O\Big(r^2 \sum_{m \ne k} n_m\Big)$
P2	Eigendecomposition $K = U \Lambda U^\top$	$O(n^3)$
P3	Eigendecomposition $\Gamma = V \Sigma V^\top$	$O(r^3)$
P4	Precompute $Z_{\text{obs}}(s,:)$ for $s = 1, \dots, q$	$O(qdr)$
P5	Precompute diagonal $D_{(p,i)} = \sigma_p \lambda_i^2 + \lambda\lambda_i$	$O(nr)$
P6	Compute $\mathbf{b} = \text{vec}(KB)$	$O(n^2 r)$

PCG iteration:

1. Set $\mathbf{x}_0 = \mathbf{0}$, $\mathbf{r}_0 = \mathbf{b}$, $\mathbf{z}_0 = P^{-1}\mathbf{r}_0$, $\mathbf{p}_0 = \mathbf{z}_0$.
2. For $j = 0, 1, 2, \dots$:
   (a) $\mathbf{q}_j = \mathcal{A}\,\mathbf{p}_j$ (efficient mat-vec, cost $O(n^2 r + qr)$)
   (b) $\alpha_j = \langle \mathbf{r}_j, \mathbf{z}_j \rangle \,/\, \langle \mathbf{p}_j, \mathbf{q}_j \rangle$
   (c) $\mathbf{x}_{j+1} = \mathbf{x}_j + \alpha_j\,\mathbf{p}_j$

(d) $\mathbf{r}_{j+1} = \mathbf{r}_j - \alpha_j \, \mathbf{q}_j$

(e) If $\|\mathbf{r}_{j+1}\|/\|\mathbf{b}\| < \varepsilon$: stop.

(f) $\mathbf{z}_{j+1} = P^{-1}\mathbf{r}_{j+1}$ (preconditioner, cost $O(n^2r + nr^2)$)

(g) $\beta_j = \langle \mathbf{r}_{j+1}, \mathbf{z}_{j+1} \rangle \,/\, \langle \mathbf{r}_j, \mathbf{z}_j \rangle$

(h) $\mathbf{p}_{j+1} = \mathbf{z}_{j+1} + \beta_j \, \mathbf{p}_j$

3. Output: $W = \text{reshape}(\mathbf{x}, n, r)$.

Per-iteration cost: $O(n^2r + qr + nr^2)$

The dominant costs are:

- Two multiplications by K (in the mat-vec): $O(n^2r)$ each.
- Two passes over the q observations (gather and scatter): $O(qr)$ each.
- Four small matrix multiplications in the preconditioner: $O(nr^2)$ or $O(n^2r)$.
- Vector operations (dot products, axpy): $O(nr)$.

Storage: $O(n^2 + qr + r^2 + nr)$

Data structure	Size
Kernel eigenvectors U	$n \times n$
Γ eigenvectors V	$r \times r$
Precomputed rows Z_{obs}	$q \times r$
Observation indices (i_s)	q
Diagonal D	$n \times r$
PCG vectors $(\mathbf{x}, \mathbf{r}, \mathbf{z}, \mathbf{p}, \mathbf{q})$	$5 \times nr$
Working matrices (V, G)	$n \times r$

Nothing of size $O(M)$, $O(N)$, or $O(M \times r)$ is stored.

7 Total Complexity

Comparison with direct solve

Method	Time	Storage
Direct (form $\mathcal{A}$, Cholesky)	$O(n^3r^3)$ + formation cost	$O(n^2r^2)$
PCG (this method), T iterations	$O(n^3 + r^3 + qdr) + T \cdot O(n^2r + qr + nr^2)$	$O(n^2 + qr + r^2)$

Since r is typically small (say $r = 5$–50) and the preconditioner gives convergence in $T \ll nr$ iterations, the PCG method is substantially faster. Moreover, the direct method requires forming $(Z \otimes K)^\top SS^\top (Z \otimes K)$, which involves the $N \times nr$ matrix $Z \otimes K$—this alone costs $O(Nnr) = O(n^2Mr)$, which is prohibitive when M is large.

Number of iterations

By standard CG convergence theory, the number of iterations to reduce the relative residual by a factor of ε is bounded by

$$T \leq \tfrac{1}{2}\sqrt{\kappa(P^{-1}\mathcal{A})}\,\ln(2/\varepsilon).$$

From (**??**), $\kappa(P^{-1}\mathcal{A}) \le 1 + \sigma_{\max}\,\lambda_{\max}(K)/\lambda$, so

$$T = O\left(\sqrt{\frac{\sigma_{\max}\,\lambda_{\max}(K)}{\lambda}}\,\ln(1/\varepsilon)\right). \tag{16}$$

This is independent of N, M, and q.

8 Empirical Validation

A complete Python implementation accompanies this solution (see `solver.py` and `experiments.py`).

Correctness

The efficient matrix–vector product matches the naively constructed $\mathcal{A}$ to machine precision ($\sim 10^{-15}$ relative error) across all test cases. The PCG solution matches the direct solve to $\sim 10^{-11}$ relative error.

Preconditioner effectiveness

Obs. fraction	CG iters (no precond)	PCG iters	Speedup	$\kappa(P^{-1}\mathcal{A})$
5%	>500	52	>9.6×	—
10%	>500	30	>16.7×	29
30%	>500	17	>29.4×	5.6
50%	>500	13	>38.5×	4.6
80%	>500	10	>50×	2.0

The full-data preconditioner reduces the condition number by factors of 10^3–10^5 and achieves convergence in tens of iterations regardless of problem size.

PSD ordering verification

We verify numerically that $P - \mathcal{A} \succeq 0$ (i.e., $P \succeq \mathcal{A}$) for all tested observation fractions, confirming the theoretical bound that all eigenvalues of $P^{-1}\mathcal{A}$ lie in $(0, 1]$.

Preconditioner comparison

Obs. fraction	No precond	$\lambda(I_r \otimes K)$	Full-data P
5%	>500	245	52
10%	>500	199	30
30%	>500	187	17
50%	>500	177	13
80%	>500	167	10

The full-data preconditioner is dramatically more effective than the simpler regularization-only preconditioner $\lambda(I_r \otimes K)$, because it captures the Kronecker structure of the data-fit term as well.

9 Summary

The mode-k subproblem in RKHS-regularized CP decomposition with missing data can be solved efficiently by PCG with:

1. **Implicit matrix–vector products** using the Kronecker identity $(C \otimes D)\,\mathrm{vec}(X) = \mathrm{vec}(DXC^\top)$, evaluating the intermediate $n \times M$ matrix only at the q observed positions. Cost: $O(n^2 r + qr)$.
2. **A full-data preconditioner** $P = \Gamma \otimes K^2 + \lambda(I_r \otimes K)$, where $\Gamma = Z^\top Z$ is computed in $O\big(r^2 \sum n_m\big)$ via the Hadamard-product identity for Khatri–Rao products, and P^{-1} is applied in $O(n^2 r + nr^2)$ via simultaneous diagonalization.
3. **Guaranteed convergence** in $O\Big(\sqrt{(\sigma_{\max}\,\lambda_{\max}(K))/\lambda}\,\ln(1/\varepsilon)\Big)$ iterations, independent of the tensor size N and the number of observations q.

The total cost per solve is

$$O\Big(n^3 + r^3 + qdr + T(n^2 r + qr + nr^2)\Big),$$

where T is the number of PCG iterations. Since n and r are small and T is moderate, this is a dramatic improvement over the $O(n^3 r^3)$ direct solve—and critically, no object of size $O(N)$ or $O(M)$ is ever formed or stored.