NONLINEARLY PRECONDITIONED OPTIMIZATION ON GRASSMANN MANIFOLDS FOR COMPUTING APPROXIMATE TUCKER TENSOR DECOMPOSITIONS

HANS DE STERCK* AND ALEXANDER HOWSE†

Abstract. Two accelerated optimization algorithms are presented for computing approximate Tucker tensor decompositions (ATDs). The first is a nonlinearly preconditioned conjugate gradient (NPCG) algorithm, wherein a nonlinear preconditioner generates a direction replacing the gradient in the nonlinear conjugate gradient iteration. The second is a nonlinear GMRES (N-GMRES) algorithm, in which a linear combination of iterates generated by a nonlinear preconditioner is minimized to produce an improved search direction. The Euclidean versions of these methods are extended to the manifold setting, where optimization on Grassmann manifolds is used to handle orthonormality constraints and to allow isolated minimizers. The higher order orthogonal iteration (HOOI), the workhorse algorithm for computing ATDs, is used as the nonlinear preconditioner in NPCG and N-GMRES. Four options are provided for the update parameter β in NPCG. Two strategies for approximating the Hessian operator applied to a vector are provided for N-GMRES. NPCG and N-GMRES are compared to HOOI, NCG, limited memory BFGS, and a manifold trust region algorithm using synthetic data and real life tensor data arising from handwritten digit recognition. Numerical results show that all four NPCG variants and N-GMRES using a difference of gradients Hessian approximation accelerate HOOI significantly for large tensors, noisy data, and when high accuracy results are required. For these problems, the proposed methods converge faster and more robustly than HOOI and the state-of-the-art methods considered.

1. Introduction. When vast quantities of information are collected for analysis, it is both useful and natural to organize these collections into multidimensional arrays, with each dimension corresponding to a component or feature of the data source and each element an observation for a particular configuration of these components or features. Such arrays are called *tensors*, and the number of *dimensions* (*modes*) is the tensor *order*. Tensors are used when large quantities of data need to be organized, stored, and analyzed, such as in data mining [21], numerical linear algebra [5], pattern and image recognition [26], and signal processing [9]. Many more examples are given in [20].

A *tensor decomposition* expresses a tensor as a sum and/or product of several components with the goal of simplifying further work involving the tensor data. *Tensor approximation problems* involve approximating a tensor $\mathcal{X}$ by a tensor $\widehat{\mathcal{X}}$ with a specified decomposition, often by minimizing $\mathcal{X}-\widehat{\mathcal{X}}$ in an appropriate norm. We consider *Tucker* decompositions, where $\mathcal{X} \in \mathbb{R}^{I_1 \times \cdots \times I_N}$ is expressed as a multilinear product (explained in Section 2) of a core tensor $\mathcal{S} \in \mathbb{R}^{R_1 \times \cdots \times R_N}$ and factor matrices $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n \times R_n}$: $\mathcal{X} = (\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) \cdot \mathcal{S}$. Of particular interest are the subset of approximate Tucker decompositions using orthonormal matrices $\mathbf{A}^{(n)}$, which introduce equality constraints into the defining optimization problem.

Two algorithms for solving this problem are discussed. The first is a nonlinearly preconditioned conjugate gradient (NPCG) method [31], in which a nonlinear preconditioner is used in place of the gradient to generate search directions in nonlinear conjugate gradient (NCG) iterations. The second is an acceleration technique called nonlinear GMRES (N-GMRES) [30, 31, 34], wherein a linear combination of past iterates, generated by a nonlinear preconditioner, is optimized to produce an improved search direction.

In this paper, these two methods are adapted to handle both equality constraints and objective function invariance properties causing non-isolated minima by using matrix manifold optimization strategies. Matrix manifold optimization for approximate Tucker decompositions has been considered before: Newton's method [15], the BFGS and limited memory BFGS quasi-Newton methods [27], NCG [18], and a trust-region method [19] have previously been developed.

We use the higher order orthogonal iteration (HOOI) [10], an alternating least squares type algorithm for Tucker tensor decomposition, as the nonlinear preconditioner. Convergence of HOOI can be slow, and we demonstrate numerically that it may be significantly accelerated by applying NPCG or N-GMRES acceleration on manifolds, resulting in robust and highly competitive methods. In particular, comparisons to HOOI [10], NCG [18], limited memory BFGS [27], and a trust-region method [19], each adapted to the matrix manifold context, show that the newly proposed methods offer significant advantages for problems with large and/or noisy source tensors.

The remainder of the paper is as follows. Section 2 covers preliminary tensor details required, including the Tucker tensor format and the standard approaches to solve this tensor approximation problem. Section 3 introduces matrix manifold optimization, which is applied to NPCG in Section 4 and N-GMRES in Section 5. Implementation details and numerical results are given in Section 6. A summary is provided in Section 7.

*School of Mathematical Sciences, Monash University, Melbourne, Australia (hans.de.sterck@monash.edu).
†Department of Applied Mathematics, University of Waterloo, Canada (ahowse@uwaterloo.ca).

2. Preliminaries. Throughout this paper, vectors are in bold lowercase ($\mathbf{x}$), matrices bold uppercase ($\mathbf{X}$), and tensors Euler script ($\mathcal{X}$). Elements are indicated by subscript or bracketed indices: $\mathcal{X}_{ijk} = \mathcal{X}(i,j,k)$.

2.1. Matrix Singular Value Decomposition. A matrix $\mathbf{M} \in \mathbb{R}^{m\times n}$ has singular value decomposition (SVD) $\mathbf{M} = \mathbf{U}\mathbf{\Sigma}\mathbf{V}^\intercal$, where $\mathbf{U} \in \mathbb{R}^{m\times m}$, $\mathbf{V} \in \mathbb{R}^{n\times n}$ are orthogonal and $\mathbf{\Sigma} \in \mathbb{R}^{m\times n}$ is diagonal with nonnegative real entries in decreasing order. The diagonal entries of $\mathbf{\Sigma}$ are the *singular values* of $\mathbf{M}$ and the columns of $\mathbf{U}$ ($\mathbf{V}$) are the *left-singular* (*right-singular*) vectors. The *rank* of $\mathbf{M}$ is equal to the number of nonzero singular values. By the Eckhart-Young theorem, the best rank-r approximation of $\mathbf{M}$ in the Frobenius norm is obtained by keeping the largest r singular values, setting the rest to zero [13].

2.2. Tensor Matricizations and Multilinear Rank. *Mode-n tensor fibers* are obtained by fixing all indices but the n^{th}. The *mode-n matricization* of $\mathcal{X}$, denoted $\mathbf{X}_{(n)}$, is obtained by taking the mode-n fibers of $\mathcal{X}$ as the columns of $\mathbf{X}_{(n)}$. In general the ordering of fibers does not matter, so long as it is consistent throughout calculations. The *n-rank* of $\mathcal{X}$ is the dimension of the vector space spanned by the mode-n fibers: $\text{rank}_n(\mathcal{X}) = \dim(\text{Col}(\mathbf{X}_{(n)}))$ [10]. The *multilinear rank* of $\mathcal{X}$ is the N-tuple $(\text{rank}_1(\mathcal{X}), \ldots, \text{rank}_N(\mathcal{X}))$.

2.3. Tensor Products and Norm. The *mode-n contravariant product* of $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ and $\mathbf{A} \in \mathbb{R}^{J\times I_n}$ is $\mathcal{Y} = (\mathbf{A})_n \cdot \mathcal{X}$ [15]: $\mathcal{Y}(i_1, \ldots, i_{n-1}, j, i_{n+1}, \ldots, i_n) = \sum_{i_n=1}^{I_n} \mathbf{A}(j, i_n)\mathcal{X}(i_1, \ldots, i_N)$. Each mode-$n$ fiber of $\mathcal{X}$ is multiplied by each row of $\mathbf{A}$. It follows that $\mathbf{Y}_{(n)} = \mathbf{A}\mathbf{X}_{(n)}$ and $(\mathbf{B})_n \cdot ((\mathbf{A})_n \cdot \mathcal{X}) = (\mathbf{B}\mathbf{A})_n \cdot \mathcal{X}$. The *mode-n covariant product* of $\mathcal{X}$ and $\mathbf{A} \in \mathbb{R}^{I_n\times J}$ is $\mathcal{Y} = \mathcal{X} \cdot (\mathbf{A})_n$ [15], where $\mathcal{Y}(i_1, \ldots, i_{n-1}, j, i_{n+1}, \ldots, i_n) = \sum_{i_n=1}^{I_n} \mathcal{X}(i_1, \ldots, i_N)\mathbf{A}(i_n, j)$. Clearly, $(\mathbf{A}^\intercal)_n \cdot \mathcal{X} = \mathcal{X} \cdot (\mathbf{A})_n$. Multiplication in different modes is commutative.

The *inner product* of $\mathcal{X}, \mathcal{Y} \in \mathbb{R}^{I_1\times\cdots\times I_n}$ is $\langle \mathcal{X}, \mathcal{Y}\rangle = \sum_{i_1=1}^{I_1} \cdots \sum_{i_N=1}^{I_N} \mathcal{X}(i_1, , \ldots, i_N)\mathcal{Y}(i_1, \ldots, i_N)$. The tensor Frobenius norm is $\|\mathcal{X}\|_F = \sqrt{\langle \mathcal{X}, \mathcal{X}\rangle}$. $\|\mathcal{X}\|_F = \left\|\mathbf{X}_{(n)}\right\|_F$ for all n, and $\|\mathcal{X}\|_F$ is invariant under orthogonal transformations $\mathbf{A}^{(n)}$: $\|\mathcal{X}\|_F = \left\|(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) \cdot \mathcal{X}\right\|_F$. Finally, the elementwise product of equal size tensors is denoted by $\mathcal{X} * \mathcal{Y}$.

2.4. The Tucker Tensor Format and the HOSVD. The Tucker format decomposition of a tensor $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ is $(\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) \cdot \mathcal{S}$, where $\mathcal{S} \in \mathbb{R}^{R_1\times\cdots\times R_N}$ and $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n\times R_n}$. If $R_n \geq \text{rank}_n(\mathcal{X})$ for all n, the decomposition is exact; otherwise, this is an approximate Tucker decomposition (ATD). A variant of the Tucker decomposition called the higher order SVD (HOSVD) was introduced in [10], which established that all tensors have such a decomposition. The HOSVD of $\mathcal{X} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ is $\mathcal{X} = (\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) \cdot \mathcal{S}$, where each $\mathbf{A}^{(n)} \in \mathbb{R}^{I_n\times I_n}$ is orthogonal and $\mathcal{S} \in \mathbb{R}^{I_1\times\cdots\times I_N}$ satisfies

(i) all-orthogonality: for all possible n, α, and β, $\alpha \neq \beta$: $\langle \mathcal{S}_{i_n=\alpha}, \mathcal{S}_{i_n=\beta}\rangle = 0$;
(ii) the ordering: $\|\mathcal{S}_{i_n=1}\|_F \geq \|\mathcal{S}_{i_n=2}\|_F \geq \cdots \geq \|\mathcal{S}_{i_n=I_n}\|_F \geq 0$ for all n.

Given a multilinear rank $(R_1, \ldots, R_N)$, a truncated HOSVD may be computed in which $\mathbf{A}^{(n)}$ contains only R_n orthonormal columns. This procedure is described in Algorithm 1 [20].

Algorithm 1 HOSVD

```
procedure HOSVD(X, R_1, ..., R_N)
    for n = 1, ..., N do
        A^(n) ← R_n leading left singular vectors of X_(n)
    end for
    S ← X · (A^(1), ..., A^(N))
    return S, A^(1), ..., A^(N)
end procedure
```

2.5. The Best Tucker Approximation Problem. While Algorithm 1 can be used to compute a HOSVD ATD with an arbitrary multilinear rank, truncating a HOSVD to a smaller multilinear rank does not give an optimal approximation [10]. To determine the best orthonormal ATD of a given $\mathcal{X}$, we minimize the approximation error in the Frobenius norm:

$$\begin{aligned} \min_{\mathcal{S},\{\mathbf{A}^{(n)}\}} \quad & \frac{1}{2}\left\|\mathcal{X} - (\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) \cdot \mathcal{S}\right\|_F^2 \\ \text{subject to} \quad & \mathcal{S} \in \mathbb{R}^{R_1\times\cdots\times R_N},\ \mathbf{A}^{(n)} \in \mathbb{R}^{I_n\times R_n} \text{ and } \mathbf{A}^{(n)\intercal}\mathbf{A}^{(n)} = \mathbf{I}_{R_n}. \end{aligned}$$

This can be shown to be equivalent [11] to the maximization problem

$$\begin{aligned} \max_{\{\mathbf{A}^{(n)}\}} \quad & \frac{1}{2}\left\|\mathcal{X}\cdot(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)})\right\|_F^2 \\ \text{subject to} \quad & \mathbf{A}^{(n)}\in\mathbb{R}^{I_n\times R_n} \text{ and } \mathbf{A}^{(n)\intercal}\mathbf{A}^{(n)}=\mathbf{I}_{R_n}, \end{aligned} \tag{2.1}$$

where $\mathcal{S}=\mathcal{X}\cdot\left(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}\right)$.

The most popular method for solving (2.1) is the higher-order orthogonal iteration (HOOI) [11], reproduced here as Algorithm 2 [20]. While simple to implement, HOOI may be slow to converge in practice, hence alternative optimization methods are desired. The maximization problem (2.1) may be treated using matrix manifold optimization strategies, as discussed in the next section.

Algorithm 2 HOOI

procedure HOOI($\mathcal{X}$,$R_1,\ldots,R_N$)
  initialize $\mathbf{A}^{(n)}\in\mathbb{R}^{I_n\times R_n}$ for $n=1,\ldots,N$ using HOSVD
  repeat
    for $n=1,\ldots,N$ **do**
      $\mathcal{Y}\leftarrow\mathcal{X}\cdot(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(n-1)},\mathbf{I},\mathbf{A}^{(n+1)},\ldots,\mathbf{A}^{(N)})$
      $\mathbf{A}^{(n)}\leftarrow R_n$ leading left singular vectors of $\mathbf{Y}_{(n)}$
    end for
  until termination condition satisfied
  $\mathcal{S}\leftarrow\mathcal{X}\cdot(\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)})$
  return $\mathcal{S},\mathbf{A}^{(1)},\ldots,\mathbf{A}^{(N)}$
end procedure

3. Matrix Manifold Concepts for NPCG and N-GMRES Acceleration. We now introduce the concepts from matrix manifold optimization used to adapt NPCG and N-GMRES to the manifold setting. This section covers the technical elements of our approach common to both methods. Sections 4 and 5 apply these specifically to NPCG and N-GMRES, respectively. As a general reference for this section, see [2].

We restrict our discussion to *Riemannian manifolds*, which have smoothly varying inner products. The *Stiefel manifold*, $\mathrm{St}(n,p)$, is the set of all $n\times p$ orthonormal matrices, $\{\mathbf{X}\in\mathbb{R}^{n\times p}|\mathbf{X}^\intercal\mathbf{X}=\mathbf{I}_p\}$. The *Grassmann manifold* (*Grassmannian*), $\mathrm{Gr}(n,p)$, is the set of p-dimensional linear subspaces of $\mathbb{R}^n$ [14]. In both contexts $p\leq n$. Each $\mathcal{Y}\in\mathrm{Gr}(n,p)$ can be represented as the column space of some $\mathbf{Y}\in\mathrm{St}(n,p)$. There is no unique representative $\mathbf{Y}$: the subset of $\mathrm{St}(n,p)$ with the same column space as $\mathbf{Y}$ is $\mathbf{Y}O_p:=\{\mathbf{YM}|\mathbf{M}\in O_p\}$, where O_p is the set of $p\times p$ orthogonal matrices. $\mathrm{Gr}(n,p)$ is thus identified with the set of matrix equivalence classes $\mathrm{St}(n,p)/O_p:=\{\mathbf{Y}O_p|\mathbf{Y}^\intercal\mathbf{Y}=\mathbf{I}_p\}$, which is induced by the equivalence relation $\mathbf{X}\sim\mathbf{Y}$ if and only if $\mathrm{Col}(\mathbf{X})=\mathrm{Col}(\mathbf{Y})$. The inner product for $\mathrm{St}(n,p)$, and hence $\mathrm{Gr}(n,p)$, is $\langle\mathbf{X},\mathbf{Y}\rangle=\mathrm{tr}(\mathbf{X}^\intercal\mathbf{Y})$.

Let $\mathcal{M}$ denote an arbitrary manifold. A *tangent vector* at $x\in\mathcal{M}$, denoted ξ_x, describes a possible direction of travel tangent to $\mathcal{M}$ at x. The vector space of all tangent vectors at x is the *tangent space*, $T_x\mathcal{M}$. A tangent vector at $\mathbf{Y}\in\mathrm{St}(n,p)$ is itself an $n\times p$ matrix. Just as $\mathbf{Y}\in\mathrm{St}(n,p)$ can represent $\mathcal{Y}\in\mathrm{Gr}(n,p)$, we can use elements of $T_\mathbf{Y}\mathrm{St}(n,p)$ to represent elements of $T_\mathbf{Y}\mathrm{Gr}(n,p)$. $T_\mathbf{Y}\mathrm{St}(n,p)$ may be expressed as the direct sum of a *vertical space* $\mathcal{V}_\mathbf{Y}$, which contains directions for movement within the equivalence class $\mathcal{Y}$; and a *horizontal space* $\mathcal{H}_\mathbf{Y}$, which contains directions for movement into a new equivalence class. Elements of $\mathcal{H}_\mathbf{Y}$ are used as unique representative tangent vectors for points on $\mathrm{Gr}(n,p)$. Specifically, $\mathcal{V}_\mathbf{Y}=\{\mathbf{YM}|\mathbf{M}=-\mathbf{M}^\intercal,\mathbf{M}\in\mathbb{R}^{p\times p}\}$ and $\mathcal{H}_\mathbf{Y}=\{\mathbf{Z}\in\mathbb{R}^{n\times p}|\mathbf{Y}^\intercal\mathbf{Z}=\mathbf{0}_p\}$, and

$$\Pi_\mathbf{Y}=\mathbf{I}-\mathbf{YY}^\intercal \tag{3.1}$$

is the orthogonal projection onto $\mathcal{H}_\mathbf{Y}$ [2, 14].

To move along $\mathcal{M}$ in the direction of ξ, one uses a mapping from $T\mathcal{M}$ to $\mathcal{M}$ known as a *retraction*. On $\mathrm{Gr}(n,p)$ we use the retraction

$$R_\mathbf{Y}(t\xi)=\mathrm{qf}(\mathbf{Y}+t\xi), \tag{3.2}$$

where $\mathrm{qf}(\mathbf{Z})$ is the $\mathbf{Q}$ factor of the thin $\mathbf{QR}$ decomposition of $\mathbf{Z}$ [2].

The direction of travel from x to y cannot be described by a vector $y - x$: this operation is not defined. A tangent vector defining the direction from x to y can be found by the *logarithmic map*. The tangent vector in $T_{\mathbf{X}}\mathrm{Gr}(n,p)$ pointing from $\mathbf{X}$ to $\mathbf{Y}$ is

$$\mathrm{Log}_{\mathbf{X}}(\mathbf{Y}) = \mathbf{U} \arctan(\mathbf{\Sigma})\mathbf{V}^\intercal, \tag{3.3}$$

where $\mathbf{U\Sigma V}^\intercal$ is the compact SVD of $\Pi_{\mathbf{X}}\mathbf{Y}(\mathbf{X}^\intercal\mathbf{Y})^{-1}$ [28]. Furthermore, tangent spaces $T_x\mathcal{M}$ and $T_y\mathcal{M}$ for $x \neq y$ are generally different vector spaces, hence linear combinations of $\xi \in T_x\mathcal{M}$ and $\eta \in T_y\mathcal{M}$ are not well defined. By using a *vector transport* mapping, we can find a $\xi' \in T_y\mathcal{M}$ which corresponds to ξ. Given $x, y \in \mathrm{Gr}(n,p)$ and $\xi \in T_x\mathrm{Gr}(n,p)$, a corresponding vector in $T_{\mathbf{Y}}\mathrm{Gr}(n,p)$ is determined using (3.1) [2]:

$$\mathcal{T}_{\mathbf{Y}}(\xi) = \Pi_{\mathbf{Y}}\xi. \tag{3.4}$$

For a manifold $\mathcal{M} = \prod_{k=1}^{N} \mathrm{Gr}(n_k, p_k)$, a Cartesian product of N Grassmannians, elements are N-tuples of linear subspaces $y = (\mathcal{Y}_1, \ldots, \mathcal{Y}_N)^\intercal$, in turn represented by N-tuples of matrices $\mathbf{y} = (\mathbf{Y}_1, \ldots, \mathbf{Y}_N)^\intercal$. The tangent space at $y \in \mathcal{M}$ is the Cartesian product of tangent spaces $T_{\mathbf{Y}_k}\mathrm{Gr}(n_k, p_k)$. The inner product on $\mathcal{M}$ is $\langle \mathbf{x}, \mathbf{y} \rangle = \sum_{k=1}^{N} \langle \mathbf{X}_k, \mathbf{Y}_k \rangle$. All other required operations are performed componentwise on $x, y \in \mathcal{M}$, using the operations defined for $\mathrm{Gr}(n_k, p_k)$.

3.1. Application to the Best ATD Problem. To solve (2.1), two points must be addressed. First, (2.1) does not have isolated maxima, as $\|\cdot\|_F$ is invariant under orthogonal transformations. Second, orthonormality of $(\mathbf{A}^{(n)})_{n=1}^{N}$ introduces a large number of equality constraints. However, if we optimize over a Cartesian product of Grassmannians, representative N-tuples of matrices from the product of Stiefel manifolds will satisfy the orthonormality constraints by definition. Furthermore, as these matrices now represent equivalence classes of matrices, we now have an unconstrained problem with isolated extrema:

$$\max_{\{\mathbf{A}^{(n)}\}} \quad \frac{1}{2} \left\| \mathcal{X} \cdot (\mathbf{A}^{(1)}, \ldots, \mathbf{A}^{(N)}) \right\|_F^2, \tag{3.5}$$

where $\mathbf{A}^{(n)} \in \mathrm{St}(I_n, R_n)$ represents $\mathcal{A}^{(n)} \in \mathrm{Gr}(I_n, R_n)$.

The general optimization strategy is to determine an ascent tangent vector for (3.5) at the current point, then carry out a linesearch along a retraction curve in the direction of this tangent vector. The Riemannian gradient and Hessian of the objective function may be required, which can be obtained from their Euclidean equivalents. We refer readers to [15, 19], where these expressions can be found.

4. Nonlinearly Preconditioned Conjugate Gradients on Grassmann Manifolds. We first explore the Euclidean NPCG algorithm, and then discuss how to extend it to a matrix manifold setting. By choosing HOOI as the nonlinear preconditioner, we obtain a complete algorithm for the ATD problem.

4.1. Nonlinear Conjugate Gradients. The conjugate gradient (CG) method is an iteration which minimizes the convex quadratic $f(\mathbf{x}) = \frac{1}{2}\mathbf{x}^\intercal\mathbf{A}\mathbf{x} - \mathbf{b}^\intercal\mathbf{x}$, where $\mathbf{A}$ is symmetric positive definite. Nonlinear conjugate gradients (NCG) is an adaptation of CG to minimize nonlinear objective functions $f(\mathbf{x})$ [24]:

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, & \mathbf{g}_k &= \nabla f(\mathbf{x_k}) \\ \mathbf{p}_{k+1} &= -\mathbf{g}_{k+1} + \beta_k \mathbf{p}_k, & \mathbf{p}_0 &= \mathbf{g}_0. \end{aligned} \tag{4.1}$$

The search direction update parameter β_k, which determines the relative weighting of the current gradient, $\mathbf{g}_{k+1}$, and the previous search direction, $\mathbf{p}_k$, may be determined by one of several possible formulae. Letting $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathbf{g}_k$, we consider the Polak-Ribière [25] and Hestenes-Stiefel [17] β formulae

$$\beta_{k+1}^{\mathrm{PR}} = \frac{\mathbf{g}_{k+1}^\intercal \mathbf{y}_k}{\mathbf{g}_k^\intercal \mathbf{g}_k}, \qquad \beta_{k+1}^{\mathrm{HS}} = \frac{\mathbf{g}_{k+1}^\intercal \mathbf{y}_k}{\mathbf{p}_k^\intercal \mathbf{y}_k}. \tag{4.2}$$

4.2. Preconditioning. Linear preconditioning can be introduced by a change of variables $\mathbf{x} = \mathbf{Sz}$ [16]. Writing the algorithm for $\mathbf{z}$, then converting back to $\mathbf{x}$ gives

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k \\ \mathbf{p}_{k+1} &= \mathbf{P}\mathbf{g}_{k+1} + \overline{\beta}_k \mathbf{p}_k, \quad \mathbf{p}_0 = \mathbf{P}\mathbf{g}_0, \end{aligned} \tag{4.3}$$

where $\mathbf{P} = \mathbf{S}\mathbf{S}^\intercal$. The formulae for $\overline{\beta}_k$ remains the same, except now $\mathbf{g}_k$ and $\mathbf{p}_k$ are replaced by $\mathbf{S}^\intercal\mathbf{g}_k$ and $\mathbf{S}^{-1}\mathbf{p}_k$. We instead introduce a fully nonlinear preconditioning function P in place of a linear transformation with matrix $\mathbf{S}$. Let $\overline{\mathbf{x}}_k = P(\mathbf{x}_k)$ and define the direction generated by the preconditioner to be $\overline{\mathbf{g}}_k = \mathbf{x}_k - \overline{\mathbf{x}}_k$. Replacing $\mathbf{g}_k$ by $\overline{\mathbf{g}}_k$, we obtain the NPCG iteration [31]

$$\begin{aligned} \mathbf{x}_{k+1} &= \mathbf{x}_k + \alpha_k \mathbf{p}_k, \\ \mathbf{p}_{k+1} &= -\overline{\mathbf{g}}_{k+1} + \beta_{k+1}\mathbf{p}_k, \quad \mathbf{p}_0 = -\overline{\mathbf{g}}_0. \end{aligned} \tag{4.4}$$

The formulae for $\overline{\beta}_{k+1}$ may be adjusted by a straight replacement of $\mathbf{g}_{k+1}$ with $\overline{\mathbf{g}}_{k+1}$ everywhere in (4.2). We denote these choices by $\widetilde{\beta}$:

$$\widetilde{\beta}^{\mathrm{PR}}_{k+1} = \frac{\overline{\mathbf{g}}_{k+1}^\intercal \overline{\mathbf{y}}_k}{\overline{\mathbf{g}}_k^\intercal \overline{\mathbf{g}}_k}, \qquad \widetilde{\beta}^{\mathrm{HS}}_{k+1} = \frac{\overline{\mathbf{g}}_{k+1}^\intercal \overline{\mathbf{y}}_k}{\overline{\mathbf{y}}_k^\intercal \mathbf{p}_k}, \tag{4.5}$$

where $\overline{\mathbf{y}}_k = \overline{\mathbf{g}}_{k+1} - \overline{\mathbf{g}}_k$. A second possibility is to consider the linearly preconditioned analog $\overline{\beta}_{k+1}$, and replace the $\mathbf{P}\mathbf{g}_{k+1}$ terms with $\overline{\mathbf{g}}_{k+1}$, retaining the gradient elsewhere [31]. We denote these choices by $\widehat{\beta}$:

$$\widehat{\beta}^{\mathrm{PR}}_{k+1} = \frac{\mathbf{g}_{k+1}^\intercal \overline{\mathbf{y}}_k}{\mathbf{g}_k^\intercal \overline{\mathbf{g}}_k}, \qquad \widehat{\beta}^{\mathrm{HS}}_{k+1} = \frac{\mathbf{g}_{k+1}^\intercal \overline{\mathbf{y}}_k}{\mathbf{y}_k^\intercal \mathbf{p}_k}. \tag{4.6}$$

Algorithm 3 Nonlinearly Preconditioned Conjugate Gradients on Manifolds

procedure NPCG($\mathbf{x_0}$)
  $\overline{\mathbf{x}}_0 \leftarrow$ one iteration HOOI($\mathbf{x}_0$)
  $\overline{\mathbf{g}}_0 \leftarrow -\mathrm{Log}_{\mathbf{x}_0}(\overline{\mathbf{x}}_0)$
  $\mathbf{p}_0 \leftarrow -\overline{\mathbf{g}}_0$
  $k \leftarrow 0$
  while $\|\mathbf{g}_k\| >$ tol **do**
    Compute α_k
    $\mathbf{x}_{k+1} \leftarrow R_{\mathbf{x}_k}(\alpha_k \mathbf{p}_k)$
    Evaluate $\mathbf{g}_{k+1} = \mathrm{grad}\, f(\mathbf{x}_{k+1})$ (the Riemannian gradient)
    $\overline{\mathbf{x}}_{k+1} \leftarrow$ one iteration HOOI($\mathbf{x}_{k+1}$)
    $\overline{\mathbf{g}}_{k+1} \leftarrow -\mathrm{Log}_{\mathbf{x}_{k+1}}(\overline{\mathbf{x}}_{k+1})$
    Compute $\overline{\beta}_{k+1}$
    $\mathbf{p}_{k+1} \leftarrow -\overline{\mathbf{g}}_{k+1} + \overline{\beta}_{k+1}\mathcal{T}_{\mathbf{x}_{k+1}}(\mathbf{p}_k)$
    $k \leftarrow k+1$
  end while
end procedure

4.3. Adaptation of NPCG to Grassmann Context. Bold lowercase letters now represent n-tuples of matrices; e.g., $\mathbf{x}_k = (\mathbf{A}_k^{(1)}, \ldots, \mathbf{A}_k^{(n)})^\intercal$. We define P to be one iteration of the inner loop of the HOOI, so $\overline{\mathbf{x}}_k = P(\mathbf{x}_k)$ describes a point on the product manifold. Let $\overline{\mathbf{g}}_k = -\mathrm{Log}_{\mathbf{x}_k}(\overline{\mathbf{x}}_k)$ (see (3.3)), the negative of the tangent vector at $\mathbf{x}_k$ in the direction of $\overline{\mathbf{x}}_k$. The manifold NPCG iteration is

$$\begin{aligned} \mathbf{x}_{k+1} &= R_{\mathbf{x}_k}(\alpha \mathbf{p}_k), \\ \mathbf{p}_{k+1} &= -\overline{\mathbf{g}}_{k+1} + \overline{\beta}_{k+1}\mathcal{T}_{\mathbf{x}_{k+1}}(\mathbf{p}_k), \quad \mathbf{p}_0 = -\overline{\mathbf{g}}_0. \end{aligned} \tag{4.7}$$

A line search is now carried out along the curve defined by a retraction $R_{\mathbf{x}_k}(\cdot)$ from $\mathbf{x}_k$ (see (3.2)), and the formula for $\mathbf{p}_{k+1}$ requires vector transport of $\mathbf{p}_k$ using $\mathcal{T}_{\mathbf{x}_{k+1}}(\cdot)$ (see (3.4, 3.1)). The $\widetilde{\beta}$ formulae become

$$\widetilde{\beta}^{\mathrm{PR}}_{k+1} = \frac{\langle \overline{\mathbf{g}}_{k+1}, \overline{\mathbf{y}}_k \rangle}{\langle \mathcal{T}_{\mathbf{x}_{k+1}}(\overline{\mathbf{g}}_k), \mathcal{T}_{\mathbf{x}_{k+1}}(\overline{\mathbf{g}}_k) \rangle}, \qquad \widetilde{\beta}^{\mathrm{HS}}_{k+1} = \frac{\langle \overline{\mathbf{g}}_{k+1}, \overline{\mathbf{y}}_k \rangle}{\langle \overline{\mathbf{y}}_k, \mathcal{T}_{\mathbf{x}_{k+1}}(\mathbf{p}_k) \rangle}, \tag{4.8}$$

where $\overline{\mathbf{y}}_k = \overline{\mathbf{g}}_{k+1} - \mathcal{T}_{\mathbf{x}_{k+1}}(\overline{\mathbf{g}}_k)$. We use vector transport so each tangent vector is in $T_{\mathbf{x}_{k+1}}\mathcal{M}$. Similarly,

$$\widehat{\beta}^{\mathrm{PR}}_{k+1} = \frac{\langle \mathbf{g}_{k+1}, \overline{\mathbf{y}}_k \rangle}{\langle \mathcal{T}_{\mathbf{x}_{k+1}}(\mathbf{g}_k), \mathcal{T}_{\mathbf{x}_{k+1}}(\overline{\mathbf{g}}_k) \rangle}, \qquad \widehat{\beta}^{\mathrm{HS}}_{k+1} = \frac{\langle \mathbf{g}_{k+1}, \overline{\mathbf{y}}_k \rangle}{\langle \mathbf{y}_k, \mathcal{T}_{\mathbf{x}_{k+1}}(\mathbf{p}_k) \rangle}, \tag{4.9}$$

where $\mathbf{y}_k = \mathbf{g}_{k+1} - \mathcal{T}_{\mathbf{x}_{k+1}}(\mathbf{g}_k)$. The resulting manifold NPCG algorithm is presented in Algorithm 3.

5. Nonlinearly Preconditioned Nonlinear GMRES on Grassman Manifolds. N-GMRES [7, 29, 34] is an acceleration technique for iterations solving nonlinear systems, where a linear combination of past approximations is optimized to produce an improved search direction. As in Section 4 we first recall N-GMRES in the Euclidean setting before extending it to manifolds.

5.1. The N-GMRES Algorithm. Given a one-step iterative update function $P(\cdot)$ and a sequence of past updates $\{\mathbf{x}_i\}_{i=1}^k$, we compute $\overline{\mathbf{x}}_{k+1} = P(\mathbf{x}_k)$, then seek an improved approximation of the form

$$\widehat{\mathbf{x}}_{k+1} = \overline{\mathbf{x}}_{k+1} + \sum_{j=0}^{k} \alpha_j (\overline{\mathbf{x}}_{k+1} - \mathbf{x}_j).$$

For the optimality condition $\nabla f(\mathbf{x}) = \mathbf{g}(\mathbf{x}) = \mathbf{0}$, we seek α_j which approximately minimize $\|\mathbf{g}(\hat{\mathbf{x}}_{k+1})\|_2$ (which is a nonlinear function of each α_j). Linearization of $\mathbf{g}(\widehat{\mathbf{x}}_{k+1})$ about $\overline{\mathbf{x}}_{k+1}$ gives:

$$\mathbf{g}(\widehat{\mathbf{x}}_{k+1}) \approx \mathbf{g}(\overline{\mathbf{x}}_{k+1}) + \sum_{j=0}^{k} \left.\frac{\partial \mathbf{g}}{\partial \mathbf{x}}\right|_{\overline{\mathbf{x}}_{k+1}} \alpha_j (\overline{\mathbf{x}}_{k+1} - \mathbf{x}_j) \approx \mathbf{g}(\overline{\mathbf{x}}_{k+1}) + \sum_{j=0}^{k} \alpha_j (\mathbf{g}(\overline{\mathbf{x}}_{k+1}) - \mathbf{g}(\mathbf{x}_j)),$$

the second approximation eliminating the need for $k+1$ Hessian-vector products. The resulting objective function describes a least squares problem:

$$\left\| \mathbf{g}(\overline{\mathbf{x}}_{k+1}) + \textstyle\sum_{j=0}^{k} \alpha_j (\mathbf{g}(\overline{\mathbf{x}}_{k+1}) - \mathbf{g}(\mathbf{x}_j)) \right\|_2.$$

Let the j^{th} column of $\mathbf{A}$ be the vector $(\mathbf{g}(\overline{\mathbf{x}}_{k+1}) - \mathbf{g}(\mathbf{x}_j))$, $\mathbf{b} = \mathbf{g}(\overline{\mathbf{x}}_{k+1})$ and $\boldsymbol{\alpha} = (\alpha_0, \ldots, \alpha_k)^\intercal$. We may describe $\boldsymbol{\alpha}$ as the solution to the normal equations, $\mathbf{A}^\intercal\mathbf{A}\boldsymbol{\alpha} = -\mathbf{A}^\intercal\mathbf{b}$, and can solve for $\boldsymbol{\alpha}$ via any method for the linear least-squares problem (e.g. QR decomposition, the Moore–Penrose pseudoinverse). If a descent direction is obtained, a line search in the direction of $\widehat{\mathbf{x}}_{k+1}$ determines x_{k+1}:

$$\mathbf{x}_{k+1} = \overline{\mathbf{x}}_{k+1} + \beta(\widehat{\mathbf{x}}_{k+1} - \overline{\mathbf{x}}_{k+1});$$

otherwise, we discard the past iterates and restart the N-GMRES process. Details on the line search used in numerical experiments are provided in Section 6.

Algorithm 4 Manifold N-GMRES

procedure MNGMRES($\mathbf{x_0}, \ldots, \mathbf{x_{w-1}}$)
  $\overline{\mathbf{x}}_w \leftarrow$ one iteration HOOI($\mathbf{x}_{w-1}$)
  Evaluate $\mathbf{b} = \mathbf{g}(\overline{\mathbf{x}}_w)$
  Form $\mathbf{A}$ by (5.1) or (5.2) using $\mathbf{x}_j$, $j = 0, \ldots, w-1$
  Solve $\mathbf{A}^\intercal\mathbf{A}\boldsymbol{\alpha} = -\mathbf{A}^\intercal\mathbf{b}$ for $\boldsymbol{\alpha}$
  $\mathbf{p}_{w-1} \leftarrow -\sum_{j=0}^{w-1} \alpha_j \mathrm{Log}_{\overline{\mathbf{x}}_w}(\mathbf{x}_j)$
  $k \leftarrow w$
  while $\|\mathbf{g}(\mathbf{x}_{k-1})\|_2 >$ tol **do**
    Compute β_{k-1} via linesearch
    $\mathbf{x}_k \leftarrow R_{\mathbf{x}_{k-1}}(\beta_{k-1}\mathbf{p}_{k-1})$
    $\overline{\mathbf{x}}_k \leftarrow$ one iteration HOOI($\mathbf{x}_k$)
    Evaluate $\mathbf{b} = \mathbf{g}(\overline{\mathbf{x}}_k)$
    Form $\mathbf{A}$ by (5.1) or (5.2) using $\mathbf{x}_j$, $j = k-(w-1), \ldots, k$
    Solve $\mathbf{A}^\intercal\mathbf{A}\boldsymbol{\alpha} = -\mathbf{A}^\intercal\mathbf{b}$ for $\boldsymbol{\alpha}$
    $\mathbf{p}_k \leftarrow -\sum_{j=k-(w-1)}^{k} \alpha_j \mathrm{Log}_{\overline{\mathbf{x}}_w}(\mathbf{x}_j)$
    $k \leftarrow k+1$
  end while
end procedure

5.2. Extending N-GMRES to Manifolds. One iteration of the inner loop of HOOI is used as the nonlinear update $P(\cdot)$. As with NPCG, a number of changes are required when adapting to manifolds. To linearize $\mathbf{g}(\mathbf{x}) = \mathrm{grad} f(\mathbf{x})$, the Riemannian gradient of $f(\mathbf{x})$, we use the approximation [2, 15]

$$\mathbf{g}(\widehat{\mathbf{x}}_{k+1}) \approx \mathrm{grad} f(\overline{\mathbf{x}}_{k+1}) + \sum_{j=0}^{k} \alpha_j \mathrm{Hess} f(\overline{\mathbf{x}}_{k+1})[\boldsymbol{\xi}_j], \tag{5.1}$$

where $\boldsymbol{\xi}_j = -\mathrm{Log}_{\overline{\mathbf{x}}_{k+1}}(\mathbf{x}_j)$. This requires a known expression for the Hessian, as well as $k+1$ evaluations of the Hessian applied to a tangent vector. We may approximate the action of the Hessian to avoid these calculations by adapting the approach from [29]:

$$\mathrm{Hess}\, f(\overline{\mathbf{x}}_{k+1})[\boldsymbol{\xi}_j] \approx \mathrm{grad} f(\overline{\mathbf{x}}_{k+1}) - \mathcal{T}_{\overline{\mathbf{x}}_{k+1}}(\mathrm{grad} f(\mathbf{x}_j)), \tag{5.2}$$

where the past gradient is transported to $\overline{\mathbf{x}}_{k+1}$. The search direction $\mathbf{p}_{k+1} = \widehat{\mathbf{x}}_{k+1} - \overline{\mathbf{x}}_{k+1}$ is given by a linear combination of tangent vectors at $\overline{\mathbf{x}}_{k+1}$: $\mathbf{p}_{k+1} = -\sum_{j=0}^{k} \alpha_j \mathrm{Log}_{\overline{\mathbf{x}}_{k+1}}(\mathbf{x}_j)$. The line search is carried out along a retraction starting at $\overline{\mathbf{x}}_{k+1}$, now defined by $\mathbf{p}_{k+1}$. This process is described in Algorithm 4.

6. Implementation and Numerical Results. NPCG and N-GMRES were used to compute ATDs for order-3 tensors by minimizing $-\frac{1}{2}\left\|\mathcal{X}\cdot(\mathbf{A}^{(1)},\mathbf{A}^{(2)},\mathbf{A}^{(3)})\right\|_F^2$ on a product of three Grassmann manifolds. The gradient and Hessian are a matrix triplet and a mapping which accepts one matrix triplet and returns another, respectively. All computations in NPCG are well defined using matrix triplet form. Vectorization is required when forming the least squares system for N-GMRES.

The linesearch used was the Moré-Thuente algorithm from the Poblano Toolbox (v1.0) [12,23], which was modified to carry out the search along a curve described by a retraction. Specifically, we provide functions for the cost and gradient, as well as for the retraction, inner product, and norm on the manifold. Cost, gradient, inner products, and norms are replaced directly, and we replace the update $\mathbf{x} = \mathbf{x} + \alpha\mathbf{p}$ by $x = R_{\mathbf{x}}(\alpha\mathbf{p})$ (see (3.2)). The linesearch parameter values used were: 10^{-4} for the sufficient decrease condition tolerance, 10^{-2} for the curvature condition tolerance, an initial step length of 1, and a maximum of 20 iterations. The algorithms were implemented using Matlab R2010a on an Intel Core i7-2630QM computer with 8GB of RAM, using the Tensor Toolbox (V2.5) [3,4] to manipulate tensors.

HOSVD truncation (Algorithm 1) generated initial points. N-GMRES requires two or more past iterates to form the least squares system. A maximum of $w = 25$ past iterates were kept. If N-GMRES produced an ascent search direction, we discarded all past iterates and replaced the search direction by the negative of the ascent direction. NPCG methods were restarted by setting $\beta = 0$ every 50 iterations. Successful termination occurred when $\|\mathbf{g}_k\|_F < \mathrm{tol}\cdot|f(\mathbf{x}_k)|$. A scaled gradient tolerance of $\mathrm{tol} = 10^{-7}$ was used. When recording computation time, we omitted time spent checking the termination conditions.

Two test problems were considered. We compared four existing methods: HOOI [10], manifold NCG [18], a manifold limited memory BFGS (LBFGS) quasi-Newton solver [27], and the manifold trust region (TR) solver from the Manopt toolbox (V1.0.7) [1,6,19]; with the newly proposed methods: manifold NPCG and manifold N-GMRES. For LBFGS we use the same termination conditions and specified that vectors from the last 5 iterations were to be used in forming the Hessian approximation. For the TR solver, excepting the termination condition parameters, we use the default parameters prescribed by the code's authors: these are recorded in Table 6.1. We chose to use a finite difference Hessian approximation rather than the exact Hessian, as any benefits from improved accuracy are outweighed by the additional computing time required.

Parameter	Description	Value
$\overline{\Delta}$	Maximum trust-region radius	$\sqrt{\sum_n \mathrm{rank}_n(\hat{\mathcal{X}})}$
Δ_0	Initial trust-region radius	$\overline{\Delta}/8$
κ	tCG linear convergence target rate	0.1
θ	tCG superlinear convergence target rate	1.0
ρ'	Accept/reject threshold	0.1

TABLE 6.1
Trust Region Parameters

Problem I. For the first problem, we generated synthetic tensor data with specified size, multilinear rank, and noise level. Similar tests have been considered in [8, 15, 19, 32]. Given a Tucker tensor $\mathcal{X} \in \mathbb{R}^{120\times120\times120}$ with multilinear rank $(40, 40, 40)$, such that $\mathcal{S}$ has standard normal distributed entries and each $\mathbf{A}^{(n)}$ has orthonormal columns, test tensors are obtained by adding noise to $\mathcal{X}$. As in [31], given $\mathcal{N}_1$ and $\mathcal{N}_2$ with standard normal distributed entries, homoskedastic and heteroskedastic noise were added according to

$$\mathcal{X}' = \mathcal{X} + \frac{1}{3}\frac{\|\mathcal{X}\|_F}{\|\mathcal{N}_1\|_F}\mathcal{N}_1 \text{ and } \mathcal{X}'' = \mathcal{X}' + \frac{1}{3}\frac{\|\mathcal{X}'\|_F}{\|\mathcal{N}_2 * \mathcal{X}'\|_F}\mathcal{N}_2 * \mathcal{X}',$$

respectively, with final test tensor $\mathcal{X}''$.

Ten trials were run. In each trial ATDs of multilinear rank $(20, 20, 20)$ were computed for a $\mathcal{X}''$ obtained from the original $\mathcal{X}$ and new $\mathcal{N}_1$, $\mathcal{N}_2$. Results obtained from HOOI, LBFGS, TR, and NCG are compared with results from manifold NPCG using four variants for the β formulae and manifold N-GMRES using both linearizations mentioned. A maximum of 2000 iterations and a maximum computation time of 1500 seconds ensured all algorithms eventually terminated.

The minimum, median, maximum, and mean time (in seconds) and number of iterations required for convergence are recorded in Table 6.2. Of particular interest are the results of each method relative to HOOI, the standard algorithm used for solving such problems. We see that, with the exception of N-GMRES using (5.1), all methods succeeded in surpassing HOOI in terms of time required, whereas N-GMRES times were approximately ten times larger than the times for HOOI. The slowness of this N-GMRES variant is unsurprising, as evaluating the Hessian applied to a vector is extremely costly.

Among the methods faster than HOOI, TR performed the best in terms of both iteration count and time required. In terms of time required, it maintains a reasonable lead over the NPCG results, which performed the best of the newly proposed methods. The results show that the β^{PR} methods slightly outperformed the β^{HS} methods, but there were no clear winners between $\widehat{\beta}$ and $\widetilde{\beta}$ variants. After another significant gap in time required come the results for NCG and N-GMRES using (5.2). The two NCG variations gave similar results in terms of both time and iteration count. N-GMRES exhibited a lower median time and a higher mean time to convergence when compared to the NCG results, thus we do not provide a relative ranking of the two methods. Finally, LBFGS lags behind the other methods significantly.

Overall, the results for this synthetic test problem are promising: the best of the new NPCG and N-GMRES methods outperform each of the existing HOOI and LBFGS methods, and are competitive with the existing TR and NCG methods, which can be somewhat faster. In what follows we will see that for more difficult test problems (noisy and of larger size) accurate results can be obtained by the newly proposed NPCG and N-GMRES methods much faster and more robustly than by the existing methods considered.

		Time				Iterations			
		Min	Med	Max	Mean	Min	Med	Max	Mean
HOOI		20.99	36.91	95.68	40.54	287	506	1322	558
LBFGS		25.77	32.84	41.03	32.71	139	180	222	180
TR		6.85	**11.99**	18.11	**11.99**	21	33	50	33
NCG	β^{PR}	11.52	18.20	23.46	17.82	130	210	253	198
	β^{HS}	11.61	18.43	24.66	18.01	130	209	266	199
NPCG	$\widehat{\beta}^{PR}$	11.14	15.81	19.49	**15.43**	58	90	114	87
	$\widehat{\beta}^{HS}$	10.88	16.28	19.94	15.70	57	90	114	87
	$\widetilde{\beta}^{PR}$	11.97	**15.23**	21.02	15.79	64	86	125	89
	$\widetilde{\beta}^{HS}$	12.60	15.46	22.16	16.13	66	87	130	90
N-GMRES	(5.1)	295.01	391.00	573.67	406.35	64	79	122	82
	(5.2)	13.49	16.98	25.23	18.38	52	68	102	71

TABLE 6.2
Problem I. Results for computing ATDs of synthetic tensor data.

Problem II. The second test problem used the MNIST Database of Handwritten Digits [22], previously used in [26,33]. This is a collection of 70,000 images, each of a digit centered in a 28×28 image. We formed a tensor $\mathcal{X} \in \mathbb{R}^{28\times 28\times 5000}$ consisting of 5000 images of the digit 5. ATDs with multilinear rank $(14, 14, 100)$ were computed by the same methods considered in the previous example. The experiment was then repeated using $\mathcal{X}' = \mathcal{X} + 2.5\frac{\|\mathcal{X}\|}{\|\mathcal{N}\|}\mathcal{N}$, where $\mathcal{N}$ has entries uniformly distributed in $[0, 1]$. Upper limits of 250 iterations and 1500 seconds were imposed. Sample convergence histories are presented in Figure 6.1.

A striking result is that NCG, LBFGS, and TR failed to converge within a reasonable amount of time, compared to the other methods considered. While NCG and LBFGS simply fail to converge, TR does in fact converge, but requires much more time than HOOI and the newly proposed methods. We may be able to obtain significant improvements in efficiency by adapting the Manopt trust region method specifically to the ATD problem, though we do not believe that an efficiency-minded implementation will eliminate the speed gap observed.

In the noise free case, HOOI converges the fastest, followed by N-GMRES and then the NPCG variants. Once noise is introduced to generate $\mathcal{X}'$, we observe that N-GMRES and the two NPCG methods all converge

faster than HOOI, satisfying the tolerance in roughly one third of the time. For this problem NPCG using $\widehat{\beta}^{HS}$ was the fastest method, followed by N-GMRES and NPCG using $\widetilde{\beta}^{HS}$. The success of NPCG using $\widehat{\beta}^{HS}$ may be due to incorporating both gradient and preconditioner direction information into the $\widehat{\beta}$ formulae.

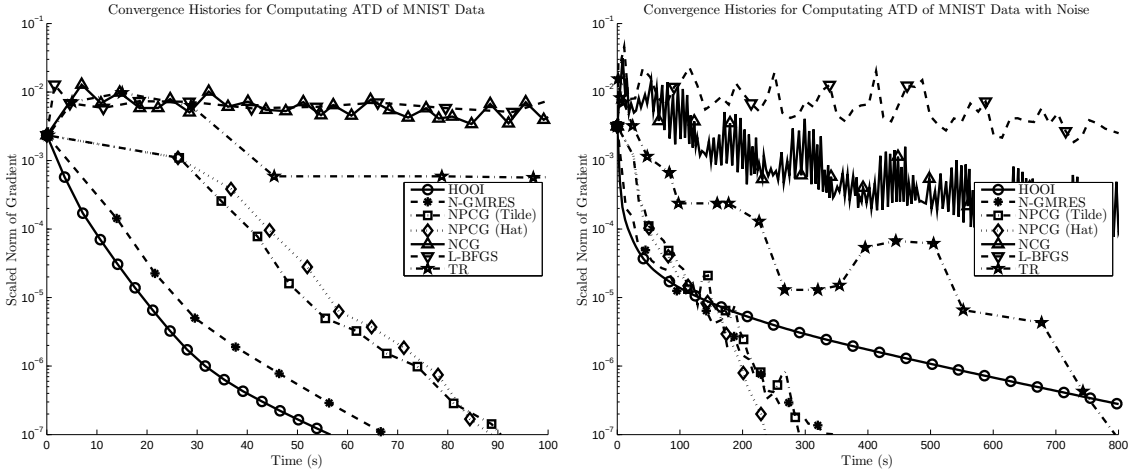

FIG. 6.1. *Problem II. Sample convergence histories based on MNIST digit input data: noise-free (left) and noisy (right).*

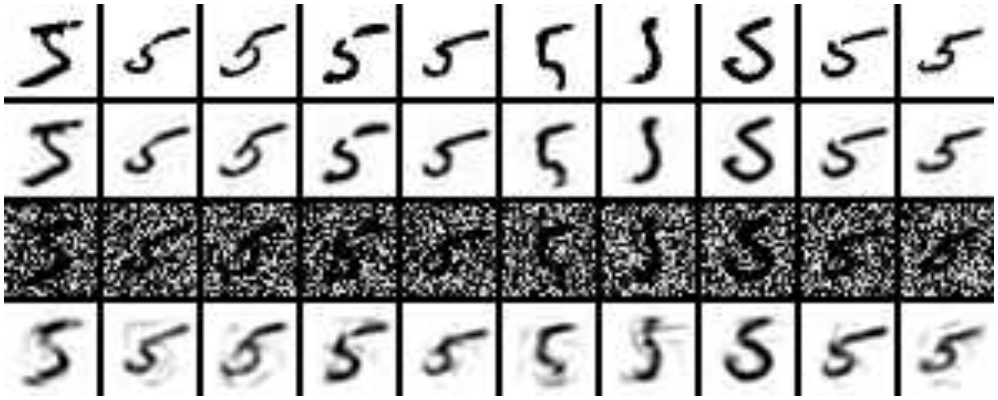

FIG. 6.2. *Problem II. From top to bottom, images are taken from: (1) $\mathcal{X}$, (2) ATD of $\mathcal{X}$, (3) $\mathcal{X}'$, (4) ATD of $\mathcal{X}'$.*

The first ten approximation images are shown in Figure 6.2. The main discrepancies in the ATD of $\mathcal{X}'$ are image artifacts in the form of dark blotches surrounding the digits. In spite of the extent to which the noise visually obfuscated the data, the ATD was able to successfully identify and recognizably display the digits of the original images.

These results suggest there are significant benefits to using manifold N-GMRES or NPCG to accelerate HOOI, these benefits having been most apparent for problems with noisy data or large dimensions. Furthermore, the advantages of our preconditioned methods over state-of-the-art methods that include manifold NCG, LBFGS, and trust region methods are significant in terms of convergence speed and robustness, as becomes apparent when moving to larger, more noisy, and more realistic data sets.

7. Concluding Remarks. The approximate Tucker decomposition is a tool commonly used in data compression and multilinear statistics and analysis. Hence, faster, more efficient methods for computing approximate Tucker decompositions will continue to be in demand. In this paper we extended acceleration methods for optimization, in particular, NPCG and N-GMRES, to the manifold setting, formulating accelerated optimization methods for approximate Tucker decompositions on manifolds. Numerical results provide evidence that HOOI accelerated by the N-GMRES algorithm approximating the action of the Hessian operator by a difference of gradients, or by NPCG iterations using any of the Polak–Ribière or Hestenes-Stiefel update rules, can significantly outperform HOOI by itself. Even more importantly, we have considered real

data wherein these newly proposed methods outperform manifold versions of NCG, limited memory BFGS quasi-Newton iterations, and a tCG based trust region algorithm, suggesting that our accelerated methods are leading contenders for efficient, approximate Tucker decompositions. The manifold versions of NPCG and N-GMRES we proposed can also be applied to other optimization problems on manifolds where simple iterative methods exist that may be accelerated.

REFERENCES

[1] P.-A. Absil, C. G. Baker, and K. A. Gallivan. Trust-region methods on Riemannian manifolds. *FoCM*, 7(3):303–330, 2007.

[2] P.-A. Absil, R. Mahony, and R. Sepulchre. *Optimization algorithms on matrix manifolds*. Princeton Press, 2009.

[3] B. Bader and T. Kolda. Algorithm 862: MATLAB tensor classes for fast algorithm prototyping. *ACM TOMS*, 32(4):635–653, December 2006.

[4] B. Bader, T. Kolda, et al. Matlab tensor toolbox version 2.5. Available online, January 2012.

[5] J. Ballani and L. Grasedyck. A projection method to solve linear systems in tensor format. *Numer. Lin. Algebra Appl.*, 20(1):27–43, 2013.

[6] N. Boumal, B. Mishra, P.-A. Absil, and R. Sepulchre. Manopt, a Matlab toolbox for optimization on manifolds. *JMLR*, 15:1455–1459, 2014.

[7] P. Brune, M. Knepley, B. Smith, and X. Tu. Composing scalable nonlinear algebraic solvers. *SIREV, forthcoming*, 2015.

[8] B. Chen, Z. Li, and S. Zhang. On optimal low rank tucker approximation for tensors: the case for an adjustable core size. *J. Global Optim.*, pages 1–22, 2014.

[9] A. Cichocki, D. Mandic, L. De Lathauwer, Guoxu Z., Qibin Z., C. Caiafa, and A.-H. Phan. Tensor decompositions for signal processing applications: From two-way to multiway component analysis. *IEEE Signal Process. Mag.*, 32(2):145–163, March 2015.

[10] L. De Lathauwer, B. De Moor, and J. Vandewalle. A multilinear singular value decomposition. *SIMAX*, 21(4):1253–1278, 2000.

[11] L. De Lathauwer, B. De Moor, and J. Vandewalle. On the best rank-1 and rank-$(r_1, r_2, \ldots, r_n)$ approximation of higher-order tensors. *SIMAX*, 21(4):1324–1342, 2000.

[12] D. Dunlavy, T. Kolda, and E. Acar. Poblano v1.0: A matlab toolbox for gradient-based optimization. Technical Report SAND2010-1422, Sandia National Laboratories, Albuquerque, NM and Livermore, CA, March 2010.

[13] C. Eckart and G. Young. The approximation of one matrix by another of lower rank. *Psychometrika*, 1(3):211–218, 1936.

[14] A. Edelman, T. Arias, and S. Smith. The geometry of algorithms with orthogonality constraints. *SIMAX*, 20(2):303–353, 1998.

[15] L. Eldén and B. Savas. A Newton-Grassmann method for computing the best multilinear rank-(r_1, r_2, r_3) approximation of a tensor. *SIMAX*, 31(2):248–271, 2009.

[16] W. Hager and H. Zhang. A survey of nonlinear conjugate gradient methods. *Pac. J. Optim.*, 2(1):35–58, 2006.

[17] M. Hestenes and E. Stiefel. *Methods of conjugate gradients for solving linear systems*, volume 49. National Bureau of Standards Washington, DC, 1952.

[18] M. Ishteva. *Numerical methods for the best low multilinear rank approximation of higher-order tensors*. PhD thesis, Department of Electrical Engineering, Katholieke Universiteit Leuven, 2009.

[19] M. Ishteva, P.-A. Absil, S. Van Huffel, and L. De Lathauwer. Best low multilinear rank approximation of higher-order tensors, based on the Riemannian trust-region scheme. *SIMAX*, 32(1):115–135, 2011.

[20] T. Kolda and B. Bader. Tensor decompositions and applications. *SIAMREV*, 51(3):455–500, 2009.

[21] T. Kolda and J. Sun. Scalable tensor decompositions for multi-aspect data mining. In *Eighth IEEE International Conference on Data Mining*, pages 363–372. IEEE, 2008.

[22] Y. Lecun and C. Cortes. The MNIST database of handwritten digits. `http://yann.lecun.com/exdb/mnist/`.

[23] J. Moré and D. Thuente. Line search algorithms with guaranteed sufficient decrease. *ACM TOMS*, 20(3):286–307, 1994.

[24] J. Nocedal and S. Wright. *Numerical Optimization*. Springer New York, 2006.

[25] E. Polak and G. Ribière. Note sur la convergence de méthodes de directions conjuguées. *ESAIM Math. Model. Numer. Anal.*, 3(R1):35–43, 1969.

[26] B. Savas and L. Eldén. Handwritten digit classification using higher order singular value decomposition. *Pattern recognition*, 40(3):993–1003, 2007.

[27] B. Savas and L.-H. Lim. Quasi-Newton methods on Grassmannians and multilinear approximations of tensors. *SISC*, 32(6):3352–3393, 2010.

[28] N. Son. A real time procedure for affinely dependent parametric model order reduction using interpolation on Grassmann manifolds. *Int. J. Numer. Meth. Eng.*, 93(8):818–833, 2013.

[29] H. De Sterck. A nonlinear GMRES optimization algorithm for canonical tensor decomposition. *SISC*, 34(3):A1351–A1379, 2012.

[30] H. De Sterck. Steepest descent preconditioning for nonlinear GMRES optimization. *Numer. Lin. Algebra Appl.*, 20(3):453–471, 2013.

[31] H. De Sterck and M. Winlaw. A nonlinearly preconditioned conjugate gradient algorithm for rank-r canonical tensor approximation. *Numer. Lin. Algebra Appl.*, 2014.

[32] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen. On the truncated multilinear singular value decomposition. *TW Reports*, TW589:1–34, 2011.

[33] N. Vannieuwenhoven, R. Vandebril, and K. Meerbergen. A new truncation strategy for the higher-order singular value decomposition. *SISC*, 34(2):A1027–A1052, 2012.

[34] T. Washio and C. Oosterlee. Krylov subspace acceleration for nonlinear multigrid schemes. *ETNA*, 6(271-290):3–1, 1997.